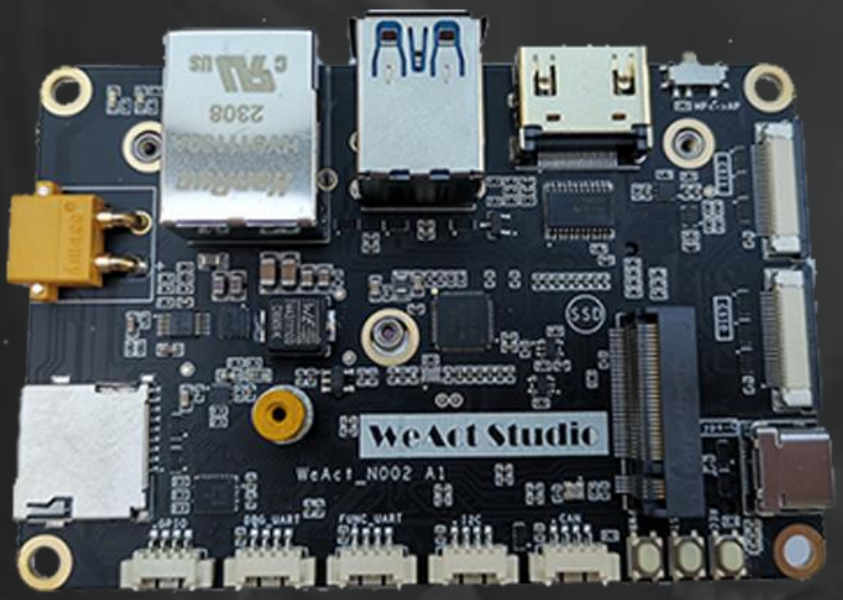


WeAct Studio

WEACT-N002

载板/底板

使用教程

WeAct Studio

目录

REVISION HISTORY

Draft Date	Revision	Description
2023.07.16	V1.0	1. 初始版本
2024.02.16	V2.0	1. 修订版本 2. 增加 GPIO 教程

1. 搭建烧写环境

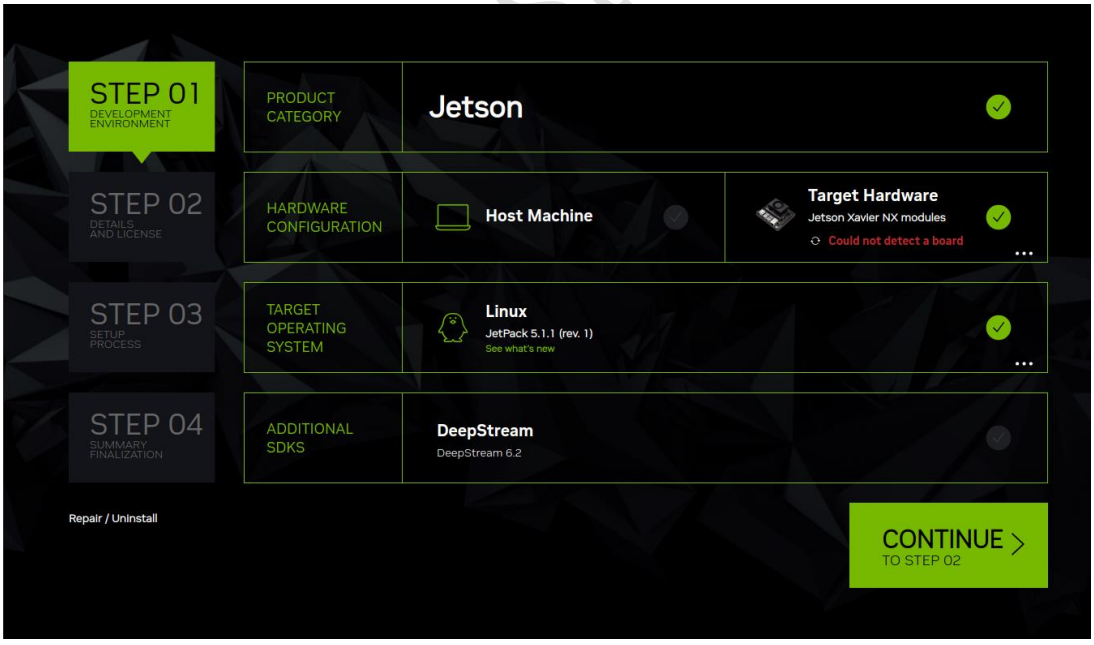
a) 需要一台装有 **Ubuntu18.04** 以上的电脑作为 HOST 端给 Nano/NX 烧写，或者可以在 Windows 上安装 VMware 来安装 Ubuntu

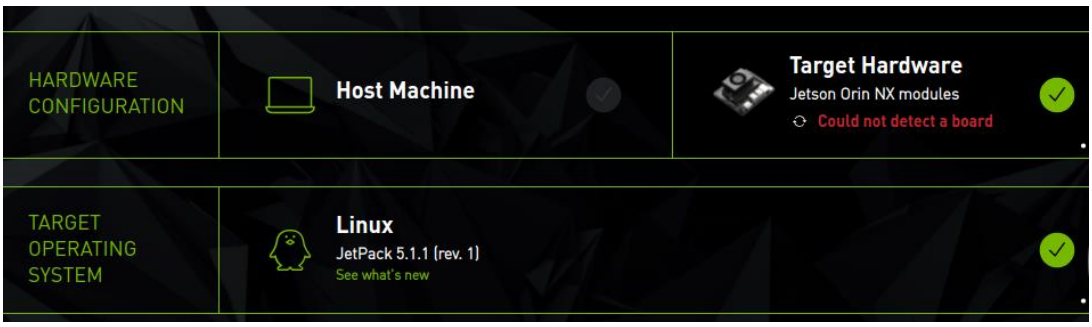
- VMware 上如何安装 Ubuntu18.04：https://blog.csdn.net/u012556114/article/details/82751089

b) 在 NVIDIA 下载最新的 **SDK-Manager** 并在 Ubuntu18.04 中安装（需要注册 NVIDIA 账号）

- SDK-Manager 下载地址：https://developer.nvidia.com/nvidia-sdk-manager

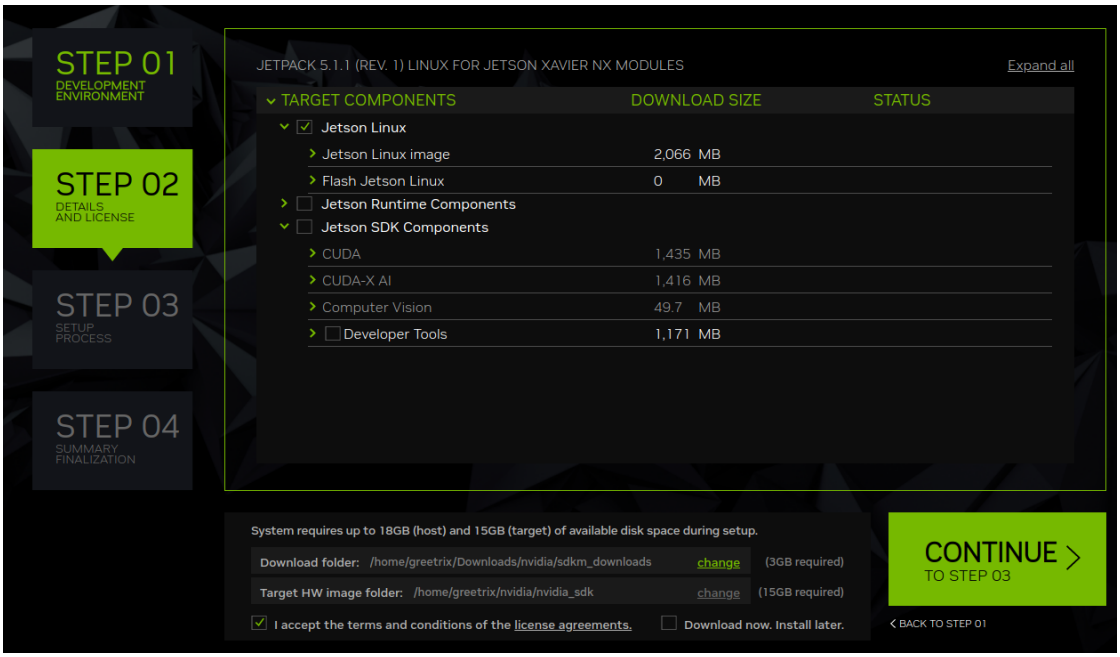
c) 选择需要 **Target Hardware** 以及 **JetPack** 版本，**不勾选 HostMachine（节省主机存储空间）**，DeepStream 根据组件需求勾选，这里以 **XavierNX（Orin 系列请看第二张图）为例**选择，点击 **Continue**



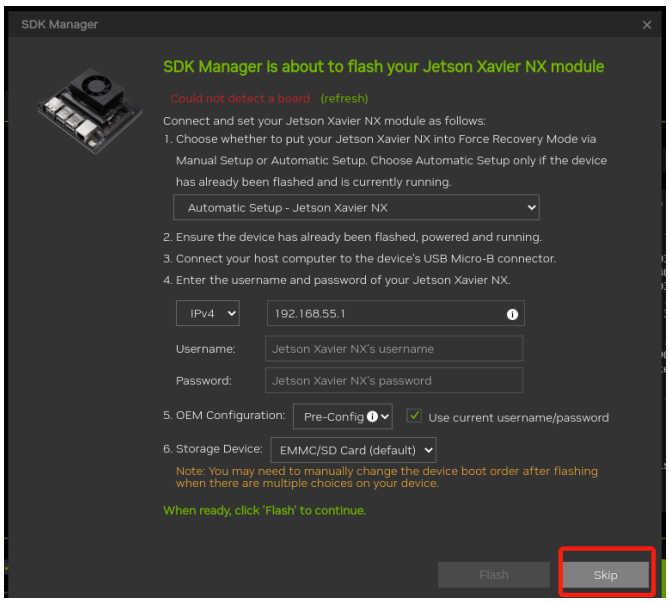


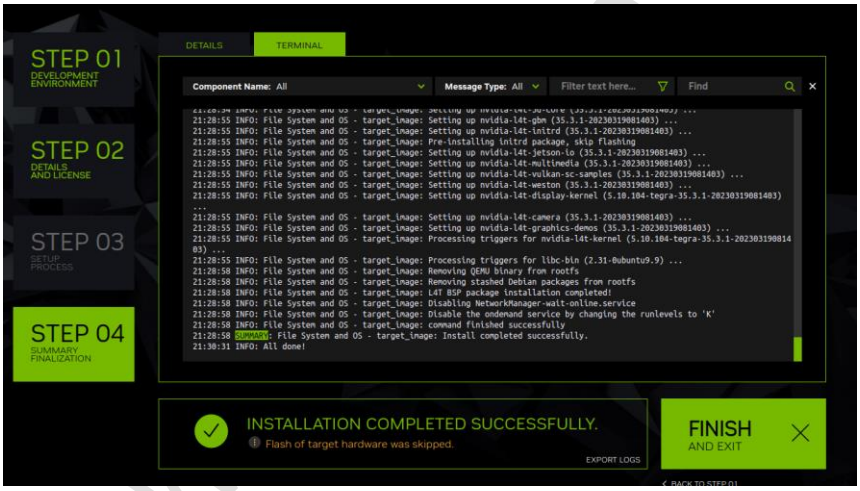
Orin Nano/NX 请选择 **Orin NX** 的 Jetpack 系统（通用）

d) **勾选 I accept the terms and conditions of the license agreements，取消勾选 Jetson Runtime 及 SDK Components（后面安装组件会重新下载并一起安装）**，点击 **CONTINUE** 进行下一步。

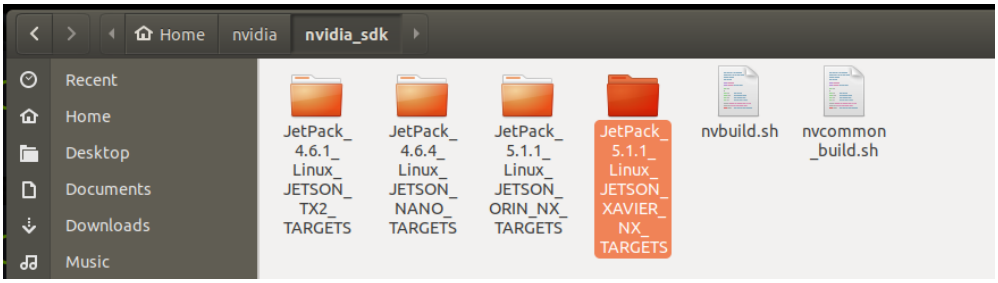


Note：请在畅通的网络环境下进行下载以及安装，下载或安装失败时，可点击 **Retry** 继续，直至**全部状态为 Installed 并且显示绿色**，安装过程中会弹出联网烧写的信息，选择 **Skip。（后续步骤通过命令来烧写**，可以兼容不同核心板）





e) 安装成功后，会在**~/nvidia/nvidia_sdk/** 下有相应版本烧写所需的文件

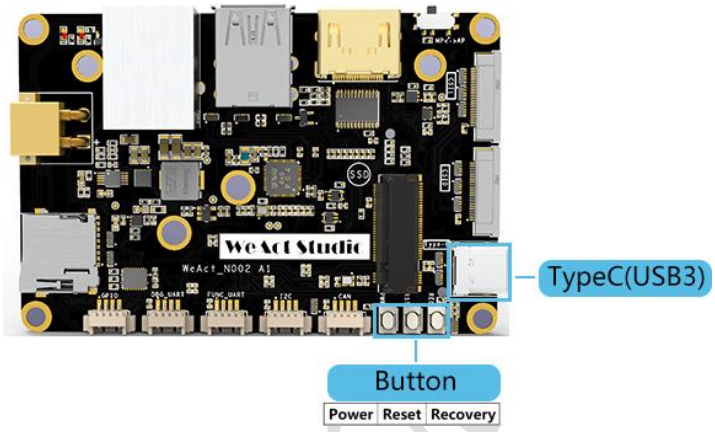


f) 在终端通过 **sudo apt-get install python** 安装 python 支持以便后续烧写环境。

2. 核心板烧写系统（以 XAVIERNX 为例）

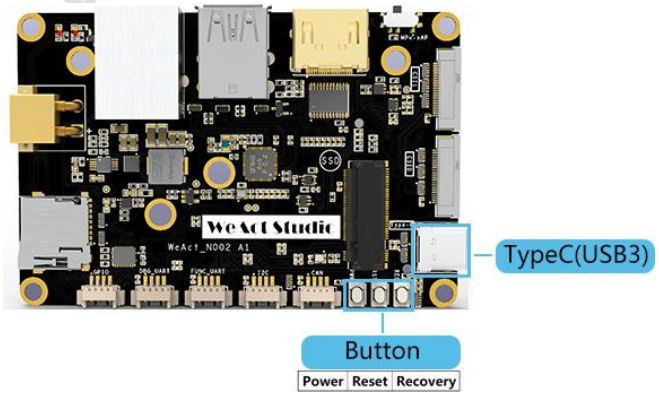
Note: 该章节用于核心板无系统，或者需要重新刷写系统，如果核心板有系统，仅更新设备树，请跳过本章节

1. 使用 **USB Type-C** 线材连接载板上的 **USB Type-C** 接口



2. 将开机键**拨至 MP（手动开机）**，两种方式进入 Recovery 模式
   a) 按住 **REC** 键，再按一下 **PWR** 键开机，松开 **REC** 键进入 Recovery 模式
   b) 按一下 **PWR** 开机，按住 **REC** 键，按一下 **RST** 键进入 Recovery 模式

   此时 VMWare 右下角会出现 **NVIDIA 的 USB 驱动标志**，或者打开终端，输入 **lsusb** 命令，会发现 **Nvidia Corp**，**同时风扇转速会到达最大。**




```
_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
```

3. Orin 系列核心板需要修改该项，并且需要插入 SSD

a) 修改文件 Linux_for_Tegra/bootloader/t186ref/BCT/tegra234-mb2-bct-misc-p3767-0000.dts

b) 删除行：cvb_eeprom_read_size = <0x100>
增加行：cvb_eeprom_read_size = <0x0>

4. 进入**~/nvidia/nvidia_sdk/JetPack_5.1.1_Linux_JETSON_XAVIER_NX_TARGETS/Linux_for_Tegra**

a) **复制资料包中 Flash_CMD/03_Jetson XavierNX/run_flash.sh 到该目录**

b) **打开终端，运行该脚本 sudo bash run_flash.sh**

等待烧录成功即可使用，其他设备脚本请参考下面表格

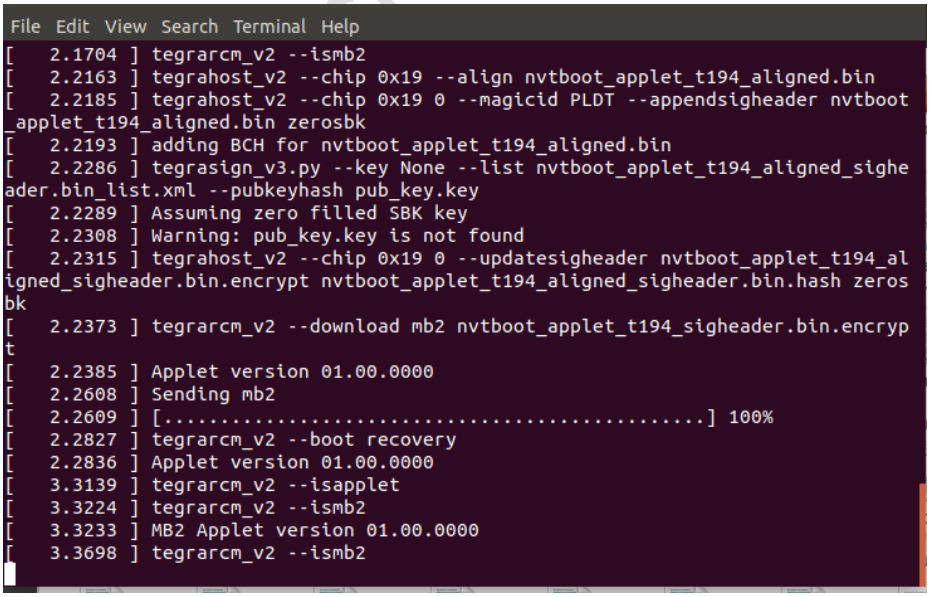
各设备刷机命令

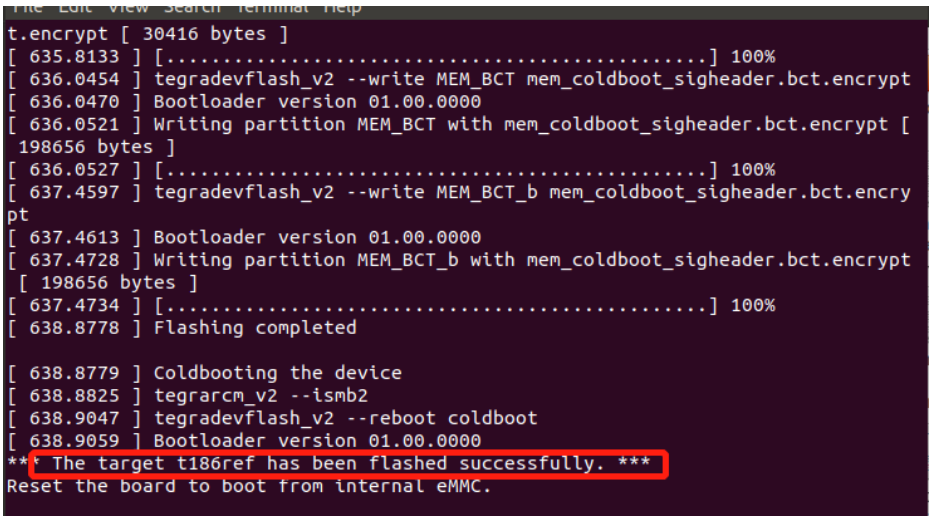
Note: -SD 用于核心板上带 SD 卡槽，-EMMC 用于核心板上不带 SD 卡槽

设备	刷机命令
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	两步: 1. 编译[不插 USB 线] Flash_CMD/04_Jetson OrinNano/01_4GB/run_compile.sh 2. 烧录[插 USB 线] Flash_CMD/04_Jetson OrinNano/01_4GB/run_flash.sh
Orin Nano 8GB	两步: 1. 编译[不插 USB 线]

	Flash_CMD/04_Jetson OrinNano/02_8GB/run_compile.sh 2. 烧录[插 USB 线] Flash_CMD/04_Jetson OrinNano/02_8GB/run_flash.sh
Orin NX 8GB	两步: 1. 编译[不插 USB 线] Flash_CMD/05_Jetson OrinNX/01_8GB/run_compile.sh 2. 烧录[插 USB 线] Flash_CMD/05_Jetson OrinNX/01_8GB/run_flash.sh
Orin NX 16GB	两步: 1. 编译[不插 USB 线] Flash_CMD/05_Jetson OrinNX/02_16GB/run_compile.sh 2. 烧录[插 USB 线] Flash_CMD/05_Jetson OrinNX/02_16GB/run_flash.sh

烧写系统成功后会有 Successfully!显示，如下图所示。





3. 核心板更新设备树（以 XAVIERNX 为例）

Note: 如果你不需要使用以下功能，不用更新设备树，请跳过此章节

各核心板 WeAct 设备树支持

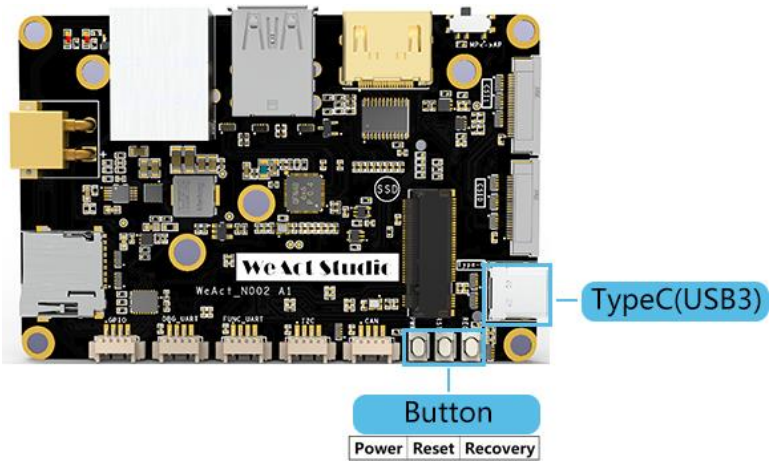
	SD Card	Function UART	Function UART+SD Card
Nano-SD	无	Jetpack4.x [更新中]	无
Nano-EMMC	Jetpack4.x	Jetpack4.x [更新中]	Jetpack4.x [更新中]
TX2NX	Jetpack4.x	Jetpack4.x [更新中]	Jetpack4.x [更新中]
XavierNX	Jetpack5.x	Jetpack5.x [更新中]	Jetpack5.x [更新中]
Orin Nano/NX	无	Jetpack5.x Jetpack6.x [更新中]	无

Note: SD Card -> 可使用 SD Card 卡槽

Function UART -> Debug 串口转为正常串口

SD Card + Function UART -> Debug 串口转为正常串口+可使用 SD Card 卡槽

1. 使用 **USB Type-C** 线连接载板上的 **USB Type-C** 接口

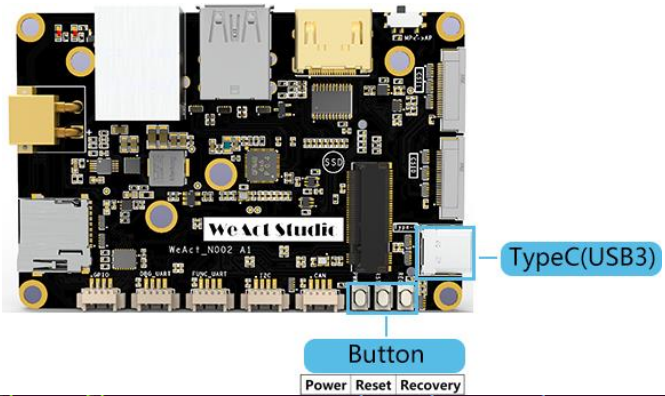


2. 将开机键**拨至 MP（手动开机）**，两种方式进入 Recovery 模式

   a) 按住 **REC** 键，再按一下 **PWR** 键开机，松开 **REC** 键进入 Recovery 模式

   b) 按一下 **PWR** 开机，按住 **REC** 键，按一下 **RST** 键进入 Recovery 模式

   此时 VMWare 右下角会出现 **NVIDIA 的 USB 驱动标志**，或者打开终端，输入 **lsusb** 命令，会发现 **Nvidia Corp**，**同时风扇转速会到达最大。**

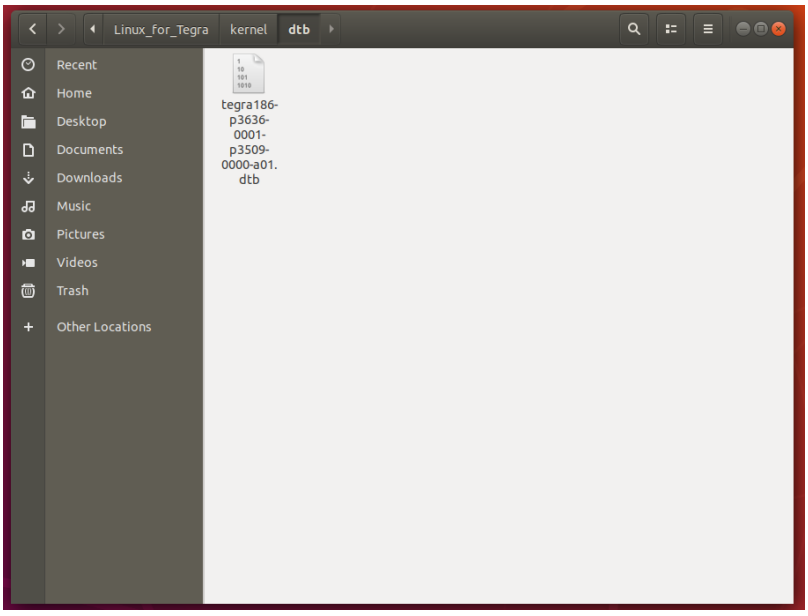



```
_XAVIER_NX_TARGETS/Linux_for_Tegra$ lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 004: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 003 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 003 Device 006: ID 0955:7e19 NVidia Corp.
Bus 003 Device 002: ID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

3. **Jetpack 4.x 版本更新设备树（Jetson Nano/TX2NX），以 TX2NX 为例**

   a) 找到相应版本的设备树

   b) 进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/kernel/dtb**，复制提供的设备树 **tegra186-p3636-0001-p3509-0000-a01.dtb** 至该目录



c) 进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：

d) 命令 **sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**，其他核心板请参考下图表格：

各设备更新设备树命令

设备	设备树更新命令
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

4. Jetpack 5.x 版本更新设备树（Jetson XavierNX)

e) 按一下 PWR 正常开机，并且进入系统

f) 将 git 对应的设备树拷入设备系统桌面（kernel_tegra194-p3668-0001-p3509-0000.dtb）

g) 在桌面右键，打开终端

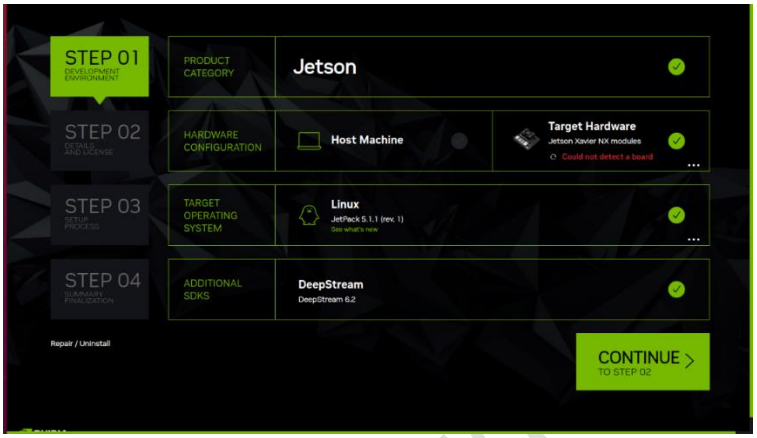
h) 输入命令：sudo rm /boot/dtb/kernel_tegra194-p3668-0001-p3509-0000.dtb

i) 输入命令：sudo cp kernel_tegra194-p3668-0001-p3509-0000.dtb /boot/dtb

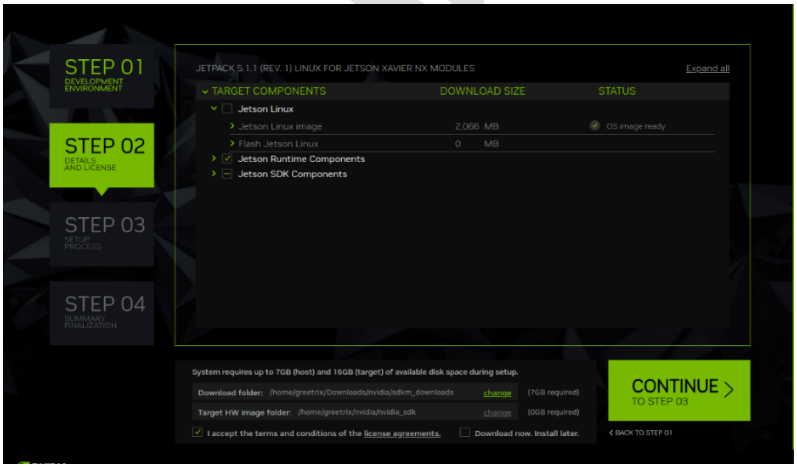
j) 重启系统即可

4. 安装 NVIDIA 组件

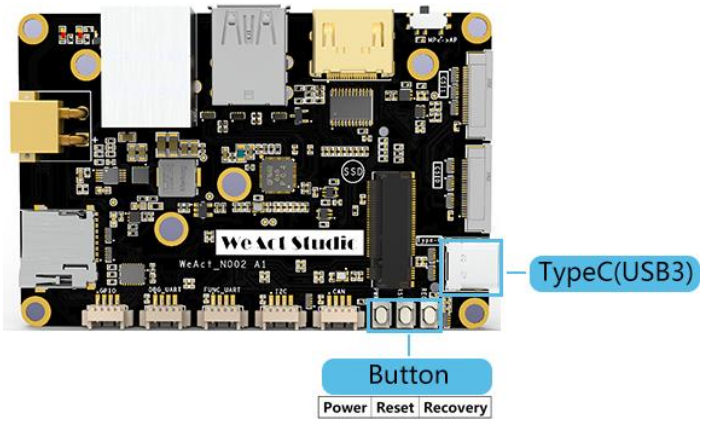
1. 选择需要 **Target Hardware** 以及 **JetPack** 版本，**不勾选 HostMachine**，这里以 **XavierNX 为例**选择，点击 **Continue**



2. 勾选所需要 **SDK 组件**，勾选 **I accept the terms and conditions of the license agreements**，点击 **CONTINUE** 进行下一步。

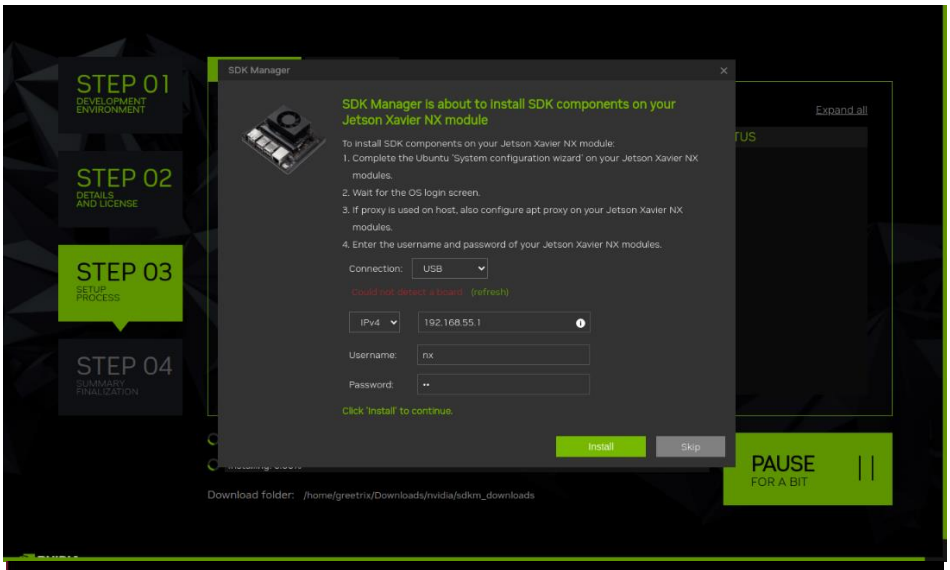


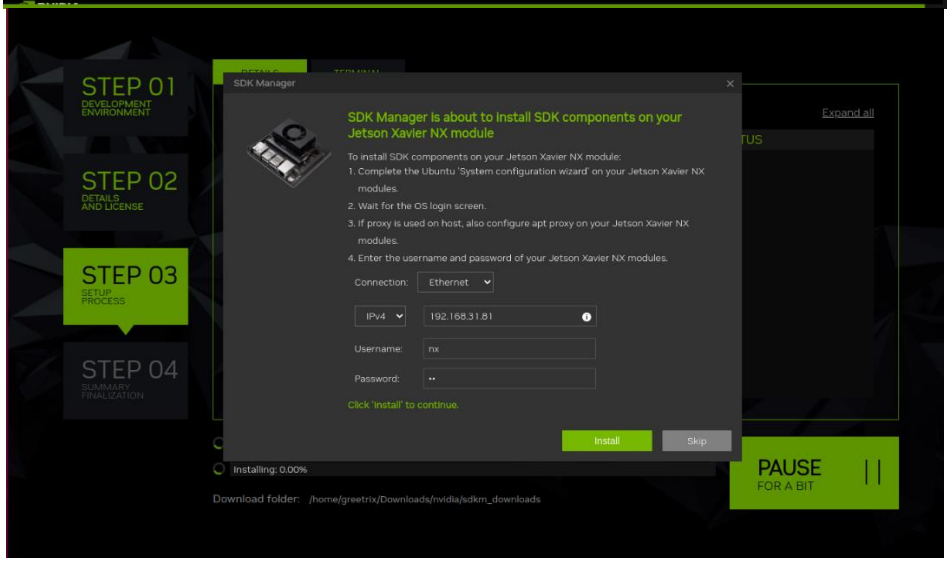
3. 使用 **USB Type-C** 线连接载板上的 **USB Type-C** 接口 。



4. 将开机键**拨至 MP（手动开机）**，按一下 **PWR** 键，正常开机并且进入系统，此时 VMWare 右下角会出现 **NVIDIA 的 USB 驱动标志**，或者打开终端，输入 **lsusb** 命令，会发现 **Nvidia Corp**。

5. 输入 **XavierNX** 账号密码，**XavierNX** 端请**保持联网状态，也可以用同局域网下的 IP（图 2，主机和 XavierNX 连接同一个路由器）**





6. 等待安装完成即可。

5. 环境备份及镜像恢复

1. 系统安装在 EMMC，以 TX2NX 为例

a) 参考**第 2 章**，无论备份还是镜像烧写，进入 Recovery 模式，注意镜像较大，请保证 Ubuntu 有充足的空间（>40G）。

b) **备份**：这里以 TX2NX 为例（其他设备参考上章内容修改 jetson 名称），对核心板现有环境进行备份。进入~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra，打开终端：

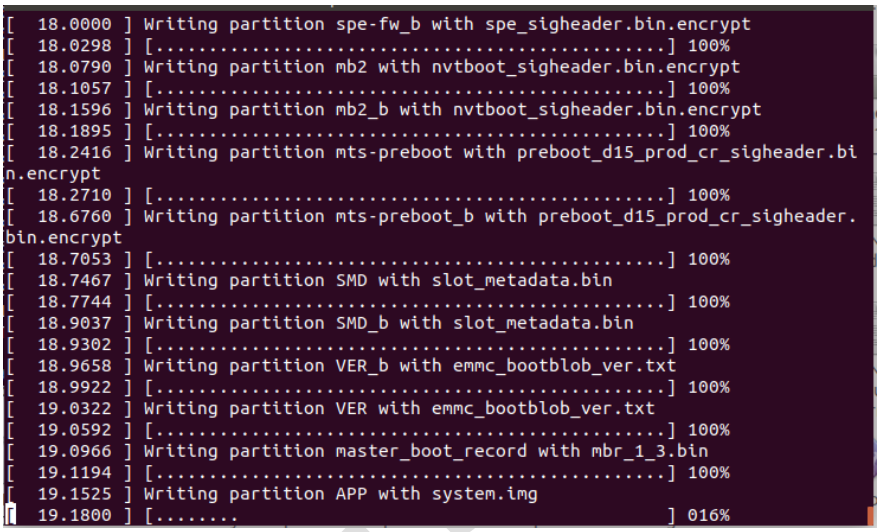
使用镜像备份命令：**sudo ./flash.sh -r -k APP -G backup.img jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**，等待备份完成即可，此时目录下会有 backup.img 的镜像（建议复制一份至其他位置备份），此时**备份已经成功。**

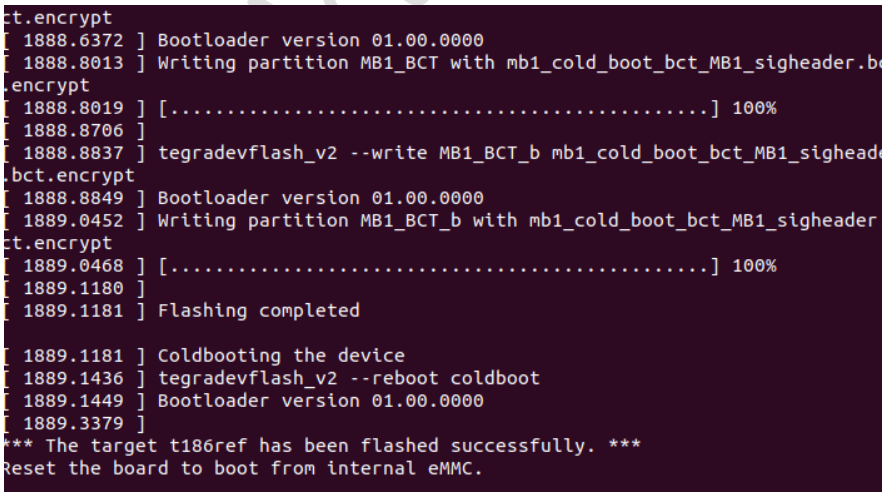
```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_T
X2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh -r -k APP -G backup.img jetson-xavie
r-nx-devkit-tx2-nx mmcblk0p1
```

```
[    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) **镜像烧写**：进入~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra，将备份好的 backup.img 拷入 Liunx_for_Tegra/bootloader/下，并重命名为 system.img，回到 Linux_for_Tegra 目录下，打开终端：

使用已有镜像烧写命令：**sudo ./flash.sh -r jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**，等待烧写完成即可。





2. 系统安装在 SD 卡或者 SSD 的设备，直接通过主机拷贝 SD 卡或者 SSD 到其他存储介质即可

6. 系统迁移至 NVME 固态硬盘

Note: 该迁移仅用于 Nano/TX2NX/XavierNX，Orin 系列插上 SSD，直接通过第二章命令烧录即可

a) WeAct-N002 载板支持 PCIE3.0 X 4 插槽，支持 2242 NVME SSD M.2 接口固态硬盘，

b) NVME 固态硬盘配置：

➢ 1. 配置前确保系统能识别到 NVME 固态硬盘，终端命令：**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. 将 NVME 设置成 GPT 格式：

i. 终端命令：**sudo parted /dev/nvme0n1** 进入 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. 终端命令：**mklabel gpt** 将磁盘 label 设置为 gpt 格式

```
(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. 终端命令：**mkpart logical 0 -1** 将磁盘 part 设置为 gpt 格式

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

iv. 终端命令：**print** 查看分区结果

```
(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. 终端命令：**quit** 退出

vi. 终端命令：**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（m for help): 输入 **N**，选择增加新分区，后面回车默认即可

```
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（m for help)：输入 **P**，查看分区结果

```
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. 终端命令：**quit** 退出

x. 终端命令：**sudo mke2fs -t ext4 /dev/nvme0n1p1**，格式化分区

```
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. 终端命令：**sudo mount /dev/nvme0n1p1 /mnt**，成功 mount 则 NVME 配置成功

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

c) **NVIDIA Jetson 系统迁移 (!!!迁移前建议参考第 3 章进行系统备份)：**

✓ **下面以 TX2NX 为例，其他设备替换命令中间的设备名称即可，设备名称可参考上面命令**

➢ 1. 终端命令： **git clone https://github.com/jetsonhacks/rootOnNVMe** 或者 https://github.com/jetsonhacks/rootOnNVMe 下载脚本

- 2. 进入 **rootOnNVMe** 文件夹，终端命令：**./copy-rootfs-ssd.sh**，复制系统文件至 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. 终端命令：**./setup-service.sh** 配置启动项

```
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. 参考第 2 章，进入 **Recovery 模式。**
- 5. （**烧录环境的 Ubuntu**，参考**前面章节**）进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：**sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx nvme0n1p1** 更新 EMMC 内部引导

```
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. 重启 TX2NX，终端命令：**df -l**， 此时系统盘已经变为 NVME SSD，并且原有 EMMC 上系统已经成功迁移。

```
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. 系统迁移至 SD 卡

a) SD 卡配置:

1. 配置前确保系统能识别到 SD 卡，终端命令：**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. 将 SD 卡设置成 GPT 格式:

i. 终端命令：**sudo fdisk /dev/mmcblk1** ，进入 sd 卡配置

```
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. 终端命令：**g**，新建 gpt 分区表

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

iii. 终端命令：**n**，新建分区，其他输入回车按默认值

```
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. 终端命令：**w**， 保存分区信息

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

v. 终端命令：**sudo mke2fs -t ext4 /dev/mmcblk1p1，格式化分区**

```
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. 终端命令：**sudo mount /dev/mmcblk1p1 /mnt**，成功 mount 则 SD 卡配置成功

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

b) NVIDIA Jetson 系统迁移 (!!!迁移前建议参考第 3 章进行系统备份)：

- ✓ **下面以 TX2NX 为例，其他设备替换命令中间的设备名称即可，设备名称可参考上面命令**
- ➢ 1. 终端命令： **git clone https://github.com/jetsonhacks/rootOnNVMe** 或者 https://github.com/jetsonhacks/rootOnNVMe 下载脚本
- ➢ 2. 修改 **copy-rootfs-ssd.sh** 文件， 注释掉 mount 命令

```
#!/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. 进入 **rootOnNVMe** 文件夹，终端命令：**./copy-rootfs-ssd.sh**，复制系统文件至 SD 卡

```
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. 参考第 2 章，进入 **Recovery 模式。**
- ➢ 5. （**烧录环境的 Ubuntu**，参考**前面章节**）进入 **~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：

  将前面章节对应的刷机命令 mmcblk0p1 换成 mmcblk1p1 更新 EMMC 内部引导

  - ◆ **如 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
```

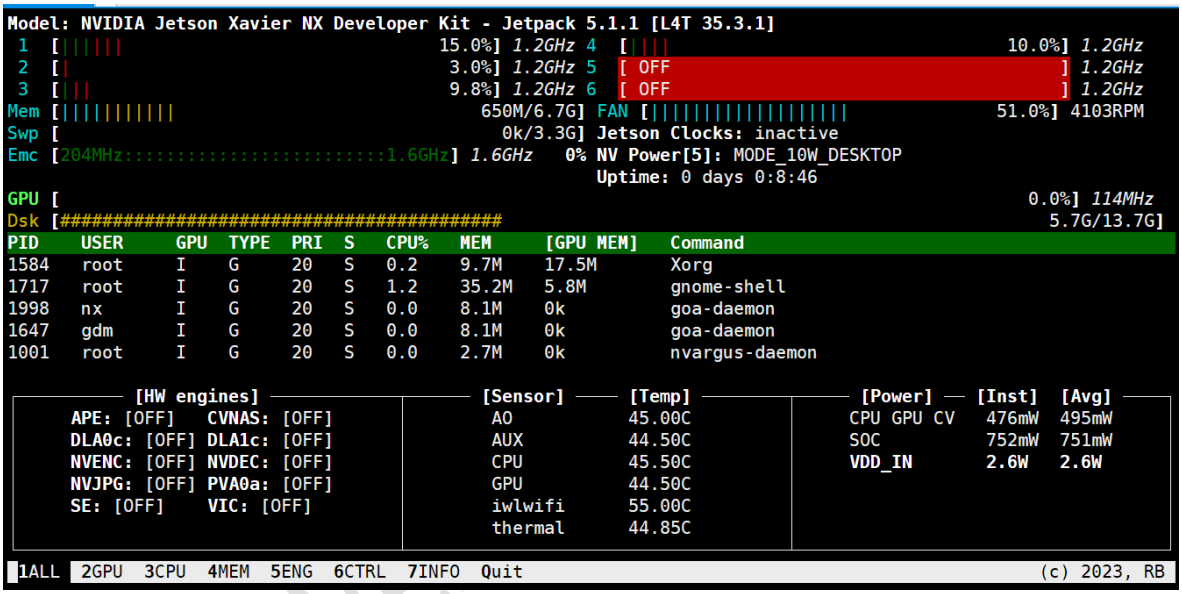
```
[   30.4511 ] Coldbooting the device
[   30.4521 ] tegradevflash_v2 --reboot coldboot
[   30.4531 ] Bootloader version 01.00.0000
[   30.6253 ]
*** The target t186ref has been flashed successfully. ***
Make the target filesystem available to the device and reset the board to boot f
rom external mmcblk1p1.
```

- 6. 重启 TX2NX，终端命令：**df -l**， 此时系统盘已经变为 SD 卡，并且原有 EMMC 上系统已经成功迁移。

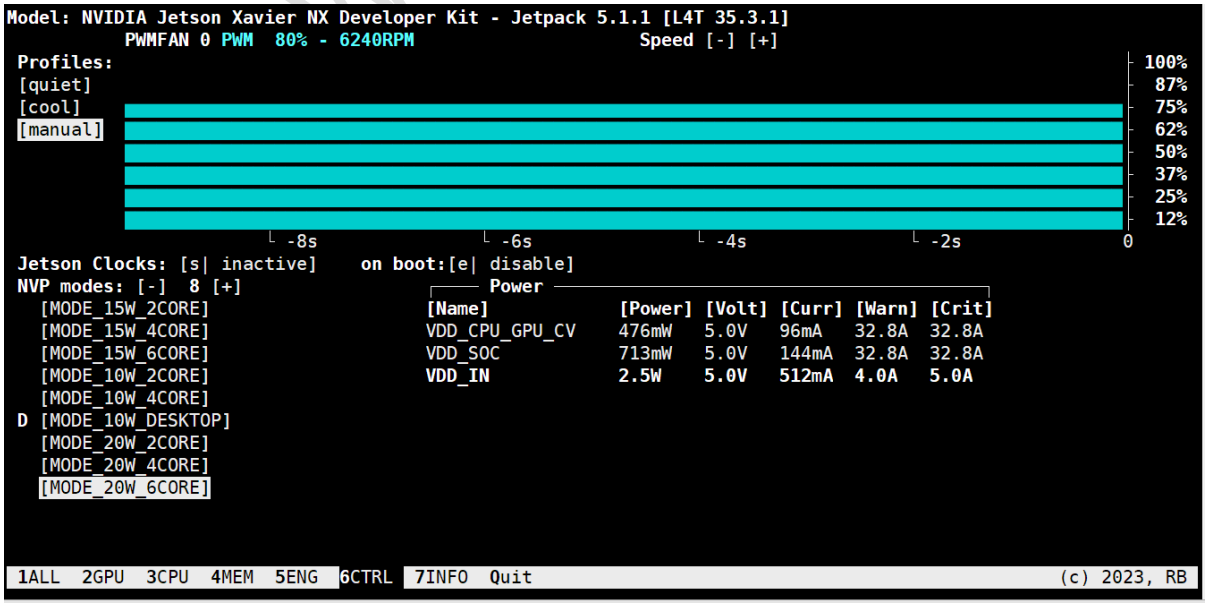
```
tx2nx@tx2nx:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk1p1   59G   12G   44G  21% /
devtmpfs        1.6G     0  1.6G   0% /dev
tmpfs           1.9G   52K  1.9G   1% /dev/shm
tmpfs           1.9G   21M  1.9G   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           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. 安装 JTOP（控制风扇，查看系统信息）

1. 打开终端，通过下面命令安装 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. 点击 6CTRL，即可手动控制风扇转速及电源方案



9. 使用 CAN 进行通信

1. Tx2-NX/XavierNX/OrinNano/OrinNX 上集成了 1 个 CAN 控制器 CAN0，另外 WeAct Studio 的载板上设计了 1 个 CAN 收发器（CAN0），可直接挂载 CAN 物理总线使用。

2. Tx2-NX/XavierNOrinNX/OrinNanoX 自带 canbus 的驱动并集成到了镜像中，已经支持 canbus 无需多做处理。我们需要安装 canbus 模块。（在终端输入下面命令或者放入 rc.local 里面开启自启）

   ```
   modprobe can        // 插入 can 总线子系统
   modprobe can-raw    //插入 can 协议模块
   modprobe can-bcm
   modprobe can-gw
   modprobe can_dev
   modprobe mttcan     //真正的 can 口支持
   ```

3. 通过 **lsmod** 检查是否安装成功。

```
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. 配置 canbus 属性，和串口的波特率设置类似。

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

5. 通过 ifconfig 查看是否配置成功。

```
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. 在一个终端通过 cansend can0(can1) ×××命令来发送数据, 另一个终端通过 candump can1(can0)完成实际信号收发测试

```
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$
```

```
nvidia@linux:~$ candump can1
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
```

10. GPIO 在 SHELL 中使用

1. Jetson Nano/TX2NX/XavierNX/OrinNano/OrinNX 可直接通过 shell 命令控制 GPIO 输入输出

	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]

以 XavierNX GPIO1 为例，PQ.05 是 GPIO ID，440 是 GPIO Number

2. GPIO 使用方法
   a) 激活 IO，命令：sudo echo <GPIO_Number> > /sys/class/gpio/export
   b) GPIO 输出：

      对于 Jetpack4.x 版本：

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

      对于 Jetpack5.x 以上版本：

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

   c) GPIO 输入

      对于 Jetpack4.x 版本：

      i. echo in > /sys/class/gpio/gpio<GPIO Number>/direction
      ii. cat /sys/class/gpio/gpio<GPIO_Number>/value

对于 Jetpack5.x 以上版本：

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

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

3. GPIO 使用例子，以 XavierNX GPIO2 [Jetpack5.1.3] 输出高电平
   a) 激活 GPIO，sudo echo 453 > /sys/class/gpio/export
   b) 设置 IO 方向，echo out > /sys/class/gpio/PS.04/direction
   c) 输出高电平，echo 1 > /sys/class/gpio/PS.04/value

   Note: 如果显示没权限，可以先使用 chmod 777 给对应的文件设置权限

4. GPIO 使用例子，以 TX2NX GPIO3 [Jetpack4.6.4] 读取电平
   a) 激活 GPIO，sudo echo 338 > /sys/class/gpio/export
   b) 设置 IO 方向，echo out > /sys/class/gpio/gpio338/direction
   c) 读取电平，cat /sys/class/gpio/ gpio338/value

联系我们

1. Github：https://github.com/WeActTC
2. 码云：https://gitee.com/WeAct-TC
3. 网站：https://www.weact-tc.cn/
4. 淘宝：https://weactstudio.taobao.com/

WeAct Studio
官方淘宝店