JETSON-NANO-DEV-KIT

From Waveshare Wiki
Jump to: navigation, search

Overview

Notice

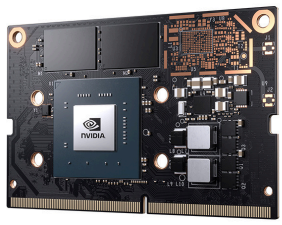
If you buy the DEV KIT from Waveshare, we have flashed a Jetpack 4.6 OS to the emmc of the Jetson Nano and enabled the SDMM3 (for TF card). If you need to modify the TF card startup, please refer to the manual to modify the startup path.
If you have special requirements for the factory image version, please contact the customer service of the Waveshare shop to communicate and confirm.
If you want to use spi and pwm, you need to reset it.

Introduction

JETSON-NANO-DEV-KIT is designed by Waveshare, based on the Jetson Nano Module, provides almost the same IOs, size, and thickness as the Jetson Nano Developer Kit (B01), more convenient for upgrading the core module. By utilizing the power of the core module, it is qualified for fields like image classification, object detection, segmentation, speech processing, etc., and can be used in sorts of AI projects.

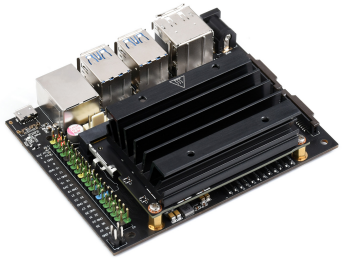
- Compared with the official B01 kit, JETSON-NANO-DEV-KIT cancels the TF card slot on the Nano core board and adds a 16GB eMMC memory chip, which is stable in reading and writing.
- The header switch of the DC power supply has been canceled, which means that we can directly insert the DC power supply into the circular interface and do not need to use a jumper cap to short the J48 interface first.
- Waceshare JETSON-NANO-DEV-KIT not only supports USB flash drive booting but also supports TF card booting as it has integrated a TF card slot on the expansion board. The reading and writing are stable, and it is almost the same as the B01 official core board with a TF card slot.

Jetson Nano Module



(/wiki/File:Jetson-Nano-Module-2.jpg)

NVIDIA Jetson Nano Module with 16G EMMC
JETSON-NANO-DEV-KIT



(/wiki/File:JETSON-NANO-DEV-KIT-A-2.jpg)

Jetson Nano dev kit
JETSON-NANO-LITE-DEV-KIT

(/wiki/File:Jetson-Nano-Module-3.jpg)

Compared with the conventional kit, JETSON-NANO-LITE-DEV-KIT simplifies the interface of the carrier board, the USB3.0 port is reduced from the original 4 to 1, and 2 USB2.0 ports are used instead, and the CSI port is reduced from the original 4 to 1. In addition, the carrier board of the Lite version also adds power and reset buttons. The carrier board of the Lite version is compatible with the original official Jetson Nano 2GB Developer Kit in terms of appearance and interface. It is suitable for users who do not require more interface resources. The core board of the Lite version also uses the Jetson Nano Module 4GB version.

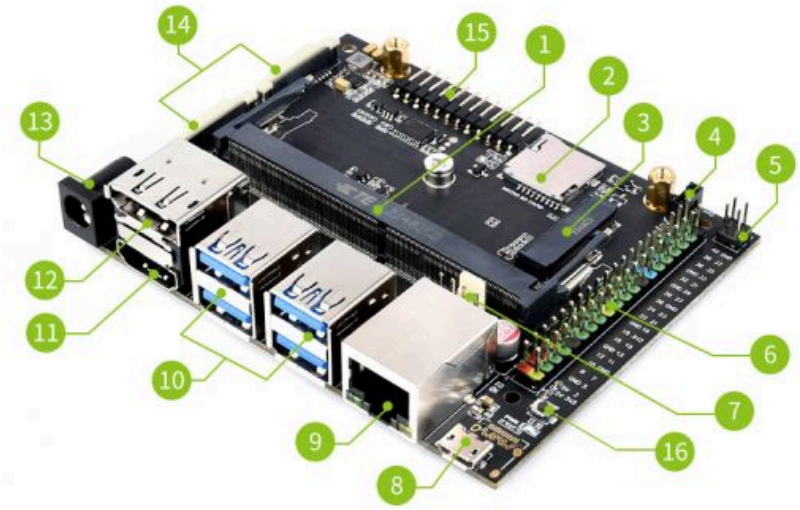
- JETSON-NANO-LITE-DEV-KIT also has a TF card slot, which is convenient for users to do TF card expansion.
- Compared with JETSON-NANO-DEV-KIT, the Lite version kit has two more onboard buttons, which are used to control the mainboard power supply and system reset respectively.
- The carrier board of JETSON-NANO-LITE-DEV-KIT is basically compatible with the official Jetson Nano 2GB Developer Kit, which is suitable for users who do not require many interfaces of the carrier boards but want the performance of 4GB Nano modules.

Jetson Nano Module Parameter

GPU	NVIDIA Maxwell™ architecture with 128 NVIDIA CUDA® cores and 0.5 TFLOPS (FP16)
CPU	Quad-core ARM® Cortex®-A57 MPCore processor
Memory	4 GB 64-bit LPDDR4 1600 MHz – 25.6 GB/s
Storage	16 GB eMMC 5.1 Flash
Video Encode	250 MP/s 1 x 4K @ 30 (HEVC) 2 x 1080p @ 60 (HEVC) 4 x 1080p @ 30 (HEVC)
Video Decode	500 MP/s 1 x 4K @ 60 (HEVC) 2 x 4K @ 30 (HEVC) 4 x 1080p @ 60 (HEVC) 8 x 1080p @ 30 (HEVC)
Camera	12-ch (3x4 or 4x2) MIPI CSI-2 D-PHY 1.1 (18 Gbps)
Connectivity	Wi-Fi requires external chip 10/100/1000 BASE-T Ethernet
Display	eDP 1.4 | DSI (1 x 2) 2 simultaneous
UPHY	1 x 1/2/4 PCIE, 1 x USB 3.0, 3 x USB 2.0

IO	3 x UART, 2 x SPI, 2 x I2S, 4 x I2C, multi GPIO headers

JETSON-IO-BASE-A Onboard Resources



(/wiki/File:JETSON-IO-BASE-A-details-intro02.jpg)

1. Core module socket

   Insert Jetson Nano core board

2. TF card slot

   TF card can be connected for TF card expansion

3. M.2 Key E connector

   AC8265 wireless network card can be connected

4. 1.25mm fan header
5. PoE pin

   PoE module is not included

6. 40PIN GPIO header

   Compatible with Raspberry Pi pins, convenient for Raspberry Pi peripherals (requires program support)

7. 2.54mm fan header
8. Micro USB port

   for 5V power input or for USB data transmission

9. Gigabit Ethernet port

   10/100/1000Base-T auto-negotiation, supports PoE if external PoE module is connected

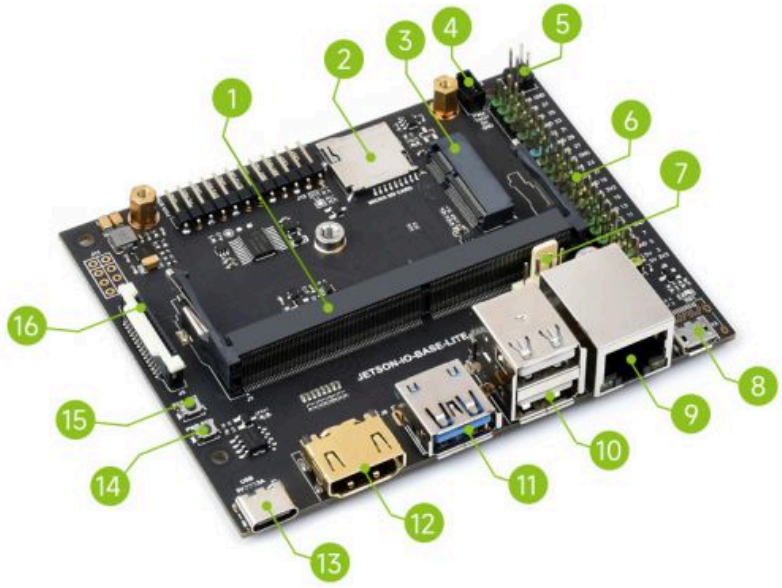
10. 4x USB 3.0 port
11. HDMI output port
12. DisplayPort connector

13. DC jack

    for 5V power input

14. 2x MIPI CSI camera connector
15. Multi-function 12PIN header
    - You can configure the Nano by short-circuiting the corresponding pins using jumpers/wires.
    - Connect PWR_BTN and GND pin to initiate power-on if Auto-Power-On disabled.
    - Connect DIS and AUTO ON to disable Auto-Power-On and require power automation press.
    - Connect FC REC and GND during power-on to put system in USB Force Recovery mode.
    - Temporarily connect GND and SYS_RST to reset system.
    - UART TXD and UART RXD are the UART debugging pins.
    - LED+ and LED- indicate System Sleep/Wake [Off when system in sleep mode].
16. PWR button
    - This button is adapted through the 12PIN, and its function is the same as the PWR BTN pin.
    - Temporarily pressing this button can power the Nano when powered off.
    - Temporarily pressing this button will initiate a soft shutdown in a normal power-on state, and it will automatically power off if no power-off option is selected for a while.
    - A long press of 15 seconds or more will force Nano to power off.

JETSON-IO-BASE-LITE Onboard Resources



(/wiki/File:JETSON-IO-BASE-A-details-intro2.jpg)

1. Core module socket
2. TF card slot
3. M.2 Key E interface
4. 1.25mm fan header
5. PoE pin

   PoE module is not included

6. 40PIN GPIO header
7. 2.54mm fan header
8. Micro USB interface

   User USB data transfer/system programming

9. Gigabit Ethernet port

   10/100/1000Base-T self-adaptive, access to PoE module can support PoE power supply

10. 2x USB 2.0 ports
11. USB 3.2 Gen1 port
12. HDMI high-definition interface
13. USB Type-C port

    For 5V 3A power input

14. PWR BTN

    Power button

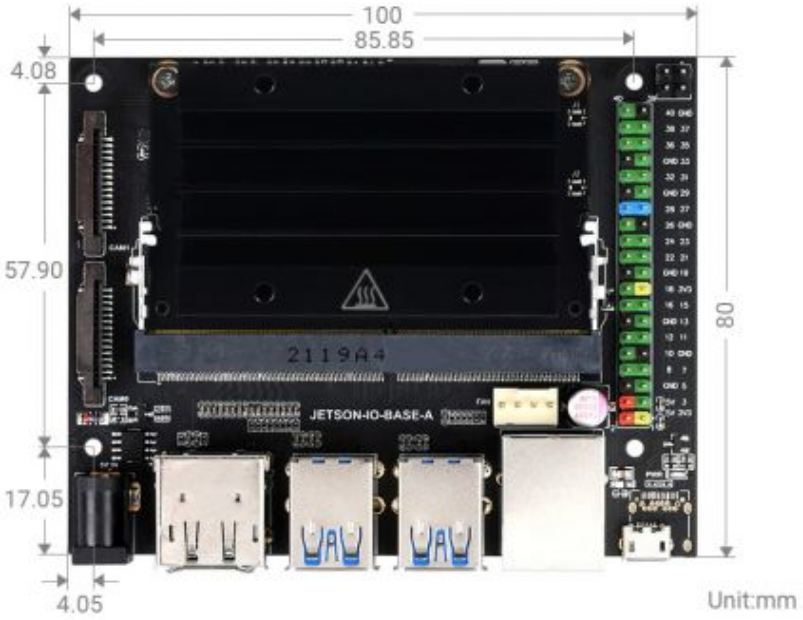
15. SYS RST

    Reset button

16. MIPI CSI camera connector

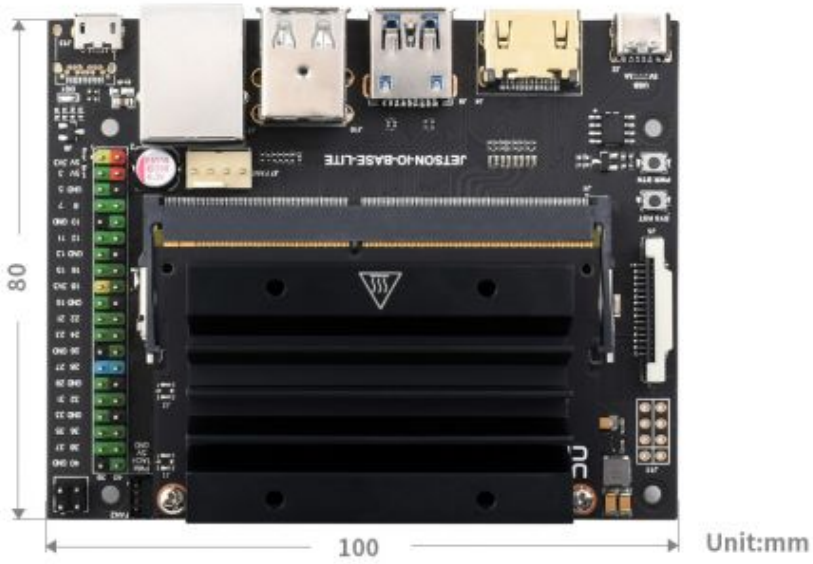
Dimension

- **JETSON-NANO-DEV-KIT**



(/wiki/File:Dimension002.jpg)

- **JETSON-NANO-LITE-DEV-KIT**



(/wiki/File:Dimension02.jpg)

User Guide

System Installation

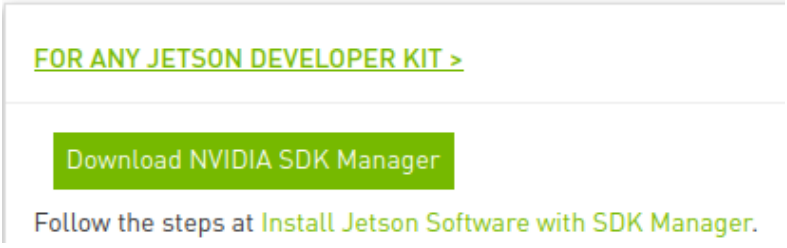
System Environment & EMMC System Programming

- Programming system requires the Ubuntu 18.04 host or virtual machine and adopts the SDK Manager tool to program.
- The system environment in this manual adopts VMware 16 virtual machine to install Ubuntu 18.04 system, only for learning.
- If you have an Ubuntu virtual machine or host but not 18.04 and can accept formatted TF cards or USB flash drives, please refer to #Method Two: Directly Download Jetpack.

Method One: Adopt SDK Manager Tool

1. Download SDK Manager: open the browser and enter the URL, click to download SDK Manager.

```
https://developer.download.nvidia.com/sdkmanager/redirects/sdkmanager-deb.html
```

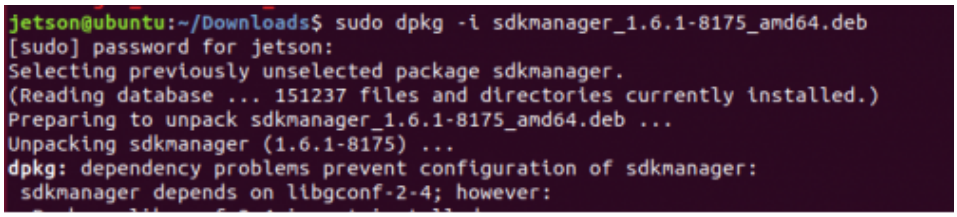


(/wiki/File:JETSON-NANO-MANUAL01.png)

You need to register an account. After logging in, we can download it successfully. If you don't know how to register, you can refer to NVIDIA-acess (https://www.waveshare.com/wiki/NVIDIA-acess).

2. After the download is complete, we enter the download path "Downloads" to install, and input the following content in the terminal:

```
sudo dpkg -i sdkmanager_1.6.1-8175_amd64.deb (enter according to your version).
```



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL02.png)

3. After the installation is complete, the system may report an error that the dependency files cannot be found. Enter the following command to solve this problem.

```
sudo apt --fix-broken install
```

4. Open the Ubuntu computer terminal and run the SDK Manager to open the software.
5. Click LOGIN, log in to the NVIDIA account, and a link will pop up in the browser, enter the previous registered email and password to log in.

(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL03.png)

6. At this point, we have successfully logged in to the SDK Manager.

(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL04.png)

Install Image on EMMC

Equipment Preparation

1. Jetson Nano board
2. Ubuntu 18.04 virtual machine (or host computer)
3. 5V/4A power adapter
4. Jumper caps (or Dupont wire)
5. USB data cable (Micro USB port, can transfer data)

Hardware Configuration (Enter Recovery Mode)

1. Use jumper caps or Dupont wires to short-circuit the FC REC and GND pins, located below the core board, as shown in the figure below.
2. Connect the DC power supply to the circular power port and wait a moment.
3. Connect the Micro USB port of the Jetson Nano to the Ubuntu host with a USB cable (note that it is a data cable).

(/wiki/File:500px-Jetson-nano-Force_recovery2-%E6%B0%B4%E5%8D%B0-1.png)

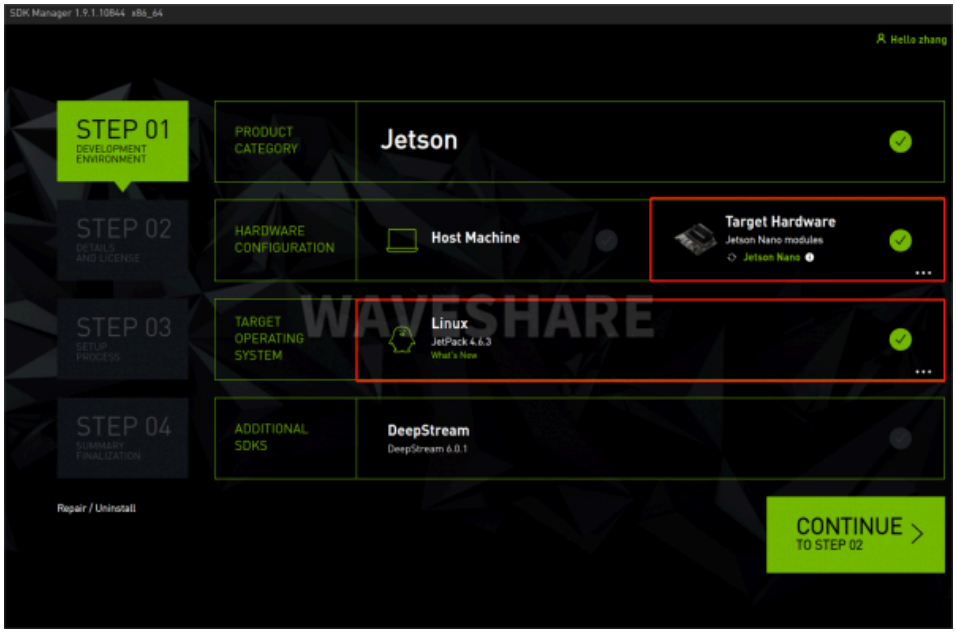
System Programming

1. Open the Ubuntu computer terminal and run the SDK Manager to open the software.
2. Using a virtual machine requires setting up the device to connect to the virtual machine.

3. Log in to your account, if the Jetson Nano is recognized normally, SDK Manager will detect and prompt options.

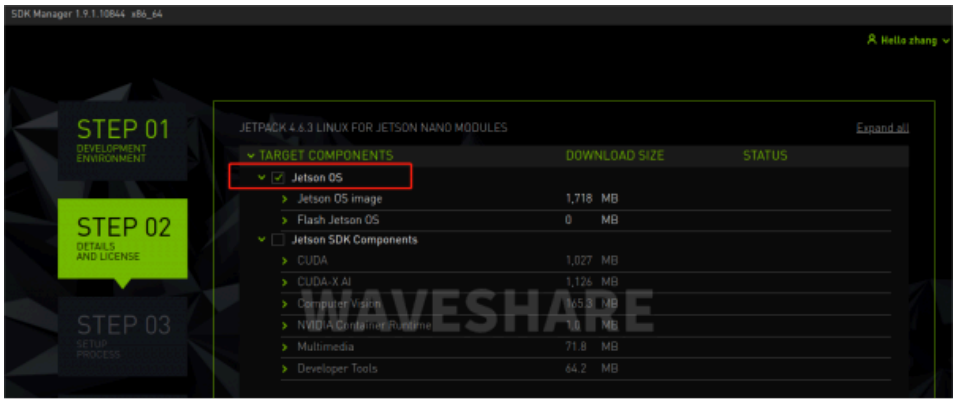
(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL09.png)

4. In the JetPack option, take the JetPack4.6 system as an example, uncheck Host Machine, and click CONTINUE.



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL10re.png)

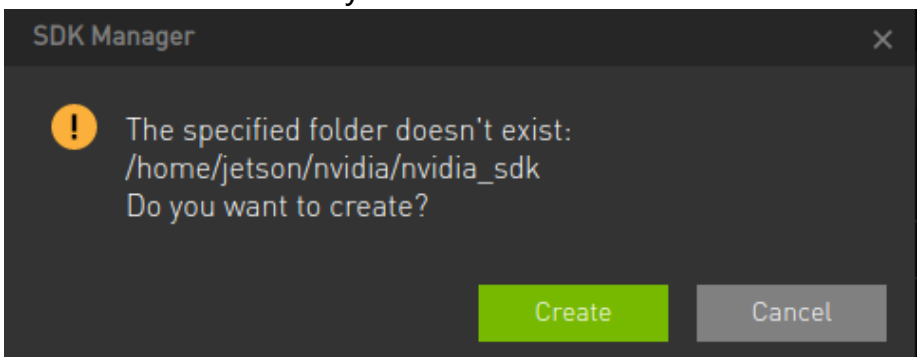
5. Select Jetson OS, and **uncheck Jetson SDK Components**. Check the protocol and click CONTINUE.



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL11re.png)

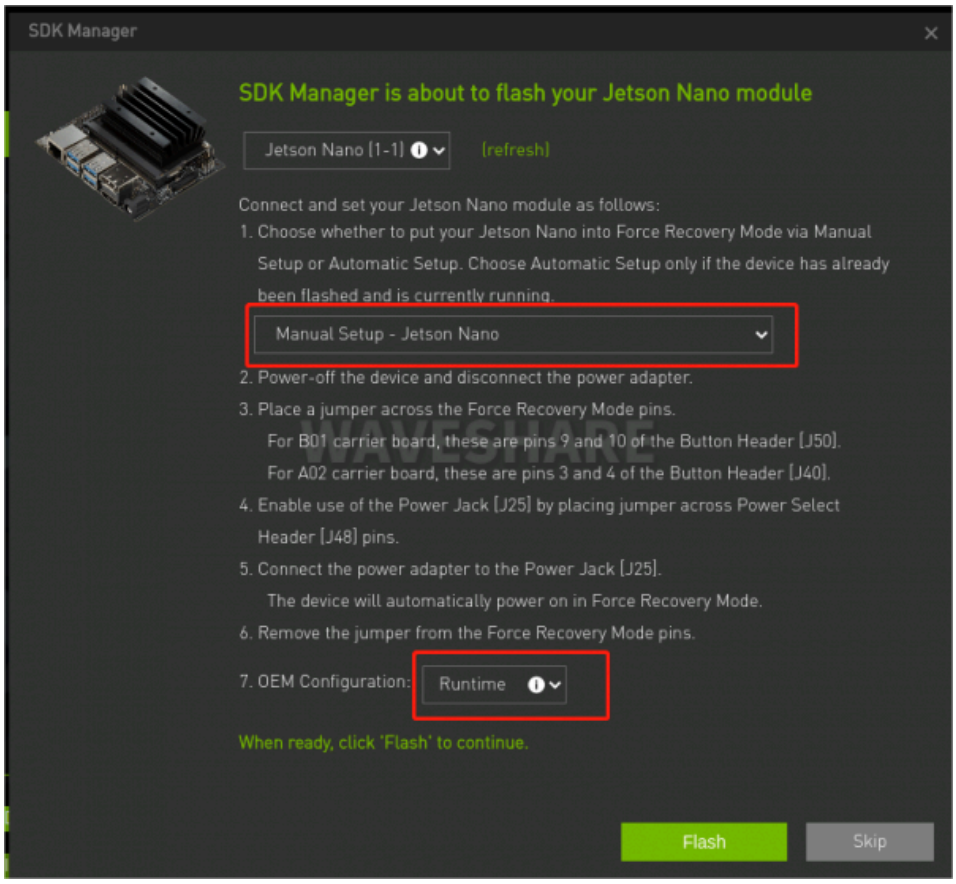
Note: Checking both will cause the download to fail.

6. The default path for saving images with HW Imager is fine. Select Create and the path will be created automatically.



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL12.png)

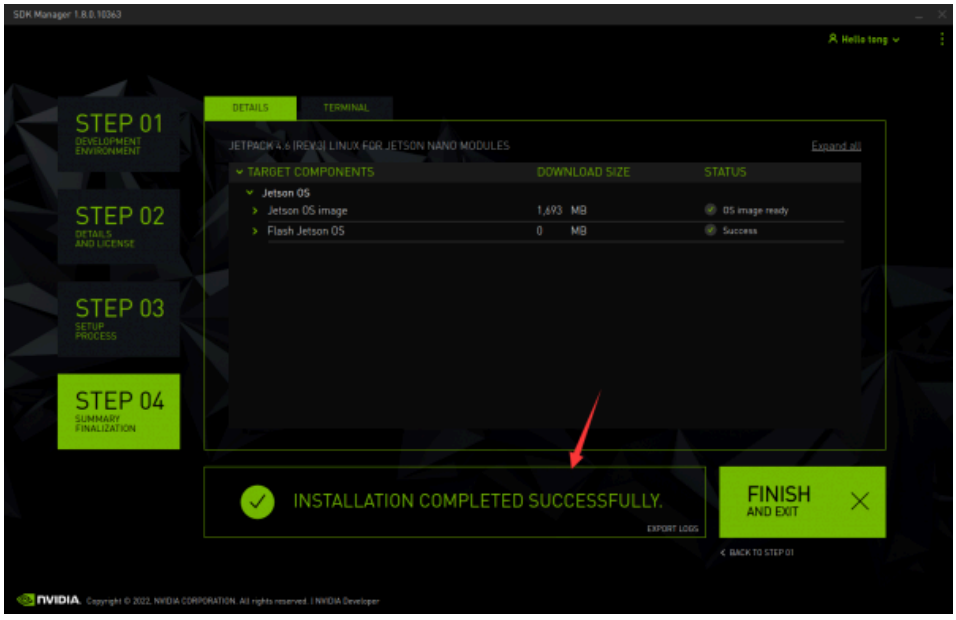
7. Starting from JetPack version 4.6.1, the pre-config window will pop up when programming the system with SDK Manager, select according to the figure below.



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL041re.png)

Note: The Pre-config option in 7. OME Configuration is to set the username and password in advance, and the Runtime is to set the username and password during the boot configuration of the Jetson Nano.

8. Enter the password of the virtual machine, wait for the download, and the programming is finished.



(/wiki/File:JETSON-NANO-DEV-KIT-MANUAL13.png)

9. Due to the official update of the memory for the Jetson Nano module, all Jetpack versions need to replace some files in order to boot up properly, otherwise they will only stop at the Nvidia logo interface:
Open the terminal

```
cd nvidia/nvidia_sdk/JetPack_4.X.X_Linux_JETSON_NANO_TARGETS (4. X.X is subject to t
he actual folder)
wget https://developer.nvidia.com/downloads/embedded/L4T/r32_Release_v7.5/overlay_3
2.7.5_PCN211181.tbz2
sudo tar -xjf overlay_32.7.5_PCN211181.tbz2
```

10. Re-execute the above programming process.
11.
12. After the programming is finished, remove the jump cap of the bottom panel, connect to the monitor, power on it again, and follow the prompts to configure the boot (if it is a pre-config set, enter the system directly after powering on).

Method Two: Directly Download Jetpack

The following Jetpack download is based on Jetpack 4.6.2 as an example, for other Jetpack version resource pack download methods, please refer to the Jetpack download method in the FAQ (https://www.waveshare.com/wiki/JETSON-NANO-DEV-KIT#accordion4).

Install System

1. Open the terminal on the Ubuntu virtual machine or host and create a new folder.

```
sudo mkdir sources_nano
cd sources_nano
```

2. Download path:

```
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/jetson-210_linux_r3
2.7.2_aarch64.tbz2
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/tegra_linux_sample-r
oot-filesystem_r32.7.2_aarch64.tbz2
```

   Move the Jetpack to a folder and extract it (in practice, try to use the tab button to automatically complete the instructions).

```
sudo mv ~/Downloads/Jetson-210_Linux_R32.7.2_aarch64.tbz2 ~/sources_nano/
sudo mv ~/Downloads/Tegra_Linux_Sample-Root-Filesystem-R32.7.2_aarch64.tbz2 ~/source
s_nano/
```

3. Unzip resource.

```
sudo tar -xjf Jetson-210_Linux_R32.7.2_aarch64.tbz2
cd Linux_for_Tegra/rootfs/
sudo tar -xjf .. /.. /Tegra_Linux_Sample-Root-Filesystem_R32.7.2_aarch64.tbz2
cd .. /
sudo ./apply_binaries.sh (If an error occurs, follow the prompts and re-enter the in
struction).
```

4. Due to the official update of the memory for the Jetson Nano module, all Jetpack versions need to replace some files in order to boot up properly, otherwise they will only stop at the Nvidia logo interface:

```
cd ..
wget https://developer.nvidia.com/downloads/embedded/L4T/r32_Release_v7.5/overlay_3
2.7.5_PCN211181.tbz2
sudo tar -xjf overlay_32.7.5_PCN211181.tbz2
```

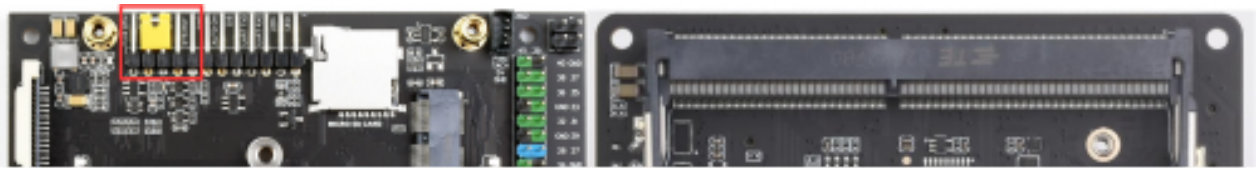
Install Image on EMMC

Equipment preparation

1. Jetson Nano board
2. Ubuntu virtual machine (or host computer)
3. 5V 4A power adapter
4. Jumper caps (or DuPont cable)
5. USB Data cable (Micro USB interface, can transfer data)

Hardware Configuration (entering recovery mode)

1. Short-connect the FC REC and GND pins with a jump cap or DuPont wire, located below the core board, as shown below.
2. Connect the DC power supply to the round power supply port and wait a while.
3. Connect the Jetson Nano's Micro USB port to the Ubuntu host with a USB cable (note that it is a data cable).



(/wiki/File:500px-Jetson-nano-Force_recovery2-%E6%B0%B4%E5%8D%B0-1.png)

System Programming

1. Programming system, Jetson Nano needs to enter recovery mode and connect to the Ubuntu computer.

```
cd ~/sources_nano/Linux_for_Tegra
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

2. After the programming is finished, remove the jumping cap of the bottom panel, connect to the monitor, power on it again, and follow the prompts to configure the boot (if it is a pre-config set, enter the system directly after powering on).

USB Flash Drive And TF Card Booting Principle

- You need to start the system on the EMMC in the module first, and then the system of the module will be booted to the USB flash drive or the TF card.
- The system in the module can use the SDK Manager in the virtual machine to program the system; the TF card system can program the system with Win32DiskImager; the system in the USB flash drive uses the virtual machine to program the system.
- Before booting the USB flash drive or TF card, you need to make sure the EMMC system has been programmed.

Boot USB Flash Drive (copy the system on eMMC)

Preparation

1. Jetson Nano board
2. USB flash drive or mobile hard disk with USB interface (USB3.0 is recommended)
3. 5V/4A power adapter

System Installation

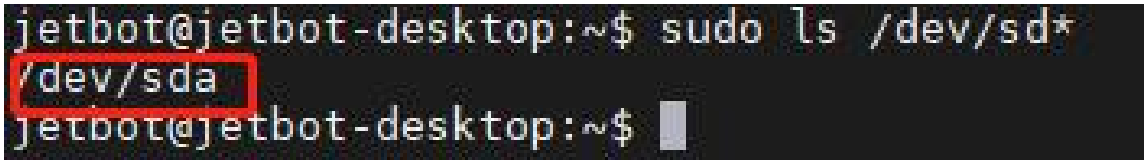
1. Connect the USB flash drive to the Jetson Nano, check the device number of the USB flash drive, such as sda, open the Jetson Nano terminal and enter.

```
ls /dev/sd*
```

2. Format the USB flash drive.

```
sudo mkfs.ext4 /dev/sda
```

Only SDA remains, as shown below:



(/wiki/File:Device_number.jpg)

3. Modify the startup path.

```
sudo vi /boot/extlinux/extlinux.conf
```

Find the statement "APPEND ${cbootargs} quiet root=/dev/mmcblk0p1 rw rootwait rootfstype=ext4 console=ttyS0,115200n8 console=tty0", modify "mmcblk0p1" to "sda".

4. Mount the USB flash drive.

```
sudo mount /dev/sda /mnt
```

5. Copy the system to the USB flash drive (the process has no information to print, please wait patiently).

```
sudo cp -ax / /mnt
```

6. After the copy is completed, uninstall the USB flash drive (do not unplug the USB flash drive).

```
sudo umount /mnt/
```

7. Restart the system.

```
sudo reboot
```

Boot TF Card

Method 1: Copy the system on eMMC directly

Note: The operation will format the TF card.

Equipment preparation

1. Jetson Nano board.
2. 5V 4A power adapter.

Burn Boot Program

1. Install the DTC software on the virtual machine.

```
sudo apt-get install device-tree-compiler
```

2. Enter the HW Imager kernel path and decompile the dts source file.
   - If you are using the SDK Manager software, use the following command:

```
cd ~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_NANO_TARGETS/Linux_for_Tegra/kernel/
dtb    #You can modify the path for different jetpacks
sudo dtc -I dtb -O dts -o tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002
-p3449-0000-b00.dtb
```

   - If you are using a resource pack, use the following command:

```
cd sources_nano/Linux_for_Tegra/kernel/dtb
sudo dtc -I dtb -O dts -o tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002
-p3449-0000-b00.dtb
```

3. Modify the device tree.

```
sudo vim tegra210-p3448-0002-p3449-0000-b00.dts
```

4. Find the sdhci@700b0400 section, change status = "disable" to okay, and add TF information below.

```
cd-gpios = <0x5b 0xc2 0x0>;
sd-uhs-sdr104;
sd-uhs-sdr50;
sd-uhs-sdr25;
sd-uhs-sdr12;

no-mmc;
uhs-mask = <0xc>;
```

```
sdhci@700b0400 {
        compatible = "nvidia,tegra210-sdhci";
        reg = <0x0 0x700b0400 0x0 0x200>;
        interrupts = <0x0 0x13 0x4>;
        aux-device-name = "sdhci-tegra.2";
        iommus = <0x30 0x1b>;
        nvidia,runtime-pm-type = <0x0>;
        clocks = <0x26 0x45 0x26 0xf3 0x26 0x136 0x26 0xc1>;
        clock-names = "sdmmc", "pll_p", "pll_c4_out2", "sdmmc_legacy_tm";
```

(/wiki/File:Nano-tf-dts_watermark.png)

5. Compile dtb files.

```
sudo dtc -I dts -O dtb -o tegra210-p3448-0002-p3449-0000-b00.dtb tegra210-p3448-0002
-p3449-0000-b00.dts
```

6. To program the system, Jetson Nano needs to enter recovery mode and connect to the Ubuntu computer.
   - If you are using the SDK Manager software, use the following command:

```
cd ~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_NANO_TARGETS/Linux_for_Tegra
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

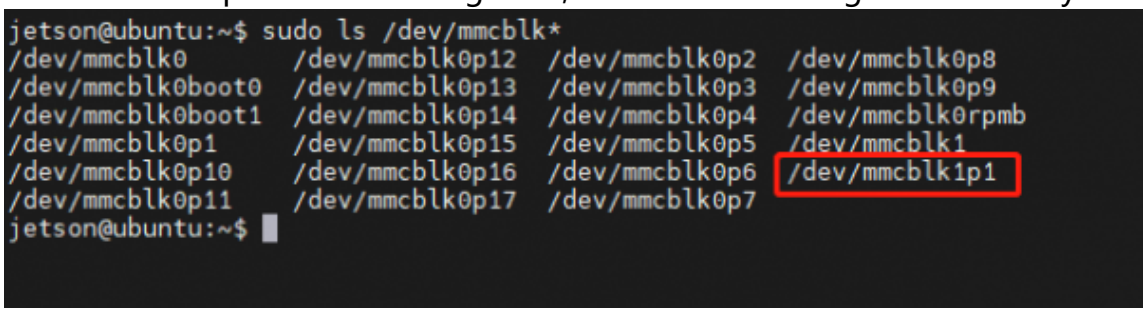
- If you are using a resource pack, use the following command:

```
cd sources_nano/Linux_for_Tegra
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

7. Check if the TF card is identified:

```
sudo ls /dev/mmcblk*
```

8. If the mmcblk1p1 device is recognized, the TF card is recognized normally.

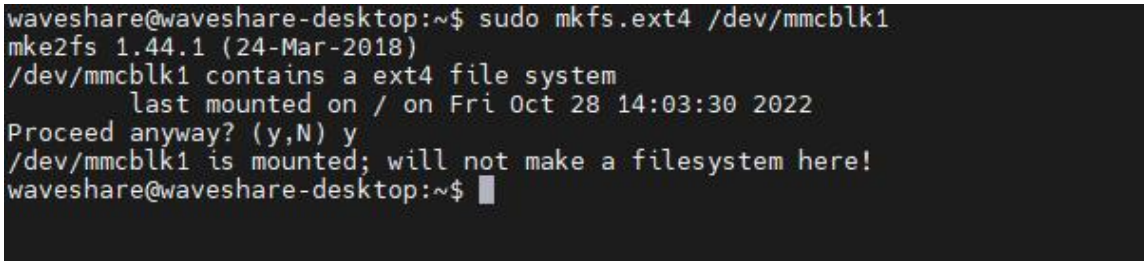


(/wiki/File:SD_card_recognition_watermark_.png)

9. Format the TF card.

```
sudo mkfs.ext4 /dev/mmcblk1
```

If the following message appears, a file system is already available.



(/wiki/File:File_system.jpg)

Unmount the TF card first:

```
sudo umount /media/ (here press the Tab key to complete automatically).
```

Format the TF card again using the format command.

After successful formatting, enter:

```
sudo ls /dev/mmcblk*
```

There is only mmcblk1, as shown below.

```
waveshare@waveshare-desktop:~$ sudo ls /dev/mmcblk*
/dev/mmcblk0       /dev/mmcblk0p12  /dev/mmcblk0p2  /dev/mmcblk0p8
/dev/mmcblk0boot0  /dev/mmcblk0p13  /dev/mmcblk0p3  /dev/mmcblk0p9
/dev/mmcblk0boot1  /dev/mmcblk0p14  /dev/mmcblk0p4  /dev/mmcblk0rpmb
/dev/mmcblk0p1     /dev/mmcblk0p15  /dev/mmcblk0p5  /dev/mmcblk1
/dev/mmcblk0p10    /dev/mmcblk0p16  /dev/mmcblk0p6
/dev/mmcblk0p11    /dev/mmcblk0p17  /dev/mmcblk0p7
waveshare@waveshare-desktop:~$
```

(/wiki/File:Config1.jpg)

10. Modify the startup path.

```
sudo vi /boot/extlinux/extlinux.conf
```

Find the statement APPEND ${cbootargs} quiet root=/dev/mmcblk0p1 rw rootwait rootfstype=ext4 console=ttyS0,115200n8 console=tty0, modify mmcblk0p1 to mmcblk1 to save.

11. Mount the TF card.

```
sudo mount /dev/mmcblk1 /mnt
```

12. Copy the system to the TF card (the process has no information to print, please wait patiently).

```
sudo cp -ax / /mnt
```

13. After the copy is complete, uninstall the TF card (do not unplug the TF card).

```
sudo umount /mnt/
```

14. Restart the system.

```
sudo reboot
```

Method 2: Another Image on the TF Card

Equipment preparation

1. Jetson Nano board.
2. 5V 4A power adapter.

Burn Boot Program

1. Install the DTC software on the virtual machine.

```
sudo apt-get install device-tree-compiler
```

2. Enter the HW Imager kernel path and decompile the dts source file (modify the corresponding path for different jetpacks).

```
cd ~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_NANO_TARGETS/Linux_for_Tegra/kernel/
dtb
sudo dtc -I dtb -O dts -o tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002
-p3449-0000-b00.dtb
```

3. Modify the device tree.

```
sudo vim tegra210-p3448-0002-p3449-0000-b00.dts
```

4. Find the sdhci@700b0400 section, change status = "disable" to okay, and add TF information below.

```
cd-gpios = <0x5b 0xc2 0x0>;
sd-uhs-sdr104;
sd-uhs-sdr50;
sd-uhs-sdr25;
sd-uhs-sdr12;

no-mmc;
uhs-mask = <0xc>;
```

```
sdhci@700b0400 {
        compatible = "nvidia,tegra210-sdhci";
        reg = <0x0 0x700b0400 0x0 0x200>;
        interrupts = <0x0 0x13 0x4>;
        aux-device-name = "sdhci-tegra.2";
        iommus = <0x30 0x1b>;
        nvidia,runtime-pm-type = <0x0>;
        clocks = <0x26 0x45 0x26 0xf3 0x26 0x136 0x26 0xc1>;
        clock-names = "sdmmc", "pll_p", "pll_c4_out2", "sdmmc_legacy_tm";
```

(/wiki/File:Nano-tf-dts_watermark.png)

5. Compile dtb files.

```
sudo dtc -I dts -O dtb -o tegra210-p3448-0002-p3449-0000-b00.dtb tegra210-p3448-0002
-p3449-0000-b00.dts
```

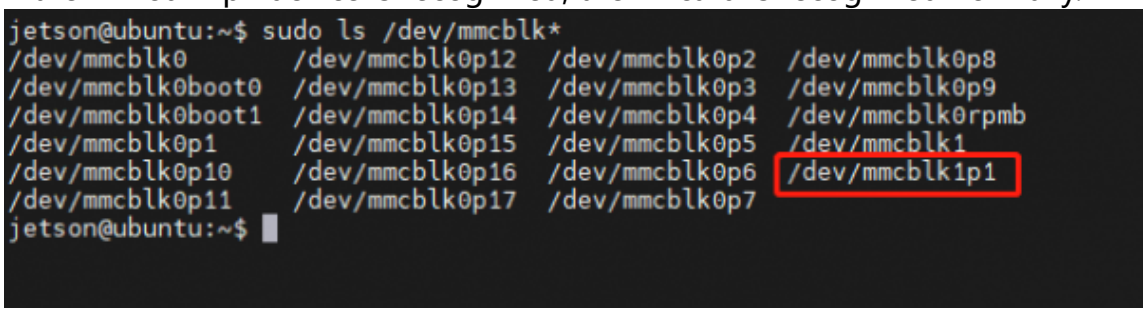
6. Programming system, Jetson Nano needs to enter recovery mode and connect to the Ubuntu computer.

```
cd ~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_NANO_TARGETS/Linux_for_Tegra
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

7. Disconnect the USB cable and jumping cap, and the Jetson Nano is powered on and configured.
8. Check if the TF card is identified:

```
sudo ls /dev/mmcblk*
```

9. If the mmcblk1p1 device is recognized, the TF card is recognized normally.



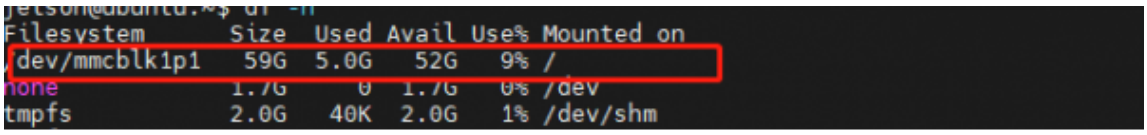
(/wiki/File:SD_card_recognition_watermark_.png)

10. Modify the boot system from the TF card (optional).

```
sudo vi /boot/extlinux/extlinux.conf
```

- Find the statement "APPEND ${cbootargs} quiet root=/dev/mmcblk0p1 rw rootwait rootfstype=ext4 console=ttyS0,115200n8 console=tty0", change "mmclk0p1" to "mmcblk1p1", save, and then restart the system.
- If you are using the image we provide, the user name after rebooting the system is jetson, and the password is jetson, if it is the image installed by the user, it is the user name and password set by the user.

Note: If the TF card memory is 64G, after entering the system, open the terminal, and enter df -h to check the disk size if the space size is not normal, please refer to the expansion image in the FAQ.



(/wiki/File:Memory_watermark.png)

Login

Offline Login

- Use the image provided by Waveshare, the user name and user password are as follows:

```
Username: waveshare
User password: waveshare
```

Remote Login

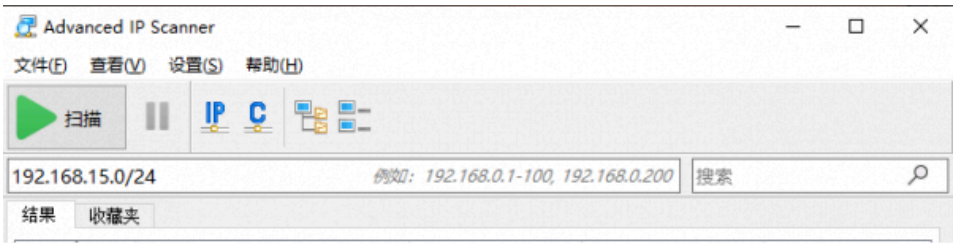
Preparation

- Please connect Jetson Nano by a network cable, and then connect the LAN port of the router with the other end of the cable.
- Please make sure Jetson Nano and your computer are under the same router or the network segment.

Get Jetson Nano IP

- Method 1: Log in to the router to find the IP of Jetson Nano.
- Method 2: You can use some LAN IP scanning tools, here is an example of Advanced IP Scanner (https://www.advanced-ip-scanner.com/).

1. Run Advanced IP Scanner.
2. Click the "Scan" button to scan the IP in the current LAN.
3. Find **all** IP with the word "NVIDIA" in Manufacture and then record.

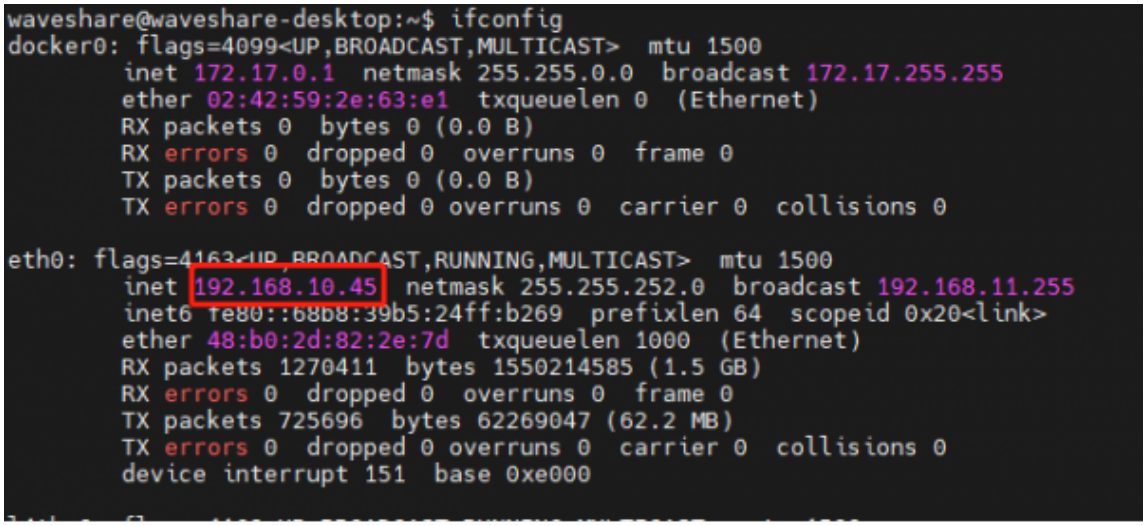


(/wiki/File:JETSON-NANO-DEV-KIT_IP01.png)

4. Power on the device and connect to the network.

5. Please click the "Scan" button again to scan the IP in the current LAN.

6. Exclude all the **previously recorded** IP addresses with the word "NVIDIA" in the Manufacturer, and the rest is your NVIDIA IP address.

- Method 3: Screen IP query
  1. Jetson Nano is connected to the Internet cable or installed a wireless network card to connect to WIFI and enter ifconfig in the terminal to view the IP address of Jetson Nano.
  2. 

(/wiki/File:Wired_Network.png)

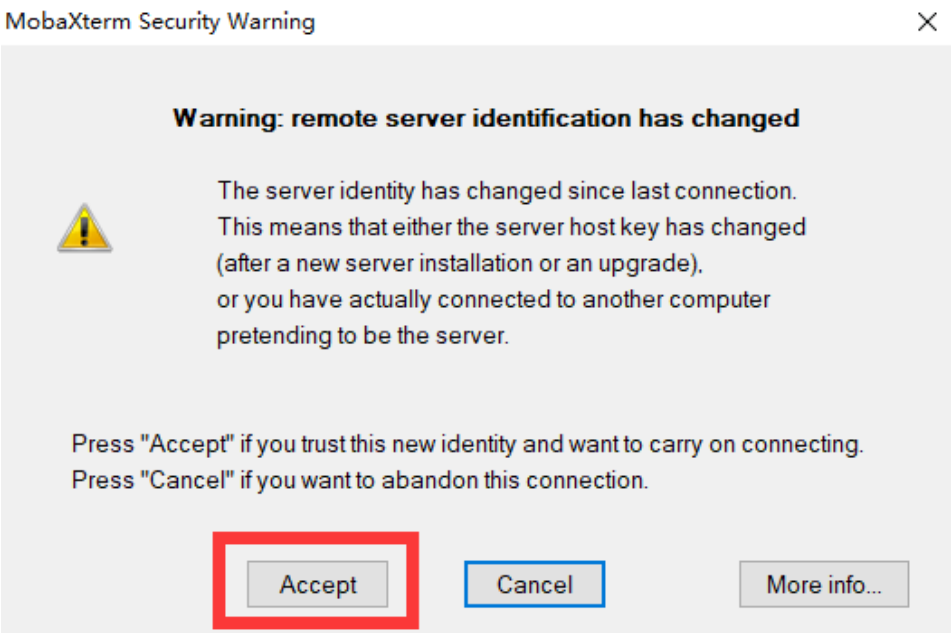
Log in with MobaXterm

Terminal Window

1. Download MobaXterm (https://files.waveshare.com/upload/c/c3/MobaXterm_Portable_v22.0.zip) and unzip it to use.
2. Open XobaXterm, click Session, and choose "ssh".
3. Enter the IP address 192.168.15.102 we queried earlier in the Remote host (fill in according to your actual IP), after filling in, click ok.

(/wiki/File:JETSON-NANO-DEV-KIT_terminal.png)

4. Click "Accept". Waveshare provides the image login name: waveshare, enter the login password: waveshare (when entering the password. It is normal that the screen does not change, and you can click "Enter" to confirm.)



(/wiki/File:JETSON-NANO-DEV-KIT_terminal2.png)

Nomachine Login

- Compared with SSH and VNC, you may not know much about Nomachine. Nomachine is a free remote desktop software.
- NoMachine basically covers all major operating systems, including Windows, Mac, Linux, iOS, Android, Raspberry, etc.

Install on Jetson Nano

1. Download and unzip NoMachine (https://files.waveshare.com/upload/5/5f/Nomachine_7.10.1_1_arm64.zip).
2. After unzipping, please use U disk or through file transmission to copy ".deb" file to Jetson Nano.

3. Install it with the following commands:

```
sudo dpkg -i nomachine_7.10.1_1_arm64.deb
```

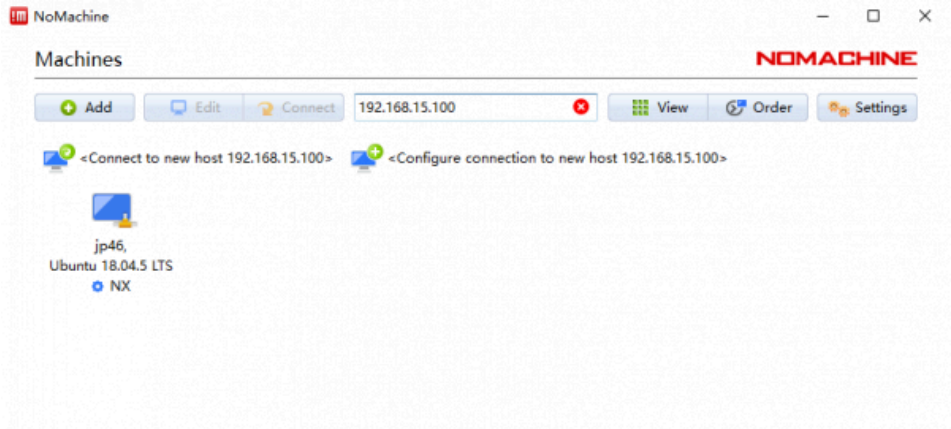
Install on Windows PC

- Download NoMachine (https://files.waveshare.com/upload/5/5f/Nomachine_7.10.1_1_arm64.zip) and then install. After clicking "Finish", you have to reboot your computer.

(/wiki/File:Nomachine.gif)

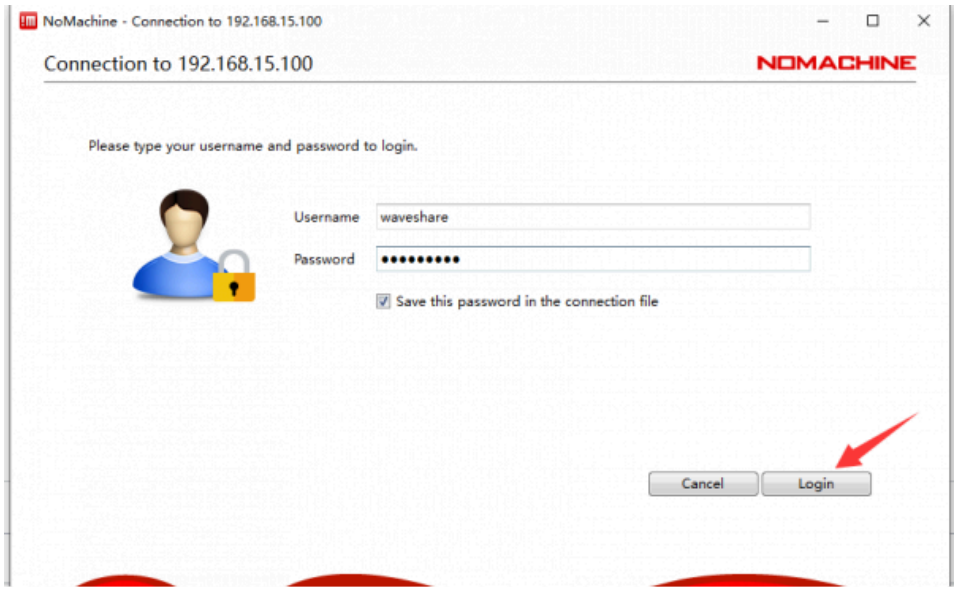
Connect to Jetson Nano

1. Open NoMachine and then enter the IP of Jetson Nano in the "Search" bar. For example, "192.168.15.100".



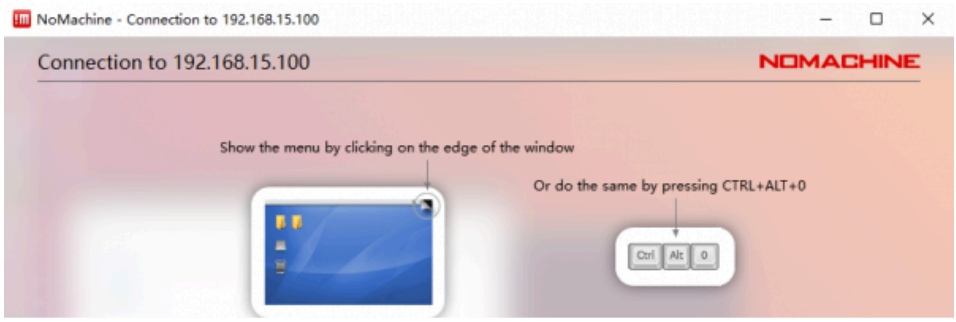
(/wiki/File:Jetson_Nano_Dev_Kit_No.png)

2. Click "Connect to new host 192.168.15.100", and then enter the username and password of Jetson Nano. Click "login".



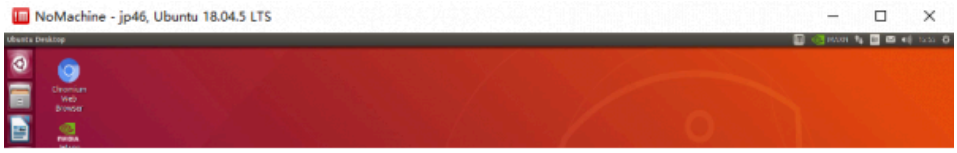
(/wiki/File:Jetson_Nano_Dev_Kit_No02.png)

3. After loading, there is an interface for software introduction, and we just need to click "OK".



(/wiki/File:Jetson_Nano_Dev_Kit_No03.png)

4. By now, we can log in to Jetson Nano successfully.



(/wiki/File:Jetson_Nano_Dev_Kit_No04.png)

VNC Login

- In the absence of a display screen, if you want to enter the desktop of the Jetson Nano, you need to use the remote desktop to log in. (It is recommended to use the display screen, VNC has a certain delay).

Configure the VNC Server

- Jetson Nano uses vino as the default VNC server, but the default settings need some modifications.

1. Configure VNC Server:

```
gsettings set org.gnome.Vino require-encryption false
gsettings set org.gnome.Vino prompt-enabled false
gsettings set org.gnome.Vino authentication-methods "['vnc']"
gsettings set org.gnome.Vino lock-screen-on-disconnect false
gsettings set org.gnome.Vino vnc-password $(echo -n "mypassword"|base64)
```

   - It should be noted that do not use sudo to run the above command, mypassword is the password to connect to VNC.

2. Set the desktop to start automatically at boot, and create a new self-starting file in the .config path.

```
 mkdir -p .config/autostart
sudo vim ~/.config/autostart/vino-server.desktop
```

   Add the following content:

```
[Desktop Entry]
Type=Application
Name=Vino VNC server
Exec=/usr/lib/vino/vino-server
NoDisplay=true
```

3. Check what manager you are currently using:

```
cat /etc/X11/default-display-manager
```

4. Edit the file:

```
sudo vim /etc/gdm3/custom.conf
```

5. Remove the comments on the following three lines, and modify the AutomaticLogin line to your own username.

```
WaylandEnable=false
AutomaticLoginEnable = true
AutomaticLogin = waveshare
```

6. Reboot Jetson Nano.

```
sudo reboot
```

Download and Install VNC Viewer

- Download and install VNC Viewer (https://files.waveshare.com/upload/4/4e/VNC-Viewer-6.21.1109-Windows.zip).

Remotely connect to Jetson Nano using VNC Viewer

1. Open VNC Viewer, enter the IP address of Jetson Nano and press Enter to confirm. For example:

```
192.168.15.102
```

2. Enter the VNC login password set earlier and click "OK".

3. At this point, you have successfully logged in to Jetson Nano.

(/wiki/File:Jetson_nano_login0012.png)

SDK Installation

Jetpack mainly includes system images, libraries, APIs, developer tools, examples and some documentation. In the SDK Manager software, we first install the OS, which is the system image, and the uninstalled part is the SDK, as shown in the figure below, you can **use the command to install directly** or use the **SDK Manger installation**:

(/wiki/File:SDK.jpg)

The SDK includes TensorRT, cuDNN, CUDA, Multimedia API, Computer Vision, and Developer Tools.

- **TensorRT**: High-performance deep learning inference runtime for image classification, segmentation, and object detection neural networks, which speeds up deep learning inference and reduces convolutional and deconvolutional neural network operations memory usage.
- **cuDNN**: The CUDA deep neural network library provides high-performance primitives for deep learning frameworks, including support for convolutions, activation functions and tensor transformations.

- **CUDA**: The CUDA toolkit provides a comprehensive development environment for C and C++ developers building GPU-accelerated applications. The toolkit includes compilers for NVIDIA GPUs, math libraries, and tools for debugging and optimizing application performance.
- **multimedia API**: Jetson Multimedia API provides a low-level API for flexible application development.
- **Computer Vision**: VPI (Vision Programming Interface) is a software library that provides computer vision/image processing algorithms implemented on PVA1 (Programmable Vision Accelerator), GPU and CPU, of which OpenCV is a leading open source library used for computer vision, image processing and machine learning, now featuring GPU acceleration for real-time operations, of which VisionWorks2 is a software development kit for computer vision (CV) and image processing.
- **Developer Tools**: The Developer Tools CUDA toolkit provides a comprehensive development environment for C and C++ developers building GPU-accelerated applications. The toolkit includes compilers for NVIDIA GPUs, math libraries and tools for debugging and optimizing application performance.

The above are some functions of the SDK.

When the previous system is installed, only the basic system is installed. Other JetPack SDK components, such as CUDA, need to be further installed after the system starts normally.

Here are the steps to install the SDK. **If you want to install this part, please ensure that the TF card or USB flash driver is the main system, because the downloaded content may cause the EMMC disk capacity to run out.**

SDK Manager Installation

Note: Because this method is too cumbersome, it is recommended to use the following commands to download, that is, to network operation on the development board.

When using the SDK Manager to install the SDK, you do not need to set the nano to recovery mode, that is, you do not need to short-circuit the pins.

- Power on the Nano normally.
- After the Jetson Nano enters the system and starts normally, connect the Micro USB port of the Jetson Nano to the Ubuntu host with a USB data cable.
- Run the sdkmanager command on the Ubuntu host computer to open SDK Manager (SDK Manager needs to be installed first).
- Similar to the previous operation of programming the system, the difference is that in the step, **instead of checking the OS option**, **check the SDK option**, and then continue to the installation.
- After downloading the resource, a pop-up window will prompt you to fill in the username and password, just fill in the username and password of the nano system.

- Wait for the SDK to be installed successfully.

Using Command To Install

Users without Ubuntu or virtual machine can choose to install directly on Jetson Nano using the following commands.

```
sudo apt update
sudo apt install nvidia-jetpack
```

If you only have less space reserved, please do not follow the above instructions to install, as it will lead to insufficient space issues. Also, do not use the SDK Manager to install components. Please follow the two instructions below instead.

```
sudo apt update
apt depends nvidia-jetpack | awk '{print $2}' | xargs -I {} sudo apt install -y {}
```

Linux Basic Operation

Common Command Introduction

File System

sudo

- The sudo command executes commands as a system administrator.
- To use the root user, log in as the waveshare user and execute the following command:

```
sudo su #Switch to super user
su waveshare #Switch common users
```

ls

- The "ls" command is used to display the contents of the specified working directory (list the subfiles and subdirectories contained in the current working directory).
- Common commands:

```
ls
ls -a #Show all files and directories (hidden files starting with . are also listed)
ls -l #In addition to the file name, it also lists the file type, permissions, owner, fil
e size and other information in detail
ls -lh #file sizes are listed in an easy-to-understand format, e.g. 4K
```

- To learn about more parameters of the command, we can use the "help" command to view:

```
ls --help
```

chmod

- The chmod command is for users to control permissions on files.
- The file calling permissions of Linux/Unix is divided into three levels: file owner (Owner), user group (Group) and other users (Other Users)
- In the figure below, the detailed file information under the Linux root directory is displayed. Among these file information, the most important is the first column, which describes in detail the permissions of files and directories, while the third and fourth columns show which user or group the file and directory belong to.

(/wiki/File:Chmod02.png)

- There are three file attributes in Linux: read-only (r), write (w), and executable (x). However, the above file attributes are divided into 10 small cells, because in addition to the first cell displaying the directory, the other three groups of three cells respectively represent the file owner permissions, permissions within the same group and other user permissions.
  - If d is displayed in the first column, it means that this is a directory; if it is a link file, l is displayed here; if it is a device file, c is displayed.
  - The first rwx field: -rwx------ indicates the permissions of the file owner.
  - The second rwx field: ---rwx--- indicates the permissions of the user within the same workgroup.
  - The third rwx field: ------rwx indicates the permissions of other users.
  - E.g:
  - -rwx rwx rwx means that no matter which user can read, write and execute this file.
  - -rw- --- --- Indicates that only the file owner has read and write permissions, but no execute permissions.
  - -rw -rw -rw means that all users have read and write permissions.
- Symbolic mode
  - who (user type)

who	User Type	Description
u	user	file owner
g	group	The file owner's group
o	others	All other users
a	all	User used, equivalent to ugo

- operator (symbol pattern table)

Operator	Description
+	Increase permissions for the specified user type
-	Remove the permission of the specified user type
=	Set the settings of the specified user permissions, that is to reset all permissions of the user type

- permission (symbol pattern table)

Mode	Name	Description
r	read	Set to readable permission
w	write	Set to writable permission
x	Execute permission	Set to executable permission
X	Special execution permission	Set the file permission to be executable only when the file is a directory file, or other types of users have executable permission
s	setuid/gid	When the file is executed, set the file's setuid or setgid permissions according to the user type specified by the who parameter
t	paste bit	Set the paste bit, only the superuser can set this bit, and only file owner u can use this bit

- Symbolic Pattern Examples

  1. Add read permissions to all users of the file:

```
chmod a+r file
```

  2. Remove execute permission for all users of the file:

```
chmod a-x file
```

3. Add read and write permissions to all users of the file:

```
chmod a+rw file
```

4. Add read, write and execute permissions to all users of the file:

```
chmod +rwx file
```

5. Set read and write permissions to the owner of the file, clear all permissions of the user group and other users to the file (space means no permission):

```
chmod u=rw,go= file
```

6. Add read permission to the user for all files in the directory waveshare and its subdirectory hierarchy, and remove read permission for the user group and other users:

```
chmod -R u+r,go-r waveshare
```

- Octal literals
  - The chmod command can use octal numbers to specify permissions. The permission bits of a file or directory are controlled by 9 permission bits, each of which is a group of three, which are the read, write and execute for the file owner (User), the read, write and execute for the user group (Group), the read, write and execute for other users (Other).

#	Permissions	rwx	Binary
7	Read + Write + Execute	rwx	111
6	Read + Write	rw-	110
5	read + execute	rwx	101
4	Read only	r--	100
3	Write + Execute	-wx	011
2	Write only	-w-	010
1	Only execute	--x	001
0	None	---	000

  - For example, 765 is interpreted as follows:
    - The owner's permission is expressed in numbers: the sum of the numbers of the owner's three permission bits. For example, rwx, which is 4+2+1, should be 7.
    - The permissions of a user group are expressed in numbers: the sum of the numbers of the permission bits that belong to the group. For example, rw-,

which is 4+2+0, should be 6.
        - The permissions of other users are expressed in numbers: the sum of the numbers of other users' permission bits. For example, r-x, which is 4+0+1, should be 5.
    - Commonly used digital permission:
        - 400 -r-------- The owner can read, no one else can operate.
        - 644 -rw-r–r-- All owners can read, but only the owner can edit.
        - 660 -rw-rw---- Both owner and group users can read and write, others cannot operate.
        - 664 -rw-rw-r-- Everyone can read, but only the owner and group users can edit.
        - 700 -rwx------ The owner can read, write and execute, no one else can operate.
        - 744 -rwxr–r-- Everyone can read, but only the owner can edit and execute.
        - 755 -rwxr-xr-x Everyone can read and execute, but only the owner can edit.
        - 777 -rwxrwxrwx Everyone can read, write and execute (this setting is not recommended).
    - For example:
        - Add read permissions to all users of the file, and the owner and group users can edit:

```
sudo chmod 664 file
```

touch

- The touch command is used to modify the time attributes of a file or directory, including access time and change time. If the file does not exist, the system will create a new file.
- For example, in the current directory, use this command to create a blank file "file.txt" and enter the following command:

```
touch file.txt
```

mkdir

- The "mkdir" command is used to create directories.
- In the working directory, create a subdirectory named waveshare:

```
sudo mkdir waveshare
```

- Create a directory named waveshare/test in the working directory.

```
sudo mkdir -p waveshare/test
```

- If the waveshare directory does not already exist, create one. (Note: If the -p parameter is not added in this example, and the original waveshare directory does not exist, an error will occur.)

cd

- Change the current working directory:

```
cd ..     #Return to the previous directory
cd /home/waveshare #Enter /home/waveshare directory
cd        #Return to user directory
```

cp

- The cp command is mainly used to copy files or directories.
- Parameter:
  - -a: This option is usually used when copying directories, it preserves links and file attributes, and copies everything under the directory. Its effect is equal to the combination of dpR parameters.
  - -d: Preserve links when copying. The links mentioned here are equivalent to shortcuts in Windows systems.
  - -f: Overwrite existing object files without prompting.
  - -i: Contrary to the -f option, give a prompt before overwriting the target file, asking the user to confirm whether to overwrite and answer "y", the target file will be overwritten.
  - -p: In addition to copying the contents of the file, also copy the modification time and access permissions to the new file.
  - -r: If the given source file is a directory file, all subdirectories and files in the directory will be copied.
  - -l: Do not copy files, just generate link files.
- Use the command cp to copy all the files in the current directory test/ to the new directory "newtest", and enter the following command:

```
sudo cp –r test/ newtest
```

mv

- The mv command is used to rename a file or directory or move a file or directory to another location.
- Parameter:
  - -b: When the target file or directory exists, create a backup of it before performing the overwrite.
  - -i: If the source directory or file specified to be moved has the same name as the target directory or file, it will first ask whether to overwrite the old file. Enter "y" to directly overwrite, and "n" to cancel the operation.
  - -f: If the source directory or file specified to be moved has the same name as the target directory or file, it will no ask, and the old file will be overwritten directly.
  - -n: Do not overwrite any existing files or directories.
  - -u: The move operation is performed only when the source file is newer than the target file or the target file does not exist.
- Use the command mv to copy the file1 file in the current directory test/ to the new directory /home/waveshare, and enter the following command:

```
sudo mv file1 /home/waveshare
```

rm

- The rm command is used to delete a file or directory.
- Parameter:
  - -i: ask for confirmation one by one before deleting.
  - -f: even if the original file attribute is set to read-only, it will be deleted directly without confirming one by one.
  - -r: delete the files in the directory and below one by one.
  - To delete a file, you can use the "rm" command directly. If you delete a directory, you must use the option "-r", for example:

    ```
    sudo rm  test.txt
    ```

  - rm: delete the general file "test.txt"? y

    ```
    sudo rm  homework
    ```

  - rm: cannot delete directory "homework": is a directory.

    ```
    sudo rm -r homework
    ```

  - rm: delete the directory "homework"? y

reboot

- The reboot command is used to restart the computer. Changing the configuration of Tinker Board 2 often requires restarting.
- Parameter:
  - -n: do not write the memory data back to the hard disk before rebooting.
  - -w: don't actually reboot, just write the log to the /var/log/wtmp file.
  - -d: do not write logs to the /var/log/wtmp file (-d is included with the -n parameter).
  - -f: force reboot, do not call shutdown command.
  - -i: stop all network-related devices before rebooting.
- reboot

```
sudo reboot
```

shutdown

- You cannot directly unplug the power cord to turn off the Jetson Nano, because the Tinker Board 2 will use the memory as a temporary storage area. If you directly unplug the power cord, some data in the memory will not have time to be written to the TF card, resulting in data loss or damage to the data on the TF card, causing the system to fail to boot.
- Parameter:
  - -t seconds: Set the shutdown procedure after a few seconds.
  - -k: Don't actually shut down, just send a warning message to all users.
  - -r: Reboot after shutdown.

    - -h: Shutdown after shutdown.
    - -n: Do not use the normal program to shut down, use the forced method to kill all the programs in execution and then shut down by itself.
    - -c: Cancel the shutdown action that is currently in progress.
    - -f: Do not do fsck when shutting down (check the Linux file system).
    - -F: Force fsck action when shutting down.
    - time: Set the shutdown time.
    - message: The warning message is sent to all users.
- Example:
    - Shut down now.

```
sudo shutdown -h now
```

    - Shutdown after specified 10 minutes.

```
sudo shutdown -h 10
```

    - Restart the computer.

```
sudo shutdown -r now
```

- No matter which command is used to shut down the system, root user permission is required. If the user uses a common user such as linaro, the sudo command can be used to temporarily obtain root permission.

pwd

- The pwd command displays the name of the current working directory: on Jetson nano, typing "pwd" will output something like "/home/waveshare".

head

- The "head" command displays the beginning of the file, which can be used with "-n" to specify the number of lines to display (default is 10), or with "-c" to specify the number of bytes.

```
head test.py -n 5
```

tail

- The tail shows the end of the file. -c bytes or -n lines specify the starting point in the file.

df

- Displays available and used disk space on mounted filesystems. Use df -h to see the output in a readable format, use M for MB instead of bytes.

```
df -h
```

tar

- The "tar" command is a tool program used to create and restore backup files. It can add and unpack files in backup files.
- Compressed file:

```
tar -cvzf waveshare.tar.gz *
```

- unzip files:

```
tar -xvzf waveshare.tar.gz
```

apt

- "apt" (Advanced Packaging Tool) is a shell front-end package manager in Debian and Ubuntu.
- The "apt" command provides commands for finding, installing, upgrading and removing a package, a group of packages or even all packages, and the commands are concise and easy to remember.
- "apt" command execution requires super administrator privileges (root).
- "apt" common commands:
  - List all updatable software inventory commands: sudo apt update.
  - Upgrade packages: sudo apt upgrade.
  - List updatable packages and version information: apt list --upgradeable.
  - Upgrade packages, delete the packages that need to be updated before upgrading: sudo apt full-upgrade.
  - Install the specified software command: sudo apt install <package_name>.
  - Install multiple packages: sudo apt install <package_1> <package_2> <package_3>.
  - Update the specified software command: sudo apt update <package_name>.
  - Display package specific information, such as version number, installation size, dependencies, etc.: sudo apt show <package_name>.
  - Remove package command: sudo apt remove <package_name>.
  - Clean up unused dependencies and libraries: sudo apt autoremove.
  - Remove packages and configuration files: sudo apt purge <package_name>.
  - Find packages command: sudo apt search <keyword>.
  - List all installed packages: apt list --installed.
  - List version information of all installed packages: apt list --all-versions.
- For example, we installed nano editor.

```
sudo apt install nano
```

Network

ifconfig

- Used to display the network configuration details of an interface on the current system when run with no arguments (ie) ifconfig.

- When connecting with SSH, you can find the IP address through ifconfig, and enter in the terminal:

```
ifconfig
```

- Check the IP address of the wired network, and enter in the terminal:

```
ifconfig eth0
```

- Check the IP address of the wireless network and enter in the terminal:

```
ifconfig wlan0
```

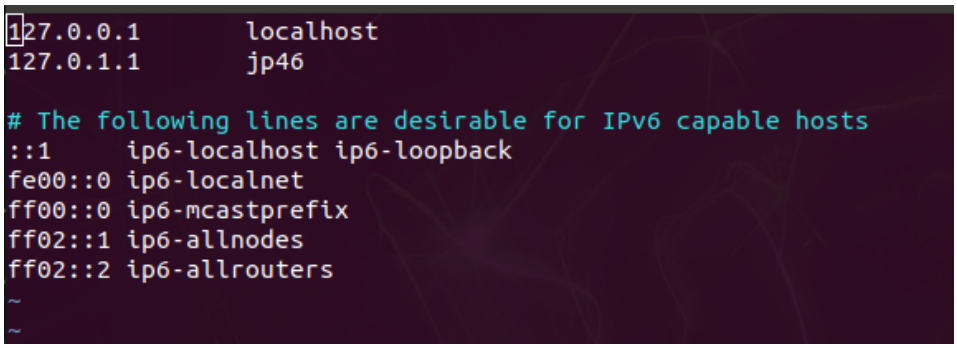
hostname

- The hostname command displays the current hostname of the system. When we use Jetson Nano, we often need to use remote tools, and the default network configuration IP address adopts dynamic allocation, which will cause the problem of IP address uncertainty.
- When the IP address of our Jetson Nano changes, it is possible to log in using the hostname.

1. Log in to Jetson Nano, and modify the host file, the command is as follows:

```
sudo vim /etc/hosts
```

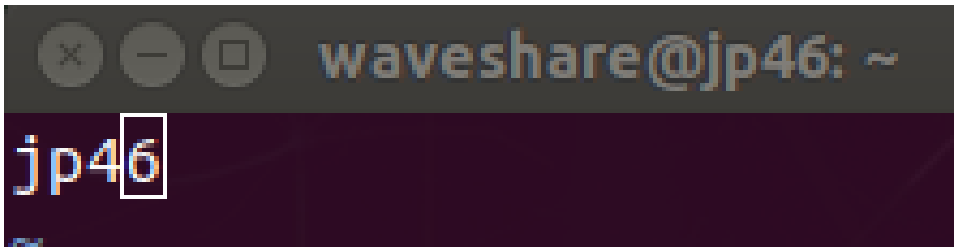
- Replace jp46 with the name to be modified, such as Waveshare, and press the keyboard ZZ to save and exit:



(/wiki/File:Jetson_nano_hostname.png)

2. Modify the hostname file, and replace the jp46 here with the name to be modified, such as waveshare, and press the keyboard ZZ:

```
sudo vim /etc/hostname
```



(/wiki/File:Jetson_nano_hostname2.png)

3. After the modification is completed, restart the Jetson Nano:

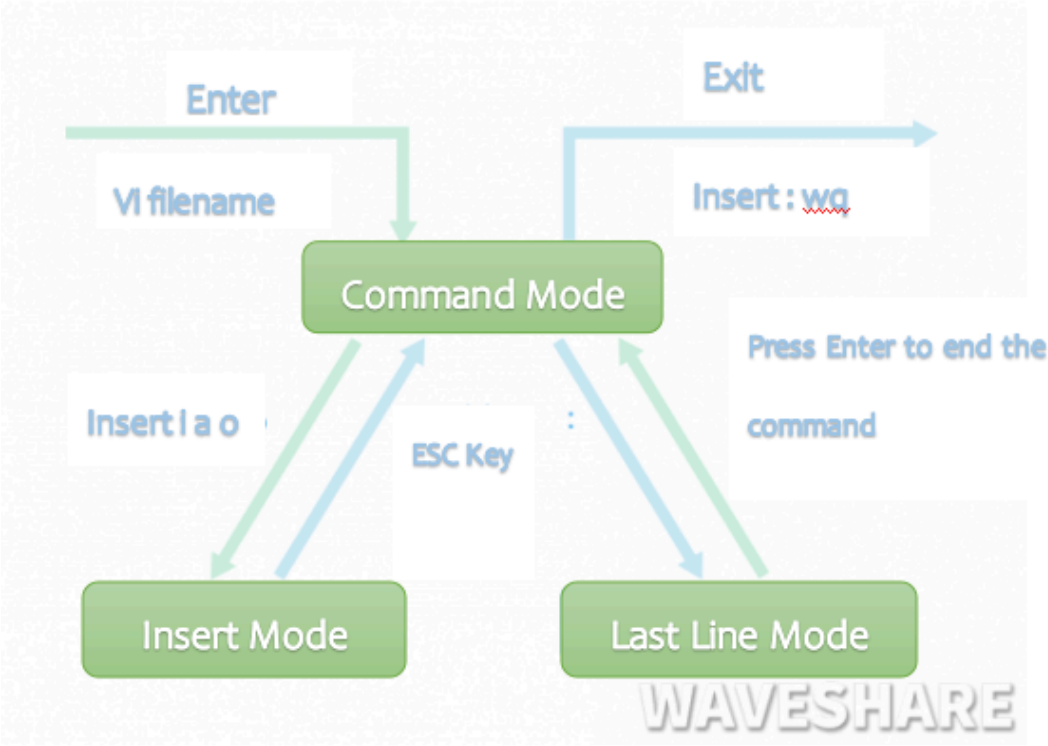
```
sudo reboot
```

4. We can also check the IP address with the following command:

```
hostname -I
```

Vim Editor User Guide

- The Vim editor is a standard editor under all Unix and Linux systems. Its power is not inferior to any of the latest text editors. Here is just a brief introduction to its usage and common commands.
- Basically vim is divided into three modes, namely Command mode, Insert mode and Last line mode.
  - Command mode: Control the movement of the screen cursor, delete characters, words or lines, move and copy a section.
  - Input Mode: Input characters and edit files in this mode.
  - Last line mode: Save the file or exit vim, you can also set the editing environment, such as finding strings, listing line numbers, etc.
  - We can think of these three modes as the icons at the bottom:



(/wiki/File:Jetson_nano_vim01.png)

1. First remove the default Vi editor:

```
sudo apt-get remove vim-common
```

2. Then reinstall Vim:

```
sudo apt-get install vim
```

3. For convenience, you have to add the following three sentences after the /etc/vim/vimrc file:

```
set nu #display line number
syntax on # syntax highlighting
set tabstop=4 #tab back four spaces
```

Common Command

- Open file, save, close file (use in vi command mode):

```
vim filename //Open the filename file
        :w // Save the file
        :q //Quit the editor, if the file has been modified please use the following
command
        :q! //Quit the editor without saving
        :wq //Exit the editor and save the file
        :wq! //Force quit the editor and save the file
        ZZ //Exit the editor and save the file
        ZQ //Exit the editor without saving
```

- Insert text or line (used in vi command mode, after executing the following command, it will enter insert mode, press ESC key to exit insert mode:

```
a //Add text to the right of the current cursor position
i //Add text to the left of the current cursor position
A //Add text at the end of the current line
I //Add text at the beginning of the current line (the beginning of the line with a
non-empty character)
O //Create a new line above the current line
o //Create a new line below the current line
R //Replace (overwrite) the current cursor position and some text behind it
J //Merge the line where the cursor is located and the next line (still in command m
ode)
```

- Delete and restore characters or lines (used in vi command mode):

```
x //Delete the current character
nx //Delete n characters from the cursor
dd //Delete the current line
ndd //Delete n lines down, including the current line
u //Undo the previous operation
U //Undo all operations on the current row
```

- Copy and paste (used in vi command mode):

```
yy //Copy the current line to the buffer
nyy //Copy the current line down n lines to the buffer
yw //Copy the characters from the cursor to the end of the word
nyw //Copy n words starting from the cursor
y^ //Copy the content from the cursor to the beginning of the line
y$ //Copy the content from the cursor to the end of the line
p //Paste the contents of the clipboard after the cursor
P //Paste the contents of the clipboard before the cursor
```

Configuration

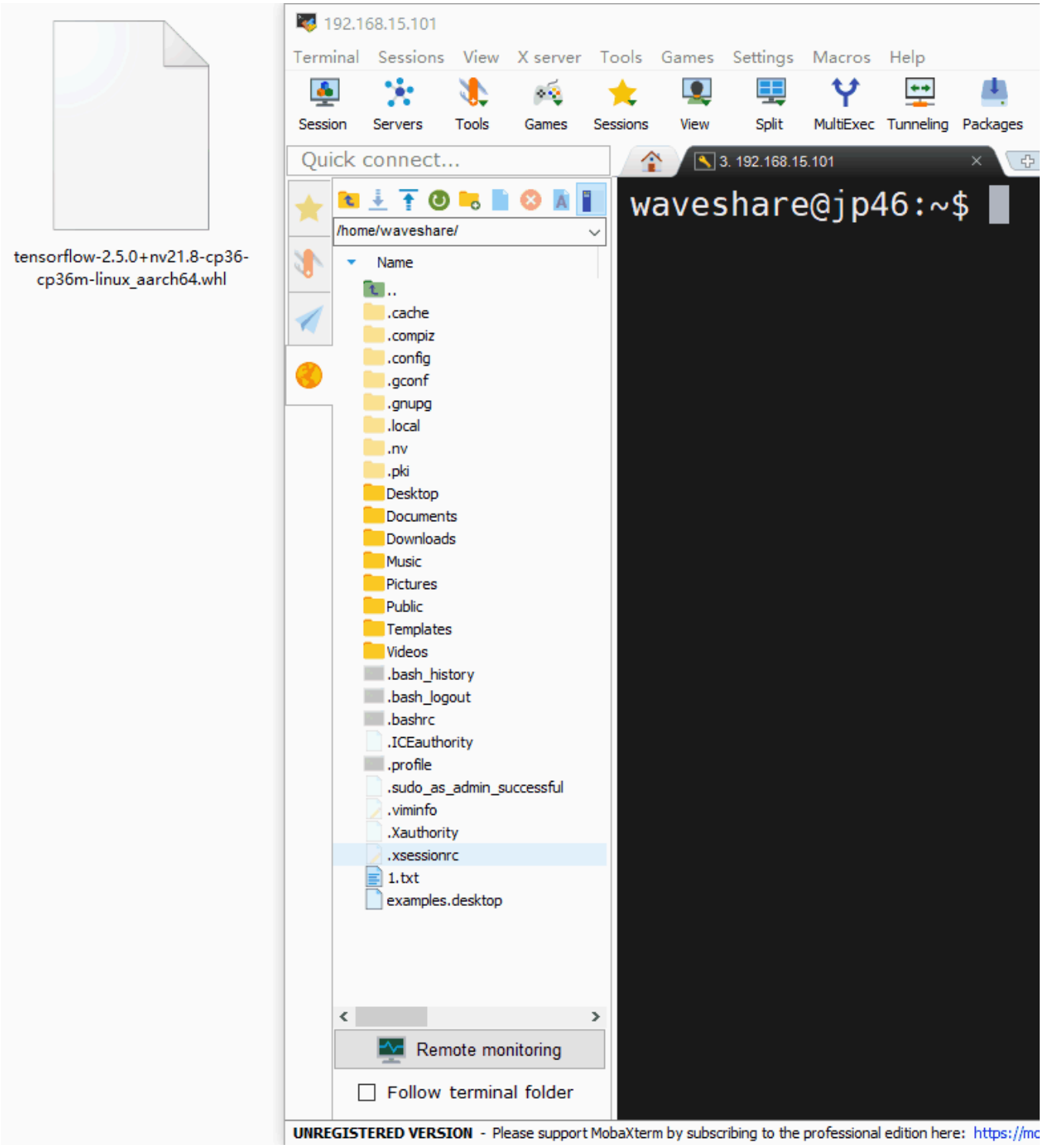
File transmission

This tutorial takes a Windows system to remotely connect to a Linux server as an example.

There are multiple ways to upload local files to the server.

MobaXterm File Transmission

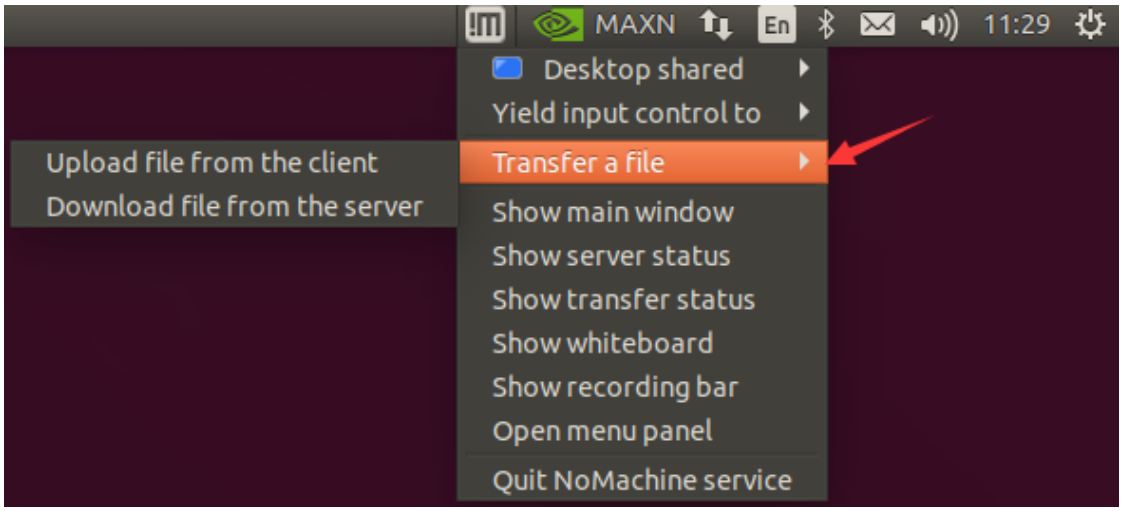
- Transferring files with the MobaXterm tool is very simple and convenient.
  - To transfer files from Windows to Raspberry Pi simply drag the files to the left directory of MobaXterm.
  - Similarly, to transfer Raspberry Pi files to Windows, just drag the files in the left directory of MobaXterm to Windows.



(/wiki/File:Jetson_nano_vim010.gif)

NoMachine File Transmission

- When the NoMachine remote connection is successful, the NoMachine icon will appear in the upper right corner of the Jetson Nano desktop. We click the icon and select Transfer a file.
  - Upload file from the client is to transfer files from a Windows computer to the Jetson Nano.
  - Download the file from the server is to transfer files from Jetson Nano to a Windows computer.



(/wiki/File:Jetson_nano_vim011.png)

SCP File Transmission

- The SCP command can be used to securely copy or encrypt transfer files and directories across Linux systems.
- Format:

```
scp +parameter +username/login name+@+hostname/IP address+ : + target file path+loca
l storage path
```

- First enter the directory where you want to store the file, hold down the keyboard Shift and right-click the blank space to open Windows PowerShell.
- Copy the file from the Jetson Nano to the local Windows and enter it in the terminal:

```
scp waveshare@192.168.10.80:file .
```

  Where "." represents the current path.
- Copy the file from the local Windows to the Jetson Nano and enter it in the terminal:

```
scp file waveshare@192.168.10.80:
```

- Copy the file folder from the Jetson Nano to the local Windows. Since the file is a directory, you need to add the parameter r and enter it in the terminal:

```
scp -r waveshare@192.168.10.80:/home/pi/file .
```

- Copy the file folder from the local Windows to the Jetson Nano and enter it in the terminal:

```
scp -r file waveshare@192.168.10.80:
```

Note: The above waveshare needs to be changed to the username of your system, and the IP address to the actual IP address of Jetson Nano.

File sharing (Samba)

File sharing is possible using the Samba service. The Jetson Nano file system can be accessed in the Windows Network Neighborhood, which is very convenient.

1. First install Samba, and enter into the terminal:

```
sudo apt-get update
sudo apt-get install samba -y
```

2. Create a shared folder sambashare in the /home/waveshare directory.

```
mkdir sambashare
```

3. After the installation is complete, modify the configuration file /etc/samba/smb.conf:

```
sudo nano /etc/samba/smb.conf
```

Pull to the end of the file and add the following statement to the end of the file.

```
[sambashare]
    comment = Samba on JetsonNano
    path = /home/waveshare/sambashare
    read only = no
    browsable = yes
```

Note: waveshare here needs to be changed to your system username. In other words, the path is the shared folder path you want to set.

4. Restart the Samba service.

```
sudo service smbd restart
```

5. Set shared folder password:

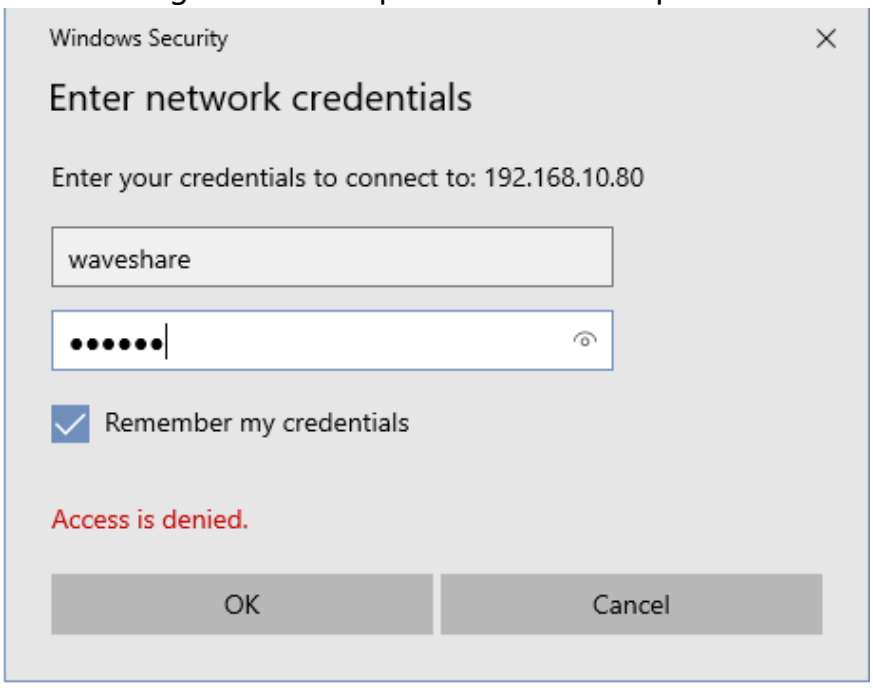
```
sudo smbpasswd -a waveshare
```

Note: The username here needs to be changed to the username of your system. If it is not the username, it will fail. You will be asked to set a Samba password here. It is recommended to use your system password directly, which is more convenient to remember.

6. After the setup is complete, on your computer, open the file manager.

```
\\192.168.10.80\sambashare
```

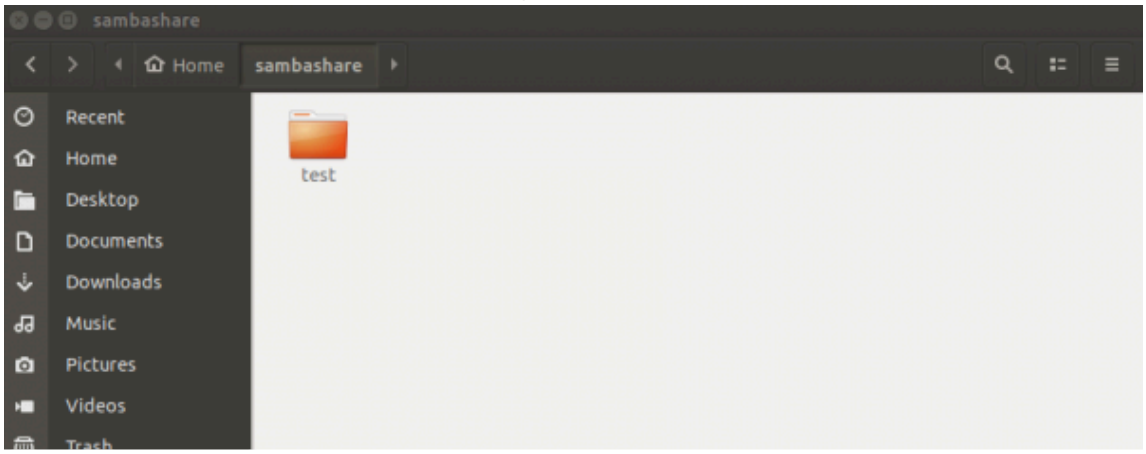
7. Enter the login name and password set in step 5 earlier.



(/wiki/File:Jetson_nano_vi012.png)

8. Let's verify, create a new test folder in Windows, and you can see the test folder in the Jetson Nano sambashare directory.

(/wiki/File:Jetson_nano_vim0132.png)



(/wiki/File:Jetson_nano_vim013.png)

System Backup

1. Create a new blank file with the .img suffix on the Windows computer desktop. Insert the TF card with the system image and select the drive letter of the corresponding TF card.
2. Open Win32DiskImager (https://files.waveshare.com/upload/7/76/Win32DiskImager.zip), click "Read", and then convert Jetson Nano's TF card file to an image.

Camera

View the first connected camera screen:

```
nvgstcapture-1.0
```

View the second connected camera screen:

```
nvgstcapture-1.0 --sensor-id=1
```

FAN

Fan speed adjustment, note that 4 wires are required to debug the fan.

```
sudo sh -c 'echo 255 > /sys/devices/pwm-fan/target_pwm'
#Where 255 is the maximum speed, 0 is to stop, modify the value to modify the speed.
cat /sys/class/thermal/thermal_zone0/temp
#Get the CPU temperature, you can intelligently control the fan through the program
#The system comes with a temperature control system, and manual control is not required i
n unnecessary situations
```

Wi-Fi connection without display (applicable to packages with wireless network card)

- After the first successful login, we want to connect to WiFi later.

1. Scan WIFI.

   ```
   sudo nmcli dev wifi
   ```

2. Connect to the WIFI network. ("wifi_name" and "wifi_password" need to be replaced with the SSID and password of your actual WiFi.)

   ```
   sudo nmcli dev wifi connect "wifi_name" password "wifi_password"
   ```

3. If "successfully" is displayed, the wireless network is successfully connected, and the motherboard will automatically connect to the WiFi you specified the next time it is powered on.

```
waveshare@jp46:~$ sudo nmcli dev wifi connect test password 12345678
Device 'wlan0' successfully activated with '6c0e415c-fe41-41a5-a991-26931941dd2d'.
```

(/wiki/File:Jetson_nano_vim016.png)

Getting Started with AI

- This tutorial is based on the JetPack4.6 system image, the Python version is Python3.6, the TensorFlow version is 2.5.0, and the Pytorch version is 1.9.0.
- Note: The TensorFlow (https://docs.nvidia.com/deeplearning/frameworks/install-tf-jetson-platform-release-notes/tf-jetson-rel.html#tf-jetson-rel) version and the Pytorch (https://docs.nvidia.com/deeplearning/frameworks/install-tf-jetson-platform-release-notes/tf-jetson-rel.html#tf-jetson-rel) version must correspond to the JetPack version.

PIP Installation

1. Python3.6 version is installed by default in Jetson Nano, directly install PIP.

```
sudo apt update
sudo apt-get install python3-pip python3-dev
```

2. After the installation is complete, we check the PIP version.

```
pip3 -V
```

```
waveshare@jp46:~$ pip3 -V
pip 9.0.1 from /usr/lib/python3/dist-packages (python 3.6)
waveshare@jp46:~$
```

(/wiki/File:Jetson_Nano_AI01.png)

3. The default installed PIP is version 9.01, you need to upgrade it to the latest version.

```
python3 -m pip install --upgrade pip
```

```
waveshare@jp46:~$ python3 -m pip install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/a4/6d/6463d49a933f547439d6
b5b98b46af8742cc03ae83543e4d7688c2420f8b/pip-21.3.1-py3-none-any.whl (1.7MB)
    100% |                                | 1.7MB 19kB/s
```

(/wiki/File:Jetson_Nano_AI02.png)

4. After the upgrade is successful, check the pip version information, and you will find some problems.

```
pip3 -V
```

```
waveshare@jp46:~$ pip3 -V
WARNING: pip is being invoked by an old script wrapper. This will fail in a futu
re version of pip.
Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the unde
rlying issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip
```

(/wiki/File:Jetson_Nano_AI03.png)

5. We use the command to solve it as follows:

```
python3 -m pip install --upgrade --force-reinstall pip
sudo reboot
```

```
waveshare@jp46:~$ pip3 -V
```

(/wiki/File:Jetson_Nano_AI04.png)

6. Install important installation packages in the field of machine learning.

```
Install important packages in the field of machine learning
sudo apt-get install python3-numpy
sudo apt-get install python3-scipy
sudo apt-get install python3-pandas
sudo apt-get install python3-matplotlib
sudo apt-get install python3-sklearn
```

Set Up The CUDA Environment

1. Install CUDA.

```
sudo apt update
sudo apt install nvidia-jetpack
```

2. Check the CUDA version, if there appears "command not found", you need to configure the environment.

```
nvcc -V
cat /usr/local/cuda/version.txt
```

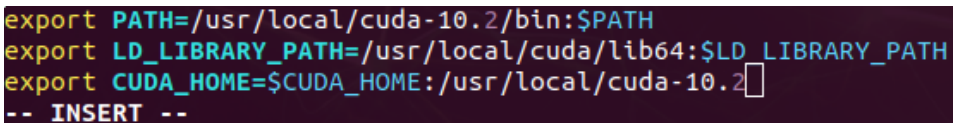
```
waveshare@jp46:~$ nvcc -V
bash: nvcc: command not found
waveshare@jp46:~$ cat /usr/local/cuda/version.txt
CUDA Version 10.2.300
waveshare@jp46:~$
```

(/wiki/File:Jetson_Nano_AI015.png)

Note: If you use the "cat" command, you can not check the version here. Please enter the "/usr/local/" directory to see if there is a CUDA directory.
If you do not install CUDA by referring to the Uninstalled CUDA section below, configure the environment after the installation is complete.

3. Set environment variables.

```
sudo vim .bashrc
Add at the end of the file:
export PATH=/usr/local/cuda-10.2/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:$LD_LIBRARY_PATH
export CUDA_HOME=$CUDA_HOME:/usr/local/cuda-10.2
```



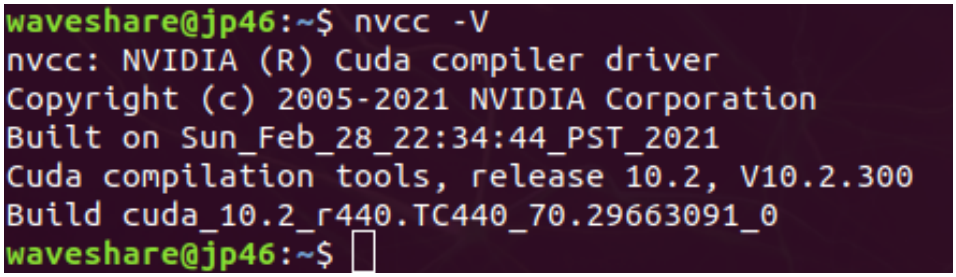
(/wiki/File:Jetson_Nano_AI016.png)

4. Update environment variables.

```
source .bashrc
```

5. Check the CUDA version again.

```
nvcc -V
```



(/wiki/File:Jetson_Nano_AI017.png)

Tensorflow GPU Environment Construction

1. Install the needed package.

```
sudo apt-get install libhdf5-serial-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpe
g8-dev liblapack-dev libblas-dev gfortran
sudo pip3 install -U pip testresources setuptools==49.6.0
```

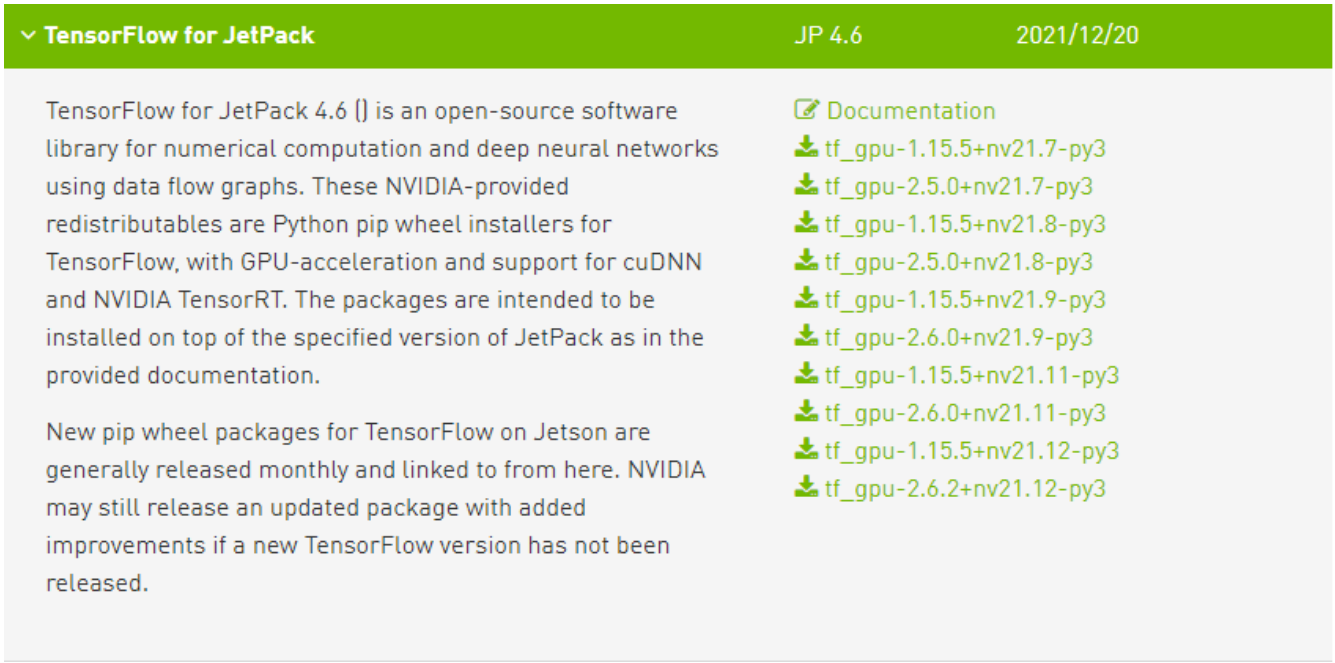
2. Install python independencies.

```
sudo pip3 install -U --no-deps numpy==1.19.4 future==0.18.2 mock==3.0.5 keras_prepro
cessing==1.1.2 keras_applications==1.0.8 gast==0.4.0 protobuf pybind11 cython pkgcon
fig packaging
sudo env H5PY_SETUP_REQUIRES=0 pip3 install -U h5py==3.1.0
```

3. Install Tensorflow (online installation often fails, you can refer to step 4 for offline installation).

```
sudo pip3 install --pre --extra-index-url https://developer.download.nvidia.com/comp
ute/redist/jp/v46 tensorflow
```

4. Finally, it is recommended to install offline, first log in to NVIDIA's official website to download the TensorFlow installation package (https://developer.nvidia.com/embedded/downloads) (take "jetpack4.6 TensorFlow2.5.0 nv21.08" as an example, it is recommended to use Firefox browser to download).

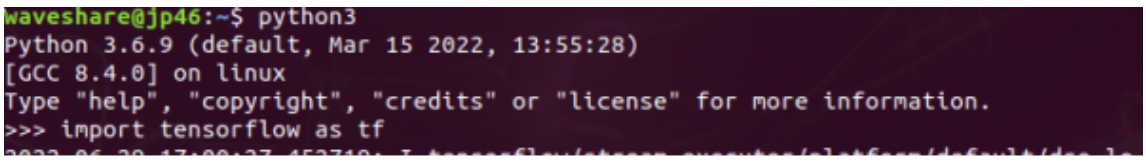


(/wiki/File:Nano_AI020.png)

```
pip3 install tensorflow-2.5.0+nv21.8-cp36-cp36m-linux_aarch64.whl
```

5. After the installation is complete, check whether the installation is successful, and enter into the terminal:

```
python3
import tensorflow as tf
```



(/wiki/File:Jetson_Nano_021.png)

6. View the version information.

```
tf.__version__
```

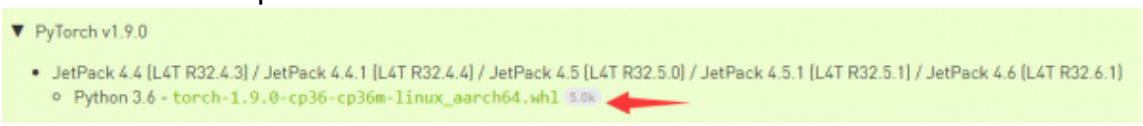
```
waveshare@jp46:~$ python3
Python 3.6.9 (default, Mar 15 2022, 13:55:28)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
2022-06-29 17:00:27.452719: I tensorflow/stream_executor/platform/default/dso_lo
ader.cc:53] Successfully opened dynamic library libcudart.so.10.2
>>> tf.__version__
'2.5.0'
```

(/wiki/File:Jetson_Nano_AI021.png)

Pytorch Environment Setting

Pytorch Installation

1. First log in to NVIDIA's official website to download Pytorch (https://forums.developer.nvidia.com/t/pytorch-for-jetson-version-1-11-now-available/72048). Here, we take Pytorch v1.9.0 as an example.



(/wiki/File:Jetson_Nano_AI022.png)

2. Download the independencies libraries.

```
sudo apt-get install libjpeg-dev zlib1g-dev libpython3-dev libavcodec-dev libavforma
t-dev libswscale-dev libopenblas-base libopenmpi-dev
```

3. Install Pytorch.

```
sudo pip3 install torch-1.9.0-cp36-cp36m-linux_aarch64.whl
```

4. Verify whether Pytorch has been installed successfully.

```
python3
import torch
x = torch.rand(5, 3)
print(x)
```

```
waveshare@jp46:~$ python3
Python 3.6.9 (default, Mar 15 2022, 13:55:28)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> x = torch.rand(5,3)
>>> print(x)
tensor([[0.4944, 0.9463, 0.0473],
        [0.6075, 0.1019, 0.3175],
        [0.2730, 0.4139, 0.5657],
        [0.3604, 0.9918, 0.0969],
        [0.8524, 0.9301, 0.2946]])
>>>
```

(/wiki/File:Jetson_Nano_AI023.png)

5. View the version information:

```
import torch
print(torch.__version__)
```

```
waveshare@jp46:~$ python3
Python 3.6.9 (default, Mar 15 2022, 13:55:28)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> print(torch.__version__)
1.9.0
```

(/wiki/File:Jetson_Nano_AI024.png)

Torchvision Installation

1. The Torchvision version should match the Pytorch version. The Pytorch version we installed earlier is 1.9.0, and Torchvision installs the v0.10.0 version.

(/wiki/File:Jetson_Nano_AI030.png)

2. Download and install torchvision.

```
git clone --branch v0.10.0 https://github.com/pytorch/vision torchvision
cd torchvision
export BUILD_VERSION=0.10.0
sudo python3 setup.py install
```

3. Verify whether Torchvision is installed successfully.

```
python3
import torchvision
```

```
waveshare@jp46:~$ python3
```

(/wiki/File:Jetson_Nano_AI031.png)

4. The error may be that the Pillow version is too high, uninstall and reinstall.

```
sudo pip3 uninstall pillow
sudo pip3 install pillow
```

```
AttributeError: module 'PIL.Image' has no attribute 'BILINEAR'
```

(/wiki/File:Jetson_Nano_AI032.png)

5. View the version information.

```
import torchvision
print(torchvision.__version__)
```

```
waveshare@jp46:~$ python3
```

(/wiki/File:Jetson_Nano_AI033.png)

Yolo V4 Environment Construction

1. First download darknet on GitHub.

```
git clone https://github.com/AlexeyAB/darknet.git
```

2. After downloading, you need to modify Makefile.

```
cd darknet
sudo vim Makefile
```

Change the first four lines of 0 to 1.

```
GPU=1
CUDNN=1
CUDNN_HALF=1
OPENCV=1
```

3. The cuda version and path should also be changed to our actual version and path, otherwise, the compilation will fail.

```
Change NVCC=nvcc to
NVCC=/usr/local/cuda-10.2/bin/nvcc
```

4. After the modification is completed, compile and enter into the terminal:

```
sudo make
```

Inference with YOLOv4

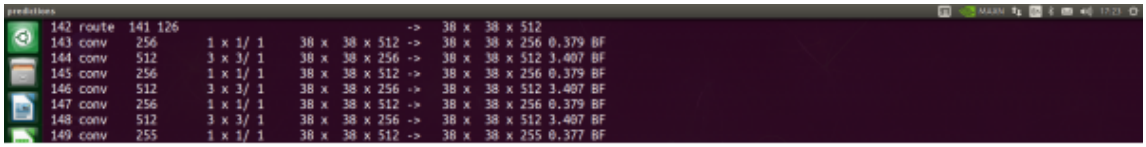
- There are three basic reasoning methods: picture, video and camera (live image).

- Choose Yolo v4 and Yolo v4-tiny (more lightweight models, suitable to run on Jetson Nano) for testing. First, you need to download the trained model weight file.

Image Test

1. Test.

```
./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/dog.jpg
```



(/wiki/File:Jetson_Nano_AI30.png)

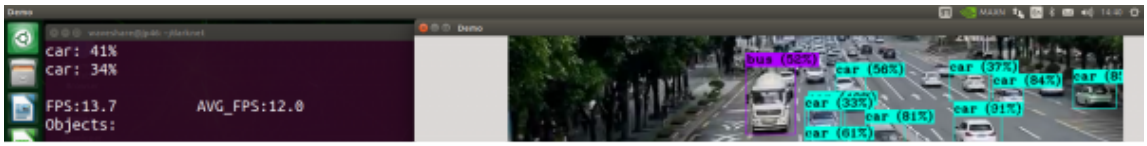
2. If you want to open the picture, you need to use the test mode, it will ask you to enter the location of the picture after execution:

```
./darknet detector test ./cfg/coco.data ./cfg/yolov4.cfg ./yolov4.weights
```

Video Test

1. Yolov4-tiny video detection (there is no video file in the data downloaded from GitHub, and the user needs to upload the video file to be detected to the data folder).

```
./darknet detector demo cfg/coco.data cfg/yolov4-tiny.cfg yolov4-tiny.weights data/x
xx.mp4
```



(/wiki/File:Jetson_Nano_AI31.png)

The frame is about 14 fps.

Live Image Test

- Check the device number of the USB camera:

```
ls /dev/video*
./darknet detector demo cfg/coco.data  cfg/yolov4-tiny.cfg  yolov4-tiny.weights /de
v/video0 "nvarguscamerasrc ! video/x-raw(memory:NVMM), width=1280, height=720, forma
t=NV12, framerate=30/1 ! nvvidconv  ! video/x-raw, width=1280, height=720, format=BG
Rx ! videoconvert ! video/x-raw, format=BGR ! appsink"
./darknet detector demo cfg/coco.data  cfg/yolov4.cfg  yolov4.weights /dev/video0 "n
varguscamerasrc ! video/x-raw(memory:NVMM), width=1280, height=720, format=NV12, fra
merate=30/1 ! nvvidconv  ! video/x-raw, width=1280, height=720, format=BGRx ! videoc
onvert ! video/x-raw, format=BGR ! appsink"
```

Hello AI World

- The Hello AI World project integrates the powerful TensroRT acceleration engine of NVIDIA, which improves performance by several times.

Environment Construction

1. Install cmake.

```
sudo apt-get update
sudo apt-get install git cmake libpython3-dev python3-numpy
```

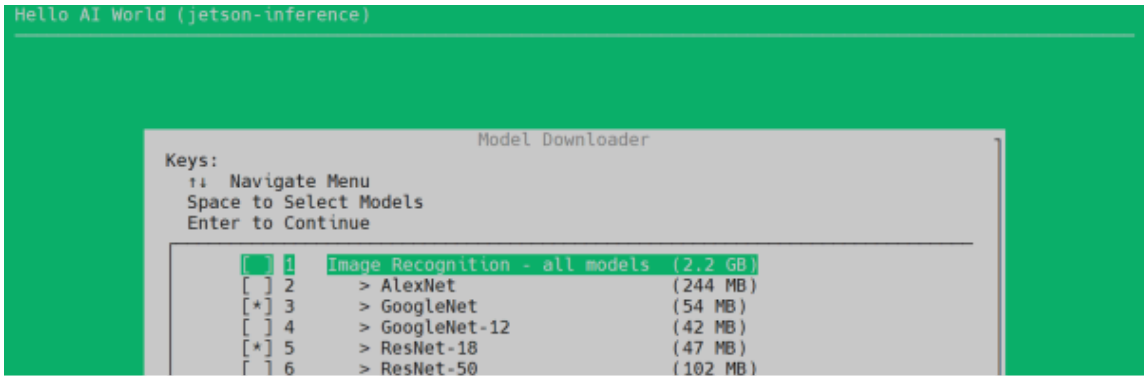
2. Get the jetson-inference open-source project.

```
git clone https://github.com/dusty-nv/jetson-inference
cd jetson-inference
git submodule update --init
```

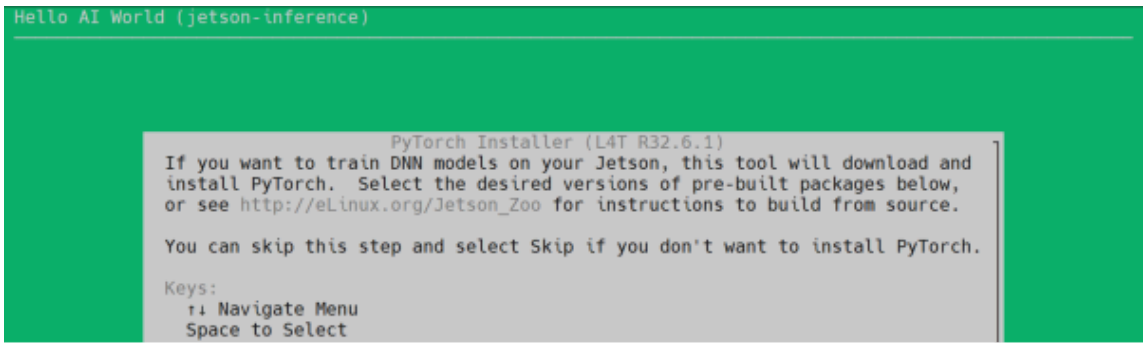
3. Create a new folder, and compile.
   Note: The following example can only be run normally if the compilation does not report an error.

```
sudo mkdir build
cd build
sudo cmake ../
```

When the download model and Pytorch interface appear, we choose to skip (Quit and Skip).



(/wiki/File:Jetson_Nano_AI33.png)



(/wiki/File:Jetson_Nano_AI34.png)

Download the model (https://github.com/dusty-nv/jetson-inference/releases), then place it in the "jetson-inference/data/networks" directory, and unzip it. To help users, here we provide

some model reference download commands:

```
cd ~/jetson-inference/data/networks/
#If you need to download more algorithms, please refer to the method here, first obtain t
he download address of the model on github, and then use wget to add commands to downloa
d. Download three models here for reference
wget https://github.com/dusty-nv/jetson-inference/releases/download/model-mirror-190618/f
acenet-120.tar.gz
wget https://github.com/dusty-nv/jetson-inference/releases/download/model-mirror-190618/G
oogleNet.tar.gz
wget https://github.com/dusty-nv/jetson-inference/releases/download/model-mirror-190618/S
SD-Mobilenet-v2.tar.gz
#The following command is to decompress the previously downloaded model, only after decom
pression can it be used
tar -zxvf facenet-120.tar.gz
tar -zxvf GoogleNet.tar.gz
tar -zxvf SSD-Mobilenet-v2.tar.gz
```

Copy files to Jetson Nano with a USB flash drive or through #MobaXterm File Transmission.

```
cd jetson-inference/build
sudo make
sudo make install
```

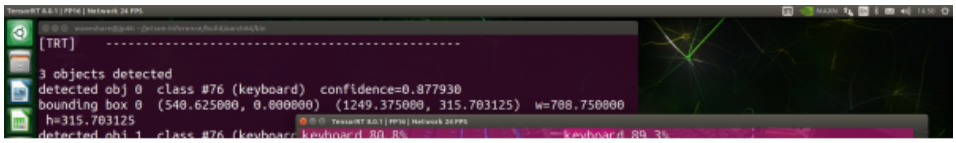
Install the v4l camera driver, and input the following command in the terminal:

```
sudo apt-get install v4l-utils
v4l2-ctl --list-formats-ext
```

DetectNet Runs Real-time Camera Detection

- Use the camera to recognize objects in the environment.

  ```
  cd ~/jetson-inference/build/aarch64/bin/
  ./detectnet-camera
  ```



(/wiki/File:Jetson_Nano_AI_detect01.png)
Use TensorRT to accelerate the frame rate to 24 fps.

- The content below is a list of pre-trained object detection networks available for download, along with the "--network" parameters for loading the pre-trained model:

```
./detectnet-camera --network=facenet # Run the face recognition network
./detectnet-camera --network=multiped # Run multi-level pedestrian/baggage detector
./detectnet-camera --network=pednet # Run the original single-stage pedestrian detec
tor
./detectnet-camera --network=coco-bottle # Detect bottles/soda cans under camera
./detectnet-camera --network=coco-dog # Detect dog under camera
```

- Let's use the "detectnet" program to locate objects in a static image. In addition to input/output paths, there are some additional command line options:
  - Change the optional "--network" flag for the detection model in use (the default is SSD-Mobilenet-v2).
  - "--overlay" flag can be comma-separated boxes, lines, labels, conf or none.
    - Default "--overlay=box,labels,conf" to show boxes, labels and confidence coefficient.
    - The box option draws filled bounding boxes, while lines only draw unfilled outlines.
  - "--alpha" value, sets the alpha mixed value used during overlay (the default is 120).
  - "--threshold" sets an optional value for the minimum threshold for detection (the default is 0.5).
  - "--camera" flag sets the camera device to use.
    - The default is to use MIPI CSI sensor 0 (--camera=0).
  - "--width" and "--height" flags set camera resolution (the default is 1280 x 720).
- The resolution should be set to a format supported by the camera, queried with "v4l2-ctl --list-formats-ext".

```
./detectnet-camera --network=facenet # Use FaceNet, default MIPI CSI camera (1280 × 720)
./detectnet-camera --camera=/dev/video1 --network=facenet # Use PedNet, V4L2 camera /dev/video1 (1280 x 720)
./detectnet-camera --width=640 --height=480 --network=facenet # Use PedNet, default MIPI CSI camera (640 x 480)
```

Hardware Control

- The Jetson TX1, TX2, AGX Xavier and Nano development boards include a 40-pin GPIO header, which is similar to the 40-pin header in the Raspberry Pi.

GPIO

- The digital inputs and outputs of these GPIOs can be controlled using the Python library provided in the Jetson GPIO library package.
- The Jetson GPIO library provides all the public APIs provided by the RPi.GPIO library. The use of each API is discussed below:

1. To import the Jetson.GPIO module, please use:

```
import Jetson.GPIO as GPIO
```

2. Pin number:
   - The Jetson GPIO library provides four methods for numbering I/O pins.
   - The first two correspond to the modes provided by the RPi.GPIO library, namely BOARD and BCM, refer to the pin number of the 40-pin GPIO header and the Broadcom SoC GPIO number respectively.
   - The remaining two modes, CVM and TEGRA_SOC use strings instead of numbers, corresponding to the CVM/CVB connector and signal names on the Tegra SoC respectively.
   - To specify the mode you use (mandatory), use the following function to call:

```
GPIO.setmode(GPIO.BOARD)
GPIO.setmode(GPIO.BCM)
GPIO.setmode(GPIO.CVM)
GPIO.setmode(GPIO.TEGRA_SOC)
```

   - To check which mode has been set, you can call:

```
mode = GPIO.getmode()
```

   The model must be GPIO.BOARD, GPIO.BCM, GPIO.CVM, GPIO.TEGRA_SOC or None.
3. If GRIO detects that a pin has been set to a non-default value, you will see a warning message.
   - You can disable warnings with the following code:

```
GPIO.setwarnings(False)
```

4. Set the channel:
   - The GPIO channel must be set before it can be used as an input or output. To configure a channel as an input, please call:

```
#(where channel is based on the pin numbering mode discussed above)
 GPIO.setup(channel, GPIO.IN)
```

   - To set the channel as output, please call:

```
GPIO.setup(channel, GPIO.OUT)
```

   - You can also specify an initial value for the output channel:

```
GPIO.setup(channel, GPIO. OUT, initial=GPIO. HIGH)
```

   - When you set a channel to output, you can also set multiple channels at the same time:

```
#add as many channels as needed. You can also use tuples: (18,12,13)
channels = [18, 12, 13]
GPIO.setup(channels, GPIO.OUT)
```

5. Input:
   - To read the value of a channel, please use:

```
GPIO.input(channel)
This will return the GPIO.LOW or GPIO.HIGH.
```

6. Output:
   - To set the value of a pin configured as an output, please use:

```
GPIO.output(channel, state)
where the state can be GPIO.LOW or GPIO.HIGH.
```

   - You can also output to a list of channels or tuples:

```
channels = [18, 12, 13] # or use tuples
GPIO.output(channels, GPIO.HIGH) # or GPIO.LOW
#Set the first channel to LOW and the rest to HIGH.
GPIO.output(channel, (GPIO.LOW, GPIO.HIGH, GPIO.HIGH))
```

7. Clean up:
   - At the end of the demo, it's a good idea to clean up the channel so that all pins are set to their default state. To clean up all used channels, please call:

```
GPIO.cleanup()
```

- If you don't want to clean up all channels, you can also clean up a single channel or a list of channels or tuples:

```
GPIO.cleanup(chan1) # cleanup only chan1
GPIO.cleanup([chan1, chan2]) # cleanup only chan1 and chan2
GPIO.cleanup((chan1, chan2)) # does the same operation as previous statement
```

8. Jetson Board Information and Library Versions:
   - To get information about Jetson modules, use/read:

```
GPIO.JETSON_INFO
```

     This provides a Python dictionary with the following keys: P1_REVISION, RAM, REVISION, TYPE, MANUFACTURER and PROCESSOR. All values in the dictionary are strings, but P1_REVISION is an integer.
   - To get information about library versions, use/read:

```
GPIO.VERSION
```

     This provides a string with the XYZ version format.
9. Interrupt:
   - In addition to polling, the library provides three additional methods to monitor input events:
   - wait_for_edge() function
   - This function blocks the calling thread until the provided edge is detected. The function can be called as follows:

```
GPIO.wait_for_edge(channel, GPIO.RISING)
```

   - The second parameter specifies the edge to detect, which can be GPIO.RISING, GPIO.FALLING or GPIO.BOTH. If you just want to limit the wait to a specified time, you can optionally set a timeout:

```
# Timeout is in milliseconds:
GPIO.wait_for_edge(channel, GPIO.RISING, timeout=500)
```

     The function returns the channel on which the edge was detected, or None if a timeout occurred.
   - event_detected() function
   - This function can be used to periodically check if an event has occurred since the last call. The function can be set and called as follows:

```
# Set rising edge detection on the channel:
GPIO.add_event_detect(channel, GPIO.RISING)
run_other_code()
if GPIO.event_detected(channel):
    do_something()
```

As before, you can detect events for GPIO.RISING, GPIO.FALLING or GPIO.BOTH.

- Callback function to run when an edge is detected.
- This function can be used to run a second thread for the callback function. Therefore, the callback function can run concurrently with your main program in response to the edge. This feature can be used as follows:

```
# define callback function
def callback_fn(channel):
    print("Callback called from channel %s" % channel)

# Add rising edge detection:
GPIO.add_event_detect(channel, GPIO.RISING, callback=callback_fn)
```

- Multiple callbacks can also be added if desired, like this:

```
def callback_one(channel):
    print("First Callback")

def callback_two(channel):
    print("Second Callback")

GPIO.add_event_detect(channel, GPIO.RISING)
GPIO.add_event_callback(channel, callback_one)
GPIO.add_event_callback(channel, callback_two)
```

In this case, the two callbacks run sequentially, not simultaneously, because only the thread runs all the callback functions.

- To prevent multiple invocations of the callback function by merging multiple events into one, optionally set a debounce time:

```
# Bouncetime set in milliseconds:
GPIO.add_event_detect(channel, GPIO.RISING, callback=callback_fn,
bouncetime=200)
```

- If edge detection is no longer needed, it can be removed as follows:

```
GPIO.remove_event_detect(channel)
```

10. Check GPIO channel function:
    - This function allows you to check the function of the provided GPIO channels:

```
GPIO.gpio_function(channel)
The function returns GPIO.IN or GPIO.OUT.
```

Turn on LED

- Sample demo:

```
import Jetson.GPIO as GPIO
import time as time

LED_Pin = 11

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(LED_Pin, GPIO.OUT)

while (True):
   GPIO.output(LED_Pin, GPIO.HIGH)
   time.sleep(0.5)
   GPIO.output(LED_Pin, GPIO.LOW)
   time.sleep(0.5)
```

Demo Use

- For the jetson.gpio library, the official also provides some simple demos.
- First download jetson-gpio:

```
git clone https://github.com/NVIDIA/jetson-gpio
```

- Once the download is complete, we can use the demos, e.g. simple_input.py to read the status of the pins:

```
cd /opt/nvidia/jetson-gpio/samples/
sudo python3 simple_input.py
```

IIC

1. Install I2Ctool first and input in the terminal:

```
sudo apt-get update
sudo apt-get install -y i2c-tools
sudo apt-get install -y python3-smbus
```

2. Check the installation and input in the terminal:

```
apt-cache policy i2c-tools
```

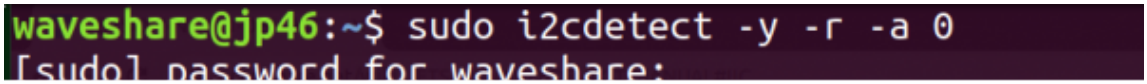
If the output is as follows, the installation is successful:

```
i2c-tools:
Installed: 4.0-2
Candidate: 4.0-2
Version list:
***4.0-2500
500 http://ports.ubuntu.com/ubuntu-ports bionic/universe arm64 Packages
100 /var/lib/dpkg/status
```

i2cdetect

- Query i2c devices:

```
sudo i2cdetect -y -r -a 0
```



(/wiki/File:Jetson_nano_hardware2.png)

  - Parameter: -y is to execute directly regardless of interaction problems, -r is the SMBus read byte command, -a is all addresses, and 0 refers to SMBus 0.
- Scan the register data:

```
sudo i2cdump -y 0 0x68
```

- Write data in register:

```
sudo i2cset -y 0 0x68 0x90 0x55
```

  - Parameter:

Parameter	Meaning
0	Represents the I2C device number
0x68	Represents the address of the I2C device
0x90	Represents the register address
0x55	Represents the data written to the register

- Register data read:

```
sudo i2cget -y 0 0x68 0x90
```

- Parameter:

Parameter	Meaning
0	Represents the I2C device number
0x68	Represents the address of the I2C device
0x90	Represents the register address

SPI

For the official B01 kit, please refer directly to #Official B01 Kit Open SPI.
The version of Jetson Nano B01 with emmc module, that is, Jetson Nano DEV KIT, cannot directly configure the 40PIN header through jetson-io.py. Here is a way to directly modify the device tree file to enable SPI1. We have only verified JetPack4.6.2 so far, this operation needs to **reinstall the system**, please operate with caution.

Hardware Preparation

1. Ubuntu computer host or virtual machine
2. Jetson Nano board
3. Micro USB cable

Install The Necessary Function Libraries

```
sudo apt-get install minicom
sudo apt-get install python-pip nano
sudo pip install pyserial
sudo pip install spidev==3.1
```

Software Setting (Host PC)

You need to modify the device tree file on the system resource. If you have installed the system image with the SDK Manager before, you can modify it directly. If not, please refer to the following resource download section to operate. The following resource pack is the Jetpack4.6.2 version. If you need to download resource packs of other versions, please refer to the resource pack download method (https://www.waveshare.com/wiki/JETSON-NANO-DEV-KIT#accordion4) in the FAQ.

Resource Download

1. Create a new folder on the Ubuntu computer:

```
sudo mkdir sources_nano
cd sources_nano
```

2. Download the following two resource packs:

```
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/jetson-210_linux_r3
2.7.2_aarch64.tbz2
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/tegra_linux_sample-r
oot-filesystem_r32.7.2_aarch64.tbz2
```

3. Move the resource pack to a folder and decompress it (in actual operation, please try to use the tag button to automatically complete the command).

```
sudo mv ~/Downloads/Jetson-210_Linux_R32.7.2_aarch64.tbz2 ~/sources_nano/
sudo mv ~/Downloads/Tegra_Linux_Sample-Root-Filesystem-R32.7.2_aarch64.tbz2 ~/source
s_nano/
```

4. Decompress resources:

```
tar -xjf Jetson-210_Linux_R32.7.2_aarch64.tbz2
cd Linux_for_Tegra/rootfs/
tar -xjf ../../Tegra_Linux_Sample-Root-Filesystem-R32.7.2_aarch64.tbz2
cd ../
sudo ./apply_binaries.sh (If an error occurs, follow the system prompts, and then en
ter the command line again)
```

Modify Device Tree

1. Install the dtc tool and nano editor:

```
sudo apt-get install -y device-tree-compiler
sudo apt-get install nano
```

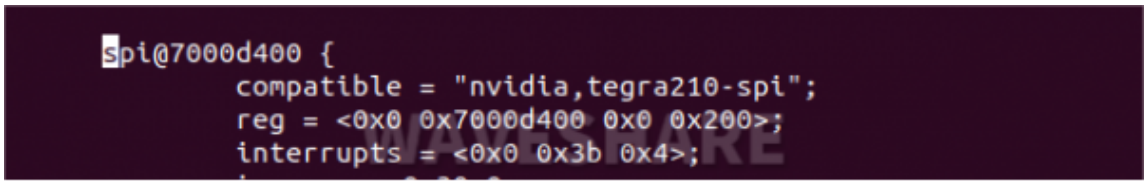
2. Decompile the dts file:

```
cd kernel/dtb
sudo dtc -I dtb -O dts -o tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002
-p3449-0000-b00.dtb
```

3. Modify the dts file:

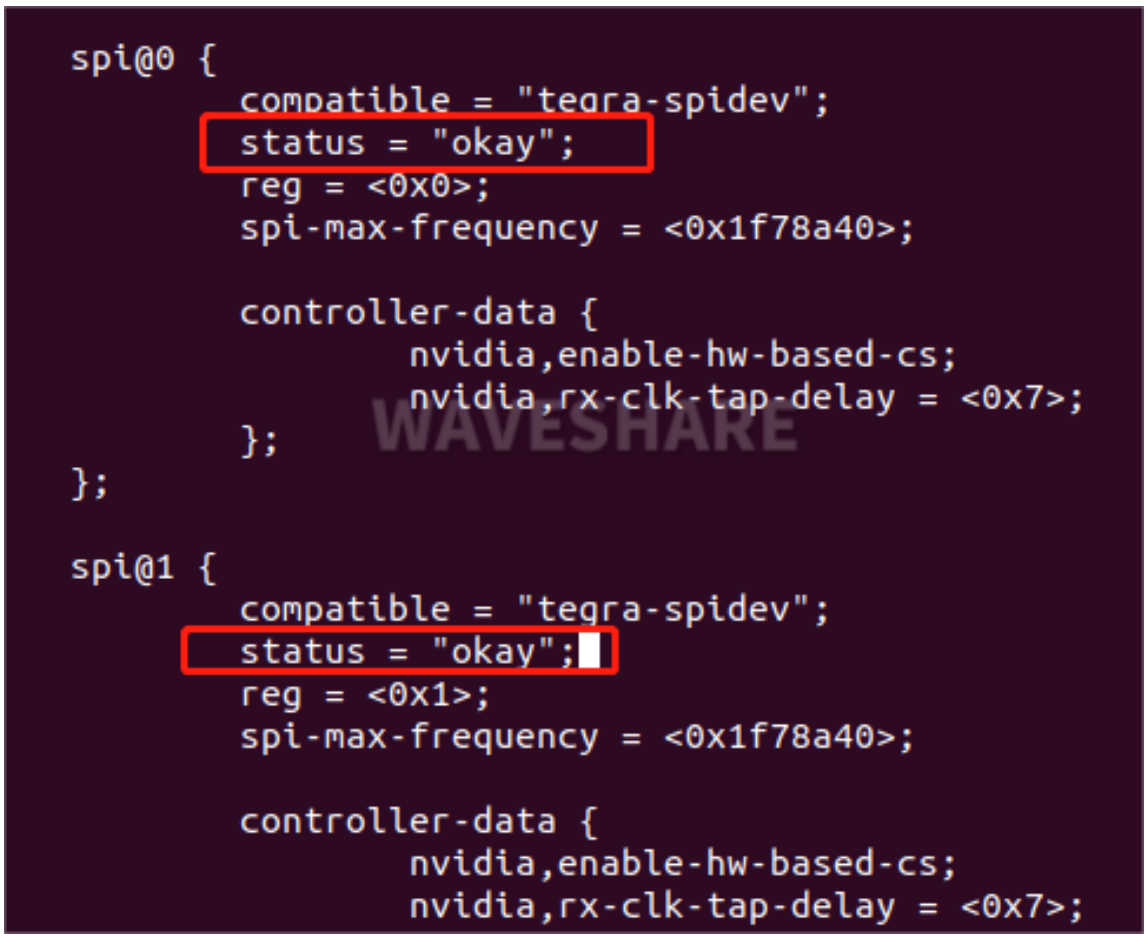
```
sudo cp tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002-p3449-0000-b00-ba
k.dts
sudo gedit tegra210-p3448-0002-p3449-0000-b00.dts
```

Press ctrl^w, type spi@7000d400 (followed by a space), press enter, and find the spi@7000d400{} part:



(/wiki/File:Config1-1re.png)

Add the statement status = "okay" in the spi@0 structure and spi@1 structure respectively;



(/wiki/File:Config1-2re.png)

Press ctrl^w, type spi1-mosi, press enter to find the pin setting of spi1, change "nvidia, function" to spi1, "nvidia,tristate" to 0x0, "Nvidia,enable-input" to 0x1. As in the picture,

all five pins should be operated.

```
spi1_mosi_pc0 {
        nvidia,pins = "spi1_mosi_pc0";
        nvidia,function = "spi1";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};

spi1_miso_pc1 {
        nvidia,pins = "spi1_miso_pc1";
        nvidia,function = "spi1";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};

spi1_sck_pc2 {
        nvidia,pins = "spi1_sck_pc2";
        nvidia,function = "spi1";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};

spi1_cs0_pc3 {
        nvidia,pins = "spi1_cs0_pc3";
        nvidia,function = "spi1";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};

spi1_cs1_pc4 {
        nvidia,pins = "spi1_cs1_pc4";
        nvidia,function = "spi1";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};
```

(/wiki/File:Config1-3re.png)

4. Ctrl+S to save the file and recompile it to dtb. Note, if you need to modify the TF card and other operations, please do it together. Not to modify the wrong position, otherwise, it will easily cause the system to fail to start normally.

```
sudo dtc -I dts -O dtb -o tegra210-p3448-0002-p3449-0000-b00.dtb tegra210-p3448-0002
-p3449-0000-b00.dts
```

Re-program System

Set nano to recovery burning mode, and connect to the Ubuntu computer. Note that only

updating the dtb partition is not supported here, so the entire system needs to be re-burned, and the boot configuration needs to be re-done after burning the system. Therefore, please connect the HDMI screen and keyboard to the Nano in advance.

```
cd ../../
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

Test SPI

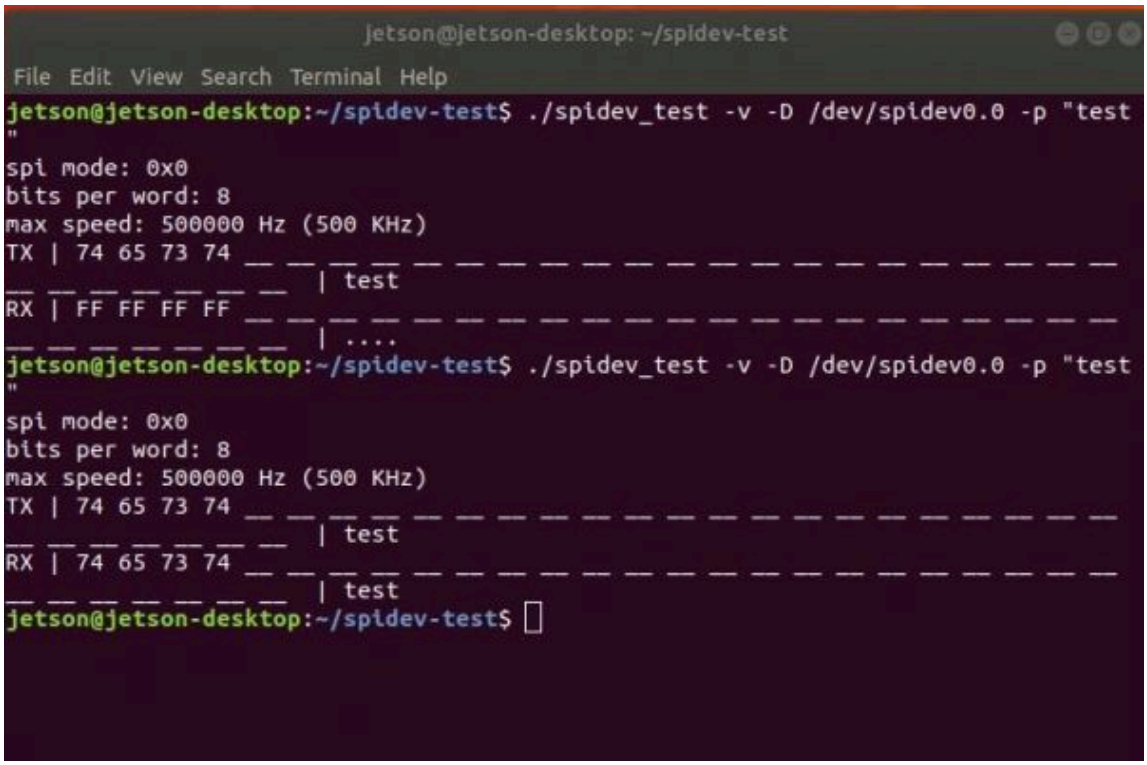
1. After the system is burned successfully, the boot configuration is completed, and the nano will restart automatically.
   Load spidev:

```
sudo modprobe spidev
git clone https://github.com/rm-hull/spidev-test
cd spidev-test/
gcc spidev_test.c -o spidev_test
```

2. Use a cable to short pins 19 and 21 of nano 40PIN, and run the program test:

```
./spidev_test -v -D /dev/spidev0.0 -p "Test"
```

If the printing information is interrupted, RX and TX can send and receive information normally.



(/wiki/File:Config1-4.jpg)

Official B01 Kit Open SPI

Add in the following file:

```
sudo nano /etc/modules-load.d/modules.conf
```

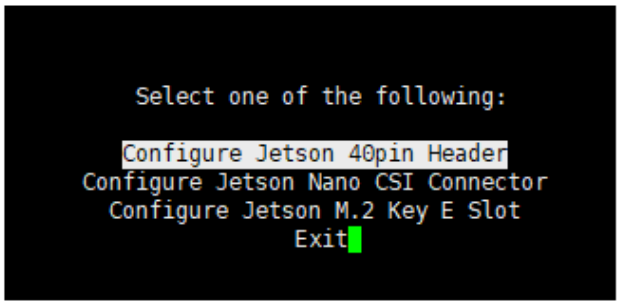
Add a line.

```
spidev
```

Press ctrl+x and then press Y, press Enter to save and exit, and then open the hardware SPI:

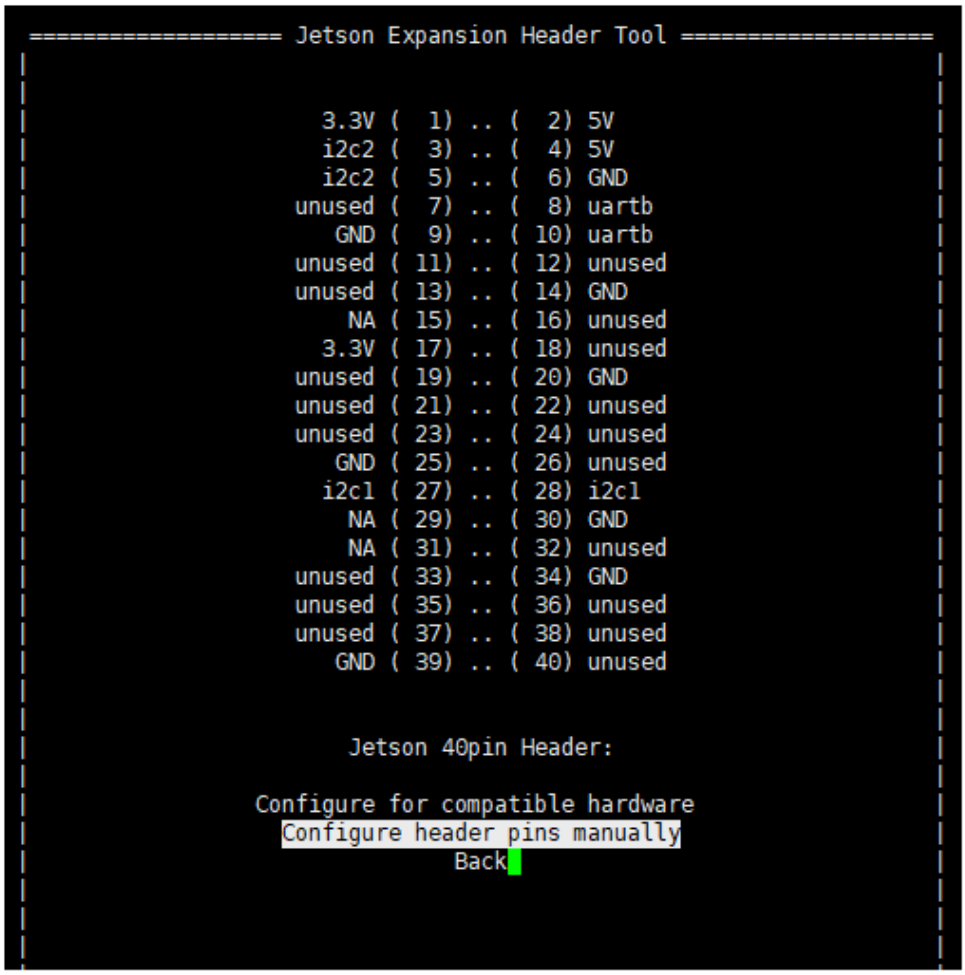
```
sudo /opt/nvidia/jetson-io/jetson-io.py
```

As shown in the picture below:
Choose to configure 40PIN header.



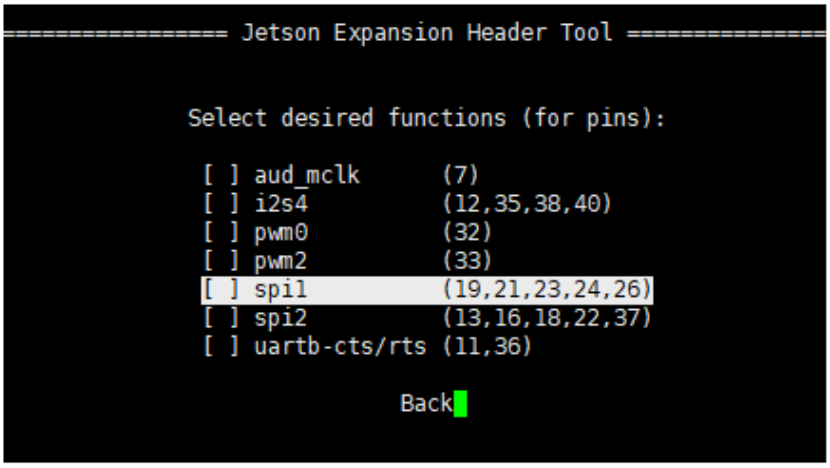
(/wiki/File:RS485-CAN-for-JetsonNano_wiki1.png)

Use the keyboard to select down, and configure pins.



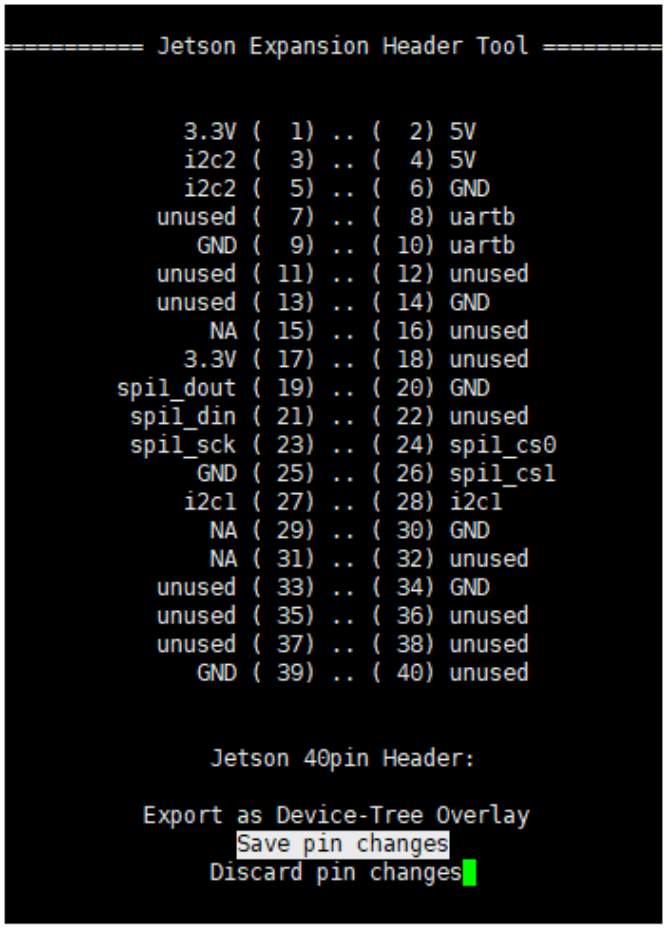
(/wiki/File:RS485-CAN-for-JetsonNano_wiki2.png)

Use the keyboard to select down, move to "SPI1", and press "Enter" to confirm, there will be an "*" sign.

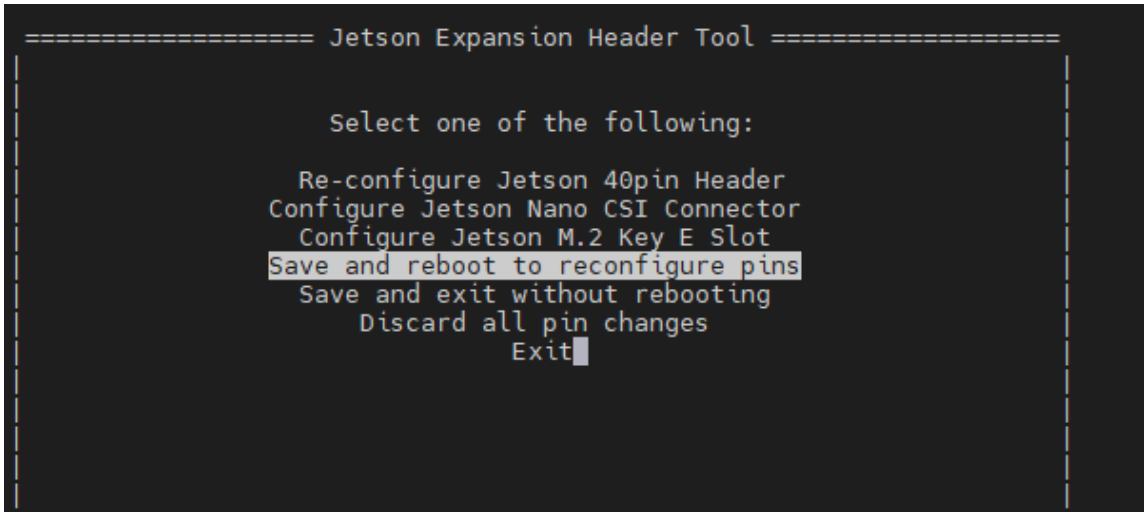


(/wiki/File:RS485-CAN-for-JetsonNano_wiki3.png)

Use the keyboard to select down, save and select restart.



(/wiki/File:RS485-CAN-for-JetsonNano_wiki4.png)



(/wiki/File:Jetson_nano_devio.png)

After restarting, ls /dev/spidev* can see the device number.

(/wiki/File:Jetson_nano_dev_pin.png)

Test SPI (https://www.waveshare.com/wiki/JETSON-NANO-DEV-KIT#Test_SPI)

PWM

For the official B01 kit, please refer directly to Open PWM (https://www.waveshare.com/wiki/JETSON-NANO-DEV-KIT#Official_B01_Kit_Turn_on_PWM).

The version of Jetson Nano B01 with emmc module, that is, Jetson Nano DEV KIT, cannot directly configure the 40PIN pin through jetson-io.py. Here is a way to directly modify the device tree file to enable pwm. The author has only verified JetPack4.6.2 so far, this operation requires **reinstalling the system**, please operate with caution.

Hardware Preparation

1. Ubuntu computer host or virtual machine
2. Jetson Nano board
3. Micro USB cable

Software Setting (Host PC)

You need to modify the device tree file on the system resource. If you have installed the system image with the SDK Manager before, you can modify it directly. If not, please refer to the following resource download section to operate first. The following resource pack is the Jetpack4.6.2 version. If you need to download resource packs of other versions, please refer to the resource pack download method (https://www.waveshare.com/wiki/JETSON-NANO-DEV-KIT#accordion4) in the FAQ.

Resource Download

1. Create a new folder on the Ubuntu computer:

```
sudo mkdir sources_nano
cd sources_nano
```

2. Download the following two resource packs:

```
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/jetson-210_linux_r32.7.2_aarch64.tbz2
https://developer.nvidia.com/embedded/l4t/r32_release_v7.2/t210/tegra_linux_sample-root-filesystem_r32.7.2_aarch64.tbz2
```

3. Move the resource pack to a folder and decompress it (in actual operation, please try to use the tag button to automatically complete the command).

```
sudo mv ~/Downloads/Jetson-210_Linux_R32.7.2_aarch64.tbz2 ~/sources_nano/
sudo mv ~/Downloads/Tegra_Linux_Sample-Root-Filesystem-R32.7.2_aarch64.tbz2 ~/sources_nano/
```

4. Decompress resources.

```
tar -xjf Jetson-210_Linux_R32.7.2_aarch64.tbz2
cd Linux_for_Tegra/rootfs/
tar -xjf ../../Tegra_Linux_Sample-Root-Filesystem-R32.7.2_aarch64.tbz2
cd ../
sudo ./apply_binaries.sh (If an error occurs, follow the system prompts, and then en
ter the command line again)
```

Modify device tree

1. Install the dtc tool:

```
sudo apt-get install -y device-tree-compiler
```

2. Decompile the dts file:

```
cd kernel/dtb
sudo dtc -I dtb -O dts -o tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002
-p3449-0000-b00.dtb
```

3. Modify the dts file:

```
sudo cp tegra210-p3448-0002-p3449-0000-b00.dts tegra210-p3448-0002-p3449-0000-b00-ba
k.dts
sudo gedit tegra210-p3448-0002-p3449-0000-b00.dts
```

Find the pe6 module

```
pe6 {
        nvidia,pins = "pe6";
        nvidia,function = "pwm2";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};
```

(/wiki/File:Pe6RE.png)

Change function to pwm2, tristate to 0x0, enable-input to 0x1.

Find the lcd_bl_pwm_pv0 module.

```
lcd_bl_pwm_pv0 {
        nvidia,pins = "lcd_bl_pwm_pv0";
        nvidia,function = "pwm0";
        nvidia,pull = <0x1>;
        nvidia,tristate = <0x0>;
        nvidia,enable-input = <0x1>;
};
```

(/wiki/File:Lcd_bl_pwm_pv0RE.png)

Change function to pwm0, tristate to 0x0, enable-input to 0x1.

4. Ctrl+S to save the file and recompile it to dtb. Note, if you need to modify the TF card and other operations, please do it together. Not to modify the wrong position, otherwise, it will easily cause the system to fail to start normally.

```
sudo dtc -I dts -O dtb -o tegra210-p3448-0002-p3449-0000-b00.dtb tegra210-p3448-0002
-p3449-0000-b00.dts
```

Reprogramming system

Set nano to recovery flashing mode, and connect to Ubuntu computer. Note that only updating the dtb partition is not supported here, so the entire system needs to be re-flashed, and the boot configuration needs to be re-done after flashing the system. Therefore, please connect the HDMI screen and keyboard to the Nano in advance.

```
cd ../../
sudo ./flash.sh jetson-nano-emmc mmcblk0p1
```

Complete boot configuration.

Enter the following command in the terminal:

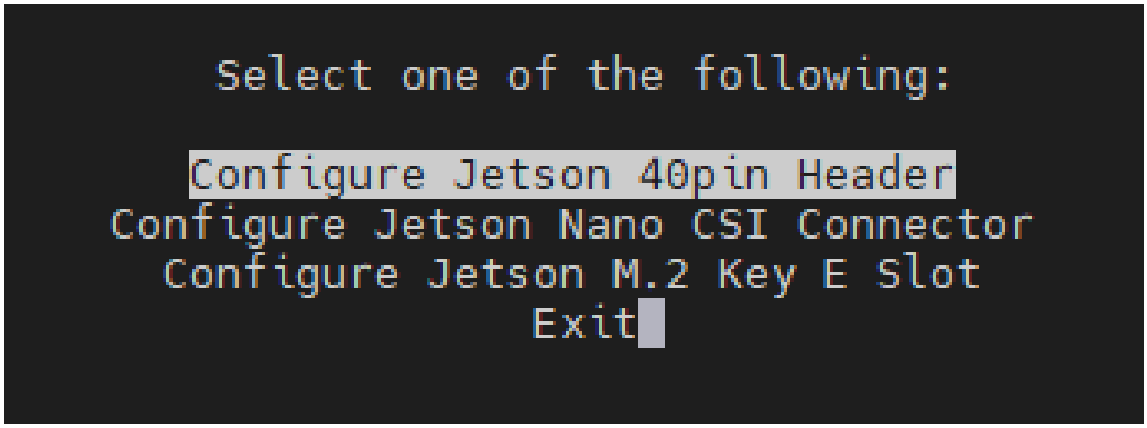
```
echo 0 > /sys/class/pwm/pwmchip0/export
echo 8333333 > /sys/class/pwm/pwmchip0/pwm0/period
echo 4166667 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
sudo cat /sys/kernel/debug/pwm
```

Official B01 Kit Turn on PWM

Input on the terminal:

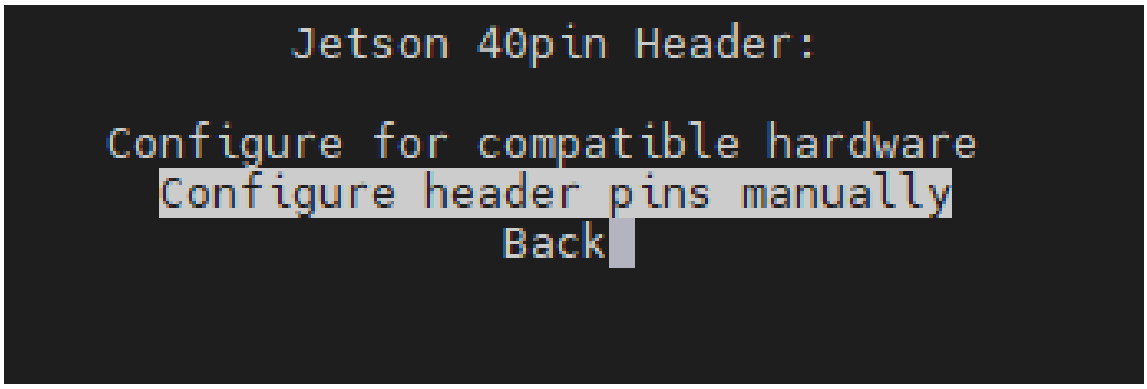
```
sudo /opt/nvidia/jetson-io/jetson-io.py
```

Select Configure Jetson 40Pin Header.



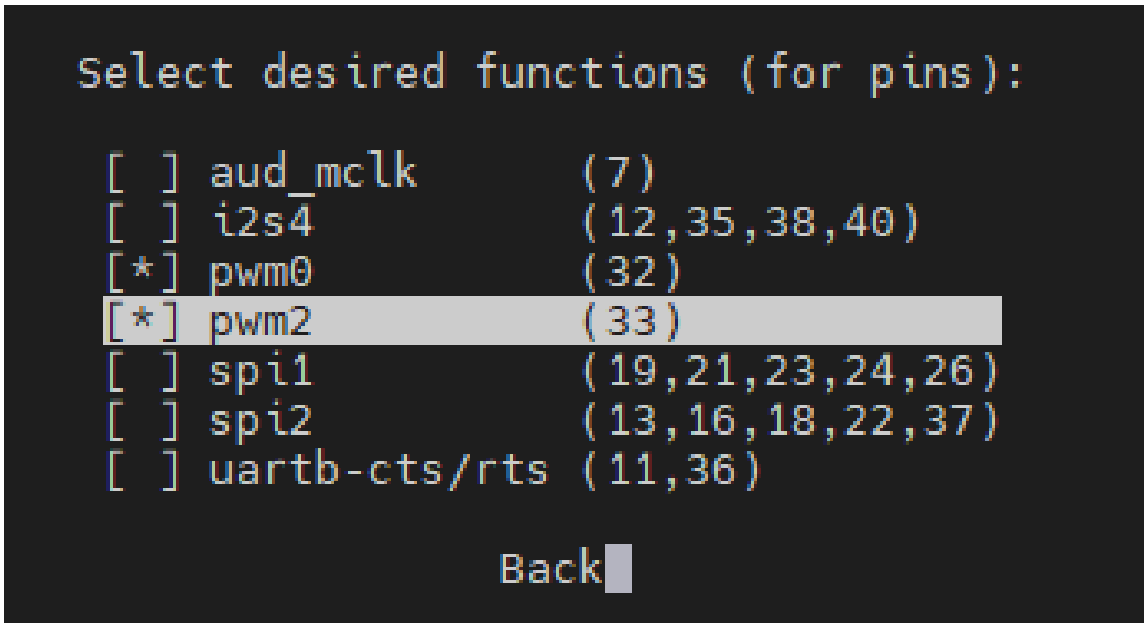
(/wiki/File:Pwm-configuration-1.png)

Select the keyboard down, configure pins.



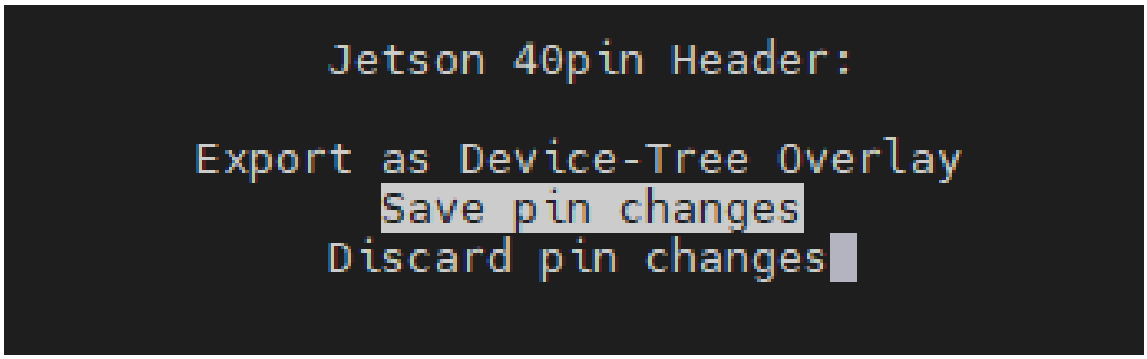
(/wiki/File:Pwm-configuration-2.png)

Select the keyboard down, move to pwm2 and pwm0, and press Enter to confirm, there will be an * sign.



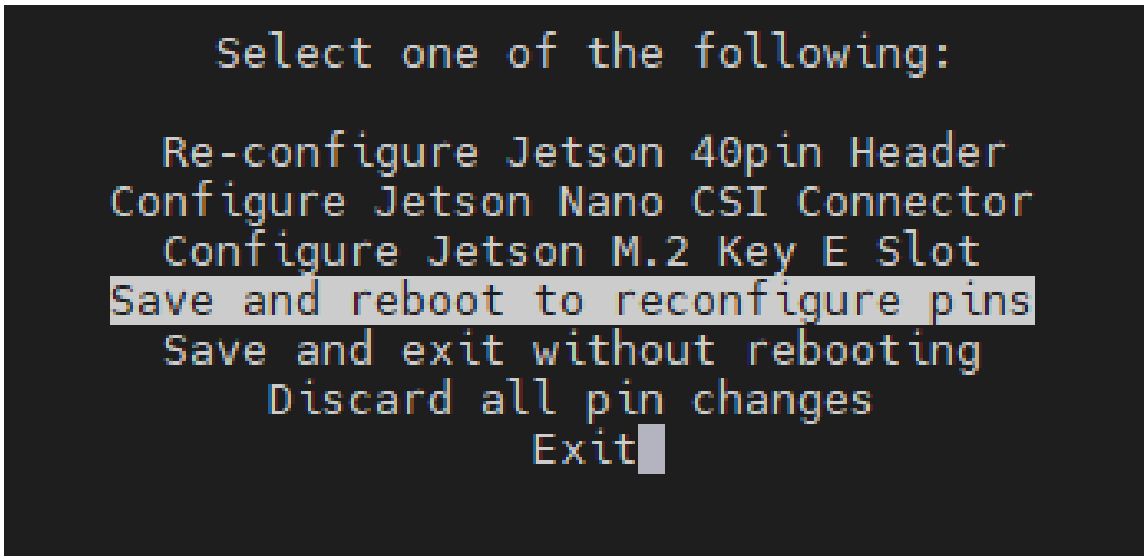
(/wiki/File:Pwm-configuration-3.png)

Save Pin.



(/wiki/File:Pwm-configuration-4.png)

Restart.



(/wiki/File:Pwm-configuration-5.png)

Enter the following commands to enable PWM.

```
echo 0 > /sys/class/pwm/pwmchip0/export
echo 8333333 > /sys/class/pwm/pwmchip0/pwm0/period
echo 4166667 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
sudo cat /sys/kernel/debug/pwm
```

Resources

Software

- Panasonic_SDFormatter (https://files.waveshare.com/upload/d/d7/Panasonic_SDFormatter.zip)
- Win32DiskImager (https://files.waveshare.com/upload/7/76/Win32DiskImager.zip)
- MobaXterm (https://files.waveshare.com/upload/c/c3/MobaXterm_Portable_v22.0.zip)
- VNC-Viewer (https://files.waveshare.com/upload/4/4e/VNC-Viewer-6.21.1109-Windows.zip)
- Advanced IP Scanner (https://www.advanced-ip-scanner.com/)

- Nomachine (https://files.waveshare.com/upload/5/5f/Nomachine_7.10.1_1_arm64.zip)

Learning Tutorial

- Jetson nano to boot the system from a USB Flash Disk (https://www.waveshare.com/wiki/Setting_up_Jetson_nano_to_boot_the_system_from_a_USB_Flash_Disk)
- Jetson Nano Case (C) (https://www.waveshare.com/wiki/Jetson_Nano_Case_(C))

Jetson Official Resources

- Jetson Nano Developer Kit User Guide (https://files.waveshare.com/upload/6/6f/Jetson_Nano_Developer_Kit_User_Guide.pdf)
- Jetson Nano Get Start (https://www.nvidia.com/JetsonNano-Start)
- Jetson Nano 3D Drawing (https://developer.nvidia.com/embedded/dlc/jetson-nano-3D-CAD-Step-Model)
- Jetson Nano Developer Kit (B01) 3D Drawing (https://developer.nvidia.com/jetson-nano-developer-kit-b01-3d-cad-step-model)
- Jetson Download Center (https://developer.nvidia.com/embedded/downloads)
- Jetson Nano Forum (https://forums.developer.nvidia.com/c/agx-autonomous-machines/jetson-embedded-systems/jetson-nano/76/)
- Jetson Github (https://github.com/dusty-nv)
- NVIDIA further study (https://www.nvidia.cn/deep-learning-ai/education/)
- NVIDIA Multimedia Description (https://docs.nvidia.com/jetson/l4t/index.html#page/Tegra%2520Linux%2520Driver%2520Package%2520Development%2520Guide%2Fmultimedia.html%23)
- JETSON-NANO-LITE-DEV-KIT 3D Model (https://developer.nvidia.com/jetson-nano-2gb-developer-kit-3d-cad-step-model)

Courses

- Free AI courses based on Jetson Nano (https://courses.nvidia.com/courses/course-v1:DLI+C-RX-02+V1/about)

3D

- JETSON-NANO-DEV-KIT-3D (https://files.waveshare.com/upload/7/75/JETSON-NANO-DEV-KIT-3D.zip)

TF Card Image

- TF Card Image (https://drive.google.com/file/d/1rc2Wa57RhhgR1eaY5o5tmFkFH-I1yJT4/view?usp=share_link)

FAQ

Question:The space allocated after image flashing on the TF card system is

less than the space of the actual storage device?

Answer:

This expansion method is for TF card or USB flash drive space with enough space, but the allocated space after image programming is less than the space of the actual storage device.
You can use the resizing script to resize the file system to the full storage size.
Open a terminal and enter the following command:

```
cd /usr/lib/nvidia/resizefs/
sudo chmod 777 nvresizefs.sh
sudo ./nvresizefs.sh
sudo reboot
```

Question:Why can't recognize the TF card after I re-flash the system?

Answer:

Refer to the steps of programming bootloader for TF card boot in Wiki to modify the device tree before recognizing the TF card, we have already programmed the system and flashed the bootloader when we shipped it.

Question:After programming the image of Jetson Nano, the TF card is not recognized on the Windows computer?

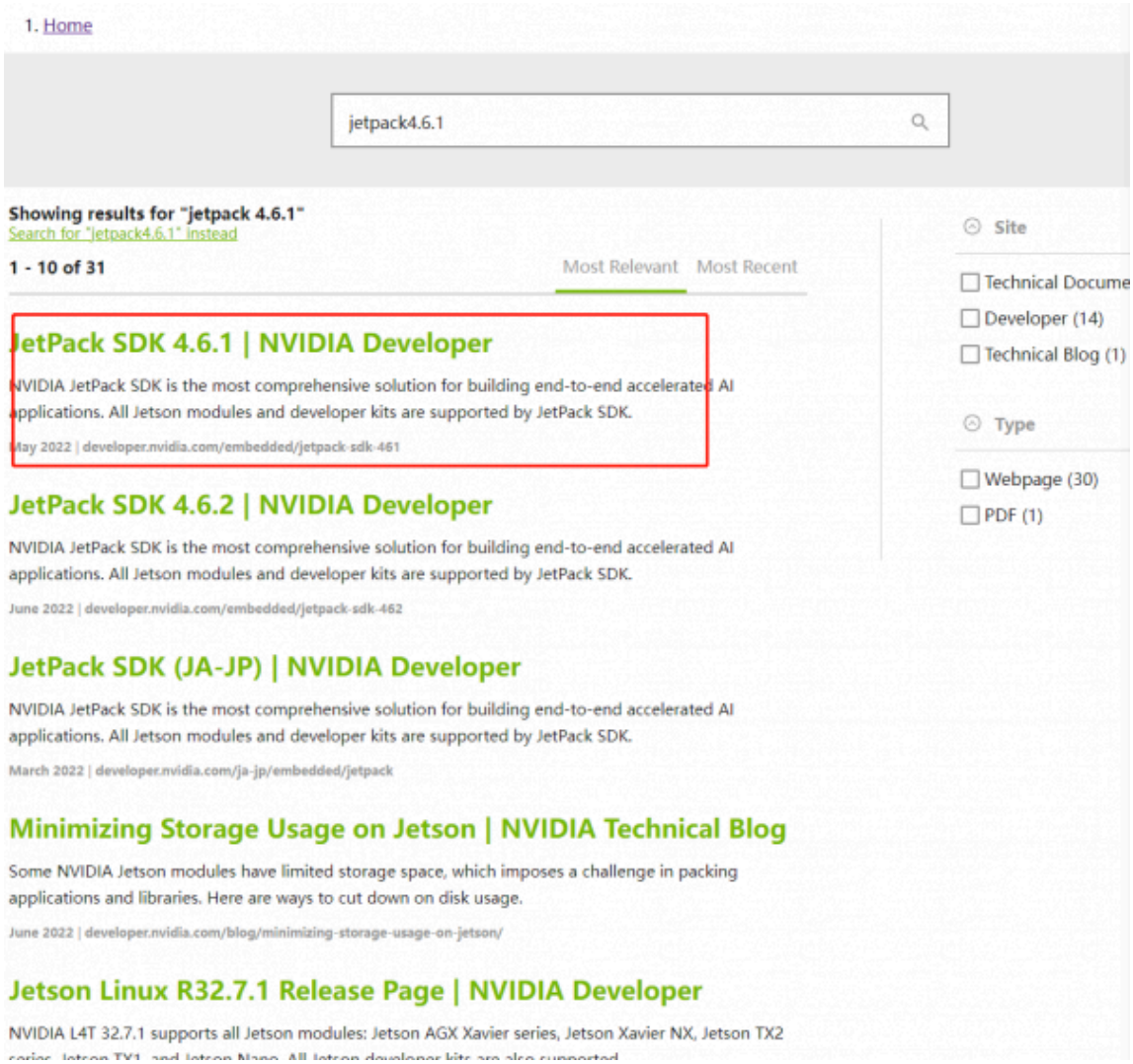
Answer:

Due to the partition problem, the TF card with the programmed image of Jetson Nano cannot recognize the drive letter normally on the Windows computer, if you need to reformat, search for Disk Management and open the disk management interface in the search bar of Windows. Find the mobile disk where the TF card is located (be careful not to mistake it for another disk), right-click Delete Volume, and then create a new volume, formatted by default. After the default formatting, the drive letter of TF will be recognized again. At this time, the TF card space memory is not correct, pay attention to the need to use formatting software to quickly format the new drive letter. After formatting, if the memory space of the TF card is normal, the new image can be re-flashed normally.

Question:How can I download the resource pack?

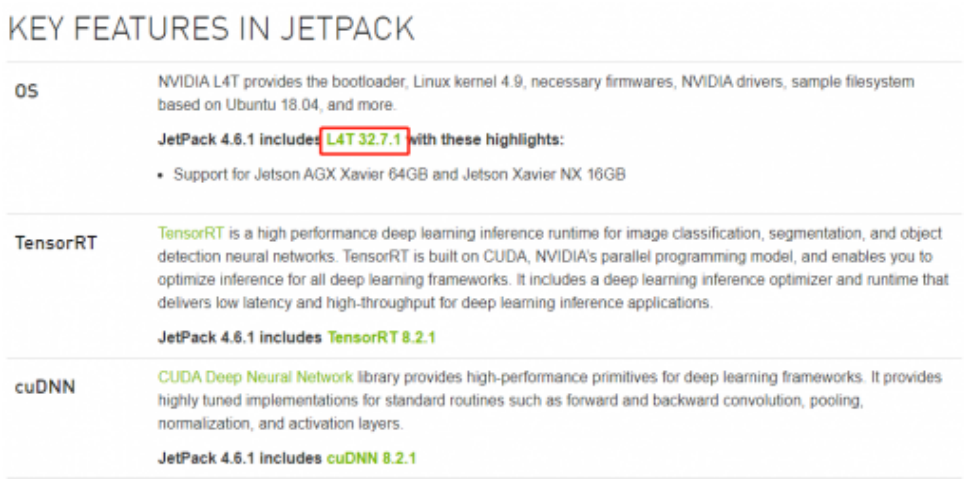
Answer:

First enter the official NVIDIA website (https://developer.nvidia.com/), click search in the upper right corner, enter the version of Jetpack you need to download, such as **jetpack 4.6.1**, and then press enter to the following interface.



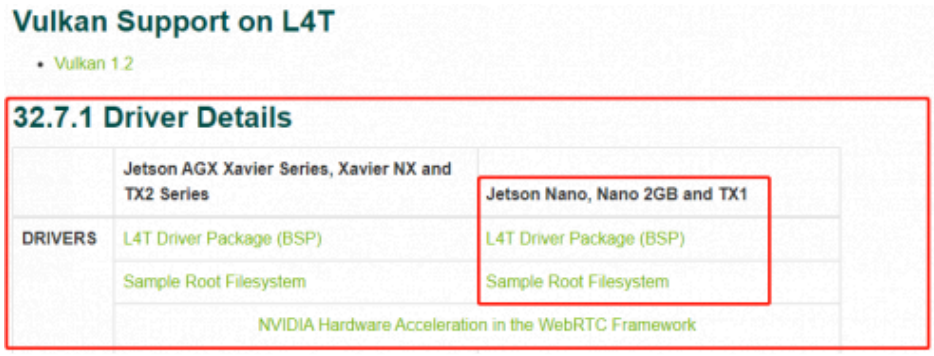
(/wiki/File:Jetson_Dev_FAQ02.png)

Click to enter, drop down to find **KEY FEATURES IN JETPACK**, click L4T 32.7.1, as shown below:



(/wiki/File:Jetson_Dev_FAQ03.png)

Find **32.7.1 Driver Details**, then find the Jetson Nano column and download the L4T Driver Package (BSP) and Sample Root Filesystem files.



(/wiki/File:Jetson_Dev_FAQ04.png)

Question:There is a TF card image in hand that is not Jetpack 4.6, and the TF card cannot start the development board normally?

Answer:

If you have a TF card image, please note that the image on EMMC needs to match the jetpack on the TF card image, and the image version below jetpack4.6 needs to reinstall by the system.

Question:Does the Jetson Nano have voltage protection?

Answer:

Yes, with overvoltage protection.

Question:What should I do if there are many pop-ups reminding me to format the TF card after programming the image we provided?

Answer:

Just click the X off in the upper right corner of the pop-up window, and the image programming will be completed to partition the TF card, no need to worry about it, just use it directly.

Question:Why can't the A02 core board with JETSON-IO-BASE-A board power on?

Answer:

The JETSON-IO-BASE-A board is not fully compatible with the A02 version, only with the official B01. Specific development board information is located in the location below the core board.

Question:Does it support PCI network port?

Answer:

No, it only supports NGFF(M.2).

Question:What is the computing power?

Answer:

0.5TFLOPS.

Question:Does the carrier board support other core board?

Answer:

No, it only supports the Jetson Nano core board.

Question:Does the M2.Key support the SSD?

Answer:

No.

Question:Any tutorials on pose estimation, motion recognition, background removal and monocular depth?

Answer:

Please refer to this link. (https://github.com/dusty-nv/jetson-inference#deploying-deep-learning)

Question:How to use the 2GB version of wifi?

Answer:

To use Wifi, you can only use usb's 802.11ac wireless adapter.

Question:What is the operating temperature of Jetson Nano Dev-Kit?

Answer:

The working temperature of the kit is 0-70°C.

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now (https://service.waveshare.com/)