AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip in Linux environment

User guide

About this document

Scope and purpose

This guide enables you to quickly set up a Wi-Fi STA (Station) using the AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip and connect to an AP (Access Point)/SoftAP. NVIDIA® Jetson Xavier NX™ developer kit, running Ubuntu 20.04, is used as the host processor for WLAN evaluation.

Note: *This document is only a draft and is subject to frequent updates.*

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **3**
2 Hardware requirements **4**
2.1 Hardware setup: PCIe 5
2.2 Hardware setup: SDIO 5
2.2.1 Rework the host Xavier NX 6
2.2.2 Rework the CYW955573M2IPA1 board 7
3 Force Recovery mode **8**
4 Verify Force Recovery mode **9**
5 Bring up CYW5557x on Jetson Xavier NX platform **10**
6 Kernel customization and CYW5557x bring-up **11**
6.1 Use a pre-configured SD card image 11
6.2 Using the Flash Tool 12
6.3 Configure Jetson Xavier NX platform 12
6.3.1 Modify the kernel 12
6.3.2 Download the root file system (rootfs) 13
6.3.3 Flash the kernel and DTB 13
6.3.3.1 Run the flash script 13
6.3.3.2 Complete the system configuration 14
6.4 Generate the flashing tool 14
6.4.1 Command-line options 14
6.4.2 Generate package 16
6.5 Verify the modifications 16
6.6 Compile the driver 16
6.7 Load the FMAC driver and firmware 17
6.8 Configure the SoftAP 18
6.9 Configure the STA 20
6.10 Ping the SoftAP and STA 22
6.11 Test the throughput 23
6.12 Bluetooth® release package 23
6.12.1 Setting up BlueZ for Linux environment 23
6.12.2 Download the Bluetooth® firmware to patch RAM 24
6.12.2.1 Use antenna configuration dLNA_ANT0 26
6.12.2.2 Use antenna configuration sLNA_ANT0 27
6.12.2.3 To use antenna configuration dLNA_BTANT 27
6.12.3 HCI UART bring up 29
Revision history **30**

1 Introduction

This guide provides step-by-step instructions to configure the Jetson Xavier NX developer kit from NVIDIA , compile and load the WLAN driver, and establish a Wi-Fi connection between a SoftAP and STA. The necessary WLAN-related files are provided as an attachment along with this guide. Contact your local Infineon Technologies distribution channel (FAE or local sales representative) to get the files.

2 Hardware requirements

1. PC with Ubuntu 18.04 and above installed
2. **Jetson Xavier NX developer kit from NVIDIA**
3. Monitor and keyboard to connect to the Jetson Xavier NX developer kit from NVIDIA
4. CYW955572M2IPA1 - AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip

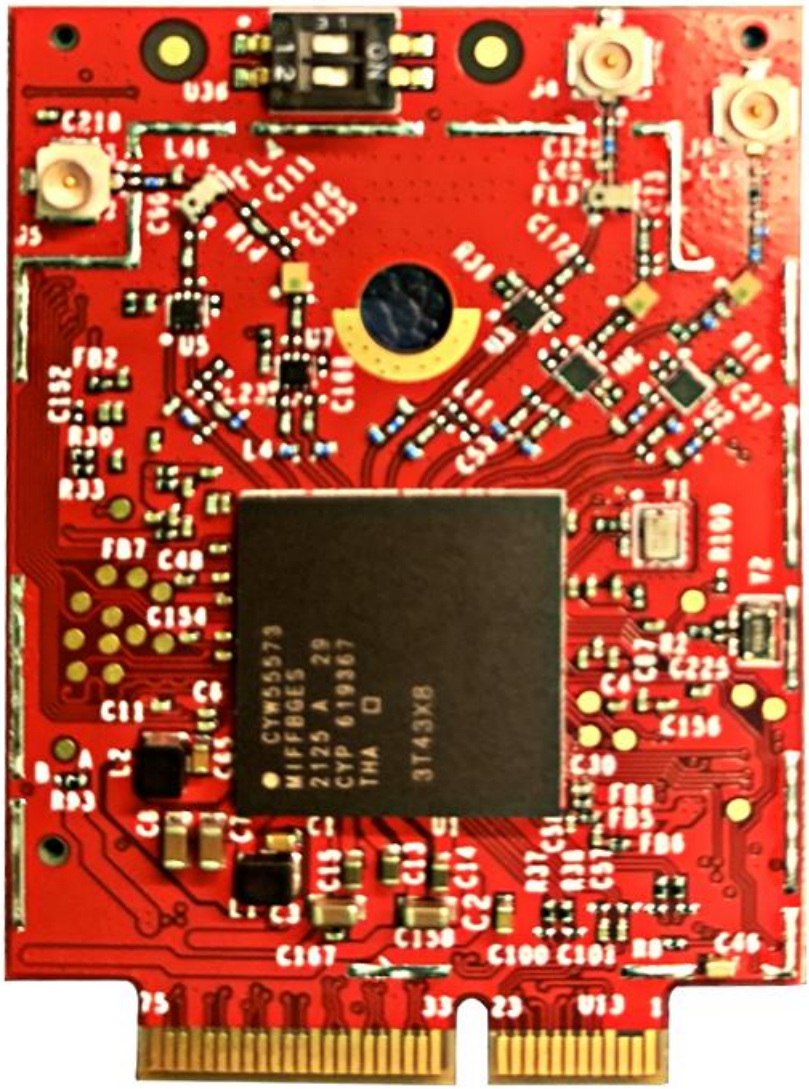


Figure 1 AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip

2.1 Hardware setup: PCIe

1. Connect the AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip to the M.2 Key E Mini-PCIe slot (J10) on the Jetson Xavier NX platform carrier board.

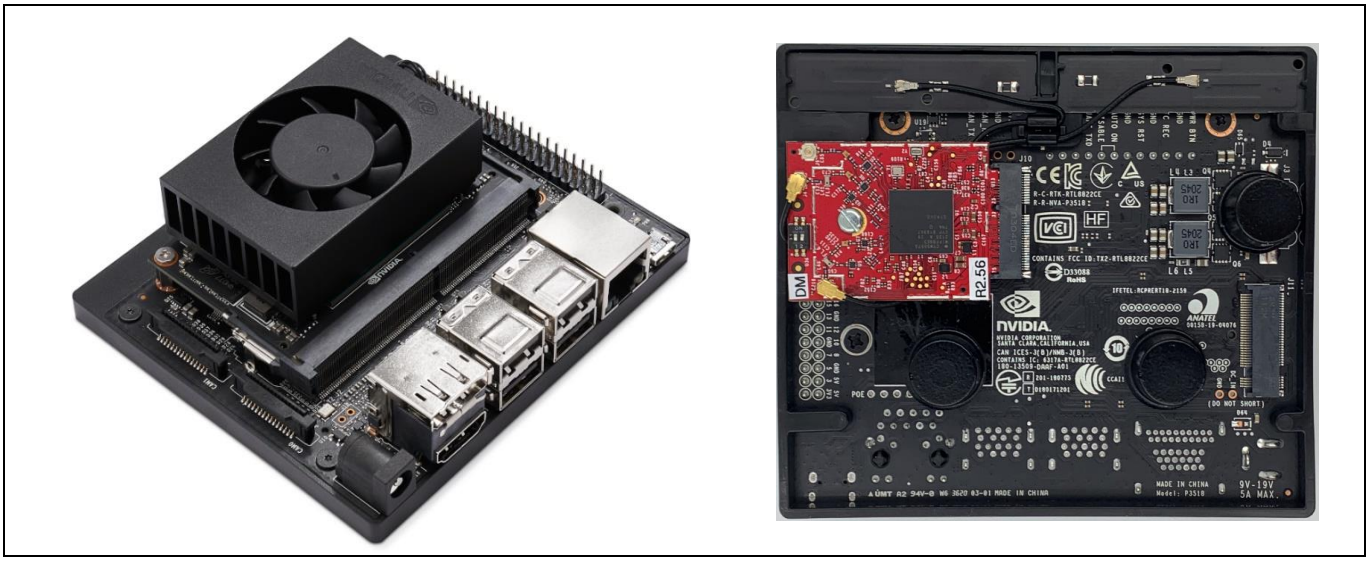


Figure 2 AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip connected to Jetson Xavier NX developer kit

2.2 Hardware setup: SDIO

Do the following rework to use the SDIO interface:

2.2.1 Rework the host Xavier NX

1. Add 0402-size 0-ohm resistors to R187, R188, R189, R190, and R191 pads.
2. Move R185 to R186.
3. Add a blue wire from Pin 23 M2.E (socket J10 - SDIO RESET) to Pin 15 (GPIO12) on the 40 pin connector (J12).
4. Add a blue wire from Pin 21 M2.E (socket J10 - SDIO WAKE) to Pin 31 (GPIO11) on the 40 pin connector (J12).

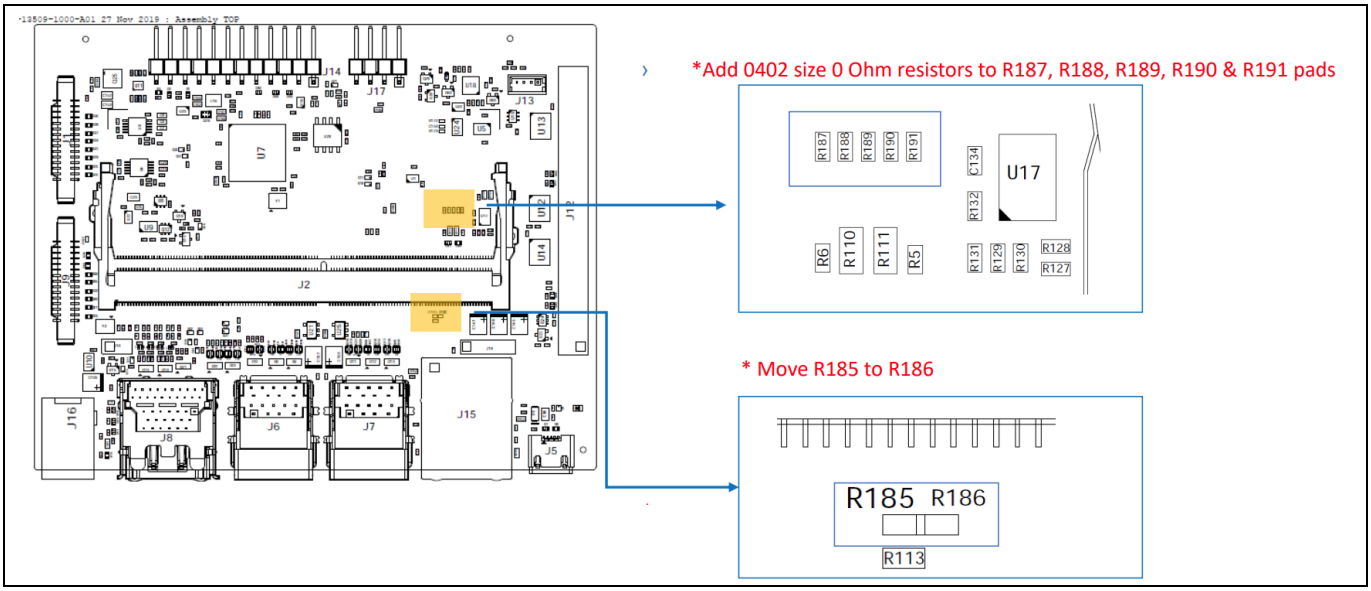


Figure 3 Xavier NX Rework Instructions on the front side for SDIO

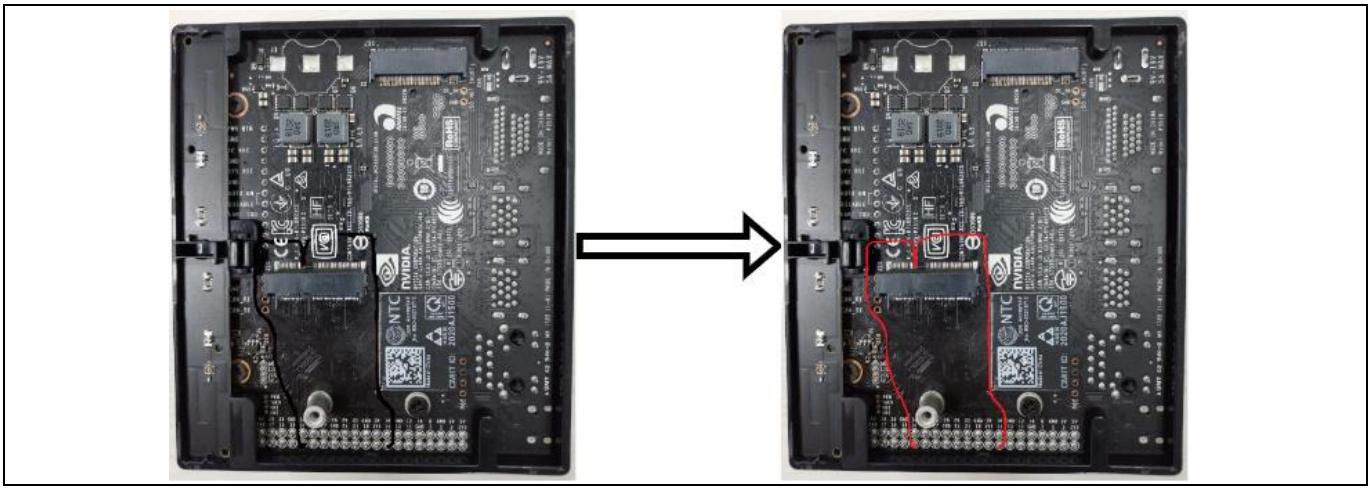


Figure 4 Xavier NX Rework Instructions on the back side for SDIO

2.2.2 Rework the CYW955573M2IPA1 board

1. Change the positioning from AC to BC on R107.
2. Rotate R92 positioning from AC to BC.
3. Depopulate U33.

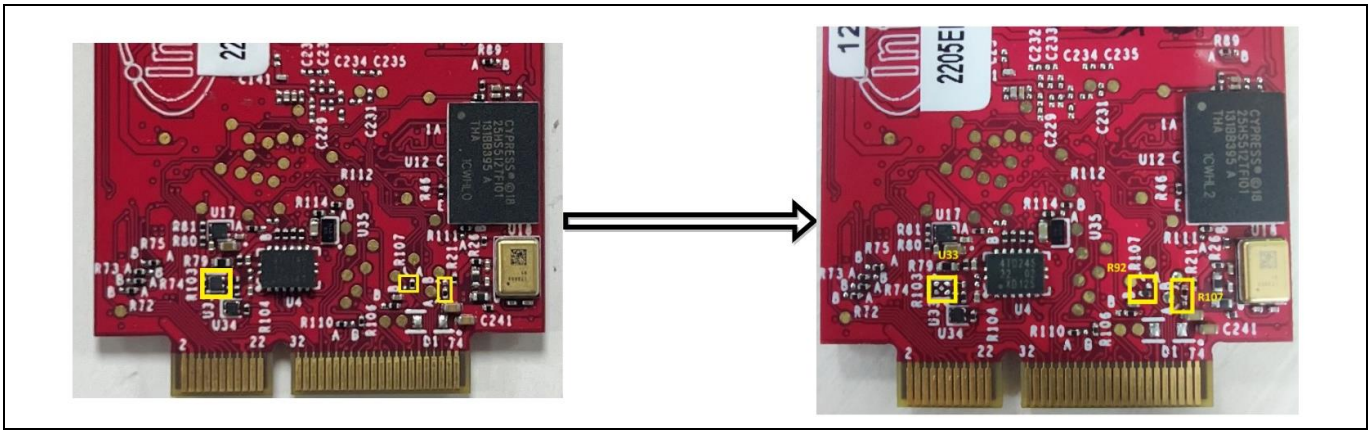


Figure 5 CYW955573M2IPA1 rework instructions for SDIO

3 Force Recovery mode

See the **Jetson Linux Developer Guide** for detailed instructions to enter recovery mode in the Jetson Xavier NX developer kit.

1. Ensure that you connect the Jetson Xavier NX developer kit with a host Ubuntu machine using a micro USB cable.
2. Ensure the kit is powered off, and that a 16 GB or larger size microSD card is inserted in the SD card slot.
3. To ensure that the developer kit starts in Force Recovery mode, place a jumper across pins 9 and 10 (FC REC and GND) of the button header (J14) located on the edge of the carrier board under the Jetson module.
4. Connect the included power adapter to J16. The developer kit powers on and enters Force Recovery mode.
5. Remove the jumper from pins 9 and 10 of the button header.

4 Verify Force Recovery mode

Do the following to determine whether the Jetson Xavier NX developer kit is in Force Recovery mode:

1. Connect a Micro-USB (J5) cable to the Ubuntu PC's USB host.
2. Check the `lsusb` command output. The Jetson Xavier NX module is in Force Recovery mode if you see the following message:

   ```
   Bus 002 Device 004: ID 0955:7e19 NVidia Corp.
   ```

   Where `7e19` indicates that the Jetson module is in force recovery mode.

5 Bring up CYW5557x on Jetson Xavier NX platform

Use one of the following four methods to bring up Infineon's AIROC™ CYW5557x on the Jetson Xavier NX developer kit:

1. Use a pre-configured SD card image (~2.5 GB).
   Get the image (*sd-blob-xavier-nx-pcie.tbz2*) from your local Infineon Technologies distribution channel (FAE or local sales representative).
   See the **Use a pre-configured SD card image** section for the steps required to flash the microSD card offline. You need to remove the microSD card from the Jetson Xavier NX developer kit and insert it into the Ubuntu PC for programming.
2. Use a pre-configured flashing tool package (~2.5 GB).
   Get the package (*flashing-tool-xavier-nx-pcie.tbz2*) from your local Infineon Technologies distribution channel (FAE or local sales representative).
   See the **Using Flash Tool** section for the steps to flash the microSD card on the Jetson Xavier NX developer kit. This method simply requires a micro USB cable to be connected to the Ubuntu PC; you do not have to remove the microSD card from the Jetson Xavier NX developer kit.
3. Use the provided kernel image and device tree blob file to create the required configuration on the Jetson Xavier NX developer kit if you do not want to flash the entire filesystem.
   The following files are required for the Wi-Fi bring-up:
   a) Kernel image file (*Image*): To use specific drivers for Wi-Fi rather than default
   b) Device tree binary (*tegra194-p3668-all-p3509-0000.dtb*): To port the kernel configuration to the Jetson Xavier NX developer kit
   c) DHD driver binary (*bcmdhd.ko*)
   d) Production Wi-Fi firmware binary (*fw_prod.trxse*)
   e) Manufacturing Wi-Fi firmware binary (*fw_mfg.trxse*)
   f) Wi-Fi NVRAM configuration file (*cyw955572m2ipa1_rev2.56.txt*)
   g) User-space utilities for Wi-Fi STA management: *wpa_supplicant* and *wpa_cli*
   h) User-space utilities for Wi-Fi SoftAP management: *hostapd* and *hostapd_cli*
   All the above files are packed in *SoftwareFiles.zip*. Contact your local Infineon Technologies distribution channel (FAE or local sales representative) for this package.
   Download and extract these files in the Jetson Xavier NX device.
   ```
   $ unzip SoftwareFiles.zip
   ```
   See the **Configure Jetson Xavier NX** section for the steps to configure the Jetson Xavier NX developer kit.
4. Use a script to generate the SD card image or flashing tool package on your own.
   Get the script files as a zip package (*nvidia_l4t_image_generator.zip*) from your local Infineon Technologies distribution channel (FAE or local sales representative). The script package contains the following:
   a) *ifx_nvidia_jetson_genimg.sh* – Main script file.
   b) *cyw5557x/* – Folder which contains all files and binaries related to WLAN and Bluetooth®.
   c) *patches/* – Folder which contains all patch files which get applied to kernel sources before build.
   Download and extract the zip file on the Ubuntu PC.
   ```
   $ unzip nvidia_l4t_image_generator.zip
   ```
   See the **Generate Flashing Tool** section for the steps to generate the SD script package on your own.

6 Kernel customization and CYW5557x bring-up

6.1 Use a pre-configured SD card image

1. Download the sd-*blob-xavier-nx-pcie.tbz2* file on to the Ubuntu PC.
2. Extract the file, which will produce the raw SD image with name *sd-blob.img*.

```
$ tar xvjf sd-blob-xavier-nx-pcie.tbz2
```

3. Connect the microSD card to your Ubuntu PC. It is recommended to use a 64-GB microSD card to provide enough space for developing applications.
4. Check the enumerated microSD device ID:

```
$ sudo parted --list
```

The device ID can be in the form of "*/dev/sdX*" (when you connect via microSD to USB connector) or "*/dev/mmcblkX*" (when you connect the microSD directly or with SD to microSD adaptor) depending on how you have connected the microSD card to the PC.

5. Flash the microSD card:

```
$ sudo dd if=sd-blob.img of=/dev/<id> bs=1M oflag=direct status=progress
  && sync
```

where `id` refers to the enumerated microSD card on the PC. This will take few minutes to complete flashing the microSD card. When completed, remove the microSD card.

Warning! Make sure the `id` you are referring to is correct because this command will simply overwrite the destination memory.

6. Ensure that your Jetson Xavier NX developer kit is powered off, and then insert the microSD into the SD card slot under the Jetson Xavier NX™ module and power it ON.
7. Connect a Micro-USB to USB cable between the Jetson Xavier NX developer kit and Ubuntu PC. This is used for programming and debugging purposes.
8. Log in to the board. Auto login is enabled by default with this SD card image; therefore, log in using the following credentials:

   username: ifx

   password: ifx
9. Go to **Section 6.5** to proceed with verification.

6.2 Using the Flash Tool

1. Download the *flashing-tool-xavier-nx-pcie.tbz2* compressed file on to the Ubuntu PC.
2. Extract the file which will produce a new folder *sd-script* and navigate to the *sd-script* folder.

```
$ tar xvjf flashing-tool-xavier-nx-pcie.tbz2
```

3. Make sure you insert the microSD card into the Jetson Xavier NX developer kit's microSD slot and a micro USB cable is connected to the Ubuntu PC.
4. Follow sections **3** and **4** to put the Jetson Xavier NX developer kit into Force Recovery mode.

   Or, if you already have console access and have logged in to the Jetson Xavier NX developer kit board, you can enter the recovery mode by running the following command:

```
$ sudo reboot --force forced-recovery
```

5. Flash the image:

```
$ sudo ./nvautoflash.sh -r
```

   It takes several minutes to complete flashing the system image. Once completed, the Jetson Xavier NX™ developer kit will auto restart.

6.3 Configure Jetson Xavier NX platform

6.3.1 Modify the kernel

1. Download the L4T BSP driver package from the **Nvidia website** to the Ubuntu laptop.

   Note:

   a) Use only NVIDIA L4T v35.1, which is the latest L4T BSP version NVIDIA has released at the time of writing this document; Infineon has tested with this version.

   b) If you have used the Jetson Xavier NX platform before, check the driver version by executing the following command on the Jetson Xavier NX developer kit board:

```
$ head -1 /etc/nv_tegra_release
```

   You can also check the driver package version in the Ubuntu PC by navigating to the *Linux_for_Tegra/kernel* directory and checking for the file names with **.deb*.

   For example: *nvidia-l4t-kernel-headers_5.10.104-tegra-35.1.0-20220810203728_arm64.deb*

   This indicates kernel version 5.10.104 and driver package version 35.1.0.

   You can skip downloading and extracting the package (steps 1 and 2) if you already have the v35.1.0 package without any additional modifications.

2. Extract the package:

```
$ cd <path to downloaded driver package>
$ tar xf jetson_linux_r35.1.0_aarch64.tbz2
```

   This creates a new folder called *Linux_for_Tegra*.

3. Replace the following existing files with the files provided in the *SoftwareFiles.zip :*

   a) `Linux_for_Tegra/kernel/Image`

   b) `Linux_for_Tegra/kernel/dtb/tegra194-p3668-all-p3509-0000.dtb`

6.3.2 Download the root file system (rootfs)

1. Download sample *rootfs* from the **NVIDIA website**.
   Ensure that the rootfs version is the same version as that of the driver package downloaded.
2. Extract rootfs in *Linux_for_Tegra/rootfs* folder and populate the debian binaries shipped with the sample rootfs.

```
$ cd <path to Linux_for_Tegra/rootfs folder>
$ sudo tar xpf ../../tegra_linux_sample-root-filesystem_r35.1.0_aarch64.tbz2
$ cd ..
$ sudo ./apply_binaries.sh
```

6.3.3 Flash the kernel and DTB

Do the following to update the kernel and dtb files:

1. Put the Jetson Xavier NX developer kit in recovery mode.
2. Ensure that the USB Micro-B cable is connected between the Jetson Xavier NX developer kit and the Ubuntu PC.
3. Follow sections **3** and **4** to put the Jetson Xavier NX developer kit into Force Recovery mode. Or, if you already have console access and have logged in to the Jetson Xavier NX platform, you can enter the recovery mode by running the following command:

```
$ sudo reboot --force forced-recovery
```

6.3.3.1 Run the flash script

This section shows how to flash the kernel image and device tree blob file individually. Execute the following commands from the *Linux_for_Tegra* folder only.

1. Enter the following command to flash the kernel image:

```
$ sudo ./flash.sh -k kernel jetson-xavier-nx-devkit mmcblk0p1
```

2. When the flashing is done, the Jetson Xavier NX platform from NVIDIA will auto-restart. To confirm whether the kit is up and running (normal mode), you can check `lsusb` on the Ubuntu PC. If it enumerates as 7020, then it means the Jetson Xavier NX developer kit is running in normal mode.

```
$ lsusb
Bus 002 Device 006: ID 0955:7020 NVidia Corp.
```

3. Log in again, enter recovery mode as explained in Section 6.3.3, and run the following command to flash the device tree blob:

```
$ sudo ./flash.sh -k kernel-dtb jetson-xavier-nx-devkit mmcblk0p1
```

Note: *Some virtual environments might not handle the command properly through USB forwarding. NVIDIA recommends using the native Ubuntu machine.*

6.3.3.2 Complete the system configuration

- Connect the monitor and keyboard to the Jetson Xavier NX developer kit. Complete the license agreement and system configuration once the system boots up.

6.4 Generate the flashing tool

This section assumes that you have followed the instructions in **Section 5** (Step 4) and extracted the *nvidia_l4t_image_generator.zip* on to your Ubuntu host machine.

The script does the following:

1. Downloads the L4T BSP from the Nvidia web server
2. Checks for pre-requisites and installs the dependent utilities automatically such as GNU Make, Git, and the aarch64 toolchain if they are not installed already
3. Cross-compiles the kernel sources and generates an SD card script package for the Jetson Xavier NX developer kit

Note: This script has tested on Jetson Xavier NX developer kit using L4T BSP v35.1.0.

6.4.1 Command-line options

The script uses various command line options as follows:

```
$ sudo ./ifx_nvidia_jetson_genimg.sh -h

usage: sudo ./ifx_nvidia_jetson_genimg.sh [-b|--bsp_dir] [-e|--
eval_kit_name] [-i|--image_type] [-w|--wifi_dir] [-u|--username] [-p|--
password] [-a|--auto_login] [-h|--help]

  -b      Specify the path to where you want the NVIDIA L4T BSP directory
(Linux_for_Tegra/) be created. (default: /home/user-
prah/jetson_xavier_nx/automation/Linux_for_Tegra)

  -t      Specify the tag name against which the BSP sources will be
pulled. (default: tegra-l4t-r35.1.0)

  -e      Specify the Nvidia evaluation kit name. Valid kit names are:
'jetson-xavier-nx-devkit', and 'jetson-tx2'.

  -i      Specify the image type to generate. Valid image types are: 'sd-
binary', 'sd-script', and 'mfi-script'.

  -w      Specify the path to Wi-Fi component directory (with directory
name). (default: ./cyw5557x)

  -u      Set Login username. If not specified, then default will be set to
'ifx'.

  -p      Set Login password. If not specified, then default will be set to
'ifx'.

  -a      Skip system configuration on boot by enabling auto login.

  -h      Display this help menu


Sample commands:

sudo ./ifx_nvidia_jetson_genimg.sh -e jetson-tx2 -i mfi-script -a

sudo ./ifx_nvidia_jetson_genimg.sh -e jetson-xavier-nx-devkit -i sd-binary
-a

sudo ./ifx_nvidia_jetson_genimg.sh -e jetson-xavier-nx-devkit -i sd-script
-a

Valid combinations:

 ------------------------------------------------

|      DEVELOPER KIT      |      IMAGE TYPE      |

 ------------------------------------------------

|       jetson-tx2        |      mfi-script      |

 ------------------------------------------------

| jetson-xavier-nx-devkit | sd-binary, sd-script |

 ------------------------------------------------

mfi-script:

        Applicable only for Jetson TX2 devkit. Massflash package which
contains scripts to flash all connected Jetson devices simultaneously. This
requires to keep the Jetson board in Recovery mode.

sd-script:

        Applicable to Jetson Xavier NX devkit. SD card package which
contains scripts to flash the connected Jetson device (one device at a
time). This requires to keep the Jetson board in Recovery mode.

sd-binary:

        Applicable to Jetson Xavier NX devkit. This is a raw SD card binary
which can be flashed to SD card directly.
```

6.4.2 Generate package

1. Navigate to the directory where the *nvidia_l4t_image_generator.zip* package was extracted before.
2. Enter the following command:

```
$ sudo ./ifx_nvidia_jetson_genimg.sh -e jetson-xavier-nx-devkit -i sd-script -a
```

   Make sure that you verify the fields displayed on the screen after you enter this command. This will start downloading the L4T BSP sources and then compile them.

   A user account with the login username 'ifx' and password 'ifx' will be created and the auto-login will be enabled. The time it takes to finish the execution depends on your internet speed and PC configuration.

   The SD card script package (*flashing-tool-xavier-nx-pcie.tbz2*) will be available in the same place from where the script was executed.
3. Follow Section **6.2** to flash the Jetson Xavier NX developer kit.

6.5 Verify the modifications

To ensure successful modifications and flash, run the `lspci` command on the target device (i.e., Jetson Xavier NX platform). This indicates that the AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip is detected by the host PCIe controller of the Jetson Xavier NX developer kit.

```
ifx@ifxhost:~$ lspci
0004:00:00.0 PCI bridge: NVIDIA Corporation Device 1ad1 (rev a1)
0004:01:00.0 Network controller: Anchor Chips Inc. Device bd31 (rev 01)
```

6.6 Compile the driver

The steps in this section is already taken care of as a part of image generation; however, you can patch the open source brcmfmac with the patch provided - *SoftwareFiles.zip/patches/ifx_brcmfmac.patch*

6.7 Load the FMAC driver and firmware

Each device in the network should have a unique WLAN MAC address configured. So, before loading the FMAC driver on each device, configure them with a unique WLAN MAC address.

1. Go to *cyw5557x/wifi/fw* and edit the NVRAM file to set the Wi-Fi MAC address. Make sure that you use the correct NVRAM file for your AIROC™ CYW5557x Wi-Fi & Bluetooth® combo chip. The following snippet shows the CYW955572M2IPA1 board as reference.

```
ifx@ifxhost:~/cyw5557x/wifi/fw$ cat cyw955572m2ipa1_rev2.56.txt
## Summary
# Board Name : cyw955572m2ipa1
# SSID : 0x85f
# All parameters were copied from cyw955572fcipa_rev2.54.txt 083121 version
# TSSI cal value updated
#
NVRAMRev=$Rev$
sromrev=11
boardrev=0x1256
boardtype=0x085f
boardflags=0x00400001
boardflags2=0x00800000
boardflags3=0x40002100
#boardnum=57410
macaddr=00:90:4c:2d:80:01
```

2. Use the following command to load the WLAN FMAC driver and firmware. This also takes care of configuring the TCP/UDP network tuning parameters for achieving better throughput.

```
 $ sudo ./start.sh <Wi-Fi firmware path> <Wi-Fi nvram path> <Wi-Fi network
IP address>
```

The full command syntax is given below. All the command arguments are optional. If the arguments are not specified, the script will assign the default values (*fw_prod.trxse* for Wi-Fi firmware, *cyw955572m2ipa1_rev2.56.txt* for Wi-Fi NVRAM file, *192.168.1.122* for Wi-Fi network IP address). In the real scenario of WLAN SoftAP and STA, you would want to set unique IP address for the wlan0 interface on both SoftAP and STA. Therefore, make sure that the IP address being passed for SoftAP and STA are not the same.

Note: Set the IP address for the wlan0 interface in such a way that it is not part of the same subnet as that of the primary eth0 interface. For example, if the primary eth0 interface has the IP address 192.168.0.26, then set the wlan0 IP address with 192.168.1.xxx. This way, you can prevent possible IP collisions with the primary interface's network.

Syntax:

```
 $ sudo ./start.sh <Wi-Fi firmware path> <Wi-Fi nvram path> <Wi-Fi network
IP address>
```

For example:

```
$ sudo ./start.sh fw_prod.trxse cyw955572m2ipa1_rev2.56.txt 192.168.1.122
```

6.8 Configure the SoftAP

hostapd is the recommended user-space Linux daemon to configure the WLAN device as a SoftAP.

It is important to understand how to pass the configuration file for hostapd. The default path for the hostapd configuration file is */home/ifx/cyw5557x/wifi/hostapd/*.

```
ifx@ifxhost:~/cyw5557x/wifi/hostapd$ ls -lrt
total 9164
-rwxrwxrwx 1 root root     419 Sep  1 09:35 hostapd_wpa2-psk_5G_80.conf
-rwxrwxrwx 1 root root     382 Sep  1 09:35 hostapd_open_5G_80.conf
-rwxrwxrwx 1 root root     236 Sep  1 09:35 hostapd.envar
-rwxrwxrwx 1 root root  463696 Sep  1 09:35 hostapd_cli
-rwxrwxrwx 1 root root 8900584 Sep  1 09:35 hostapd
```

Select the configuration file to be used by the hostapd in the file *hostapd.envar* as shown below:

```
# Start the hostapd with Open network security configuration,
# and operates in 5GHz band with 80MHz bandwidth.
HOSTAPD_CONFIG_FILE=hostapd_open_5G_80.conf
```

By default, the *hostapd_open_5G_80.conf* file is selected, which is configured to start with an open network security on the 5-GHz channel with 80-MHz bandwidth with SSID *JETSON_AP_AX*. However, you can choose a different configuration or create your own hostapd configuration file and update your selection in file *hostapd.envar*.

Use the following Linux SystemD commands to control the operation of hostapd:

1. Start hostapd service:

```
$ sudo systemctl start hostapd@wlan0.service
```

2. Stop hostapd service:

```
$ sudo systemctl stop hostapd@wlan0.service
```

3. Restart hostapd service:

```
$ sudo systemctl restart hostapd@wlan0.service
```

4. Check the status of hostapd service:

```
$ sudo systemctl status hostapd@wlan0.service
```

Sample output:

```
ifx@ifxhost:~/cyw5557x/wifi$ sudo systemctl start hostapd@wlan0.service
ifx@ifxhost:~/cyw5557x/wifi$ sudo systemctl status hostapd@wlan0.service
● hostapd@wlan0.service - IFX: hostapd for wlan0
   Loaded: loaded (/etc/systemd/system/hostapd@wlan0.service; disabled;
vendor preset: enabled)
   Active: active (exited) since Wed 2022-09-07 11:42:12 UTC; 2min 29s ago
  Process: 5950 ExecStart=/home/ifx/cyw5557x/wifi/systemd_start_hostapd.sh
$HOSTAPD_CONFIG_FILE (code=exited, status=0/SUCCESS)
 Main PID: 5950 (code=exited, status=0/SUCCESS)
    Tasks: 1 (limit: 8924)
  Memory: 4.5M
   CGroup: /system.slice/system-hostapd.slice/hostapd@wlan0.service
           └─10846 /home/ifx/cyw5557x/wifi/hostapd/hostapd -B
/home/ifx/cyw5557x/wifi/hostapd/hostapd_open_5G_80.conf -dd

Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: nl80211: TX queue
param set: queue=2 aifs=3 cw_min=15 cw_max=63 burst_time=0 --> res=-95
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: Failed to set TX
queue parameters for queue 2.
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: nl80211: TX queue
param set: queue=3 aifs=7 cw_min=15 cw_max=1023 burst_time=0 --> res=-95
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: Failed to set TX
queue parameters for queue 3.
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: wlan0: interface
state UNINITIALIZED->ENABLED
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: wlan0: AP-ENABLED
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: wlan0: Setup of
interface done.
Sep 07 11:42:12 ifxhost systemd_start_hostapd.sh[5950]: ctrl_iface not
configured!
Sep 07 11:42:12 ifxhost sudo[10834]: pam_unix(sudo:session): session closed
for user root
Sep 07 11:42:12 ifxhost systemd[1]: Started IFX: hostapd for wlan0.
```

Verify the Hostapd started successfully by checking the state of Hostapd using *hostapd_cli* utility.

```
ifx@ifxhost:~$ sudo hostapd_cli status | grep state
state=ENABLED
ifx@ifxhost:~$ sudo hostapd_cli status | grep ssid
bssid[0]=00:90:4c:2d:80:01
ssid[0]=JETSON_AP_AX
```

5. To start WPA3 SoftAP, navigate to */home/ifx/cyw5557x/wifi* folder and run the script as shown below:

```
ifx@ifxhost:~/cyw5557x/wifi$ sudo ./wpa3_ap.sh
```

This starts Wi-Fi AP in WPA3-Personal security mode with SSID *JETSON_AP_WPA3*.

6.9 Configure the STA

wpa_supplicant is the recommended user-space Linux daemon to handle the WLAN authentication, association, disassociation, and deauthentication processes for STAs.

Like hostapd, it is important to understand how to pass configuration files for controlling the WPA supplicant. The default path for the WPA supplicant configuration file is */home/ifx/cyw5557x/wifi/wpa_supplicant/*.

```
ifx@ifxhost:~/cyw5557x/wifi/wpa_supplicant$ ls -lrt
total 13880
-rwxrwxrwx 1 root root      425 Sep  1 09:35 wpa_supplicant_wpa2-psk.conf
-rwxrwxrwx 1 root root      418 Sep  1 09:35 wpa_supplicant.envar
-rwxrwxrwx 1 root root 13566056 Sep  1 09:35 wpa_supplicant
-rwxrwxrwx 1 root root   620904 Sep  1 09:35 wpa_cli
-rwxrwxrwx 1 root root      310 Sep  1 09:35 p2p_supplicant.conf
-rwxrwxrwx 1 root root      352 Sep 7  09:38 wpa_supplicant.conf
```

1. Select the configuration file to be used by the WPA Supplicant in file *wpa_supplicant.envar* as shown below:

```
# Start the WPA supplicant with Open network security configuration.
WPA_SUPPLICANT_CONFIG=wpa_supplicant.conf
```

By default, the *wpa_supplicant.conf* file is selected, which is configured to start with an open network security. However, you can choose a different configuration or create your own WPA supplicant configuration file and update your selection in the file *wpa_supplicant.envar*.

Note: Check all the settings in the configuration file set in the wpa_supplicant.envar before proceeding to the following steps.

Use the following Linux SystemD commands to control the operation of WPA supplicant:

2. Start WPA supplicant service:

```
$ sudo systemctl start wpa_supplicant@wlan0.service
```

3. Stop WPA supplicant service:

```
$ sudo systemctl stop wpa_supplicant@wlan0.service
```

4. Restart WPA Supplicant service:

```
$ sudo systemctl restart wpa_supplicant@wlan0.service
```

5. Check the status of WPA Supplicant service:

```
$ sudo systemctl status wpa_supplicant@wlan0.service
```

Sample output

```
$ sudo systemctl start wpa_supplicant@wlan0.service

$ sudo systemctl status wpa_supplicant@wlan0.service

ifx@ifxhost:~/cyw5557x/wifi/wpa_supplicant$ sudo systemctl status
wpa_supplicant@wlan0.service

● wpa_supplicant@wlan0.service - IFX: WPA supplicant for wlan0

     Loaded: loaded (/etc/systemd/system/wpa_supplicant@wlan0.service;
disabled; vendor preset: enabled)

     Active: active (exited) since Wed 2022-09-07 12:06:54 UTC; 36s ago

    Process: 5143
ExecStart=/home/ifx/cyw5557x/wifi/systemd_start_supplicant.sh
$WPA_SUPPLICANT_CONFIG (code=exited, status=0/SUCCESS)

   Main PID: 5143 (code=exited, status=0/SUCCESS)

      Tasks: 1 (limit: 8924)

     Memory: 5.8M

     CGroup: /system.slice/system-
wpa_supplicant.slice/wpa_supplicant@wlan0.service

             └─5151 /home/ifx/cyw5557x/wifi/wpa_supplicant/wpa_supplicant -
iwlan0 -Dnl80211 -c/var/run/wpa_supplicant.conf -
m/var/run/p2p_supplicant.conf -puse_p2p_group_interface=1p2p_device=1 -
e/var/run/entropy.bin -f/var/run/p2p_supp.log -ddddd -t -B


Sep 07 12:06:54 ifxhost systemd[1]: Starting IFX: WPA supplicant for
wlan0...

Sep 07 12:06:54 ifxhost sudo[5146]:     root : TTY=unknown ; PWD=/ ;
USER=root ; COMMAND=/home/ifx/cyw5557x/wifi/wpa_supplicant/wpa_supplicant -
iwlan0 -Dnl80211 -c/var/run/wpa_supplicant.conf -
m/var/run/p2p_supplicant.conf -puse_p2p_group_interface=1p2p_device=1 -
e/var/run/entropy.bin -f/var/run/p2p_supp.log -ddddd -t -B

Sep 07 12:06:54 ifxhost sudo[5146]: pam_unix(sudo:session): session opened
for user root by (uid=0)

Sep 07 12:06:54 ifxhost sudo[5146]: pam_unix(sudo:session): session closed
for user root

Sep 07 12:06:54 ifxhost systemd[1]: Finished IFX: WPA supplicant for wlan0.
```

6. Verify that the WPA supplicant has associated to the SoftAP using *wpa_cli* utility:

```
ifx@ifxhost:~/cyw5557x/wifi/wpa_supplicant$ sudo wpa_cli status
Selected interface 'wlan0'
bssid=00:90:4c:2d:80:01
freq=5180
ssid=JETSON_AP_AX
id=0
mode=station
wifi_generation=6
pairwise_cipher=NONE
group_cipher=NONE
key_mgmt=NONE
wpa_state=COMPLETED
ip_address=192.168.1.68
p2p_device_address=02:90:4c:2d:80:68
address=00:90:4c:2d:80:68
uuid=386a0c89-6f9f-50c2-939d-e05145f3f93f
ieee80211ac=1
```

7. Disconnect from the SoftAP:

```
$ sudo wpa_cli disconnect
```

8. Reconnect to the SoftAP:

```
$ sudo wpa_cli reconnect
```

To associate a Wi-Fi station to WPA3-Personal Access Point, navigate to the */home/ifx/cyw5557x/wifi/* folder, modify the SSID you want to connect to, and run the script as shown below:

```
ifx@ifxhost:~/cyw5557x/wifi$ sudo ./wpa3_sta.sh
```

6.10 Ping the SoftAP and STA

Ping the STA and SoftAP to check if they are connected.

1. From STA, ping *<IP address of the SoftAP>*

   `$ ping -I wlan0 <IP address of softAP>`

2. From the SoftAP, ping *<IP address of the STA>*

   `$ ping -I wlan0 <IP address of STA>`

Note:

1. *Verify that the MAC address of the wireless interface of STA and AP are not same.*
2. *Ensure there is no IP address conflict with the Ethernet interface.*

3. *Verify the wireless interface name using the* `ifconfig` *command.*

6.11 Test the throughput

Throughput test can also be done using two WLAN devices.

1. Device 1 can be configured as a SoftAP by following the instructions mentioned in **Configure the SoftAP**. If the script is used with the default parameters, it will set the device up in the 5-GHz band with 80-MHz bandwidth (channel 36).
2. Device-2 can be configured as STA by following the instructions mentioned in **Configure the STA**.

TCP throughput testing commands	
Device-1-TCP Server (hosted on SoftAP interface)	`iperf -s -f m -i 1 -w 8m -l 8k`
Device-2-TCP Client (hosted on STA interface)	`iperf -c <Server's WLAN IP address> -i 1 -f m -t 30 -w 8m -l 8k`

Note: Adjust the TCP window size or try setting it to 256k (i.e., "-w 256k" in the iperf TCP server and client commands) when you see the throughput dropping to zero.

UDP throughput testing commands	
Device-1-UDP Server (hosted on SoftAP interface)	`iperf -s -u -i 1`
Device-2-UDP Client (hosted on STA interface)	`iperf -c <Server's WLAN IP address> -u -i 1 -f m -t 30 -b 1200m`

6.12 Bluetooth® release package

6.12.1 Setting up BlueZ for Linux environment

Do the following to install the BlueZ stack on the Jetson Xavier NX developer kit:

1. Download the latest version of BlueZ from **here** to the Jetson Xavier NX board on a folder of your choice.

Note: At the time of this documentation release, BlueZ-5.61 version is available.

2. Navigate to the folder where you downloaded the BlueZ. Follow these commands to extract it.

```
> tar -xf bluez-5.61.tar.xz
> cd bluez-5.61
```

3. Run the following command from the terminal to check for all the dependencies required to compile BlueZ:

```
> ./configure
```

For example, you may get errors similar to the one shown below:

```
Configure: error : D-Bus >= 1.6 is required.
```

To fix such errors, install libraries such as *libdbus-1-dev, libudev-dev*, *libical-dev,* and *libreadline-dev* using the following command:

```
sudo apt-get install libdbus-1-dev libudev-dev libical-dev libreadline-dev
```

Run the following command to install the required library if you get a "configure: error rst2man is required":

```
>     sudo apt-get install python-docutils
```

4. After resolving dependency issues, compile the BlueZ stack with the following command:

```
> make
```

Note: The above make command should complete without any errors.

If the compile operation is successful, you get the *bluetoothd* binary in */usr/local/libexec/Bluetooth.*

5. Install *bluetoothd* with the following command:

```
> sudo make install
```

6. Copy the installed bluetoothhd folders to the */usr/lib/Bluetooth* folder using the following command:

```
> sudo cp /usr/local/libexec/bluetooth/bluetoothd /usr/lib/bluetooth
```

Note: If a "Text file busy" message is displayed during this copy operation, this means that there is already a *bluetoothd* running. This process must be killed.

Get the PID number of this process using `pidof bluetoothd` and then kill it using `sudo kill -9 pidno` (pidno to be replaced by the PID of *bluetoothd*)

7. After all the above operations are successful, start *bluetoothd* by executing the following command:

```
> sudo systemctl start bluetooth
```

6.12.2 Download the Bluetooth® firmware to patch RAM

The Bluetooth® firmware programming is done automatically using the scripts present in the "*cyw5557x/scripts/*" folder (discussed in detail as follows). Here for information purpose, we are describing the Timing diagram and sequence involved in Bluetooth® firmware programming to Patch RAM.

1. Pull the UART CTS line of CYW5557x low and toggle the bt_reg_on pin from LOW to HIGH.
   This ensures that the device is in auto baud mode.
2. Send the HCI Reset command to the device.
   This will auto detect the baud rate.
3. Initiate the firmware download within three seconds of completing the previous step.
4. The following timing diagram is to be followed for Bluetooth® firmware downloading to Patch RAM.

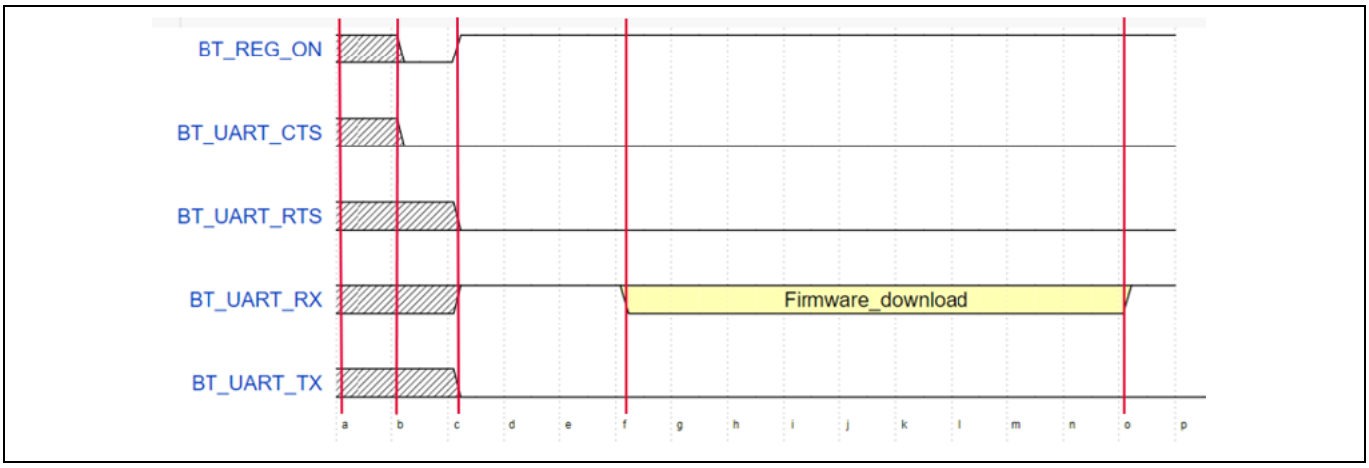


Figure 6 Timing diagram for Bluetooth® firmware downloading.

Do the following as indicated in this diagram:

1. Drive BT_UART_CTS from HIGH to LOW, see (b).
2. Drive the BT_REG_ON from LOW to HIGH, see (c).
3. Initiate the Bluetooth® firmware download within three seconds from the previous step; see (c) to (f) in the diagram.

Note:

1. *The UART CTS of cyw5557x/CYW8x570 is connected to UART RTS of Jetson Nano Linux host board and BT_REG_ON of cyw5557x/CYW8x570 is connected to GPIO3_PI.02(GPIO66) of Jetson Nano Linux host board.*

As mentioned in the **Use a pre-configured SD card image** and **Generate the flashing tool** sections, *cyw5557x/BT/* and *cyw5557x/scripts/* folders are created automatically in the home directory. These folders contain all files required for the Bluetooth® device.

The Bluetooth® release package consists of the following resources:

```
cyw5557x
|
+--script
|      |
|      bt_autobaud.sh
|      bt_boot.sh
|      bt_load.sh
|      bt_read_addr.sh
|      bt_reg_onoff.sh
|      bt_uart_cts.sh
|      include.sh
|      mbt
|      Readme
|
+--BT
|    |
|    +--CYW55560A1_001.002.087
|                  |
|                  +--fcbga_iPA_dLNA_ANT0
|                  |          |
|                  |          CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_dLNA_ANT0.hcd
|                  |
|                  +--fcbga_iPA_dLNA_BTANT
|                  |          |
|                  |          CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_dLNA_BTANT.hcd
|                  |
|                  +--fcbga_iPA_sLNA_ANT0
|                  |          |
|                  |          CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_sLNA_ANT0.hcd
```

Figure 7 Bluetooth® and scripts package structure

The following script files are available in the *scripts* folder:

Script name	Description
bt_uart_cts.sh	Controls the GPIO signal which is mapped to the CTS line of the CYW5557x device.
bt_boot.sh	Boots Bluetooth® as normal mode. This script must run on every power cycle of the platform.
bt_audobaud.sh	Transitions the Bluetooth® chip into “PatchRam FW” download mode.
include.sh	Contains the environment used by the other scripts.
bt_reg_onoff.sh	Controls the GPIO signal which is mapped to BT_REG_ON so that it can control the power to the Bluetooth® chip.
bt_load.sh	Triggers programming of the Bluetooth® firmware based on the antenna selected.

In addition to these script files, the *scripts* folder also contains the Manufacturing Bluetooth® test tool (MBT) binary. This tool is used to send/receive commands to/from the Bluetooth® chip.

The firmware version of Bluetooth® is CYW5557XA1_001.002.087. Navigate to the scripts folder using the following command:

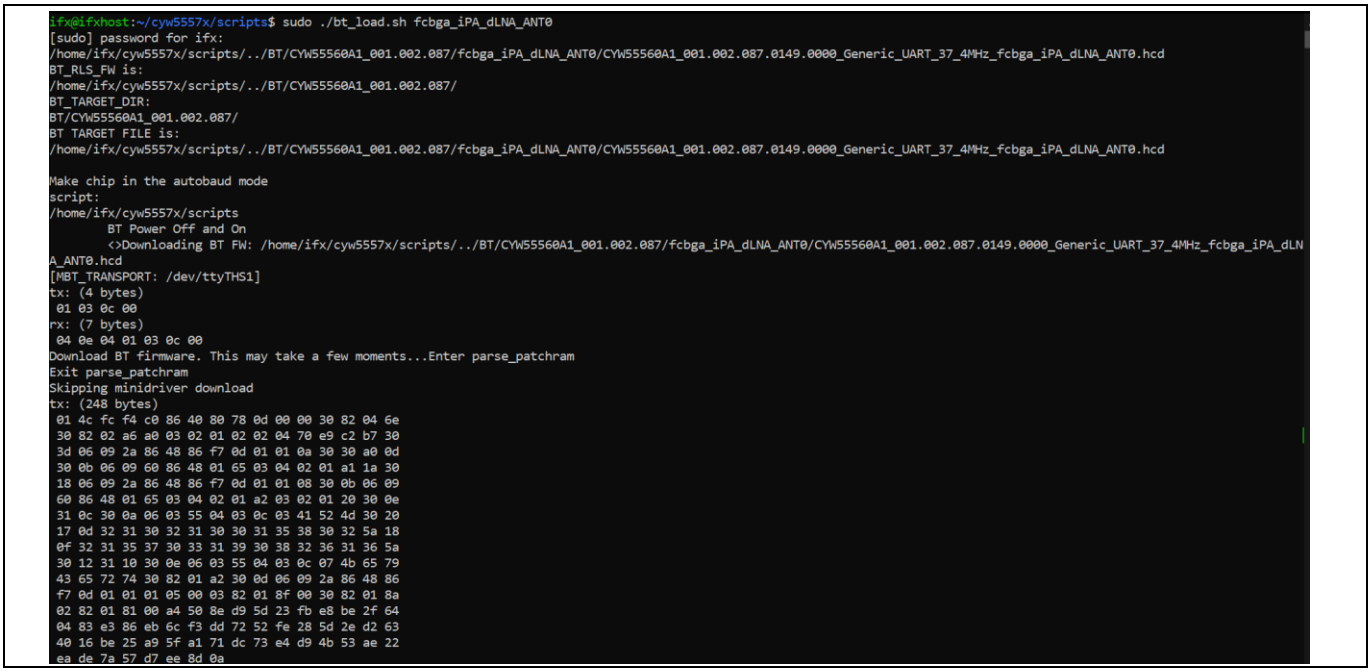
```
> cd ~/cyw5557x/scripts/
```

The Bluetooth® firmware contains three different co-existence antenna configurations.

6.12.2.1 Use antenna configuration dLNA_ANT0

Navigate to the *scripts* folder and run the following command to use dLNA_ANT0:

```
>     sudo ./bt_load.sh fcbga_iPA_dLNA_ANT0
```



Note: *No CY5557X board rework is required.*

6.12.2.2 Use antenna configuration sLNA_ANT0

Navigate to the *scripts* folder and run the following command to use sLNA_ANT0:

```
>     sudo ./bt_load.sh fcbga_iPA_sLNA_ANT0
```

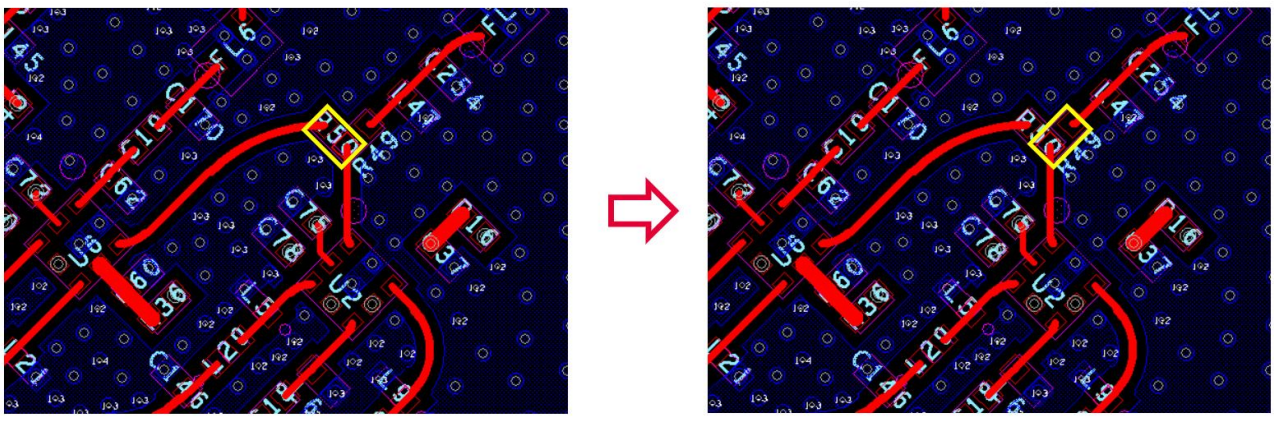
```
ifx@ifxhost:~/cyw5557x/scripts$ sudo ./bt_load.sh fcbga_iPA_sLNA_ANT0
BT_RLS_FW is:
/home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/
BT_TARGET_DIR:
BT/CYW55560A1_001.002.087/
BT TARGET FILE is:
/home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/fcbga_iPA_sLNA_ANT0/CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_sLNA_ANT0.hcd

Make chip in the autobaud mode
script:
/home/ifx/cyw5557x/scripts
       BT Power Off and On
       <>Downloading BT FW: /home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/fcbga_iPA_sLNA_ANT0/CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_sLN
A_ANT0.hcd
[MBT_TRANSPORT: /dev/ttyTHS1]
tx: (4 bytes)
 01 03 0c 00
rx: (7 bytes)
 04 0e 04 01 03 0c 00
Download BT firmware. This may take a few moments...Enter parse_patchram
Exit parse_patchram
Skipping minidriver download
tx: (248 bytes)
 01 4c fc f4 c0 86 40 80 78 0d 00 00 30 82 04 6e
 30 82 02 a6 a0 03 02 01 02 02 04 70 e9 c2 b7 30
 3d 06 09 2a 86 48 86 f7 0d 01 01 0a 30 30 a0 0d
 30 0b 06 09 60 86 48 01 65 03 04 02 01 a1 1a 30
 18 06 09 2a 86 48 86 f7 0d 01 01 08 30 0b 06 09
 60 86 48 01 65 03 04 02 01 a2 03 02 01 20 30 0e
 31 0c 30 0a 06 03 55 04 03 0c 03 41 52 4d 30 20
 17 0d 32 31 30 32 31 30 30 31 35 38 30 32 5a 18
 0f 32 31 35 37 30 33 31 39 30 38 32 36 31 36 5a
 30 12 31 10 30 0e 06 03 55 04 03 0c 07 4b 65 79
 43 65 72 74 30 82 01 a2 30 0d 06 09 2a 86 48 86
 f7 0d 01 01 01 05 00 03 82 01 8f 00 30 82 01 8a
 02 82 01 81 00 a4 50 8e d9 5d 23 fb e8 be 2f 64
 04 83 e3 86 eb 6c f3 dd 72 52 fe 28 5d 2e d2 63
 40 16 be 25 a9 5f a1 71 dc 73 e4 d9 4b 53 ae 22
 ea de 7a 57 d7 ee 8d 0a
```

Note: *No CY5557X board rework is required.*

6.12.2.3 To use antenna configuration dLNA_BTANT

1. Follow these steps to perform the hardware rework (in CY5557x board) in order to use dLNA_BTANT antenna configurations.

1. Top Layer : Remove R50 and Insert R49 0ohm



Note: *R49 and R50 are both 'zero' ohm.*

2. Navigate to the *scripts* folder and run the following command to use dLNA_BTANT:

```
> sudo ./bt_load.sh fcbga_iPA_dLNA_BTANTy
```

```
ifx@ifxhost:~/cyw5557x/scripts$ sudo ./bt_load.sh fcbga_iPA_dLNA_BTANT
BT_RLS_FW is:
/home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/
BT_TARGET_DIR:
BT/CYW55560A1_001.002.087/
BT TARGET FILE is:
/home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/fcbga_iPA_dLNA_BTANT/CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_dLNA_BTANT.hcd

Make chip in the autobaud mode
script:
/home/ifx/cyw5557x/scripts
       BT Power Off and On
       <>Downloading BT FW: /home/ifx/cyw5557x/scripts/../BT/CYW55560A1_001.002.087/fcbga_iPA_dLNA_BTANT/CYW55560A1_001.002.087.0149.0000_Generic_UART_37_4MHz_fcbga_iPA_dL
NA_BTANT.hcd
[MBT_TRANSPORT: /dev/ttyTHS1]
tx: (4 bytes)
 01 03 0c 00
rx: (7 bytes)
 04 0e 04 01 03 0c 00
Download BT firmware. This may take a few moments...Enter parse_patchram
Exit parse_patchram
Skipping minidriver download
tx: (248 bytes)
 01 4c fc f4 c0 86 40 80 78 0d 00 00 30 82 04 6e
 30 82 02 a6 a0 03 02 01 02 02 04 70 e9 c2 b7 30
 3d 06 09 2a 86 48 86 f7 0d 01 01 0a 30 30 a0 0d
 30 0b 06 09 60 86 48 01 65 03 04 02 01 a1 1a 30
 18 06 09 2a 86 48 86 f7 0d 01 01 08 30 0b 06 09
 60 86 48 01 65 03 04 02 01 a2 03 02 01 20 30 0e
 31 0c 30 0a 06 03 55 04 03 0c 03 41 52 4d 30 20
 17 0d 32 31 30 32 31 30 30 31 35 38 30 32 5a 18
 0f 32 31 35 37 30 33 31 39 30 38 32 36 31 36 5a
 30 12 31 10 30 0e 06 03 55 04 03 0c 07 4b 65 79
 43 65 72 74 30 82 01 a2 30 0d 06 09 2a 86 48 86
 f7 0d 01 01 01 05 00 03 82 01 8f 00 30 82 01 8a
 02 82 01 81 00 a4 50 8e d9 5d 23 fb e8 be 2f 64
 04 83 e3 86 eb 6c f3 dd 72 52 fe 28 5d 2e d2 63
 40 16 be 25 a9 5f a1 71 dc 73 e4 d9 4b 53 ae 22
 ea de 7a 57 d7 ee 8d 0a
```

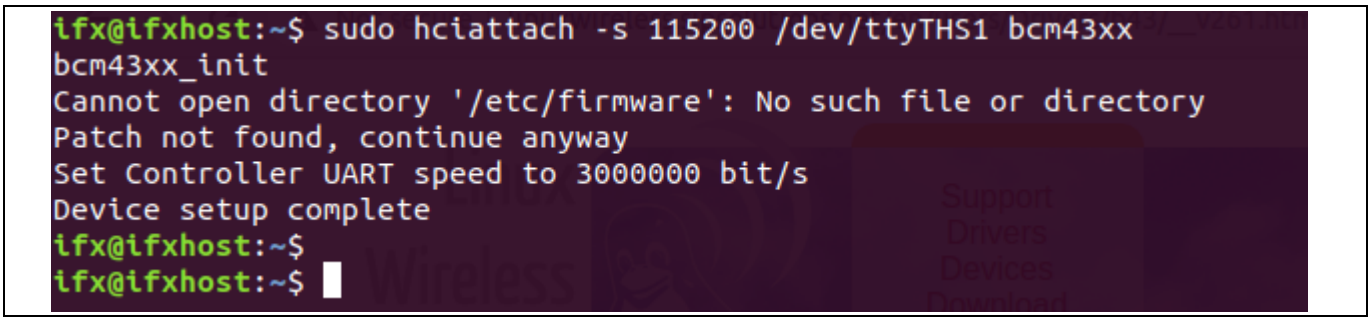
3. Wait for the download to complete. It may take approximately 10-15 seconds. If success, messages such as the following appear: This is common for all antenna configurations.

```
rx: (7 bytes)
 04 0e 04 01 4c fc 00
tx: (8 bytes)
 01 4e fc 04 ff ff ff ff
rx: (7 bytes)
 04 0e 04 01 4e fc 00
Exit proc_patchram
tx: (4 bytes)
 01 03 0c 00
rx: (7 bytes)
 04 0e 04 01 03 0c 00
Current state: Completed successfully
ifx@ifxhost:~/cyw5557x/scripts$
```

6.12.3 HCI UART bring up

After all earlier steps are performed, attach the BlueZ stack on to the UART. Run the following commands:

```
>      sudo hciattach -s 115200 /dev/ttyTHS1 bcm43xx
```



If the above message is displayed after running the `hciattach` command, the installation of BlueZ and Firmware download is successfully done and the Jetson Xavier NX board, along with CYW5557x, is ready for use with standard BlueZ tools such as *bluetoothctl* and *hcitool*.

Revision history

Date	Version	Description
2021-02-17	**	New quick start guide for CYW55572.
2021-05-31	*A	Added new section 6.4 Generate the flashing tool. Added new section 4.3 Generate a massflash image using script. Updated sections 6.8 and 6.9. Updated various sections based on new massflash image. Added a new section 6.12 Bluetooth® release package.
2021-09-09	*B	Replaced instructions with the Jetson Xavier NX developer kit from NVIDIA instead of Jetson TX2. Added sections **2.2**, **4**, and **6.2**. Changed usage from CYW55572 to CYW5557x. Updated sections **6.8** and **6.9** to use system infrastructure instead of manual script files. Changed the usage of FCIPA to M2IPA and updated the board images accordingly. Updated **Figure 1** and **Figure 2**. Added notes inside section **6.11** Added instructions for BlueZ support, and Bluetooth firmware installation guide on CYW5557x.
2021-12-22	*C	Added steps to use Bluetooth® firmware with Co-Existance Antenna configurations in section **6.13**
2022-09-29	*D	Updated document for Ebirah release, added Bluetooth® firmware programming diagram

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2022-09-29
Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Go to www.infineon.com/support

Document reference
002-31595 Rev. *D

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”) .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.