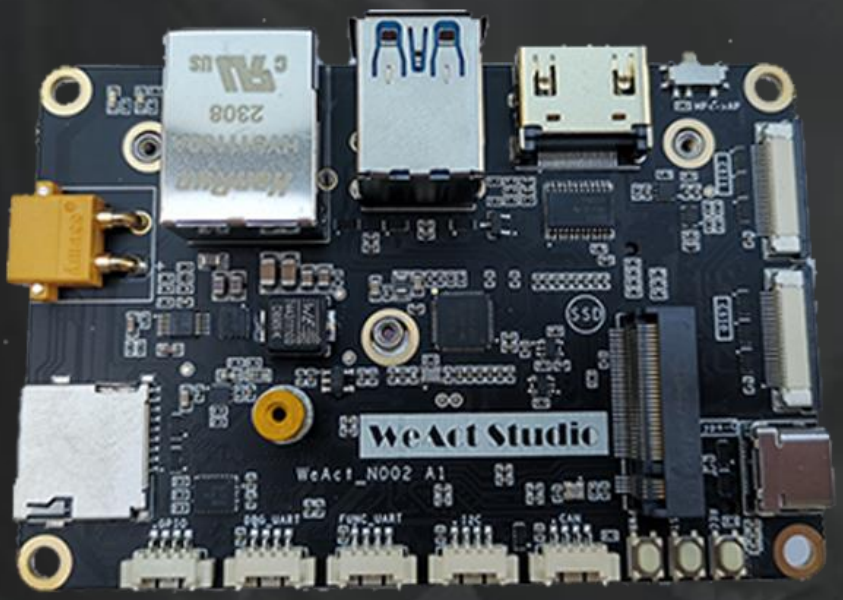


WeAct Studio

WEACT-N002 CARRIER BOARD

Tutorial

WeAct Studio

Directory

Revision History 3

1. Build a Flash environment 4

2. FLASH SYSTEM image(Example with XavierNX) 7

3. Update Device For Core Board （Example with XavierNX） 11

4. Install NVIDIA ComponenT 15

5. Environment backup and image recovery 18

6. System migration to nvme SSD 21

7. System migration to SD Card 24

8. Install JTOP (control fan, view system information) 27

9. Using CAN for communicatioN 28

10. Using GPIO in SHELL 30

ConTACT US 32

REVISION HISTORY

Draft Date	Revision	Description
2023.07.16	V1.0	1. Initial
2024.02.16	V2.0	1. Revise 2.Add GPIO Tutorial

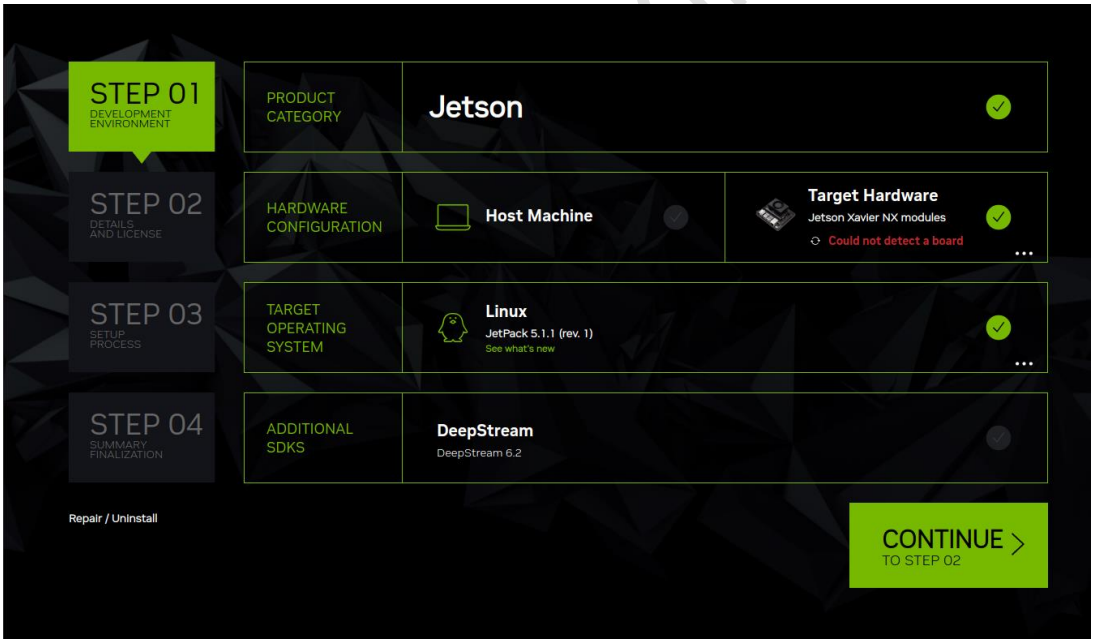
1. BUILD A FLASH ENVIRONMENT

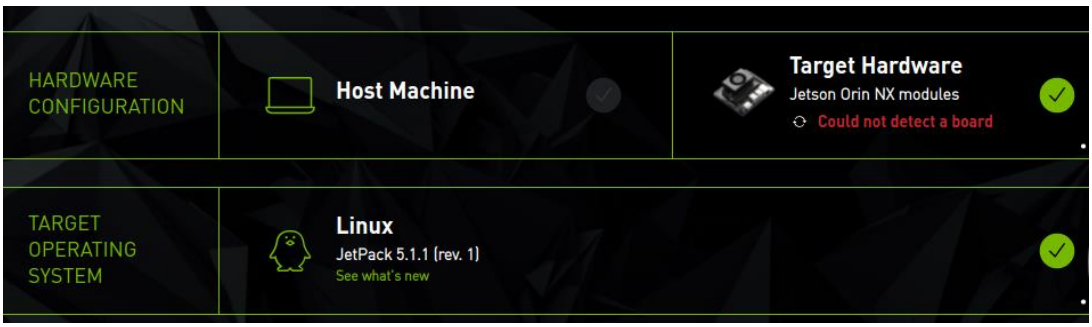
a) First, you need a computer with **Ubuntu 18.04** or above as the host to flash NANO/NX, or you can install VMware on windows.

b) Download the latest SDK manager from NVIDIA and install it in Ubuntu 18.04 (You need to register an NVIDIA account, which will also be used later).

- SDK-Manager Link: https://developer.nvidia.com/nvidia-sdk-manager

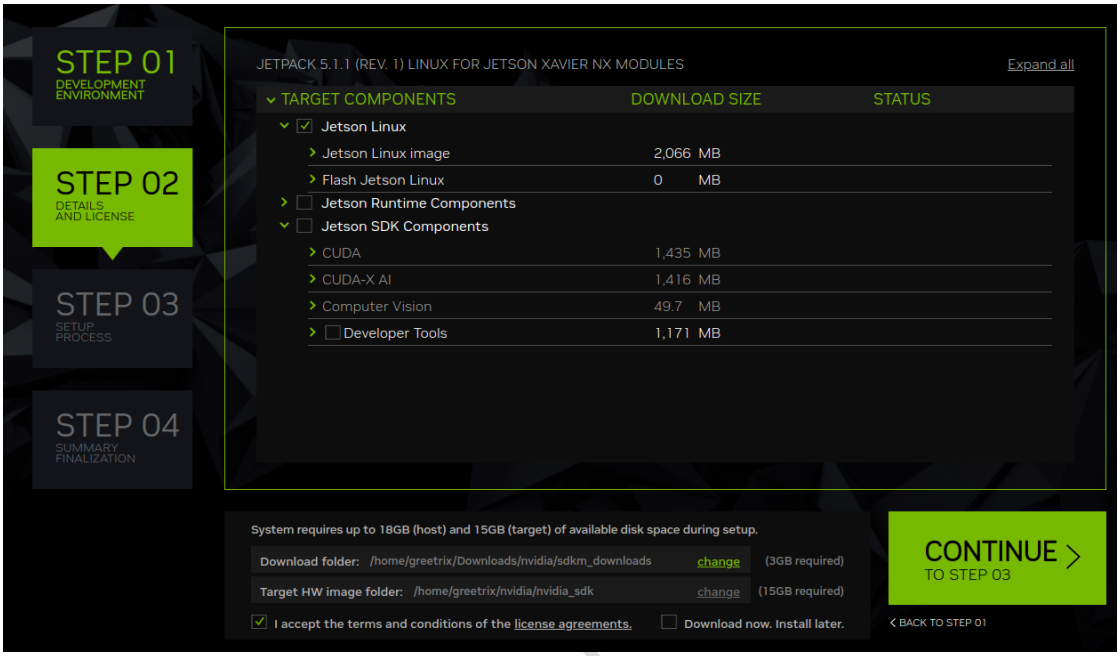
c) Select the relative **Target Hardware** and **JetPack** version, uncheck the **HostMachine(Save the disk memory),** take **XavierNX** as an example, and click **continue.**



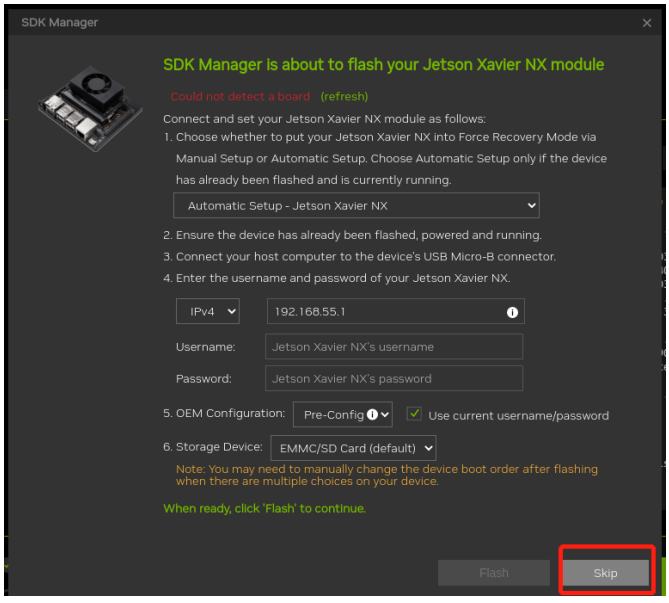


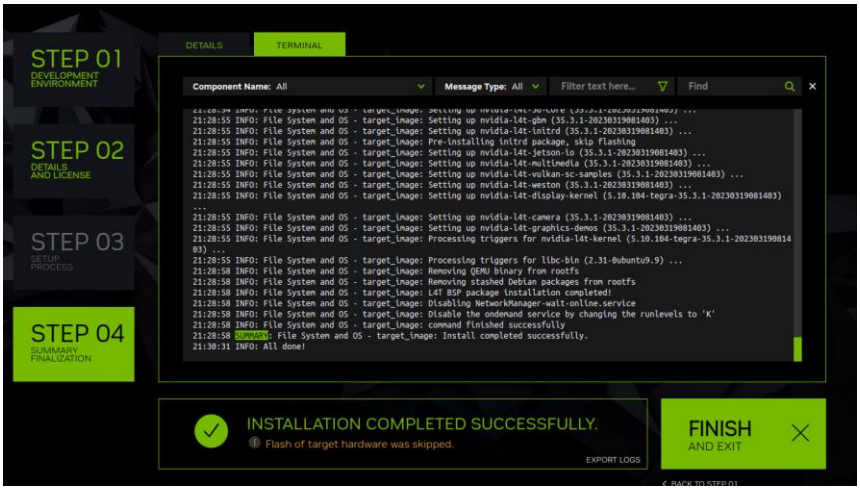
For **Orin Nano/NX**, Please use the **Orin NX** latest jetpack System (Common)

d) **Check** I accept the terms and conditions of the license agreements, **uncheck** the Jetson SDK components, and click continue to proceed to the next step(SDK will install in the next chapter indepent).

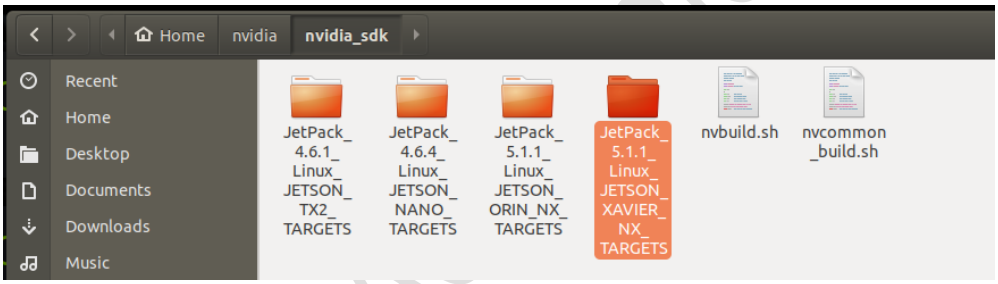


Note：Please download and install in a smooth network environment. When the download or installation fails, click **Retry** to continue until **all the status is installed and green is displayed**. During the installation process, a network burning message will pop up and select **skip**. (**We will use the script to flash the system for different core board adaptive but not SDK manager**).





e) After the installation is successful, the required files will be burned with the corresponding version under **~/NVIDIA/nvidia_sdk/** .

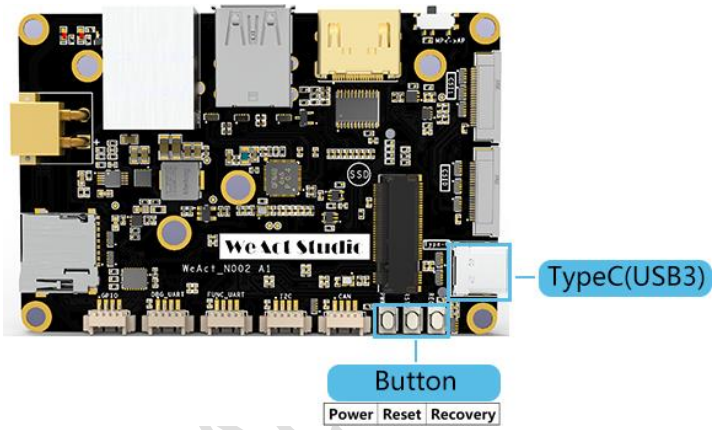


f) Install Python support through **sudo apt get install python** on the terminal for subsequent environment burning.

2. FLASH SYSTEM IMAGE(EXAMPLE WITH XAVIERNX)

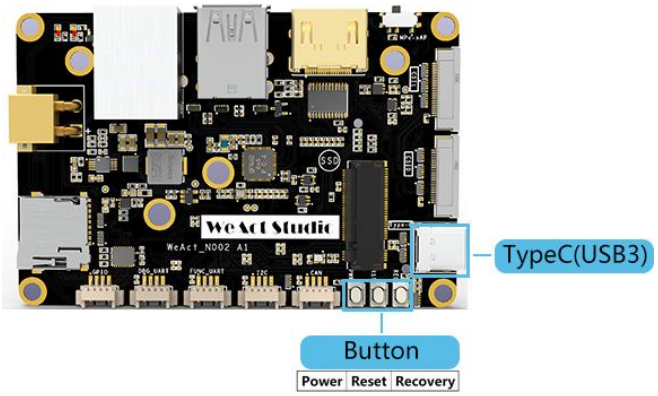
Note: This chapter is used for core board without system or if the system needs to be rewritten. If the core board has a system, only the device tree will be updated. Please skip this chapter.

1. Connect **USB Type-C** cable to carrier board **USB Type-C** connector.



1. Switch slide button to MP **(Manual Power On)**, 2 ways enter Recovery mode below:
   a) Long press **REC** key，press **PWR** to power on，release **REC** key that make the system to enter recovery mode.
   b) Press **PWR** to power on，long press **REC** Key，press **RST** to enter recovery mode.

   NVIDIA USB Driver will appear in the lower right of VMware, or open the terminal and enter **lsusb** command, the NVIDIA Corp will be found and the **fan speed will be set to max**.




```
_XAVIER_NX_TARGETS/Linux_for_Tegra$ lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 004: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 003 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 003 Device 006: ID 0955:7e19 NVidia Corp.
Bus 003 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 002: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

2. **Orin Serial (Orin NX/Orin Nano) need to modify this and put SSD in the carrier board M.2 slot.**
   a) Modify Linux_for_Tegra/bootloader/t186ref/BCT/tegra234-mb2-bct-misc-p3767-0000.dts
   b) Delete line: cvb_eeprom_read_size = <0x100>
      Add line: cvb_eeprom_read_size = <0x0>

3. Enter **~/nvidia/nvidia_sdk/JetPack_5.1.1_Linux_JETSON_XAVIER_NX_TARGETS/Linux_for_Tegra**:
   a) **Copy the git Flash_CMD/03_Jetson XavierNX/run_flash.sh to this directory.**
   b) **Open the terminal and run the script: sudo bash run_flash.sh.**

   Wait the flashing finishing and others CoreBoard flashing command below.

Different CoreBoard Flashing Command

Note: -SD is used to core board which with SD slot，-EMMC is used to core board which without SD slot

Device	Flashing Command
Nano-SD	Flash_CMD/01_Jetson Nano/01_Nano_SD/run_flash.sh

Nano-EMMC	Flash_CMD/01_Jetson Nano/01_Nano_EMMC/run_flash.sh
TX2-NX	Flash_CMD/02_Jetson TX2NX/run_flash.sh
Xavier-SD	Flash_CMD/03_Jetson XavierNX/01_XavierNX_SD/run_flash.sh
Xavier-EMMC	Flash_CMD/03_Jetson XavierNX/01_XavierNX_EMMC/run_flash.sh
Orin Nano 4GB	2 steps: 1. Compile [Do not connect the USB] Flash_CMD/04_Jetson OrinNano/01_4GB/run_compile.sh 2. Flash [Please connect the USB] Flash_CMD/04_Jetson OrinNano/01_4GB/run_flash.sh
Orin Nano 8GB	2 steps: 1. Compile [Do not connect the USB] Flash_CMD/04_Jetson OrinNano/02_8GB/run_compile.sh 2. Flash [Please connect the USB] Flash_CMD/04_Jetson OrinNano/02_8GB/run_flash.sh
Orin NX 8GB	2 steps: 1. Compile [Do not connect the USB] Flash_CMD/05_Jetson OrinNX/01_8GB/run_compile.sh 2. Flash [Please connect the USB] Flash_CMD/05_Jetson OrinNX/01_8GB/run_flash.sh
Orin NX 16GB	2 steps: 1. Compile [Do not connect the USB] Flash_CMD/05_Jetson OrinNX/02_16GB/run_compile.sh 2. Flash [Please connect the USB] Flash_CMD/05_Jetson OrinNX/02_16GB/run_flash.sh

After updating the device tree, it will be successful! Display, as shown in the following figure.

```
File  Edit  View  Search  Terminal  Help
[    2.1704 ] tegrarcm_v2 --ismb2
[    2.2163 ] tegrahost_v2 --chip 0x19 --align nvtboot_applet_t194_aligned.bin
[    2.2185 ] tegrahost_v2 --chip 0x19 0 --magicid PLDT --appendsigheader nvtboot
_applet_t194_aligned.bin zerosbk
[    2.2193 ] adding BCH for nvtboot_applet_t194_aligned.bin
[    2.2286 ] tegrasign_v3.py --key None --list nvtboot_applet_t194_aligned_sighe
ader.bin_list.xml --pubkeyhash pub_key.key
[    2.2289 ] Assuming zero filled SBK key
[    2.2308 ] Warning: pub_key.key is not found
[    2.2315 ] tegrahost_v2 --chip 0x19 0 --updatesigheader nvtboot_applet_t194_al
igned_sigheader.bin.encrypt nvtboot_applet_t194_aligned_sigheader.bin.hash zeros
bk
[    2.2373 ] tegrarcm_v2 --download mb2 nvtboot_applet_t194_sigheader.bin.encryp
t
[    2.2385 ] Applet version 01.00.0000
[    2.2608 ] Sending mb2
[    2.2609 ] [.............................................] 100%
[    2.2827 ] tegrarcm_v2 --boot recovery
[    2.2836 ] Applet version 01.00.0000
[    3.3139 ] tegrarcm_v2 --isapplet
[    3.3224 ] tegrarcm_v2 --ismb2
[    3.3233 ] MB2 Applet version 01.00.0000
[    3.3698 ] tegrarcm_v2 --ismb2
```

```
t.encrypt [ 30416 bytes ]
[ 635.8133 ] [.............................................] 100%
[ 636.0454 ] tegradevflash_v2 --write MEM_BCT mem_coldboot_sigheader.bct.encrypt
[ 636.0470 ] Bootloader version 01.00.0000
[ 636.0521 ] Writing partition MEM_BCT with mem_coldboot_sigheader.bct.encrypt [
 198656 bytes ]
[ 636.0527 ] [.............................................] 100%
[ 637.4597 ] tegradevflash_v2 --write MEM_BCT_b mem_coldboot_sigheader.bct.encry
pt
[ 637.4613 ] Bootloader version 01.00.0000
[ 637.4728 ] Writing partition MEM_BCT_b with mem_coldboot_sigheader.bct.encrypt
 [ 198656 bytes ]
[ 637.4734 ] [.............................................] 100%
[ 638.8778 ] Flashing completed

[ 638.8779 ] Coldbooting the device
[ 638.8825 ] tegrarcm_v2 --ismb2
[ 638.9047 ] tegradevflash_v2 --reboot coldboot
[ 638.9059 ] Bootloader version 01.00.0000
*** The target t186ref has been flashed successfully. ***
Reset the board to boot from internal eMMC.
```

3. UPDATE DEVICE FOR CORE BOARD (EXAMPLE WITH XAVIERNX)

Note: If you don’ t need to used the below function, please skip this chapter.

Different core board with WeAct decive tree supported

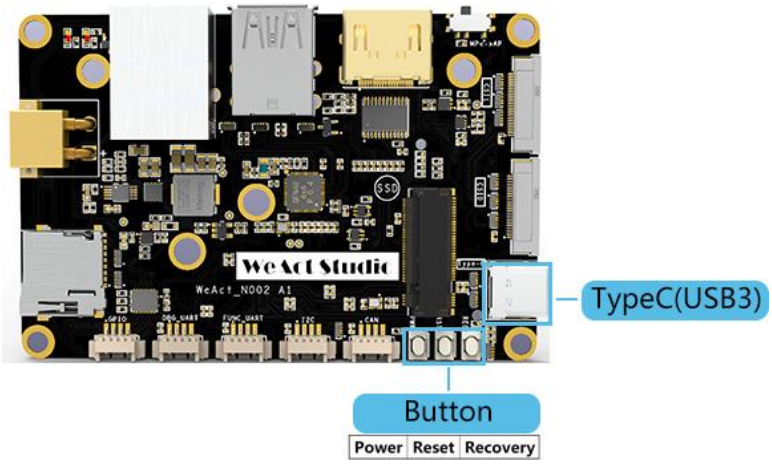
	SD Card	Function UART	Function UART+SD Card
Nano-SD	None	Jetpack4.x [Updating]	None
Nano-EMMC	Jetpack4.x	Jetpack4.x [Updating]	Jetpack4.x [Updating]
TX2NX	Jetpack4.x	Jetpack4.x [Updating]	Jetpack4.x [Updating]
XavierNX	Jetpack5.x	Jetpack5.x [Updating]	Jetpack5.x [Updating]
Orin Nano/NX	None	Jetpack5.x Jetpack6.x [Updating]	None

Note: SD Card -> Can use the SD Card slot

Function UART -> Debug UART change to Normal function UART

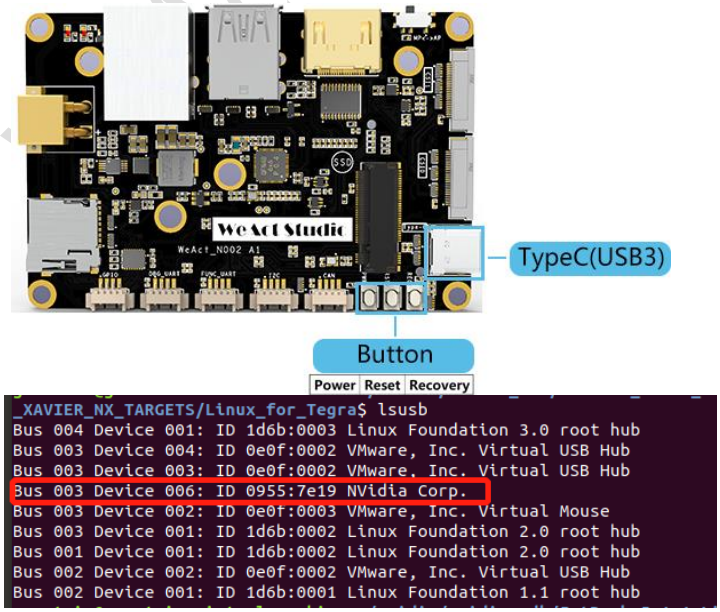
SD Card + Function UART -> Debug UART change to Normal function UART +Can use the SD Card slot

1. Connect **USB Type-C** cable to carrier board **USB Type-C** connector



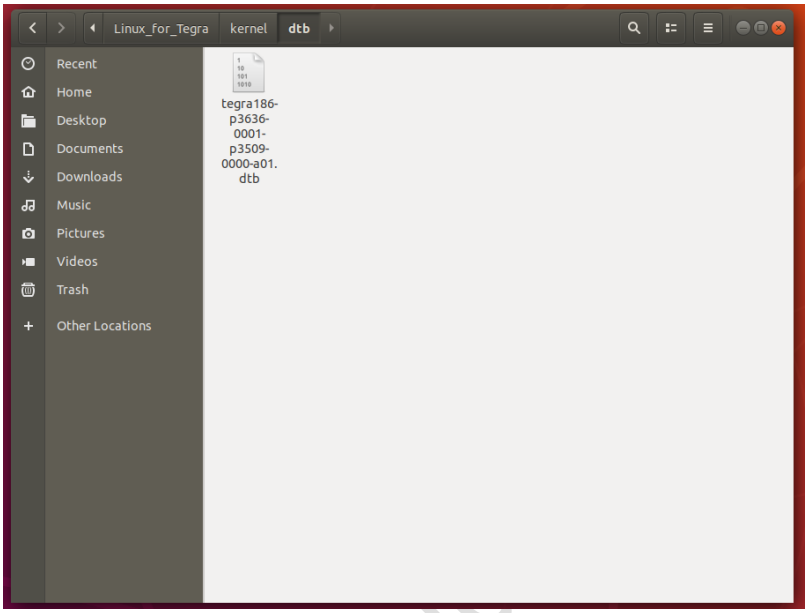
1. Switch slide button to MP **(Manual Power On)**, 2 ways enter Recovery mode below:
   a) Long press **REC** key，press **PWR** to power on，release **REC** key that make the system to enter recovery mode.
   b) Press **PWR** to power on，long press **REC** Key，press **RST** to enter recovery mode.

   NVIDIA USB Driver will appear in the lower right of VMware, or open the terminal and enter **lsusb** command, the NVIDIA Corp will be found and the **fan speed will be set to max**.



1. **Jetpack 4.x version update device-tree(Jetson Nano/TX2NX), example with TX2NX.**
   a) Find the relative decive-tree.

b) Enter **~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/kernel/dtb**, Copy the device-tree(**tegra186-p3636-0001-p3509-0000-a01.dtb)** to this directory.



c) Enter **~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**, open the terminal:

d) Command: **sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-tx2-nx mmcblk0p1,** other core board refer the below figure:

Different Core Board Update Device-Tree Command in Jetpack4.x

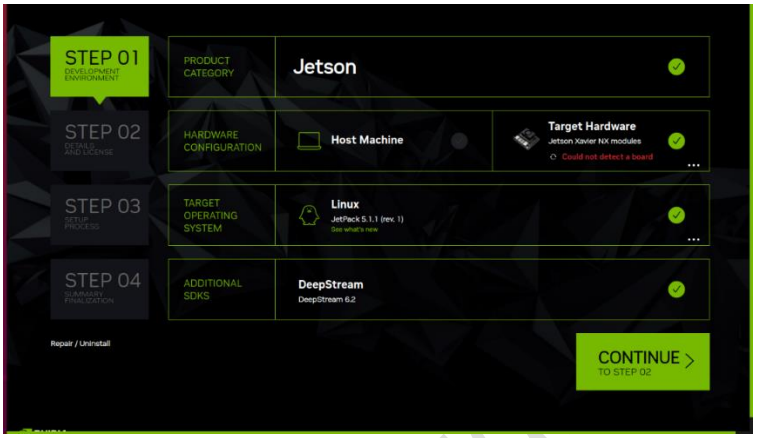
Device	Command
Nano-SD	sudo ./flash.sh -r -k DTB jetson-nano-qspi-sd mmcblk0p1
Nano-EMMC	sudo ./flash.sh -r -k DTB jetson-nano-emmc mmcblk0p1
TX2-NX	sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-tx2-nx mmcblk0p1

3. Jetpack 5.x version update device-tree (Jetson XavierNX)

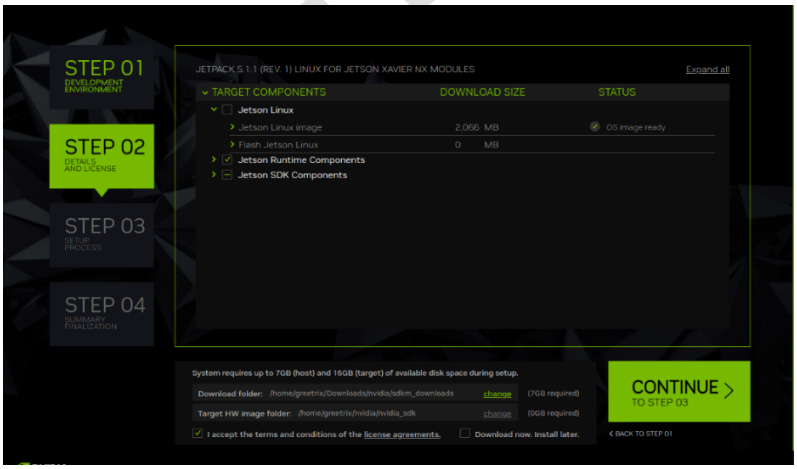
e) Press the PWR button and power on directly, login the system.
f) Copy the relative Device-Tree from Git to the System Desktop (kernel_tegra194-p3668-0001-p3509-0000.dtb).
g) Right Click in Desktop and open the terminal.
h) Command: **sudo rm /boot/dtb/kernel_tegra194-p3668-0001-p3509-0000.dtb.**
i) Command: **sudo cp kernel_tegra194-p3668-0001-p3509-0000.dtb /boot/dtb**.
j) Reboot the system

4. INSTALL NVIDIA COMPONENT

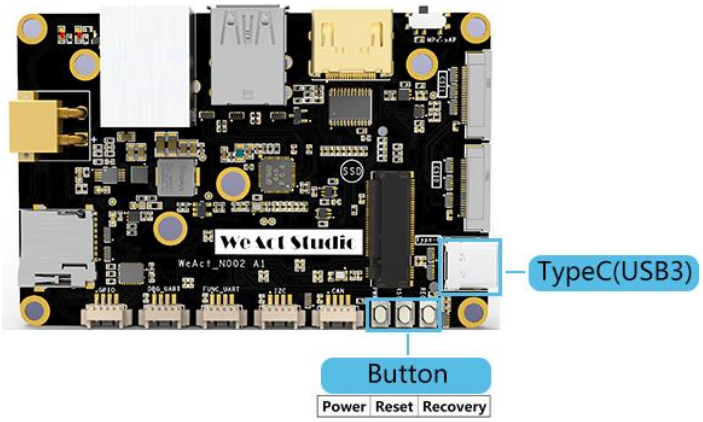
1. Selete the relative **Target Hardware** and **JetPack** version, **uncheck HostMachine**, take **XavierNX** as an example, and click **continue.**



2. Check the **required SDK components**, check **I accept the terms and conditions of the license agreements**, and click **continue** to proceed to the next step.

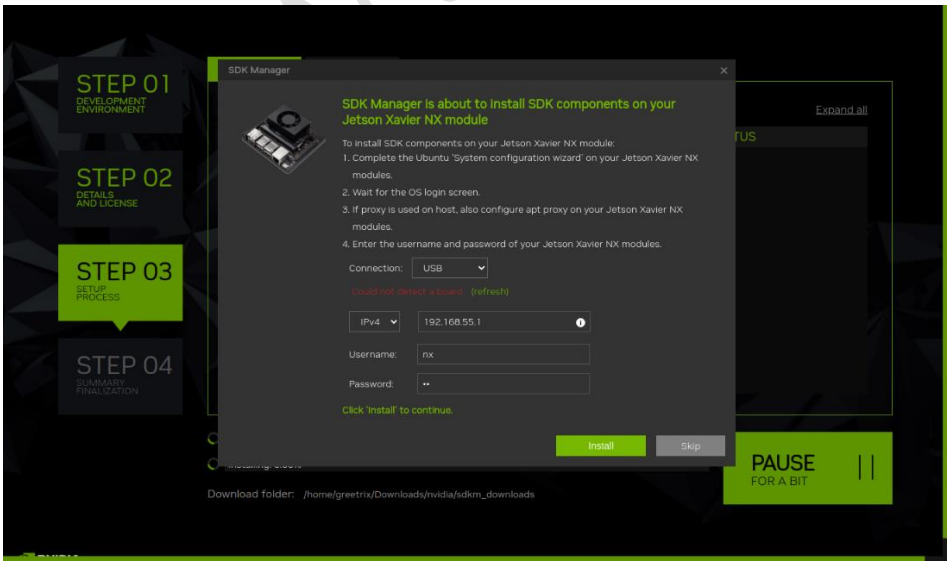


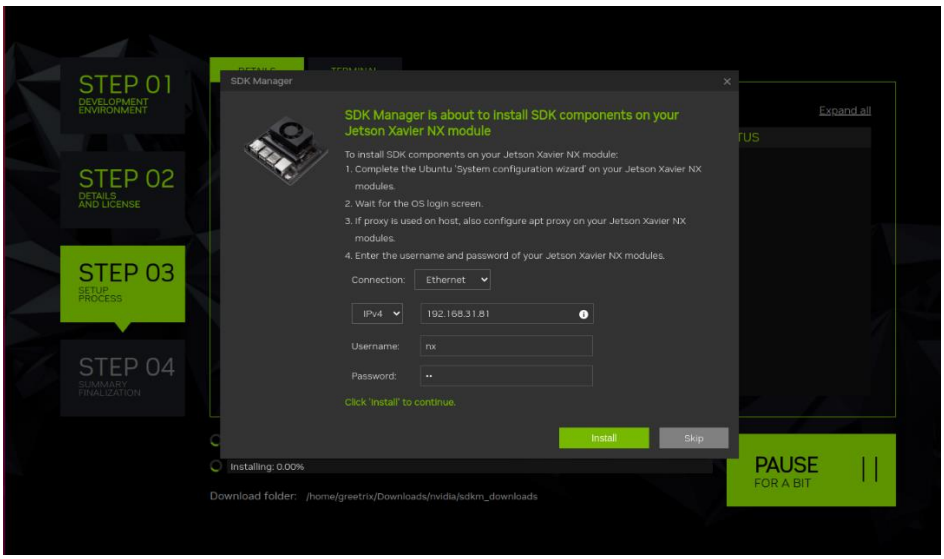
3. Use the **USB Type-C** cable to connect the USB Type-C connector.



4. Turn the **power on key to MP(manual power on)** and press **PWR** key to power on. At this time, **NVIDIA USB drive sign** will appear in the lower right corner of VMware, or open the terminal and enter **lsusb** command to find **NVIDIA Corp**.

5. Enter the **Xavier NX** account and password. **Please keep the Xavier NX connected to the internet or use an IP from the same local area network** (as shown below, the host and Xavier NX are connected to the same router)**.**





6. Wait for the installation to complete.

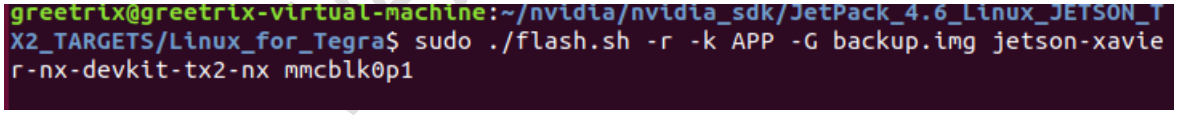
5. ENVIRONMENT BACKUP AND IMAGE RECOVERY

1. **System install in EMMC, take an example with TX2NX**

   a) Referring to Chapter 2, whether it is backup or image burning, enter Recovery mode and note that the image is large. Please ensure that Ubuntu has sufficient space (>40GB).

   b) **Backup:** Taking TX2NX as an example (for other devices, please refer to the previous chapter to modify the Jetson name), backup the existing environment of the core board. Enter ~/nvidia/nvidia_sdk/JetPack_4.6-Linux_JETSON-TX2-TARGETS/Linux_for-Tegra, and open the terminal

Using the mirror backup command: **sudo ./flash.sh -r -k APP -G backup.img jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**, wait for the backup to complete. At this time, there will be a backup.img image in the directory (it is recommended to copy a copy to another location for backup), and the **backup has been successful.**



```
[    9.1920 ] tegrarcm_v2 --boot recovery
[    9.1966 ] Applet version 01.00.0000
[    9.3692 ]
[   10.3763 ] tegrarcm_v2 --isapplet
[   10.3793 ] USB communication failed.Check if device is in recovery
[   10.5068 ]
[   10.8536 ] tegradevflash_v2 --iscpubl
[   10.8565 ] Cannot Open USB
[   11.3572 ]
[   12.3617 ] tegrarcm_v2 --isapplet
[   12.5109 ]
[   12.5142 ] tegradevflash_v2 --iscpubl
[   12.5163 ] Bootloader version 01.00.0000
[   12.6843 ] Bootloader version 01.00.0000
[   12.7463 ]
[   12.7464 ] Reading partition
[   12.7492 ] tegradevflash_v2 --read APP /home/greetrix/nvidia/nvidia_sdk/JetPac
k_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/backup.img
[   12.7511 ] Bootloader version 01.00.0000
[   12.9183 ] [..............................................] 100%
[ 2216.5426 ]
*** The [APP] has been read successfully. ***
        Converting RAW image to Sparse image... greetrix@greetrix-virtual-machin
X2_TARGETS/Linux_for_Tegra$ k_4.6_Linux_JETSON_TX
```

c) **Image Recovery**: Enter ~/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra, Copy the backup.img to Liunx_for_Tegra/bootloader/, and rename to system.img, back to the Linux_for_Tegra directory, open the terminal:

Use the flash command: **sudo ./flash.sh -r jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**, wait it to finish.

```
[   18.0000 ] Writing partition spe-fw_b with spe_sigheader.bin.encrypt
[   18.0298 ] [..............................................] 100%
[   18.0790 ] Writing partition mb2 with nvtboot_sigheader.bin.encrypt
[   18.1057 ] [..............................................] 100%
[   18.1596 ] Writing partition mb2_b with nvtboot_sigheader.bin.encrypt
[   18.1895 ] [..............................................] 100%
[   18.2416 ] Writing partition mts-preboot with preboot_d15_prod_cr_sigheader.bi
n.encrypt
[   18.2710 ] [..............................................] 100%
[   18.6760 ] Writing partition mts-preboot_b with preboot_d15_prod_cr_sigheader.
bin.encrypt
[   18.7053 ] [..............................................] 100%
[   18.7467 ] Writing partition SMD with slot_metadata.bin
[   18.7744 ] [..............................................] 100%
[   18.9037 ] Writing partition SMD_b with slot_metadata.bin
[   18.9302 ] [..............................................] 100%
[   18.9658 ] Writing partition VER_b with emmc_bootblob_ver.txt
[   18.9922 ] [..............................................] 100%
[   19.0322 ] Writing partition VER with emmc_bootblob_ver.txt
[   19.0592 ] [..............................................] 100%
[   19.0966 ] Writing partition master_boot_record with mbr_1_3.bin
[   19.1194 ] [..............................................] 100%
[   19.1525 ] Writing partition APP with system.img
[   19.1800 ] [........                                      ] 016%
```

2. For devices installed on SD cards or SSDs, simply copy the SD card or SSD directly from the host to other storage media.

6. SYSTEM MIGRATION TO NVME SSD

Note: This migration only used in Nano/TX2NX/XavierNX, for the Orin series, please refer to chapter 2.

a) WeAct-N002 carrier board support PCIE3.0 X 4 slot, support 2242 NVME SSD M.2 interfaces SSD

b) NVME Solid HardDisk Confiugration:

- 1. Before configuration, ensure that the system can recognize the nvme SSD. The terminal command is **sudo fdisk -lu.**

```
Disk /dev/nvme0n1: 119.2 GiB, 128035676160 bytes, 250069680 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
```

- 2. Set the NVME with GPT format:

i. Command: **sudo parted /dev/nvme0n1,** enter the parted.

```
tx2nx@tx2nx:~$ sudo parted /dev/nvme0n1
[sudo] password for tx2nx:
GNU Parted 3.2
Using /dev/nvme0n1
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)
```

ii. Command: **mklabel gpt**, make the label with **GPT** format.

```
(parted) mklabel gpt
Warning: The existing disk label on /dev/nvme0n1 will be destroyed and all data on this disk will be lost. Do you want to
Yes/No? Yes
```

iii. Command: **mkpart logical 0 -1,** make the **part** with GPT format.

```
(parted) mkpart logic 0 -1
Warning: The resulting partition is not properly aligned for best performance.
Ignore/Cancel? Ignore
```

iv. Command: **print**, see the result.

```
(parted) print
Model: KBG40ZNS128G NVMe TOSHIBA 128GB (nvme)
Disk /dev/nvme0n1: 128GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End    Size   File system  Name   Flags
 1      17.4kB  128GB  128GB               logic
```

v. Command: **quit**

vi. Command: **sudo fdisk /dev/nvme0n1**

```
(parted) quit
Information: You may need to update /etc/fstab.

tx2nx@tx2nx:~$ sudo fdisk /dev/nvme0n1

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help):
```

vii. Command: Enter **N**, add new part, Enter next.

```
Command (m for help): n
Partition number (2-128, default 2): 2
First sector (250067728-250069646, default 250068992):
Last sector, +sectors or +size{K,M,G,T,P} (250068992-250069646, default 250069646):

Created a new partition 2 of type 'Linux filesystem' and of size 327.5 KiB.
```

viii. Command: Enter **P**, check the filesystem status.

```
Device             Start       End   Sectors   Size Type
/dev/nvme0n1p1        34 250067727 250067694 119.2G Linux filesystem
/dev/nvme0n1p2 250068992 250069646       655 327.5K Linux filesystem
```

ix. Command: **quit**

x. Command: **sudo mke2fs -t ext4 /dev/nvme0n1p1**, format the part

```
tx2nx@tx2nx:~$ sudo mke2fs -t ext4 /dev/nvme0n1p1
mke2fs 1.44.1 (24-Mar-2018)
/dev/nvme0n1p1 contains a ext4 file system
        last mounted on / on Sun Dec 26 11:04:07 2021
Proceed anyway? (y,N)
```

xi. Command: **sudo mount /dev/nvme0n1p1 /mnt**, if success, configurate NVME ok

```
tx2nx@tx2nx:~$ sudo mount /dev/nvme0n1p1 /mnt
tx2nx@tx2nx:~$
```

c) NVIDIA Jetson system migration

- ✓ **Taking tx2nx as an example, other devices can replace the device name in the middle of the command, and the device name can refer to the above command**
- ➢ 1. Command: **git clone https://github.com/jetsonhacks/rootOnNVMe** or

https://github.com/jetsonhacks/rootOnNVMe, download the script

- 2. Go to **rootOnNVMe** directory, command： **./copy-rootfs-ssd.sh**, copy system file to NVME SSD.

```
tx2nx@tx2nx:/home/script/rootOnNVMe-master$ ./copy-rootfs-ssd.sh
mount: /mnt: /dev/nvme0n1p1 already mounted on /mnt.
     17,380,838   0%    2.40MB/s    0:00:06 (xfr#39, ir-chk=1015/44887)
```

- 3. Command： **./setup-service.sh** configure the boot options.

```
tx2nx@tx2nx:/home/script/rootOnNVMe-master$ ./setup-service.sh
==== AUTHENTICATING FOR org.freedesktop.systemd1.reload-daemon ===
Authentication is required to reload the systemd state.
Authenticating as: tx2nx,,, (tx2nx)
Password: Failed to reload daemon: Method call timed out
polkit-agent-helper-1: pam_authenticate failed: Authentication failure
Created symlink /etc/systemd/system/default.target.wants/setssdroot.service → /etc/systemd/system/setssdroot.service.
Service to set the rootfs to the SSD installed.
Make sure that you have copied the rootfs to SSD.
Reboot for changes to take effect.
```

- 4. Refer to chapter 2 and enter the **Recovery** Mode.
- 5. (**The flash environment of Ubuntu**, Refer to last chapter), enter **~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**, command： **sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx nvme0n1p1** update the EMMC boot.

```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx nvme0n1p1
```

```
[   37.3739 ] Coldbooting the device
[   37.3775 ] tegradevflash_v2 --reboot coldboot
[   37.3788 ] Bootloader version 01.00.0000
[   37.5711 ]
*** The target t186ref has been flashed successfully. ***
Make the target filesystem available to the device and reset the board to boot from external nvme0n1p1.
```

- 6. Reboot the TX2NX, Command： **df -l**, now the boot storage space change to SSD.

```
tx2nx@tx2nx:~$ df -l
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/nvme0n1p1 122547172 11949920 104329176  11% /
none             1578060        0   1578060   0% /dev
tmpfs            1962748       52   1962696   1% /dev/shm
tmpfs            1962748    20764   1941984   2% /run
tmpfs               5120        4      5116   1% /run/lock
tmpfs            1962748        0   1962748   0% /sys/fs/cgroup
tmpfs             392548       12    392536   1% /run/user/120
tmpfs             392548        0    392548   0% /run/user/1000
```

7. SYSTEM MIGRATION TO SD CARD

a) SD Card Configuration:

1. Make sure the system can recognize the SD card, command: **sudo fdisk -lu**

```
Disk /dev/mmcblk1: 59.5 GiB, 63864569856 bytes, 124735488 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

2. Set the SD with **GPT** format:

i. Command: **sudo fdisk /dev/mmcblk1, enter the** SD card configuration.

```
tx2nx@tx2nx:~/Desktop$ sudo fdisk /dev/mmcblk1

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

ii. Command: **g**, new a GPT disk label.

```
Command (m for help): g
Created a new GPT disklabel (GUID: E39DF30E-48FE-B041-A6FA-5EFAEC223CEA).
```

iii. Command: **n**, new partition, enter next.

```
Command (m for help): n
Partition number (1-128, default 1):
First sector (2048-124735454, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-124735454, default 124735454):
```

iv. Command: **w,** save the configuration.

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

v. Command: **sudo mke2fs -t ext4 /dev/mmcblk1p1,** format the partition.

```
nx@nx-desktop:~$ sudo mke2fs -t ext4 /dev/mmcblk1p1
mke2fs 1.45.5 (07-Jan-2020)
Discarding device blocks: done
Creating filesystem with 15591675 4k blocks and 3899392 inodes
Filesystem UUID: fd266d8c-21ae-4dbd-89a6-6e5cd0f6be67
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 26542
08,
        4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (65536 blocks): done
Writing superblocks and filesystem accounting information: done
```

vi. Command：**sudo mount /dev/mmcblk1p1 /mnt**

```
tx2nx@tx2nx:~/Desktop$ sudo mount /dev/mmcblk1p1 /mnt
```

b) NVIDIA Jetson system migration

- ✓ **Taking tx2nx as an example, other devices can replace the device name in the middle of the command, and the device name can refer to the above command**
- ➢ 1. Command： **git clone https://github.com/jetsonhacks/rootOnNVMe** or https://github.com/jetsonhacks/rootOnNVMe download the script
- ➢ 2. Modify **copy-rootfs-ssd.sh**, comment the mount command.

```
#!/bin/bash
# Mount the SSD as /mnt
# sudo mount /dev/nvme0n1p1 /mnt
# Copy over the rootfs from the SD card to the SSD
sudo rsync -axHAWX --numeric-ids --info=progress2 --exclude={"/dev/","/proc/","/sys/","/tmp/"
,"/run/","/mnt/","/media/*","/lost+found"} / /mnt
# We want to keep the SSD mounted for further operations
# So we do not unmount the SSD
```

- ➢ 3. Enter **rootOnNVMe** directory, Command：**./copy-rootfs-ssd.sh**, copy the system file to SD card.

```
nx@nx-desktop:~/rootOnNVMe$ sudo bash ./copy-rootfs-ssd.sh
    276,613,277  37%   50.12MB/s    0:00:05 (xfr#2089, ir-chk=1622/4960)
```

- ➢ 4. Refer to chapter 2 and enter **recovery mode.**
- ➢ 5. Enter **~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**, open the terminal：

  Replace the previous chapter with the corresponding flash command mmcblk0p1to mmcblk1p1 and update the EMMC bootloader.

  - ◆ **TX2NX-EMMC:** sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx **mmcblk1p1**

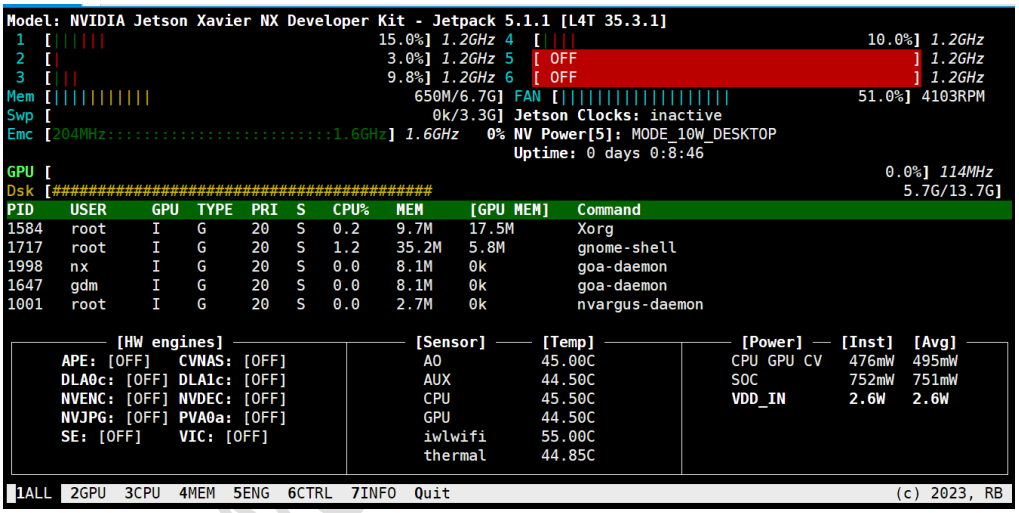
```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_T
X2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx mmcbl
k1p1
[   30.4511 ] Coldbooting the device
[   30.4521 ] tegradevflash_v2 --reboot coldboot
[   30.4531 ] Bootloader version 01.00.0000
[   30.6253 ]
*** The target t186ref has been flashed successfully. ***
Make the target filesystem available to the device and reset the board to boot f
rom external mmcblk1p1.
```

- 6. Reboot TX2NX, terminal command： **df -l**, At this point, the system disk has become an SD card, and the system on the original EMMC has been successfully migrated.

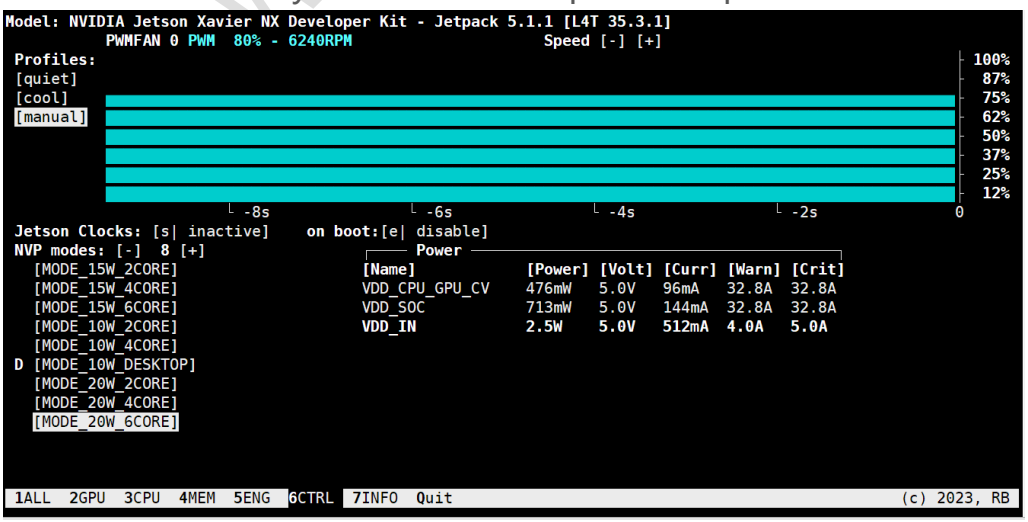
```
tx2nx@tx2nx:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk1p1   59G   12G   44G  21% /
devtmpfs        1.6G     0  1.6G   0% /dev
tmpfs           1.9G   52K  1.9G   1% /dev/shm
tmpfs           1.9G   21M  1.9G   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
tmpfs           384M   12K  384M   1% /run/user/120
tmpfs           384M     0  384M   0% /run/user/1000
```

8. INSTALL JTOP (CONTROL FAN, VIEW SYSTEM INFORMATION)

1. Open the terminal, install using the following command Jtop
   a) sudo apt-get install python3-pip python3-dev -y
   b) sudo -H pip3 install jetson-stats
   c) sudo systemctl restart jtop.service
   d) sudo jtop



2. Click on 6CTRL to manually control the fan speed and power scheme.



9. USING CAN FOR COMMUNICATION

1. TX2-NX/XavierNX/OrinNX/OrinNano integrates one CAN controller CAN0, and WeAct Studio has designed one CAN transceiver (CAN0) on the carrier board, which can be directly mounted on the CAN physical bus for use.

2. TX2-NX/XavierNX/OrinNX/OrinNano comes with a canbus driver and is integrated into the image, supporting canbus without further processing. We need to install the canbus module. (Enter the following command on the terminal or put it into rc.local to enable self start)

   modprobe can
   modprobe can-raw
   modprobe can-bcm
   modprobe can-gw
   modprobe can_dev
   modprobe mttcan

3. Check if the installation was successful through **lsmod**.


```
nvidia@localhost:~$ lsmod
Module                  Size  Used by
fuse                  103841  2
mttcan                 66251  0
can_dev                13306  1 mttcan
can_gw                 10919  0
can_bcm                16471  0
can_raw                10388  0
can                    46600  3 can_raw,can_bcm,can_gw
zram                   26166  6
overlay                48691  0
bcmdhd                934274  0
cfg80211              589351  1 bcmdhd
spidev                 13282  0
nvgpu                1575721  20
bluedroid_pm           13912  0
ip_tables              19441  0
x_tables               28951  1 ip_tables
```

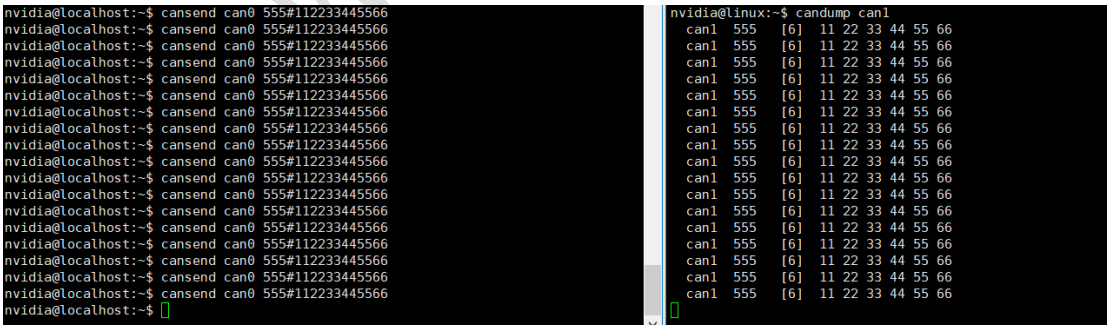
4. Configure canbus properties, similar to the baud rate setting of the serial port.

   sudo ip link set can0 type can bitrate 500000
   sudo ip link set up can0

5. Check if the configuration was successful through ifconfig.

```
nvidia@localhost:~$ ifconfig
can0: flags=193<UP,RUNNING,NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 131
```

6. Through cansend can0 (can1) on a terminal ××× Command to send data, and another terminal completes the actual signal transmission and reception test through candump can1 (can0)



10. USING GPIO IN SHELL

1. Jetson Nano/TX2NX/Xavier NX/OrinNo/OrinNX can directly control GPIO input and output through shell commands.

	Nano	TX2NX	XavierNX	OrinNano&OrinNX
GPIO1	PS.05[149]	PN.01[425]	PQ.05[440]	PQ.05[453]
GPIO2	PBB.00[216]	PJ.04[396]	PS.04[453]	PAC.06[492]
GPIO3	PY.02[194]	PC.02[338]	PCC.04[321]	PN.01[433]
GPIO4	PE.06[38]	PU.05[269]	PN.01[419]	PH.00[391]

Take an example with XavierNX GPIO1, PQ.05 is GPIO ID，440 is GPIO Number.

2. GPIO Usage:
   a) Activate IO Command：sudo echo <GPIO_Number> > /sys/class/gpio/export
   b) GPIO Output：

      For Jetpack4.x version:

      i. echo out > /sys/class/gpio/gpio<GPIO Number>/direction
      ii. echo 1 > /sys/class/gpio/gpio<GPIO_Number>/value

      For Jetpack5.x version and above:

      i. echo out > /sys/class/gpio/<GPIO ID>/direction
      ii. echo 1 > /sys/class/gpio/<GPIO ID>/ value

   c) GPIO Input

      For Jetpack4.x version:

      i. echo in > /sys/class/gpio/gpio<GPIO Number>/direction

ii. cat /sys/class/gpio/gpio<GPIO_Number>/value

For Jetpack5.x version and above:

iii. echo in > /sys/class/gpio/<GPIO ID>/direction

iv. cat /sys/class/gpio/<GPIO ID>/ value

3. GPIO Example，take an example with XavierNX GPIO2 [Jetpack5.1.3] and output high level
   a) Activate IO GPIO: sudo echo 453 > /sys/class/gpio/export
   b) Set IO direction: echo out > /sys/class/gpio/PS.04/direction
   c) Output the high level: echo 1 > /sys/class/gpio/PS.04/value

   Note: If there is no permission displayed, you can first use chmod 777 to set per missions for the corresponding file.

4. GPIO Example，take an example with TX2NX GPIO3 [Jetpack4.6.4] and read the voltage level.
   a) Activate GPIO: sudo echo 338 > /sys/class/gpio/export
   b) Set IO direction: echo out > /sys/class/gpio/gpio338/direction
   c) Read the voltage level: cat /sys/class/gpio/ gpio338/value

CONTACT US

1. Github：https://github.com/WeActTC
2. Aliexpress：https://pt.aliexpress.com/item/1005006558131261.html?spm=a2g0o.detail.1000023.1.1572Oc5yOc5yOi