Thundercomm
AIOT ENABLER

Qualcomm Robotics SDK Manager

User Guide

Rev. O
Oct 14, 2024

Revision History

Revision	Date	Description
A	Jul 30, 2020	Initial release.
B	Oct 27, 2020	• Add information on Windows 10 Professional and Windows 10 Enterprise operation. • Update TROUBLESHOOTING.
C	Dec 14, 2020	Optimize file construction.
D	Jul 29, 2022	Add contents to make this user guide compatible with RB6.
E	Jan 12, 2023	• Add disk requirements in Chapter 2. • Update Step 5 in Chapter 5. • Update Step 4 in Chapter 6. • Update Step 9 - 4) in Chapter 7.
F	Mar 31, 2023	Add contents to make this document compatible with RB2 platform: • Update Chapter 1. Overview. • Update the download link of SDK Manager in Chapter 4. SDK Manager Download and Unzip. • Update Chapter 5. • Add Figure 5-3, Figure 5-4, Figure 7-3 and Figure 7-4. • Fix command font problem.
G	Apr 10, 2023	• Restructure the document. • Add RB5 LU2.0 relevant info. throughout this document: ▫ Chapter 1. Overview ▫ Chapter 3. System and Disk Requirements ▫ Chapter 4. Download SDK Manager ▫ Chapter 5. On Ubuntu Host ▫ Chapter 6. Generate Ubuntu Docker Image ▫ Step 4 and 5 in Chapter 7. On Windows 10 (64-bit) Host
H	Apr 19, 2023	Update the note at the beginning of Chapter 5. On Ubuntu Host.
I	May 08, 2023	Update Chapter 7. On Windows 10 (64-bit) Host.
J	June 09, 2023	• Restructure the document. • Update 4.1.1. OS version is recommended version. • Update 4.2. On Windows 10 (64-bit) Host. • Update Table 5-1. Troubleshooting information. • Update Table 6-1. For additional reference please refer to: .
K	Aug 21, 2023	• Update Chapter 1. Overview. • Update 3.1. OS requirements. • Update Chapter 4. SDK Manager Operation Process .

Revision	Date	Description
L	Dec 19, 2023	• Update the download address in Step 1 of Chapter 4. SDK Manager Operation Process . • Update the commands in Step 4 - 1) of Section 4.1.1. • Update the notes in Step 4 - 7) of Section 4.1.1. • Update in Step 9 - 1) and 3) of 4.2. On Windows 10 (64-bit) Host. • Update Table 5-1. Troubleshooting information.
M	Mar 14, 2024	• Update the following chapters/section: ▫ Chapter 1. Overview ▫ 3.1. OS requirements ▫ Chapter 4. SDK Manager Operation Process ▫ Chapter 6. Reference Documents • Add the following figures: ▫ Figure 4-5 ▫ Figure 4-6 ▫ Figure 4-12 ▫ Figure 4-13
N	Aug 05, 2024	• Update the following chapters/section: ▫ Chapter 1. Overview ▫ 3.1. OS requirements ▫ Chapter 4. SDK Manager Operation Process ▫ Chapter 6. Reference Documents • Add the following figures: ▫ Figure 4-7 ▫ Figure 4-15
O	Oct 14, 2024	Update the commands in 4.1.1. For recommended OS versions.

Table List

About This Document

- Illustrations in this documentation might look different from your product.
- Depending on the model, some optional accessories, features, and software programs might not be available on your device.
- Depending on the version of operating systems and programs, some user interface instructions might not be applicable to your device.
- Documentation content is subject to change without notice. Thundercomm makes constant improvements on the documentation of the products, including this guidebook.
- Function declarations, function names, type declarations, attributes, and code samples appear in a different format, for example, `cp armcc armcpp`.
- Code variables appear in angle brackets, for example, `<number>`.
- Button, tool, and key names appear in bold font, for example, click **Save** or press **Enter**.
- Commands to be entered appear in a different font; on the host computer use $ as shell prompt, while on the target device use # as shell prompt, for example,

```
$ adb devices
# logcat
```

- Part of the code that does not contain instructions appear in a different format, for example,

```
SUBSYSTEM=="usb", ATTR{idVendor}=="18d1", MODE="0777", GROUP="adm"
```

- Folders, path and files are formatted in italic, for example, *turbox_flash_flat.sh*.

Table of Contents

Chapter 1. Overview

SDK Manager provides a complete set of tools for generating and flashing the RBx and C6490 firmware, supporting systems including Ubuntu 16.04, Ubuntu 18.04, Ubuntu 20.04, Ubuntu 22.04, Windows 10 Professional (64-bit) and Windows 10 Enterprise (64-bit).

V4.2.0 supports the following products:

- C6490 Platform
- Robotics RB1 Platform
- Robotics RB2 Platform
- Robotics RB3 Gen2 Platform
- Robotics RB5 Platform
- Robotics RB5 Gen2 Platform
- Robotics RB5N (Non-Pop) Platform
- Robotics RB6 Platform

Chapter 2. Read This First

- To register a Thundercomm Account, go to http://www.thundercomm.com.
- Keep the internet connected during the image generation.
- The full process lasts for at least 40 minutes, depending on Internet speed.
- A working directory is needed to be built with the write and read permission in SDK Manager. For Docker container user, create your target directory under */home/hostPC/*.
- Docker Desktop is only supported on Windows 10 Professional (64-bit) and Windows 10 Enterprise (64-bit) system.
- Before flashing the full build, generate the image first.
- USB 3.0 port and USB 3.0 cable are recommended for flashing images.
- When flashing the device on a Linux host, run the command below before connecting the device to the host.

```
$ sudo systemctl stop ModemManager
```

- Plug in a USB device before starting Option 2 (**EDL** programming sequence), if an Ubuntu 18.04 host is running the SDK Manager by the Ubuntu 18.04 Docker.

Chapter 3. System and Disk Requirements

3.1. OS requirements

- **For RB5 Gen2 platform**
  - Recommended OS (Operating System): Ubuntu 22.04.
  - Alternatively, run an Ubuntu 22.04 Docker on a host of Ubuntu 16.04, Ubuntu 18.04, Ubuntu 20.04, Windows 10 Professional (64-bit), or Windows 10 Enterprise (64-bit) system.
- **For C6490, RB3 Gen2, RB5 LU2.0, and RB5N LU2.0 platforms**
  - Recommended OS (Operating System): Ubuntu 20.04.
  - Alternatively, run an Ubuntu 20.04 Docker on a host of Ubuntu 16.04, Ubuntu 18.04, Ubuntu 20.04, Windows 10 Professional (64-bit), or Windows 10 Enterprise (64-bit) system.
- **For RB1, RB2, RB5 LU1.0, and RB6 platforms**
  - Recommended OS: Ubuntu 18.04.
  - Alternatively, run an Ubuntu 18.04 Docker on a host of Ubuntu 16.04, Ubuntu 18.04, Ubuntu 20.04, Windows 10 Professional (64-bit), or Windows 10 Enterprise (64-bit) system.

3.2. Disk requirements

Make sure that the following minimum disk requirements are met.

- At least 1.5GB disk space to download a software version.
- It requires at least 50GB disk space to download LU resources and generate *system.img* with current release.

Chapter 4. SDK Manager Operation Process

Step 1. Download SDK Manager via the following link:

https://thundercomm.s3.ap-northeast-1.amazonaws.com/uploads/web/common/TC-sdkmanager-4.2.0.zip

Step 2. Unzip the SDK Manager file with the following command:

```
$ unzip TC-sdkmanager-x.x.x.zip
```

4.1. On Ubuntu Host

4.1.1. For recommended OS versions

➲ **NOTE:** It is required to observe the following requirements on package version.

- **For RB5 Gen2 Platform (recommended OS: Ubuntu 22.04)**
  - Required minimum package version: coreutils 8.32, fakechroot 2.20, fakeroot 1.28, kmod 29-1ubuntu1, libc6-arm64-cross 2.35, python 2.7.18, qemu-user-static 1:6.2+dfsg-2ubuntu6.21, udev 249.11-0ubuntu3.7, unzip 6.0, wget 1.21.2.
  - Run the following commands to create soft links:

```
$ sudo rm -rf /lib/ld-linux-aarch64.so.1
$ sudo ln -sf /usr/aarch64-linux-gnu/lib/ld-linux-aarch64.so.1 /lib/ld-linux-aarch64.so.1
$ sudo ln -sf /bin/bash /bin/sh
$ sudo apt-get install -y cron rsyslog rtkit chrony pulseaudio python2
$ sudo groupadd diag
```

- **For C6490, RB3 Gen2, RB5 LU2.0 and RB5N LU2.0 Platforms (recommended OS: Ubuntu 20.04)**
  - Required minimum package version: coreutils 8.30, fakechroot 2.19, fakeroot 1.24, kmod 27-1ubuntu2.1, libc6-arm64-cross 2.31, python 2.7.18, qemu-user-static 1:7.2+dfsg-5ubuntu1, udev 245.4-4ubuntu3.20, unzip 6.0, wget 1.20.3.
  - Run the following commands to create soft links:

```
$ sudo rm -rf /lib/ld-linux-aarch64.so.1
$ sudo ln -sf /usr/aarch64-linux-gnu/lib/ld-2.31.so /lib/ld-linux-aarch64.so.1
$ sudo ln -sf /bin/bash /bin/sh
$ sudo dpkg -P qemu-user-static
$ wget http://archive.ubuntu.com/ubuntu/pool/universe/q/qemu/qemu-user-static_6.2+dfsg-2ubuntu6_amd64.deb
$ sudo dpkg -i qemu-user-static_6.2+dfsg-2ubuntu6_amd64.deb
```

- **For RB5 LU1.0, RB6, RB1, RB2 Platforms (recommended OS: Ubuntu 18.04)**

  Required minimum package version: coreutils 8.28, fakechroot 2.19, fakeroot 1.22, kmod 24-1ubuntu3.2, libc6-arm64-cross 2.27, python 2.7.15, qemu-user-static 1:2.11+dfsg- 1ubuntu7.28, udev 237-3ubuntu10.42, unzip 6.0, wget 1.19.4.

Operation Procedure:

Step 1. Install the dependency libraries to the host computer:

```
$ sudo apt-get install coreutils fakechroot fakeroot \
  kmod libc6-arm64-cross  python2.7 qemu-user-static wget udev openssh-server
```

Step 2. Unzip *TC-sdkmanager-x.x.x.zip* and navigate to *TC-sdkmanager-x.x.x* directory from a terminal window, and install or re-install SDK Manager:

```
$ sudo dpkg -i tc-sdkmanager-vx.x.x_amd64.deb
```

Step 3. Launch SDK Manager.

```
$ sdkmanager
```

Step 4. Run SDK Manager.

1) Provide Thundercomm login credentials:

```
Thundercomm Account Checking ...
Enter your Thundercomm user email:
Enter your Thundercomm password:
```

2) To change the installation path, you should specify a working directory (for example, "*/home/user*") when prompted for a target directory. Then, enter the absolute target directory (default directory: */home/user*) where the SDK Manager will overwrite any existing files.

```
Enter absolute target directory for Image resources (overwrites existing files,
default: /home/user/):
```

➲ **NOTE:** Docker users must specify a working directory as */home/hostPC/[workingdirectory]*.

3) Enter the number of the selected product, for example, **1**.

```
Select your product:
1: RB1
2: RB2
3: RB3 Gen2
4: RB5
5: RB5 Gen2
6: RB5N (Non-Pop)
7: RB6
8: C6490
Select one number of product ( 1 | 2 | 3 ...) to continue with:
```

➲ **NOTE:** If the product is only compatible with a single platform, SDK Manager will automatically omit Step 4 - 4) and proceed directly to Step 4 - 5).

4) Enter the number of the available platform for Robotics RBx device, for example, **1**.

```
Choose a platform for Robotics RBx device
Enter 1 to use LU platform, 2 to use LE platform:
```

5) Enter the number of the available version for image repack, for example, **1**:

```
Checking current versions of release ...
Available versions:
1: QRB5165.x.x.x-xxxxxx
… …
Select one number of available version ( 1 | 2 | 3 ...) to continue with:
```

6) Enter **1** when the message below appears on your screen:

```
--------------------------------------------------------------------------------
SDK has been successfully set up and is ready to be used
Type 'help' for commands
--------------------------------------------------------------------------------
>
```

➲ **NOTES:**

- This step lasts for at least 40 minutes.

- Enter `help` for more information:

```
> help
commands:
help = Show usage help for LU platform
1 = Download LU resources and generate system.img with current release
2 = Flash full build (require system.img generation first)
q = exit sdk manager
```

7) The system images are successfully generated in the working directory with the following messages displayed:

```
--------------------------------------------------------------------------------
Move sparse images to full_build ...done
You may proceed to flash full_build to your device
--------------------------------------------------------------------------------
```

➲ **NOTES:**

- For Docker users, the system image is generated at */home/hostPC/[workingdirectory]*.
- In case of any error, please refer to Chapter 5. Troubleshoot.
- The default support of 6490DK is 8G DDR + 128G UFS.

Step 5. Disconnect the device from the computer, then follow the steps below to flash the full build:

➲ **NOTE:** When flashing the device on a Linux host, run the command below before connecting the device to the host.

```
$ sudo systemctl stop ModemManager
```

1) Power off the device by disconnecting the power cable and USB cable.

2) Press the **F_DL Key**.

3) Power on the device (required voltage: 12 V).

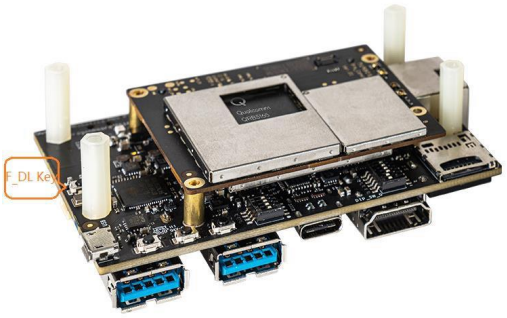

Figure 4-1. RB5 F_DL Key

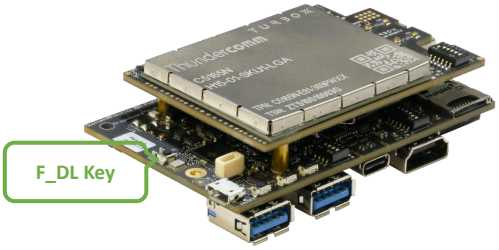

Figure 4-2. RB6/RB5N F_DL Key

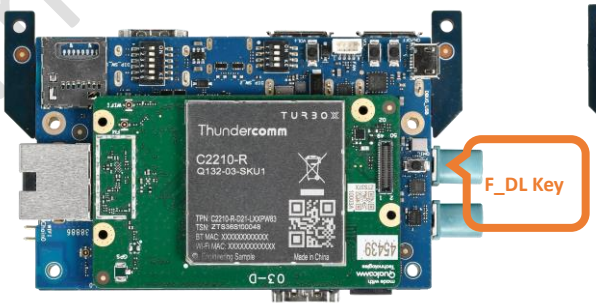

Figure 4-3. RB1 F_DL Key

Figure 4-4. RB2 F_DL Key

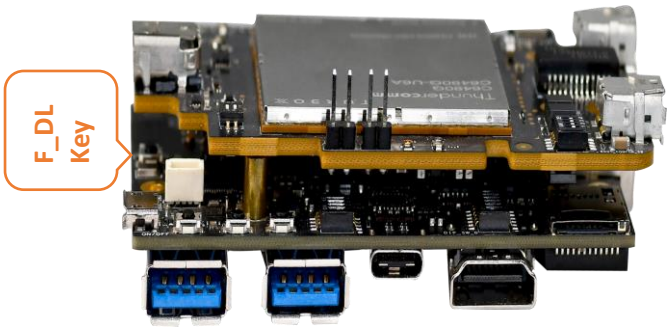

Figure 4-5.RB3 Gen2 F_DL Key

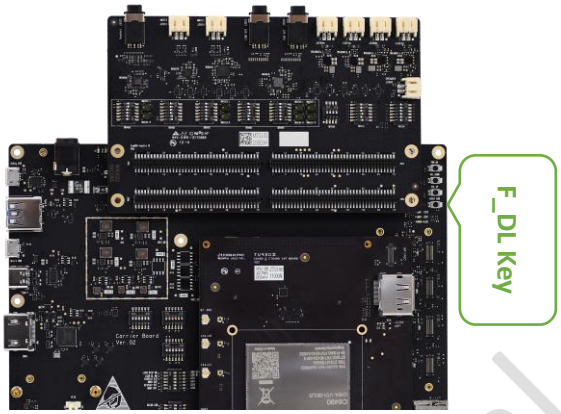

Figure 4-6.C6490 F_DL Key

Figure 4-7.RB5 Gen2 F_DL Key

4) Keep pressing the **F_DL Key** while connecting the board to your computer with a USB Type-C cable (This step will switch the device to **EDL** mode).

5) Release the **F_DL Key** after connecting the board to your computer.

6) Start flashing process from the SDK Manager with "`Flash full build`".

7) SDK Manager shall detect the device and start the flashing process automatically.

8) After the flashing process is complete, the board will reboot automatically. This may take some time.

```
Flashing image ... done
Waiting for device to reboot ...
* daemon not running; starting now at tcp:5037
* daemon started successfully
Waiting for boot up, time elapsed: 10s
Waiting for boot up, time elapsed: 20s
Waiting for boot up, time elapsed: 30s
Waiting for boot up, time elapsed: 40s
Waiting for boot up, time elapsed: 50s


========================================================================
RB5 device is ready to use.
Open another terminal and enter 'adb shell' to interact with your device.
========================================================================
```

Figure 4-8.Board Reboot

9) Once your device has completed the booting process, please open a new terminal window on your host computer and enter the following command:

```
$ adb wait-for-device shell
```

4.1.2. For other OS versions

➲ **NOTES:** Different OS versions require different Docker images.

- **RB5 Gen2 Platform**

  For an Ubuntu 16.04/18.04/20.04 host or other, an Ubuntu 22.04 Docker image is required.

- **C6490, RB3 Gen2, RB5 LU2.0 and RB5N LU2.0 Platforms**

  For an Ubuntu 16.04/18.04/22.04 host or other, an Ubuntu 20.04 Docker image is required.

- **RB1, RB2, RB5 LU1.0 and RB6 Platforms**

  For an Ubuntu 16.04/20.04/22.04 host or other, an Ubuntu 18.04 Docker image is required.

Operation Procedure:

Step 1. Install qemu-user-static, openssh-server, udev to the host PC.

```
$ sudo apt-get install qemu-user-static openssh-server udev -y
```

Step 2. To install Docker, refer to: https://docs.docker.com/engine/install/ubuntu/.

Step 3. Generate Ubuntu 18.04/20.04/22.04 docker image:

Unzip *TC-sdkmanager-x.x.x.zip* and navigate to the *TC-sdkmanager-x.x.x* directory from a new terminal window, then execute the following commands:

```
# Ubuntu terminal #
# Generate Ubuntu 18.04 docker image #
$ ln -sf Dockerfile_18.04 Dockerfile
$ sudo docker build -t ubuntu:18.04-sdkmanager .

# Generate Ubuntu 20.04 docker image #
$ ln -sf Dockerfile_20.04 Dockerfile
$ sudo docker build -t ubuntu:20.04-sdkmanager .

# Generate Ubuntu 22.04 docker image #
$ ln -sf Dockerfile_22.04 Dockerfile
$ sudo docker build -t ubuntu:22.04-sdkmanager .
-----------------------------------------------------------------------------
```

➲ **NOTES:**

- Make sure to include the space and full stop at the end of the command: **.**.
- Generated Docker image name: “ubuntu:18.04(20.04/22.04)-sdkmanager”.
- For users in mainland China, download the Ubuntu image by using the following commands, and then execute the above commands to build the sdkmanager image.

```
# Supports three version:ubuntu-18.04.tar, ubuntu-20.04.tar, ubuntu-22.04.tar
# For example: 22.04
$ wget https://thundercomm.s3.ap-northeast-1.amazonaws.com/uploads/web/sdkmanager/docker/ubuntu-22.04.tar
$ sudo docker load -i ubuntu-22.04.tar
-----------------------------------------------------------------------------
```

Step 4. Create a Docker container:

```
# Ubuntu 18.04 docker image #
$ sudo docker run -v /home/${USER}:/home/hostPC/ --privileged -v /dev/:/dev -v
/run/udev:/run/udev -d --name sdkmanager_container -p 36000:22
ubuntu:18.04-sdkmanager

# Ubuntu 20.04 docker image #
$ sudo docker run -v /home/${USER}:/home/hostPC/ --privileged -v /dev/:/dev -v
/run/udev:/run/udev -d --name sdkmanager_container -p 36000:22
ubuntu:20.04-sdkmanager

# Ubuntu 22.04 docker image #
$ sudo docker run -v /home/${USER}:/home/hostPC/ --privileged -v /dev/:/dev -v
/run/udev:/run/udev -d --name sdkmanager_container -p 36000:22
ubuntu:22.04-sdkmanager
--------------------------------------------------------------------------------
Host PC's /home/${USER} is mounted on /home/hostPC in Docker container
sdkmanager_container: container name
```

➲ **NOTE**: With the above commands, a Docker container name will be generated after `sdkmanager_container:`.

Step 5. Launch SDK Manager in the Docker container.

```
$ sudo docker exec -it sdkmanager_container sdkmanager
```

Step 6. Run SDK Manager. Refer to Step 4 of 4.1.1. OS version is recommended version.

Step 7. Disconnect the device from the computer and then proceed to flash the full build by following Step 5 of 4.1.1. OS version is recommended version.

Step 8. After your device has successfully booted up, open a new terminal window on the host computer and enter the following command:

```
$ adb wait-for-device shell
```

4.2. On Windows 10 (64-bit) Host

Step 1. To download Docker Desktop, go to:

https://hub.docker.com/editions/community/docker-ce-desktop-windows/

Step 2. Open **Dashboard** from the Docker notification menu to launch Docker Desktop.

Step 3. Open Windows PowerShell and enter `docker images` to verify docker installation.

➲ **NOTE:** If an error is displayed in the PowerShell console, it indicates that either the installation or operation of Docker Desktop has failed.

Step 4. Generate the Ubuntu docker image.

1) Unzip the *TC-sdkmanager-x.x.x.zip* file and navigate to the *TC/sdkmanager/x.x.x* directory from a Windows PowerShell.

2) Execute the following commands:

```
# Windows PowerShell #
# For RB1, RB2, RB5 LU1.0 and RB6 Platforms : Generate Ubuntu 18.04 docker image #
$ rm .\Dockerfile
$ cmd /c mklink Dockerfile Dockerfile_18.04
$ docker build -t ubuntu:18.04-sdkmanager .

# For C6490, RB3 Gen2, RB5 LU2.0 and RB5N LU2.0 Platform : Generate Ubuntu 20.04
docker image #
$ rm .\Dockerfile
$ cmd /c mklink Dockerfile Dockerfile_20.04
$ docker build -t ubuntu:20.04-sdkmanager .

# For RB5 Gen2 Platform : Generate Ubuntu 22.04 docker image #
$ rm .\Dockerfile
$ cmd /c mklink Dockerfile Dockerfile_22.04
$ docker build -t ubuntu:22.04-sdkmanager .
-------------------------------------------------------------------------------
```

➲ **NOTES**:

- Make sure to include the space and period at the end of the command.
- Generated docker image name: “ubuntu:18.04-sdkmanager” or “ubuntu:20.04-sdkmanager”.

Step 5. Create a docker container.

```
# For RB1, RB2, RB5 LU1.0 and RB6 Platforms : Generate Ubuntu 18.04 docker image #
$ docker run -it -d --name sdkmanager_container ubuntu:18.04-sdkmanager

# For C6490, RB3 Gen2,RB5 LU2.0 and RB5N LU2.0 Platform : Generate Ubuntu 20.04
docker image #
$ docker run -it -d --name sdkmanager_container ubuntu:20.04-sdkmanager

# For RB5 Gen2 Platform : Generate Ubuntu 22.04 docker image #
$ docker run -it -d --name sdkmanager_container ubuntu:22.04-sdkmanager
```

➲ **NOTE:** With the above commands, a docker container name can be generated after `sdkmanager_container`.

Step 6. Launch SDK Manager.

```
$ docker exec -it sdkmanager_container sdkmanager
```

Step 7. Run SDK Manager. Refer to Step 4 of 4.1.1. OS version is recommended version.

Step 8. Copy the full build from the Docker container to a Windows Host computer.

```
$ docker cp  sdkmanager_container:[target_directory]/[name_of_selected_release]/full_build [destionation path on Windows host PC]
--------------------------------------------------------------------------------
Example: docker cp sdkmanager_container:/home/hostPC/demo_0803/QRB5165.x.x.x-xxxxxx/full_build D:\
--------------------------------------------------------------------------------
```

Step 9. Flash the device.

1) Download the *thundercomm-tflash-windows.msi* file and install **Tflash** **(TurboX Flash)**.

2) Proceed with either option to check if your device is in EDL (Emergency Download) mode:

Option 1: Enter `adb reboot edl`.

Option 2: Press the **F_DL Key** to power on your device.

➲ **NOTE:** Check if the device has been recognized as Qualcomm HS-USB QLoader 9008 (COMx) in the Device Manager. If it is not recognized, it may be necessary to download and install the appropriate USB drivers.

3) Flash the full build with **Tflash**:

a) Launch **Tflash**.

b) Select "UFS" as the **storage type**.

c) Click the **Browse** button to select the programmer file (*prog_firehose_ddr.elf*) and load XML files. When prompted to select XML files, choose all XML files and all Patch files in the *full_build ufs* folder. Leave all other settings as default.

d) Disconnect the device from the computer and power it off.

e) Press the **F_DL Key**.

f) Power on the device (required voltage: 12 V).

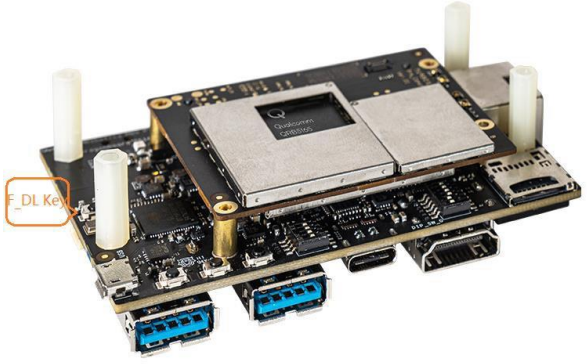

Figure 4-9.RB5 F_DL Key

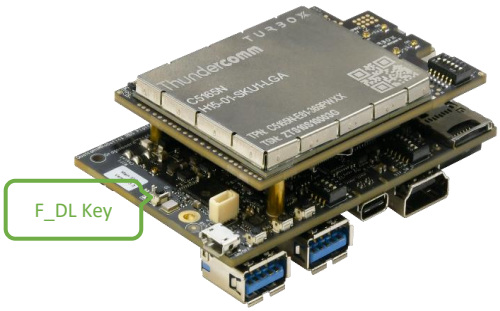

Figure 4-10. RB6/RB5N F_DL Key

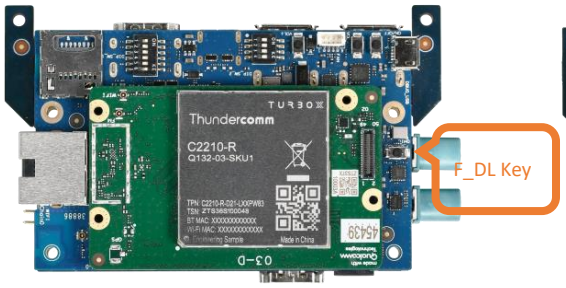

Figure 4-11.RB1 F_DL Key

Figure 4-12.RB2 F_DL Key

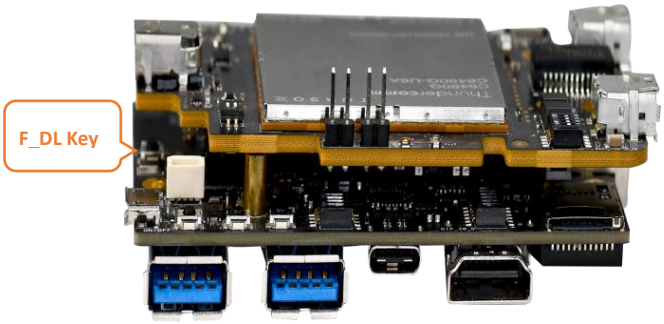

Figure 4-13.RB3 Gen2 F_DL Key

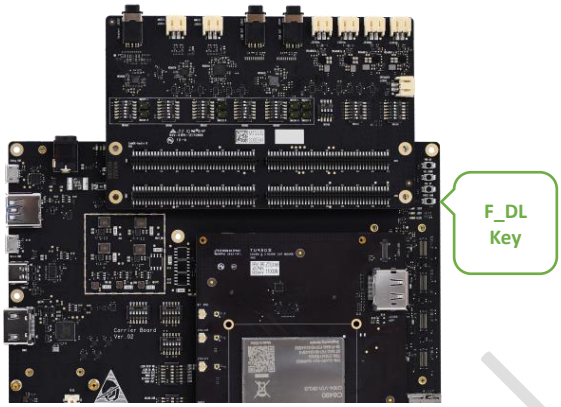

Figure 4-14.C6490 F_DL Key

Figure 4-15.RB5 Gen2 F_DL Key

g) Press and hold the **F_DL Key** while connecting the board to your computer with a USB Type-C cable.

➲ **NOTE:** This step will switch the device to EDL mode.

h) Release the **F_DL Key** after the board has been connected to your computer.

i) Select the device by clicking the **UpdatePort** button.

j) Start flashing by clicking the **Download** button that corresponds to your device port in the **Tflash** window. Upon completion of the flashing process, the board will reboot automatically.

➲ **NOTE:** This step may take some time.

Chapter 5. Troubleshoot

Refer to Table 5-1 for the solutions to problems that have definite symptoms.

Table 5-1. Troubleshooting information

Problem	Solution

Internet Timeout Issue:
Internet timeout issue may occur during the image generation process, such as “Unable to fetch”.
	Try to run **Command 1** again.
	`commands:`
	`help = Show usage help for XX platform`
	`1 = Download LU resources and generate system.img with current release`
	`2 = xxxx`
	`>1`

APT Source Issue	If the download fails, check the internet connection and the source list.
Device Boot Up Issue: SDK Manager cannot detect the device after reboot.	• If Ubuntu 18.04 is used on Docker, check whether `adb kill-server` is entered on the host PC before flashing image. • Reboot your device manually, open a terminal, then enter `adb shell`. • Check if any Debian packages are modified.
Process Issue: The flashing process of Ubuntu system does not function well.	Copy the full folder to a computer with Windows system, then flash the image using **TurboX Flash**. For further information, refer to: TurboX Flash User Guide.
SDK Manager Flash Issue	Enter the following command on the host machine before restarting the flash: `$ sudo systemctl stop ModemManager`
Generating System Image Issue: The execution of `chroot` command failed. For example: `/usr/sbin/chroot:failed to run command '/bin/bash': Exec format error`	Enter the following command on the host machine before generating system image: `$ docker run --rm --privileged multiarch/qemu-user-static:register --reset`

Chapter 6. Reference Documents

- **Robotics C6490 Platform:**
  - Quick Start Guide: https://docs.qualcomm.com/bundle/80-64868-300/resource/80-64868-300_REV_AB_QCS6490_UBUN_1_0_Linux_Ubuntu_Quick_Start_Guide.pdf
- **Robotics RB1/RB2 Platform:**
  - Quick Start Guide: https://developer.qualcomm.com/hardware/qualcomm-robotics-rb1-rb2-kits/quick-start-guides
  - Hardware Reference Guide: https://developer.qualcomm.com/hardware/qualcomm-robotics-rb1-rb2-kits/hardware-reference-guide
- **Robotics RB5 Platform:**
  - Quick Start Guide: https://developer.qualcomm.com/qualcomm-robotics-rb5-kit/quick-start-guide
  - Hardware Reference Guide: https://developer.qualcomm.com/qualcomm-robotics-rb5-kit/hardware-reference-guide
  - Software Reference Manual: https://developer.qualcomm.com/qualcomm-robotics-rb5-kit/software-reference-manual
- **Robotics RB5 Gen2 Platform:**
  - Quick Start Guide: https://docs.qualcomm.com/bundle/resource/topics/80-79054-300/introduction-to-linux-ubuntu-for-qcs8550-ubun-1-0_1.html

Appendix 1. Notices

Thundercomm may have patents or pending patent programs covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries to service@thundercomm.com.

THUNDERCOMM PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

Changes are made periodically to the information herein; these changes will be incorporated in new editions of the publication. To provide better service, Thundercomm reserves the right to improve and/or modify the products and software programs described in the manuals, and the content of the manual, at any time without additional notice.

The software interface and function and hardware configuration described in the manuals included with your development board or system on module might not match exactly the actual configuration of that you have purchased. For the configuration of the product, refer to the related contract (if any) or product packing list, or consult the distributor for the product sales. Thundercomm may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Thundercomm product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Thundercomm or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary.

The information of this document should not be as any invitation for offer or any advice to the visitors. Please consult the professional comments from the sales consultant prior to do any actions of investment or purchase.

Thundercomm may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Thundercomm Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Thundercomm product, and use of those Web sites is at your own risk. Thundercomm shall not be responsible for the content of the third party.

Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This document is copyrighted by Thundercomm and the property right of the date mentioned in this document, including but not limited trademarks, patents, copyrights, trade name etc. are not covered by any open-source license. Thundercomm may update this document at any time without notice.

Anyone doesn’t have the right to amend, reprint, republication, reproduce, transmit, distribute or any other way to use this document in business or public purpose without the prior written consent by Thundercomm.

E-mail messages sent to Thundercomm via the Internet are not guaranteed to be completely secure.Thundercomm shall not be liable for any loss incurred by the surfer when transmitting any information over the Internet or for any loss incurred by Thundercomm when sending any information over the Internet at

your request.

Thundercomm has all rights under other relevant exemptions provided by laws and regulations, and Thundercomm's failure to claim or delay in claiming such rights shall not be deemed to be a waiver of such rights by Thundercomm.

Thundercomm reserves the right of final interpretation of this document.

Appendix 2. Trademarks

Thundercomm, Thundercomm TurboX, TURBOX, Thundersoft turbox are trademarks of Thundercomm Corporation or its associate companies in China and/or other countries. Intel, Intel SpeedStep, Optane, and Thunderbolt are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Microsoft, Windows, Direct3D, BitLocker, and Cortana are trademarks of the Microsoft group of companies. Mini DisplayPort (mDP), DisplayPort, and VESA are trademarks of the Video Electronics Standards Association. The terms HDMI and HDMI High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC in the United States and other countries. Wi-Fi, Wi-Fi Alliance, WiGig, and Miracast are registered trademarks of Wi-Fi Alliance. USB-C is a registered trademark of USB Implementers Forum. All other trademarks are the property of their respective owners.