**安裝JetPack:**

Tx2使用arm cpu，CUDA以及CUDNN官方不支援arm cpu，因此需要透過JetPack將套件安裝到裝置上。

事前準備:

* 準備一台與tx2作業系統相同的電腦(ubuntu 16.04)作為host端對target端(tx2)進行刷機。

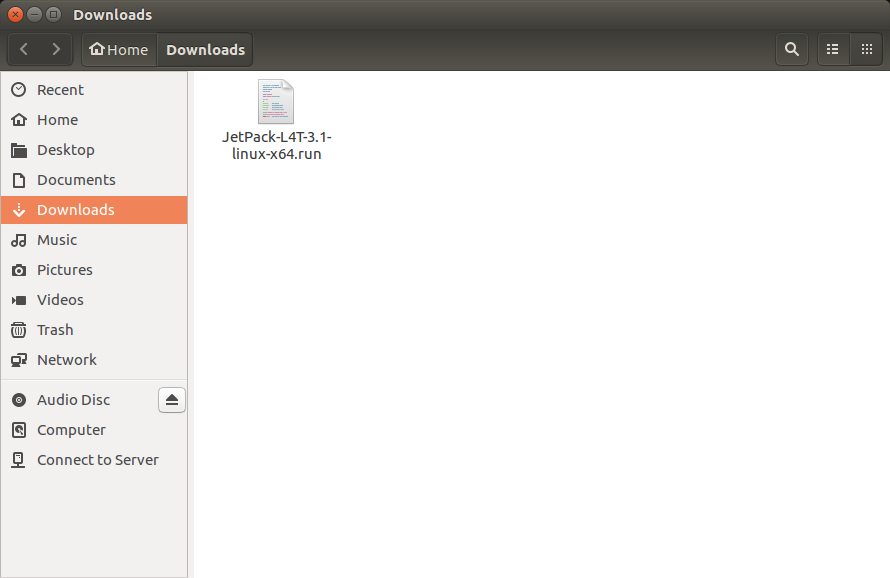
＊如果沒有ubuntu系統的電腦，也可以使用vmware安裝虛擬系統用於刷機(目前知道vmware刷機沒問題，但oracle virtualbox會出錯！！！)

* 在host端下載官方jetpack(目前版本為3.3)，需要申請nvidia帳號才能下載

安裝流程：

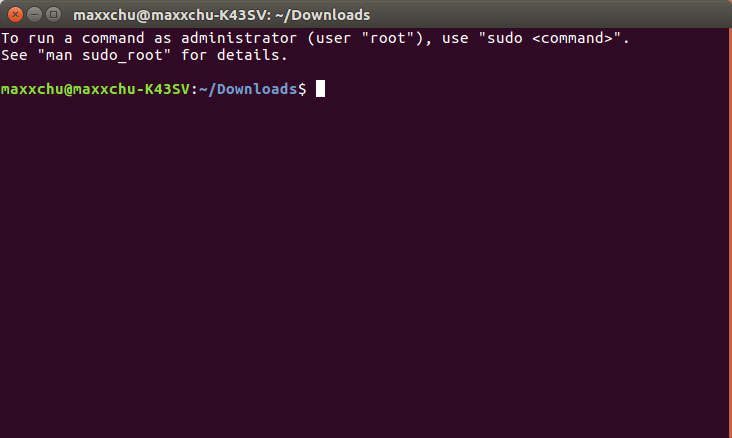
(以下內容節錄至 “<http://www.honghutech.com/nvidia-jeston-tx2/flashtx2>”)

Ubuntu預設的瀏覽器是Firefox打開瀏覽器進入＂<https://developer.nvidia.com/embedded/jetpack>＂網址 下載回來的檔案應該會在Home/hostname/Downloads下

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/jetpack01%20(2).png?attredirects=0)

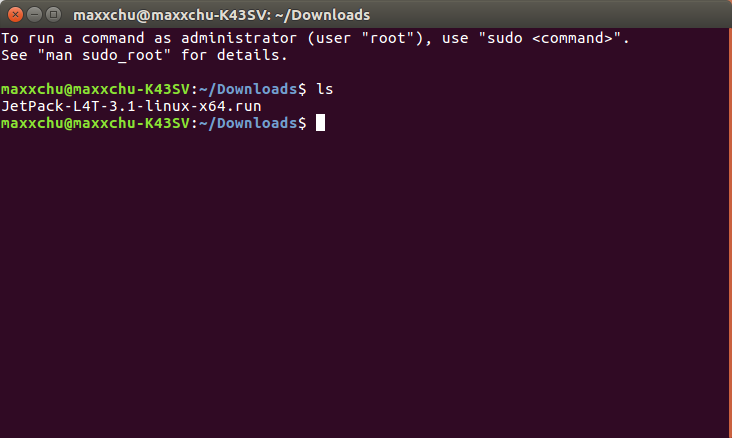
在Downloads目錄下有下載了3.1版的run檔案

按下滑鼠右鍵跳出功能表選擇"Open in Terminal"



下載回來的JetPack run file是沒有執行權限的

可以鍵入ls按下Enter查看 如果可執行 檔案會是綠色的

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/jetpack03%20(2).png?attredirects=0)

Add exec permission for the JetPack-${VERSION}.run

鍵入下列文字後按enter ${VERSION}依照你下載的版本號修改

改完再以ls察看就變成綠色了

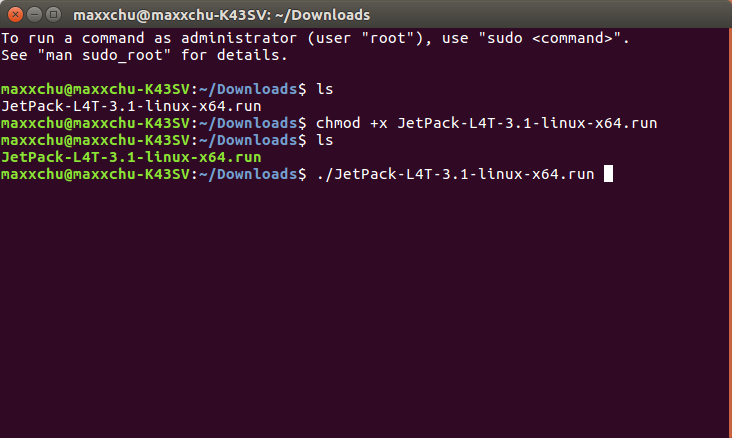
chmod +x JetPack-${VERSION}.run

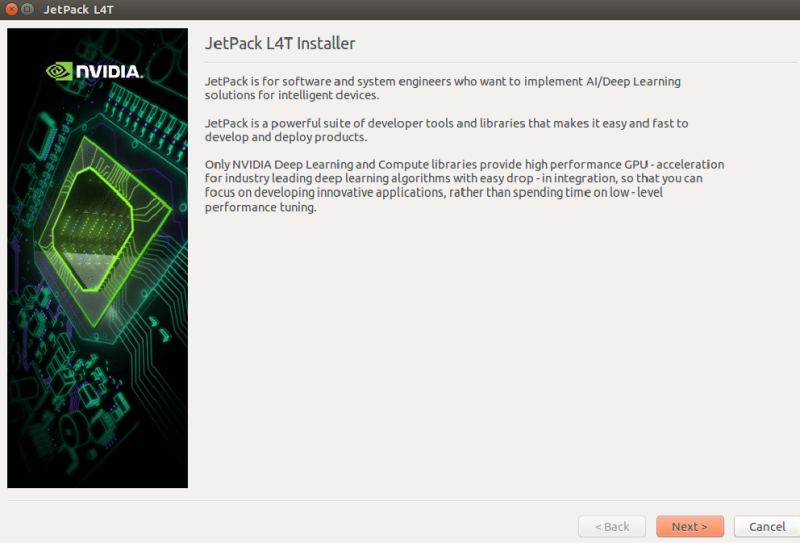
接下來就可以執行這個run檔了

就是輸入./連著檔名就可執行

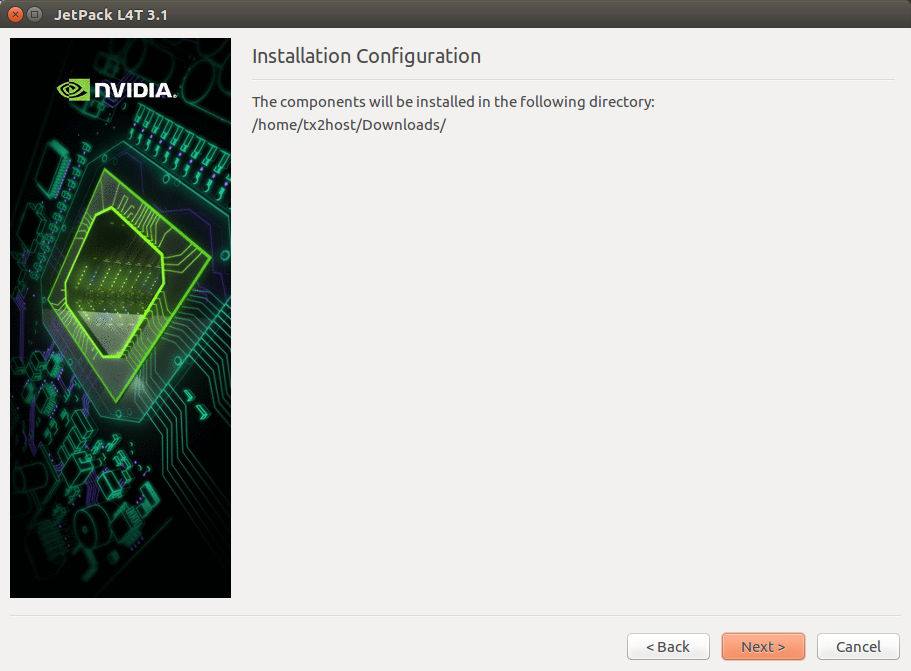
執行這個run file就會跳出下面這個畫面

./JetPack-${VERSION}.run

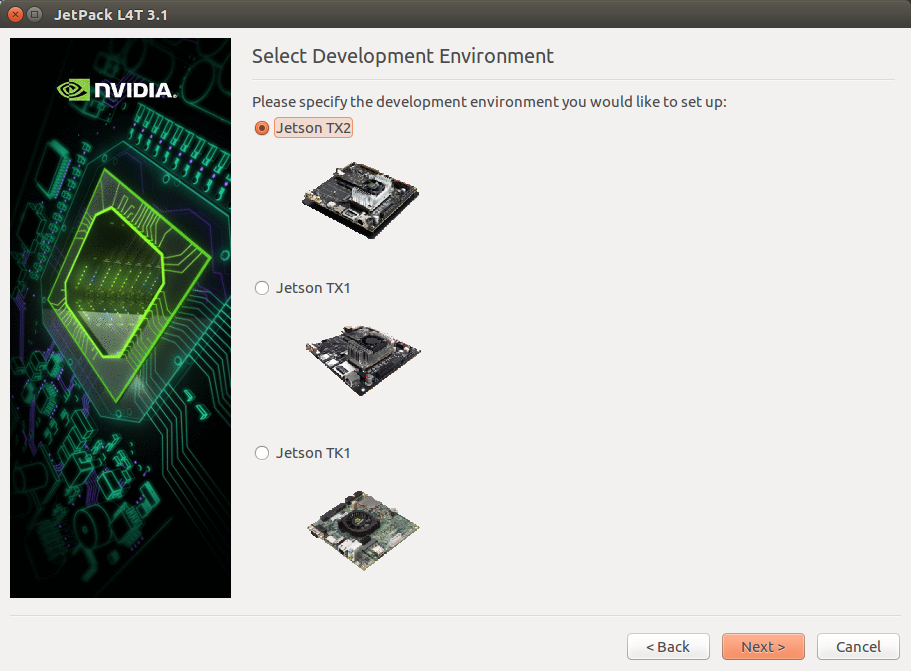
[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/jetpack05%20(2).png?attredirects=0)

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/jetpack_l4t_description.001_800x543.png?attredirects=0)

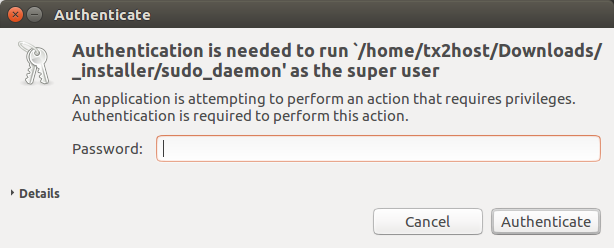
選擇NEXT會出現下圖告訴你這些元件會安裝的位置

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack02.png?attredirects=0)

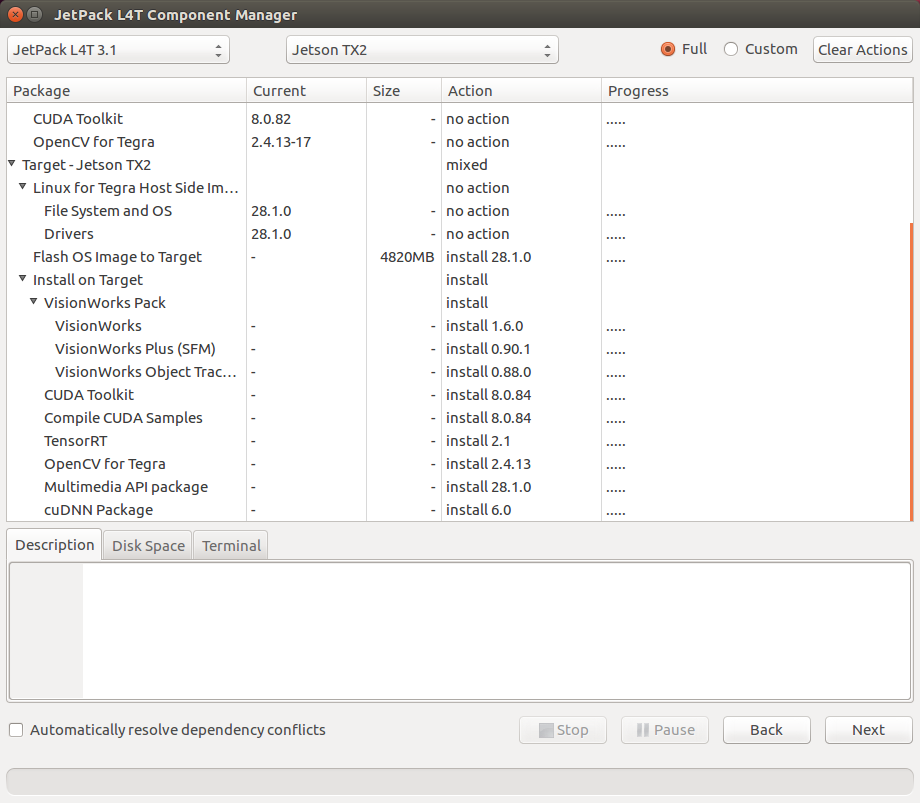
接著會詢問你想要安裝的是哪塊開發板 現在還不用擔心連接的問題 選好按下一步

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack03.png?attredirects=0)

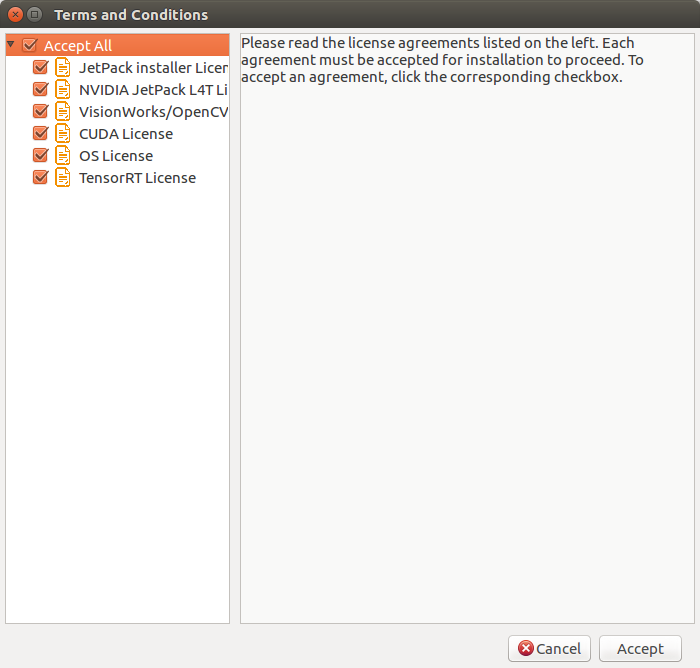
接下來會詢問你的密碼 因為要使用super user 的權限才能安裝

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack04.png?attredirects=0)

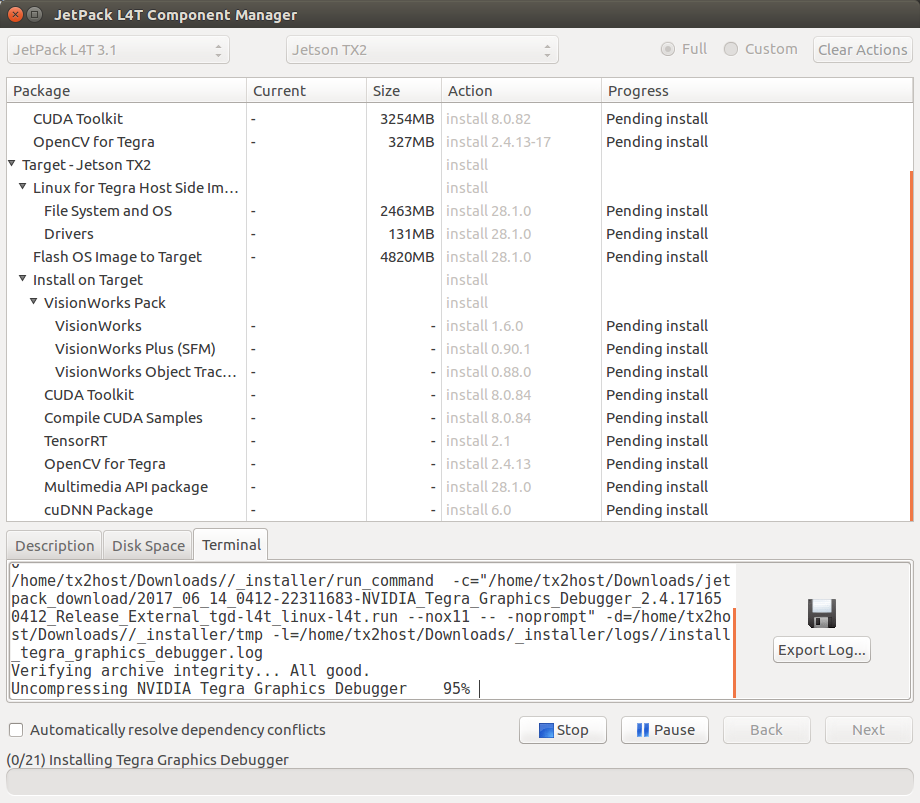
接著安裝程式會把JetPack L4T的元件列出來，直接選擇下一步

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack05-1.png?attredirects=0)

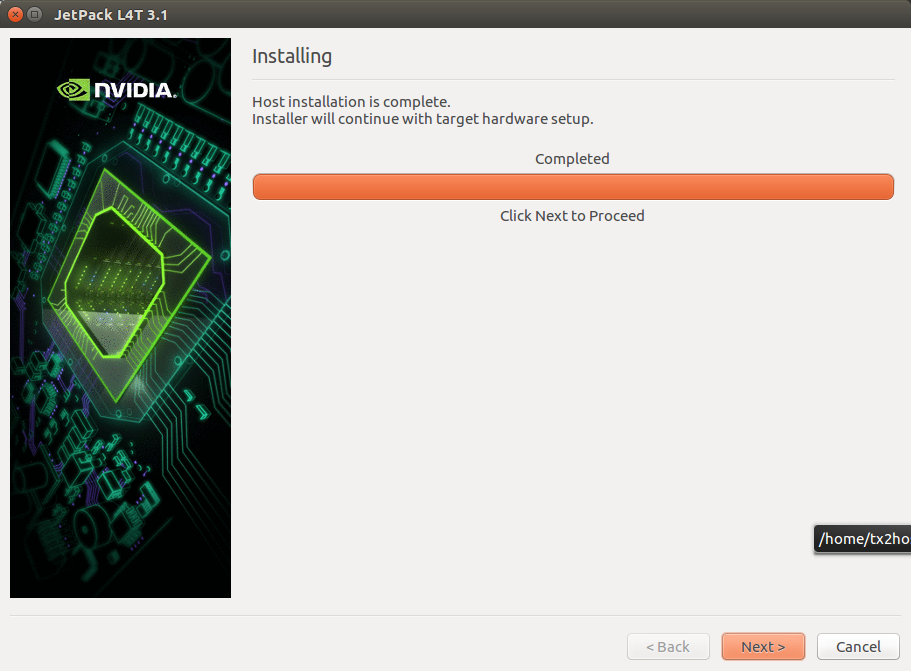
請勾選 Accept All，勾選後選擇右下 Accept才可作用

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack07.png?attredirects=0)

等待下載完所有套件，會在host端直接安裝(需要注意硬碟空間足夠，否則會出錯)

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack10.png?attredirects=0)

.

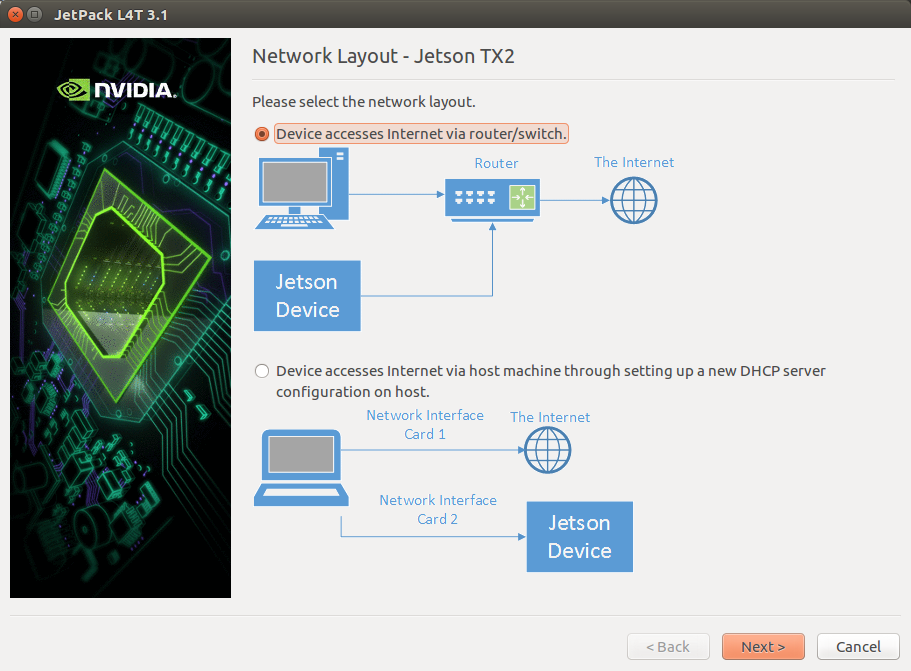
[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack11.png?attredirects=0)

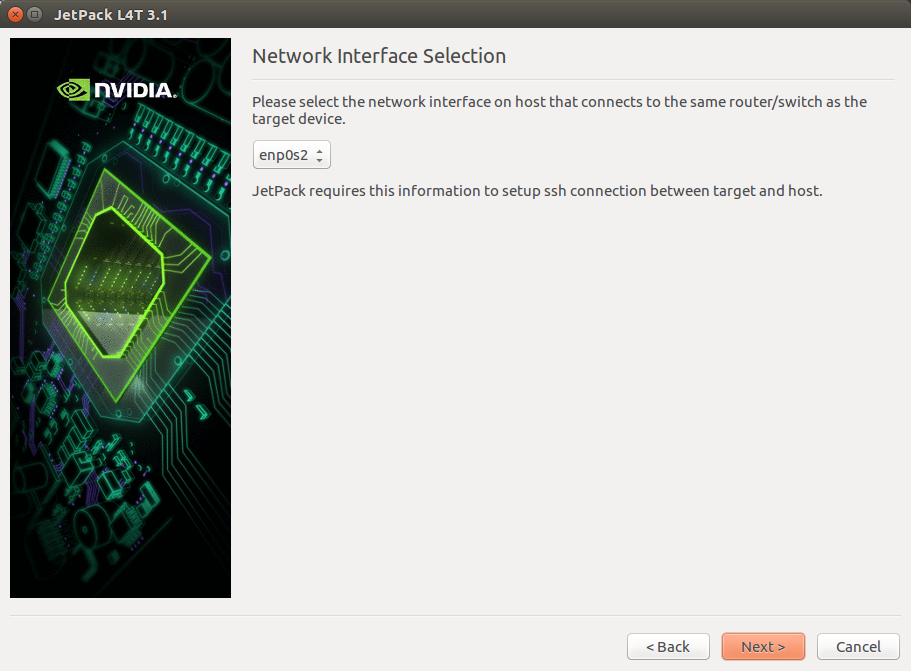
因為我們有選擇 **Flash OS**在 [Component Manager](http://docs.nvidia.com/jetpack-l4t/content/developertools/mobile/jetpack/l4t/3.0/jetpack_l4t_install.htm#component_manager), 所以我們會需要選擇一下網路的架構

上方是TX2與HOST都以網路線連至同一顆switch(同一網段)

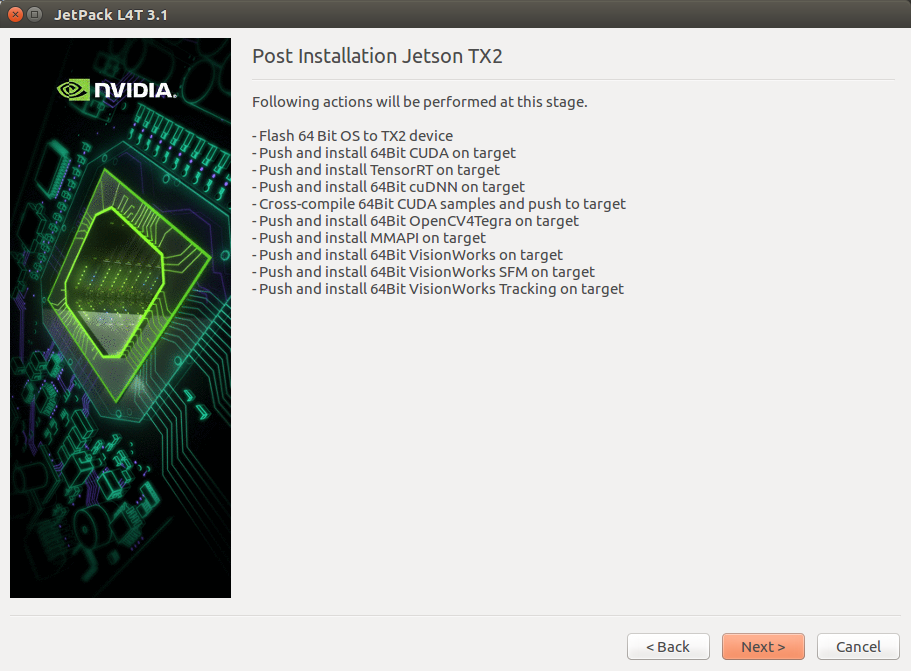
下方則是TX2是透過HOST設定DHCP伺服服務而連至Internet(目前已知這種連接方式是有BUG的 所以請不要選擇此種方式)

我們建議使用第一種(如果分享器就在你附近又剛好有兩個空的網路孔)

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack12.png?attredirects=0)

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack13.png?attredirects=0)

接著它說明了它繼續會安裝哪些程式元件到TX2

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack14.png?attredirects=0)

在這個時候可以把鍵盤滑鼠(自備)的接收器插到 TX2上,還有用HDMI線(自備)連到螢幕(自備),網路線插上連至switch

接著會跳出一個視窗(下圖)是告訴你要如何讓TX2進入強制USB復原模式

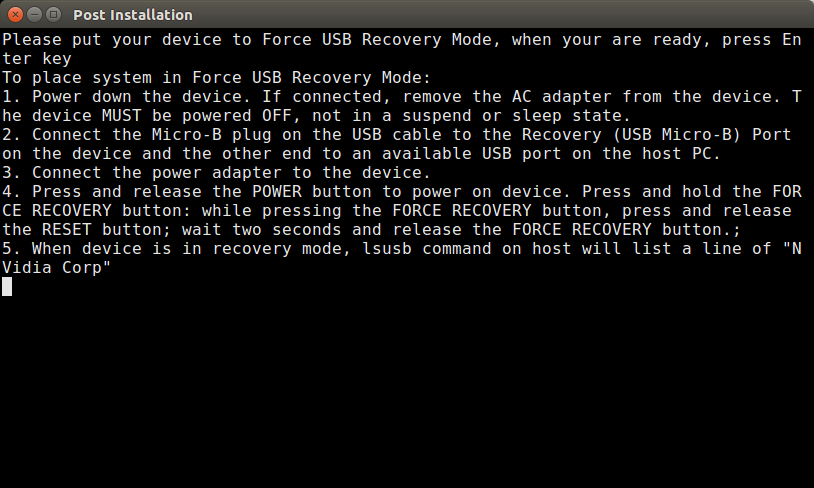
1.TX2要斷電.如果變壓器有插著也要拔掉.TX2必須是關機而不是待機或休眠狀態.

2.把附帶的USB線Micro-B的那頭插入到TX2相對應的插孔,另一頭插入HOST

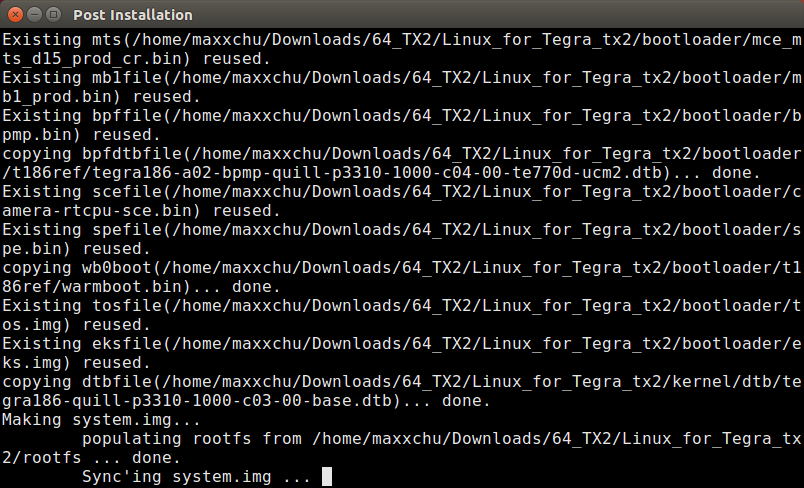
3.把變壓器插頭插入TX2.(剛插上電源 風扇沒轉是正常的,它是溫控的 有做大量運算溫度高時才會啟動 以達到省電效果)

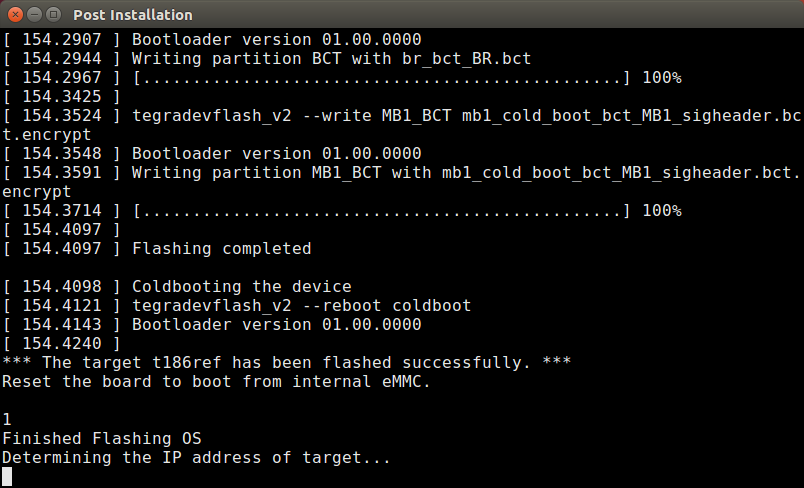
4.按一下TX2上的POWER BTN鈕讓TX2啟動。按住REC鈕不放(FORCE RECOVER)並按一下RST鈕放開(RESET)大約2秒後再放開REC鈕

5.當TX2已經進入復原模式 你可以在HOST鍵入lsusb後按下enter會有一行字有"NVidia Corp"的字眼。繼續按下enter就會開始安裝程式元件到TX2

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack15.png?attredirects=0)

Host開始建立要燒入到tx2的檔案，要跑個好幾分鐘

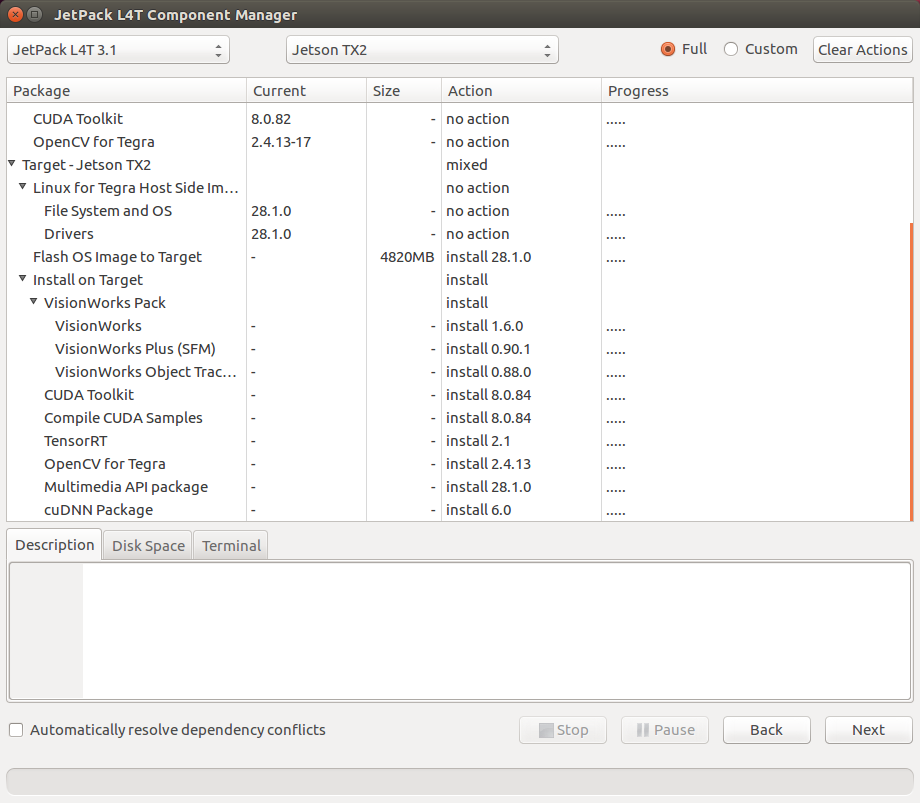
[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/jetpack%20install.png?attredirects=0)



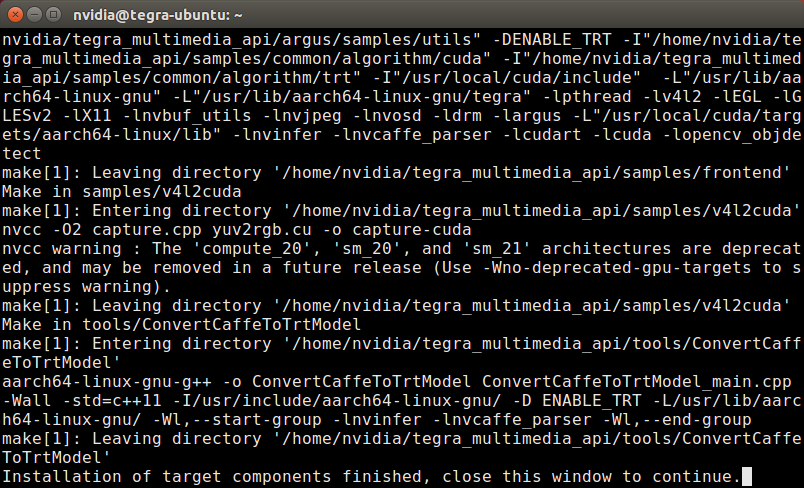
看到Finished Flashing OS不要高興得以為已經完畢了

到這邊可以直接按ctrl C將視窗關掉，重新再執行一次.run檔案，將前面的步驟再執行一次。

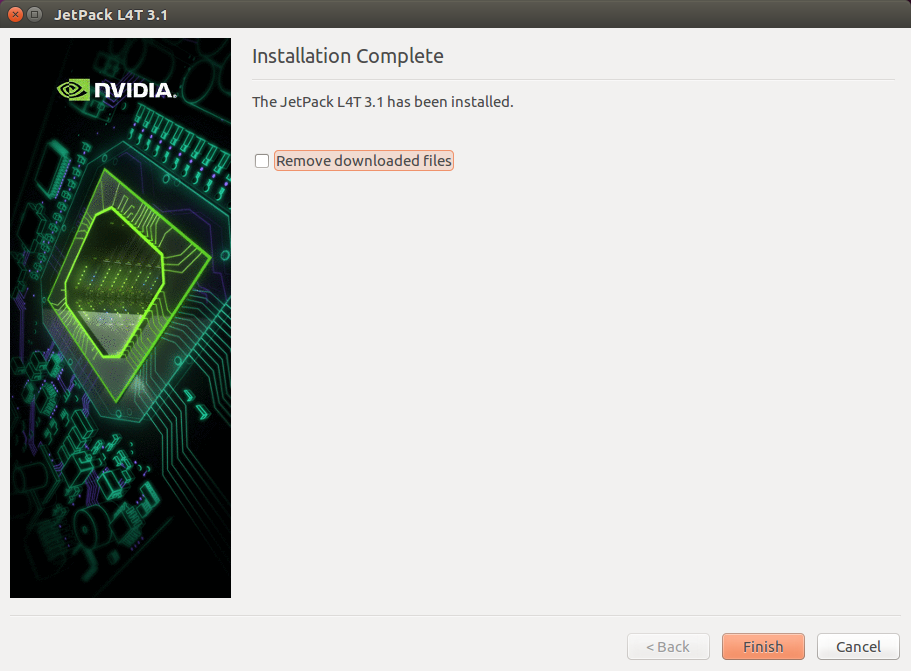
注意此步驟要將Flash OS Image to Target由install改為no action

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/Jetpack05-1.png?attredirects=0)

繼續按下Next並接受license後，會跳出畫面要求手動輸入tx2的IP以及帳號密碼(在tx2打開終端機輸入ifconfig可以查看IP，帳密預設都是nvidia)，輸入完成再按下一步，就會透過區網開始在tx2安裝套件，大約再等待等個15分鐘就會出現下圖。

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/finish.png?attredirects=0)

要看到close this window to continue並把這個視窗關掉

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/hostFInish.png?attredirects=0)

跳回這個視窗按下Finish

現在TX2連著的螢幕應該會重新啟動並進入Ubuntu16.04的桌面環境

預設的帳號和密碼是nvidia/nvidia

[](http://www.honghutech.com/nvidia-jeston-tx2/flashtx2/start%20up.png?attredirects=0)

**安裝Tensorflow、keras :**

以下內容節入至[github](https://gist.github.com/vellamike/7c26158c93e89ef155c1cc953bbba956)，下列安裝指令都在tx2執行

Jetpack 3.2 includes Cuda 9 and CuDNN 7 so it is necessary to compile it from source.

**Step 1 - Get Java**

sudo apt-get install openjdk-8-jdk

**Step 2 - Get some dependencies**

sudo apt-get install python3-numpy swig python3-dev python3-pip python3-wheel -y

**Step 3 - Get Bazel**

wget --no-check-certificate https://github.com/bazelbuild/bazel/releases/download/0.10.0/bazel-0.10.0-dist.zip

unzip bazel-0.10.0-dist.zip -d bazel-0.10.0-dist

cd bazel-0.10.0-dist

./compile.sh

cp output/bazel /usr/local/bin

**Step 4 - clone tensorflow**

git clone https://github.com/tensorflow/tensorflow # You may wish to check out a speicfic branch here

**Step 5 - configure tensorflow**

cd tensorflow

# This will prompt you with various questions, my answers are included as an example of what worked for me.

./configure

You have bazel 0.10.0- (@non-git) installed.

Please specify the location of python. [Default is /usr/bin/python]: /usr/bin/python3

Found possible Python library paths:

/usr/local/lib/python3.5/dist-packages

/usr/lib/python3/dist-packages

Please input the desired Python library path to use. Default is [/usr/local/lib/python3.5/dist-packages]

Do you wish to build TensorFlow with jemalloc as malloc support? [Y/n]:

jemalloc as malloc support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Google Cloud Platform support? [Y/n]: n

No Google Cloud Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Hadoop File System support? [Y/n]: n

No Hadoop File System support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Amazon S3 File System support? [Y/n]: n

No Amazon S3 File System support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Apache Kafka Platform support? [y/N]: n

No Apache Kafka Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with XLA JIT support? [y/N]: n

No XLA JIT support will be enabled for TensorFlow.

Do you wish to build TensorFlow with GDR support? [y/N]:

No GDR support will be enabled for TensorFlow.

Do you wish to build TensorFlow with VERBS support? [y/N]:

No VERBS support will be enabled for TensorFlow.

Do you wish to build TensorFlow with OpenCL SYCL support? [y/N]:

No OpenCL SYCL support will be enabled for TensorFlow.

Do you wish to build TensorFlow with CUDA support? [y/N]: y

CUDA support will be enabled for TensorFlow.

Please specify the CUDA SDK version you want to use, e.g. 7.0. [Leave empty to default to CUDA 9.0]:

Please specify the location where CUDA 9.0 toolkit is installed. Refer to README.md for more details. [Default is /usr/local/cuda]: /usr/local/cuda-9.0

Please specify the cuDNN version you want to use. [Leave empty to default to cuDNN 7.0]:

Please specify the location where cuDNN 7 library is installed. Refer to README.md for more details. [Default is /usr/local/cuda-9.0]:

Do you wish to build TensorFlow with TensorRT support? [y/N]:

No TensorRT support will be enabled for TensorFlow.

Please specify a list of comma-separated Cuda compute capabilities you want to build with.

You can find the compute capability of your device at: https://developer.nvidia.com/cuda-gpus.

Please note that each additional compute capability significantly increases your build time and binary size. [Default is: 3.5,5.2]

Do you want to use clang as CUDA compiler? [y/N]:

nvcc will be used as CUDA compiler.

Please specify which gcc should be used by nvcc as the host compiler. [Default is /usr/bin/gcc]:

Do you wish to build TensorFlow with MPI support? [y/N]:

No MPI support will be enabled for TensorFlow.

Please specify optimization flags to use during compilation when bazel option "--config=opt" is specified [Default is -march=native]:

Would you like to interactively configure ./WORKSPACE for Android builds? [y/N]:

Not configuring the WORKSPACE for Android builds.

Preconfigured Bazel build configs. You can use any of the below by adding "--config=<>" to your build command. See tools/bazel.rc for more details.

--config=mkl # Build with MKL support.

--config=monolithic # Config for mostly static monolithic build.

--config=tensorrt # Build with TensorRT support.

Configuration finished

**Step 6 - Build tensorflow**

# Build a whl

bazel build --config=opt --config=cuda //tensorflow/tools/pip\_package:build\_pip\_package

**Step 7 - Make a whl**

bazel-bin/tensorflow/tools/pip\_package/build\_pip\_package /tmp/tensorflow\_pkg #put the whl in /tmp/tensorflow\_pkg

**Step 8 - install the whl**

pip3 install /tmp/tensorflow\_pkg/tensorflow-1.6.0rc0-cp35-cp35m-linux\_aarch64.whl #depending on your tensorflow/python version

**其他可能會遇到的問題 :**

1. **DAB100無法認得裝置:**

由於DAB100使用的Tx2與原廠的硬體設計不同，硬體設備只有micro USB沒有typeA USB，因此剛燒完OS要使用micro USB當作typeA USB的時候可能會出現該port無效的問題(鍵盤、滑鼠、隨身碟等裝置都無法讀取，且不過電)。

原因是Tx2的flash os kernel沒有給usb2.0電，且預設的擴充管理會將USB的設定覆蓋掉(<https://elinux.org/Jetson/TX2_USB>)，參考前述網址將device tree檔案修改設即可。

**修改方法:**

在host端將tegra186-quill-p3310-1000-c03-00-base.dtb輸出成可讀的dtsi格式並內容修改。

**Step1.** 找到/Linux\_for\_Tegra\_tx2/kernel/dtb/tegra186-quill-p3310-1000-c03-00-base.dtb

sudo apt-get install device-tree-compiler

**Step2.** dtc -I dtb -O dts -o extracted.dtsi kernel/dtb/tegra186-quill-p3310-1000-c03-00-base.dtb(dtc要透過apt下載)

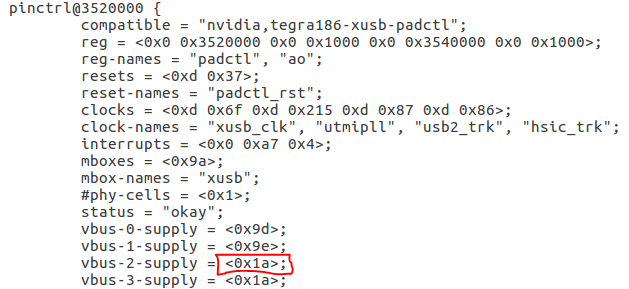
**Step3.1** 在extracted.dtsi修改usb2.0給電設定

Usb需修改為 5v : <0x26>

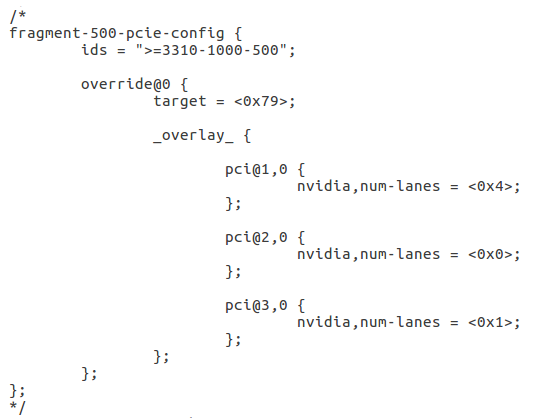
//<0x26> 先16進轉2進

<https://www.convertworld.com/zh-hant/numerals/hexadecimal.html>

//用2進(0~7) 查表 看offset 範例: <http://danielwippermann.github.io/resol-vbus/vbus-specification.html>



**Step3.2** 註解fragment-500-pcie-config、fragment-500-xusb-config、fragment-500-e3325-pcie區段



**Step4.** 儲存extracted.dtsi，編譯回tegra186-quill-p3310-1000-c03-00-base.dtb

sudo dtc -I dts -O dtb -o 檔案名稱.dtb extracted.dtsi // dtc input dts output dtb

**Step5.** 將新編譯的tegra186-quill-p3310-1000-c03-00-base.dtb取代下列三個檔案

“Linux\_for\_Tegra\_tx2/rootfs/boot/tegra186-quill-p3310-1000-c03-00-base.dtb”

“Linux\_for\_Tegra\_tx2/bootloader/tegra186-quill-p3310-1000-c03-00-base.dtb”

“Linux\_for\_Tegra\_tx2/kernel/dtb/tegra186-quill-p3310-1000-c03-00-base.dtb”

**Step6.**

開啟Tx2並進入recovery mode，執行Linux\_for\_Tegra\_tx2/flash.sh

sudo ./flash -r -k kernel-dtb jetson-tx2 mmcblk0p1