Fibocom
PERFECT WIRELESS EXPERIENCE

FG132 OPENSDK Compilation Environment Building Guide

V1.1

Disclaimer

Any action you take in the course of using this document is at your own risk, and Fibocom shall not be liable for any damages or losses under any circumstances. Due to product version upgrade or other reasons, Fibocom reserves the right to modify any information in this document at any time without prior notice and any responsibility. Unless otherwise agreed, all statements, information and suggestions in this document do not constitute any express or implied guarantee.

This document may include the third-party information covering products, services, software, data, and so on. Fibocom does not control and assumes no responsibility for the third-party content, including but not limited to the accuracy, compatibility, reliability, availability, legitimacy, appropriateness, performance, non-infringement, and status update, unless otherwise specified in this document. Fibocom does not provide any guarantee or authorization for the third-party content mentioned or referenced in this document. If you need a third-party license, obtain it in an authorized or legal way, unless otherwise specified in this document.

Copyright Notice

Unless specially authorized by Fibocom, the recipient of the documents shall keep the documents and information received confidential, and shall not use them for any purpose other than the implementation and development of this project. Without the written permission of Fibocom, no unit or individual shall extract or copy part or all of the contents of this document without authorization, or transmit them in any form. Fibocom has the right to investigate legal liabilities for any offense and tort in connection with violation of confidentiality obligations, or unauthorized use or malicious use of the said documents and information in other illegal forms.

Trademark Statement

The trademark is registered and owned by Fibocom Wireless Inc.

Other trademarks, product names, service names and company names appearing in this document are owned by their respective owners.

Contact Information

Website: https://www.fibocom.com

Address: 10/F-14/F, Block A, Building 6, Shenzhen International Innovation Valley, Dashi First Road, Xili Community, Xili Subdistrict, Nanshan District, Shenzhen

Tel: 0755-26733555

Contents

Applicable Model

No.	Applicable Model	Description
1	FG132 series	SDX35 platform

Change History

V1.1 (2024-06-14)	Optimized document descriptions, unified the names of the same objects, and unified language structures, etc.
V1.0 (2024-05-16)	Initial version

1 Overview

This document mainly introduces the environment building, system image compilation and programming of FG132- OPEN-SDK.

2 Development Environment Requirements

- A Linux host connected to the Internet with at least 100 GB of available space is required.
- The host should be installed with Ubuntu system (for code compilation), and version 22.04 is recommended (the environment setup for other distribution versions may be slightly different, and Fibocom has not verified it).
- Execute the following code to install the following software packages:

```
$ sudo apt update

$ sudo apt install build-essential clang flex bison g++ gawk gcc-multilib \
g++-multilib gettext git libncurses-dev libssl-dev python3-distutils rsync \
unzip zlib1g-dev file wget liblzo2-dev uuid-dev swig time xsltproc \
libxml-simple-perl ccache ecj fastjar java-propose-classpath libelf-dev \
libncurses5-dev libncursesw5-dev python2.7-dev python3 \
python3-setuptools python3-dev subversion openjdk-8-jdk

$ sudo ln -sf /bin/bash /bin/sh
$ javac -version
javac 1.8.0_402 //Ensure that javac is 1.8

$ git config --global user.email "you@example.com"
$ git config --global user.name "Your Name"
```

For more details about the development environment, see:

https://openwrt.org/docs/guide-developer/toolchain/install-buildsystem

3 Downloading the SDK

Contact Fibocom technical support to apply for Git permissions.

The FG132 series SDK can be downloaded as follows:

```
git clone "ssh://git_user@xa.fibocom.com:29418/iot/fg132-open-sdk"
```

git_user is the Git account applied for by the customer.

4 Introduction to SDK Directory

After cloning the code from the Git repository, its directory structure is as shown in the figure below.

Figure 1. sdx35-sdk code directory structure

The following table describes information about the directories:

Table 1. Key directories and files

Directory and File	Description
build*.sh	Compilation script.
commom	Path for storing the project image signature tool.
projects	Path for storing the projects, including compilable projects. You can create your own internal projects based on Fibocom project templates.
projects/PRJ_XX/config_data	Partition configuration XML for compiling f_cust_devcfg.bin and

	f_def_devcfg.bin.
projects/PRJ_XX/config_patch/	Path for storing overlay configuration or code. Files under this path will overwrite the files in the corresponding path of the SDK root directory during compilation.
projects/PRJ_XX/project_config.xml	Project property configuration, which cannot be modified.
owrt_workspace/owrt/	Path for storing OpenWRT.
owrt_workspace/vendor/fibocom/	Path for storing Fibocom open source code and binary files.
tools	Path for storing tools.

There is an inheritance relationship between projects. Therefore, not all project names are stored under **projects**. Check **projects/Readme.txt** for the inheritance relationship.

For example, **PRJ_FG132-GL-00-00** can be used to compile **PRJ_FG132-CN-00-00**.

5 Code Compilation

5.1 Compilation Command Description

The compilation script **buid.sh** is located in the SDK root directory. The command format is as follows:

```
build.sh [<project_name> [<build_menu>]]
```

- **project_name**: The name of project model supported by the current SDK.
- **build_menu**: Compilation menu options.

5.1.1 Project Information Query

Run **./build.sh** to view all project information.

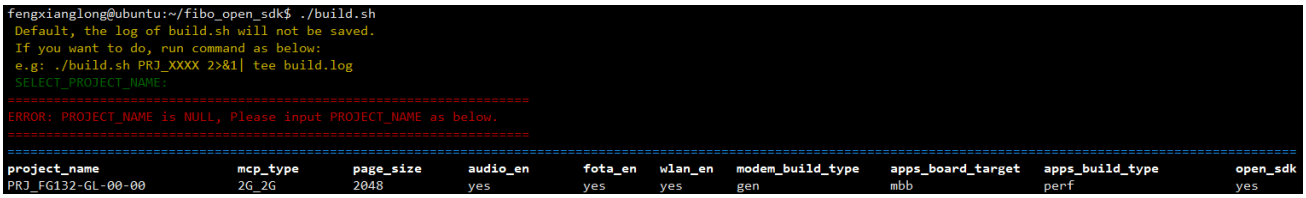

Figure 2. Viewing the project information

5.1.2 Compilation Menu

Method 1: Directly enter **./build.sh project_name build_menu** to specify the project for compilation. For the description of **build_menu**, see Table 2.

Table 2. Description of build_menu

Compilation SN	Compilation Name	Description
1	buildall	Full compilation
2	build_apps_proc	Compile **abl.elf**, **boot.img** and **sysfs-2k.ubi** images.
3	build_apps_recovery_proc	Compile **boot-recovery.img** and **recoveryfs-2k.ubi** images.
4	build_ota_target_files	Package **FOTA tagert**.
5	build_common_image	Package signed **modem** image. This function is under development.
6	build_devcfg_image	Compile Fibocom configuration image and custom configuration image, which are **f_def_devcfg.bin** and **f_cust_devcfg.bin**.
7	build_partition_image	Compile the partition table.
8	build_boot_image	Compile **boot.img** separately.

9	build_abl_image	Compile abl.elf separately.

Method 2: Run **./build.sh project_name** and select the corresponding compilation subsystem from the pop-up compilation menu, as shown in Figure 3.

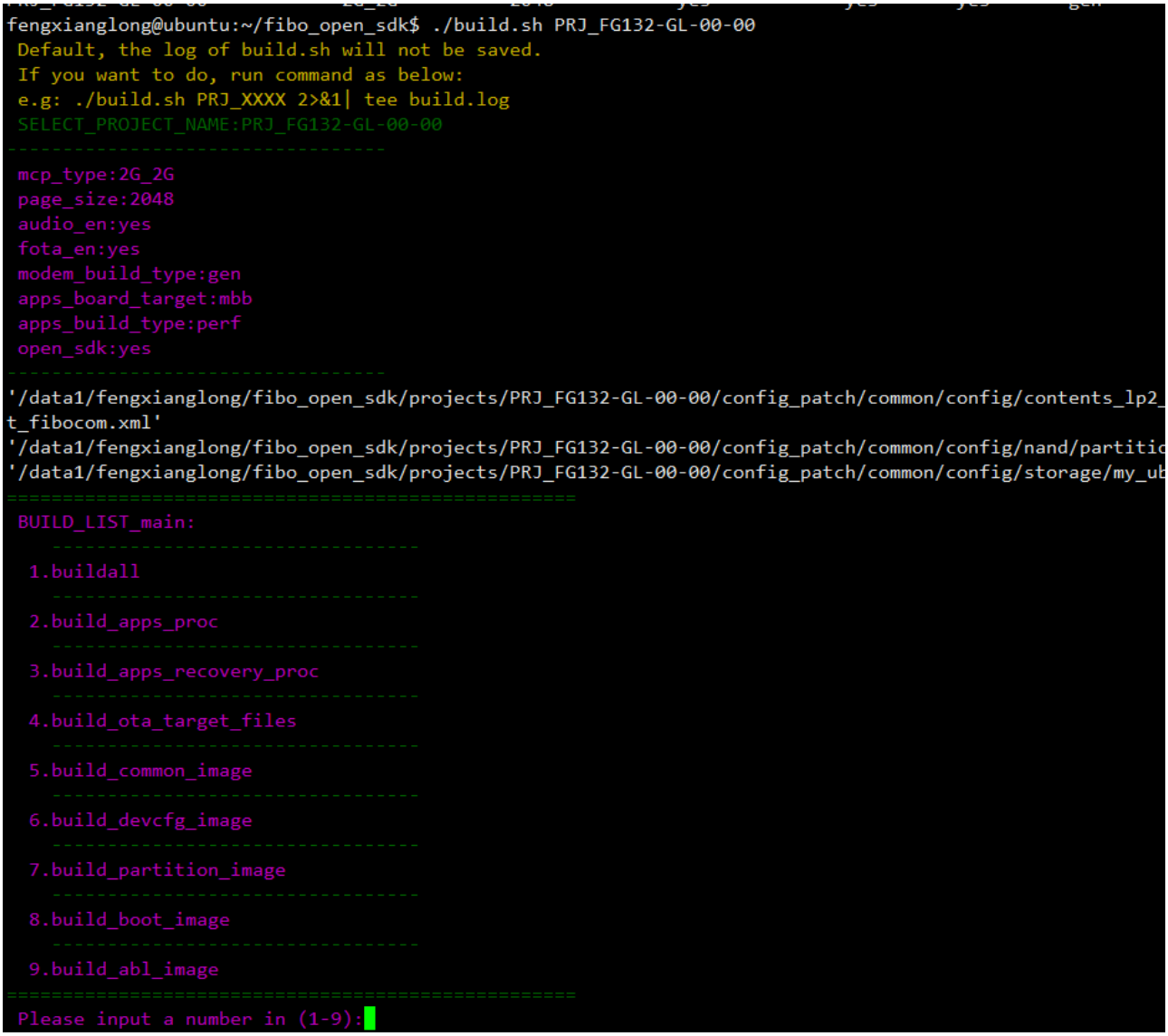

Figure 3. Compilation submenu

5.2 Full compilation

Use the following command to compile the full image of FG132-GL-00-00.

```
./build.sh PRJ_FG132-GL-00-00 1
```

The compilation logs are stored in **owrt_workspace/owrt/build*.log**.

The compiled image output is as follows:

```
image/Build_Version/
```

```
├── abl.elf
├── boot.img
├── boot-recovery.img
├── .cust
│   ├── partition_complete_p2K_b128K.mbn
│   ├── partition.mbn
│   ├── partition_nand.xml
│   ├── patch_p2K_b128K.xml
│   ├── rawprogram_nand_p2K_b128K_for_upgradedevconfigonly_windows.xml
│   ├── rawprogram_nand_p2K_b128K_for_upgrade_linux.xml
│   └── rawprogram_nand_p2K_b128K_for_upgrade_windows.xml
├── .fact
│   ├── partition_complete_p2K_b128K.mbn
│   ├── partition.mbn
│   ├── partition_nand.xml
│   ├── patch_p2K_b128K.xml
│   ├── rawprogram_nand_p2K_b128K_for_erase_dl.xml
│   ├── rawprogram_nand_p2K_b128K_for_upgradedevconfigonly.xml
│   └── rawprogram_nand_p2K_b128K_for_upgrade_dl.xml
├── f_cust_devcfg.xml
├── f_def_devcfg.bin
├── f_def_devcfg.xml
├── ota-target-2024-04-01-13-27-30.zip
├── recoveryfs-2k.ubi
└── sysfs-2k.ubi
```

Table 3. Image description

Image Name	Description
`abl.elf`	Qualcomm abl image, which the boot program used to start the kernel after UEFI.
`boot.img`	Kernel image.
`boot-recovery.img`	Recovery system kernel image.
`.cust/partition_complete_p2K_b128K.mbn` `.fact/partition_complete_p2K_b128K.mbn`	Partition table image.

`.cust/partition.mbn`
`.cust/patch_p2K_b128K.xml`
`.fact/partition.mbn`
`.fact/patch_p2K_b128K.xml`
	Boot image downloaded from USB, which will not be downloaded to the module.

Image Name	Description
`.cust/rawprogram_nand_p2K_b128K_for*.xml` `.fact/rawprogram_nand_p2K_b128K_for*.xml`	Configuration downloaded from USB, including downloaded images and erased partitions.
`f_def_devcfg.bin`	Default configuration image of Fibocom.
`f_cust_devcfg.bin`	Configuration image of customer.
`ota-target-*.zip`	FOTA base package, used to make DFOTA package.
`recoveryfs-2k.ubi`	File system image of the recovery system.
`sysfs-2k.ubi`	System image of system file.

5.3 Separate Compilation

Taking the boot.img image as an example and referring to Table 2, run the following command in the root directory.

```
./build.sh PRJ_FG132-GL-00-00 8
```

The compiled image output is as follows:

```
image/Build_Version/
├── boot.img
```

6 Image Programming

6.1 fastboot Programming

The fastboot programming of each image is as follows (due to platform restrictions, some partitions do not support fastboot programming):

Table 4. Commands for fastboot programming

Image Name	Programming Command
abl.elf	fastboot flash abl abl.elf
boot.img	fastboot flash boot boot.img
boot-recovery.img	fastboot flash recovery boot-recovery.img
partition_complete_p2K_b128K.mbn	Not supported
f_def_devcfg.bin	Not supported
f_cust_devcfg.bin	Not supported
recoveryfs-2k.ubi	fastboot flash recoveryfs recoveryfs-2k.ubi
sysfs-2k.ubi	fastboot flash system sysfs-2k.ubi

Example: To use fastboot to program **boot.img**, you need to run the **AT+GTUSBMODE** command to set the USB combination that supports the adb port. For details, see the *Fibocom_FG132_AT Command Manual*.

Use tools such as adb fastboot in **C:\Program Files (x86)\FibocomCustomizedDriver\Tools**. Other Android tools may have compatibility issues.

1. After the module is powered on, enter the following command through adb to make the module in bootloader mode.

```
adb root
adb reboot bootloader
```

2. Erase the partition where the kernel is located (which is named **boot**).

```
fastboot erase boot
```

3. Program the kernel to the partition where it is located.

```
fastboot flash boot boot.img
```

4. Restart the module after successful programming.

```
fastboot reboot
```

6.2 Programming in USB Download Mode

In order to preserve the modem calibration parameters, Fibocom recommends to use the SDX35_Windows_Multi_Upgrade_Tool to download the image. Since SDK can only compile partial firmware, contact Fibocom technical support to obtain the full software package of the model and

overwrite the compiled image into the package.

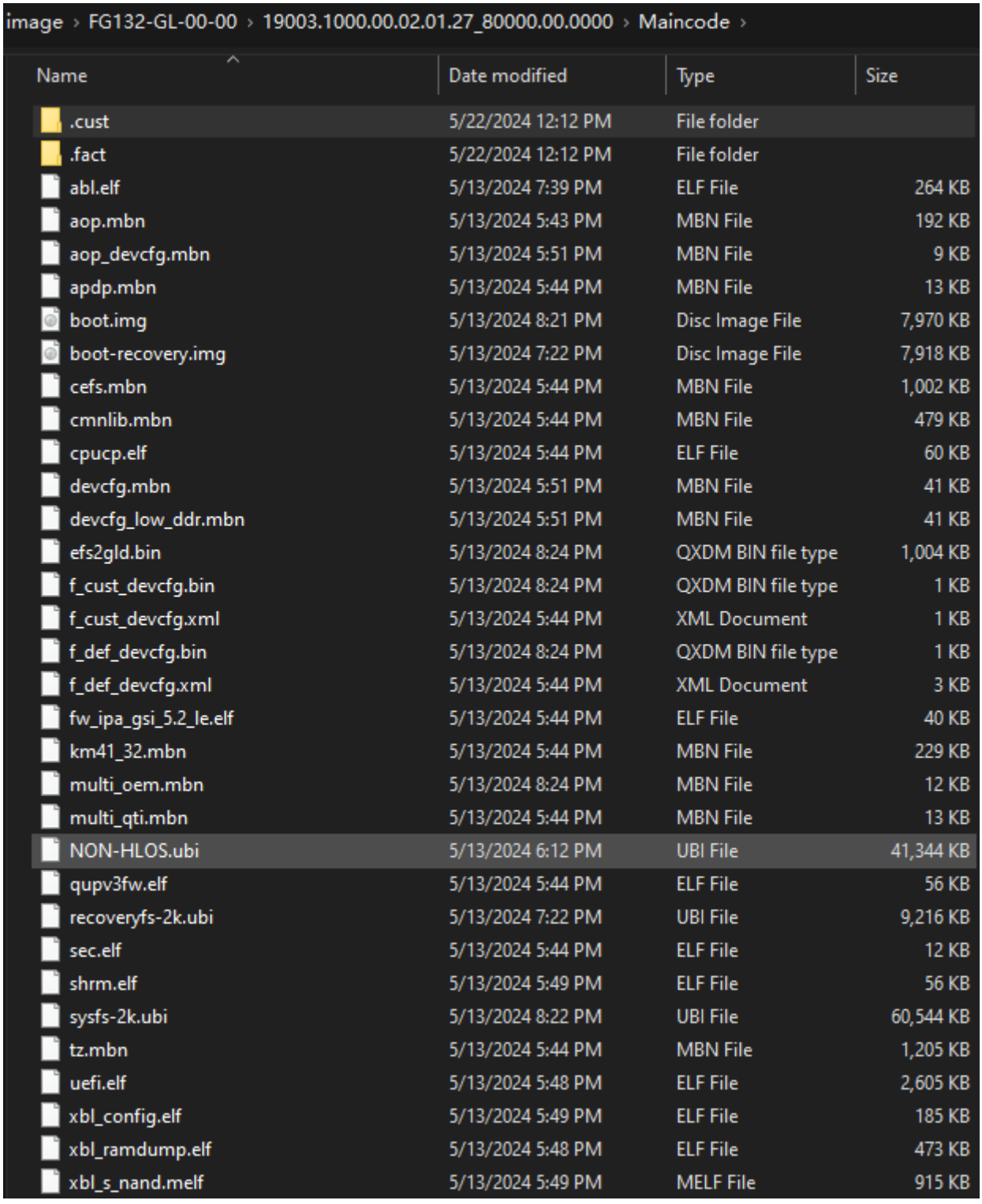

Figure 4. Full software base package

Here takes FG132-GL-00 as an example:

1. Pull PIN22 USB_BOOT up to 1.8 V. Press the power button to start the module. Connect the module to the PC through USB. The module will enter the download mode, as shown below:

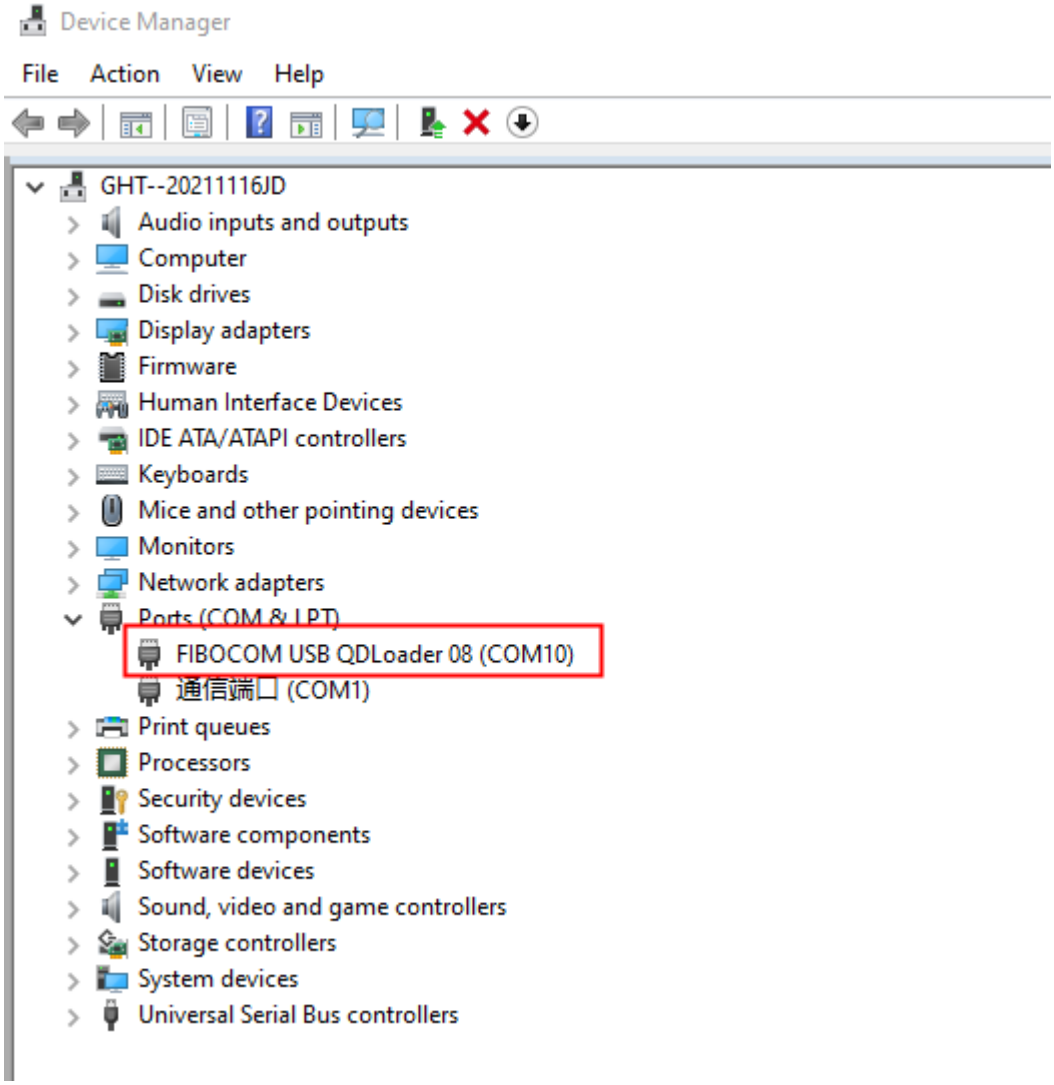

Figure 5. Download port

2. Enable the SDX35_Windows_Multi_Upgrade_Tool.

Use .cust/rawprogram_nand_p2K_b128K_for_upgrade_windows.xml to configure the firmware update, as shown below:

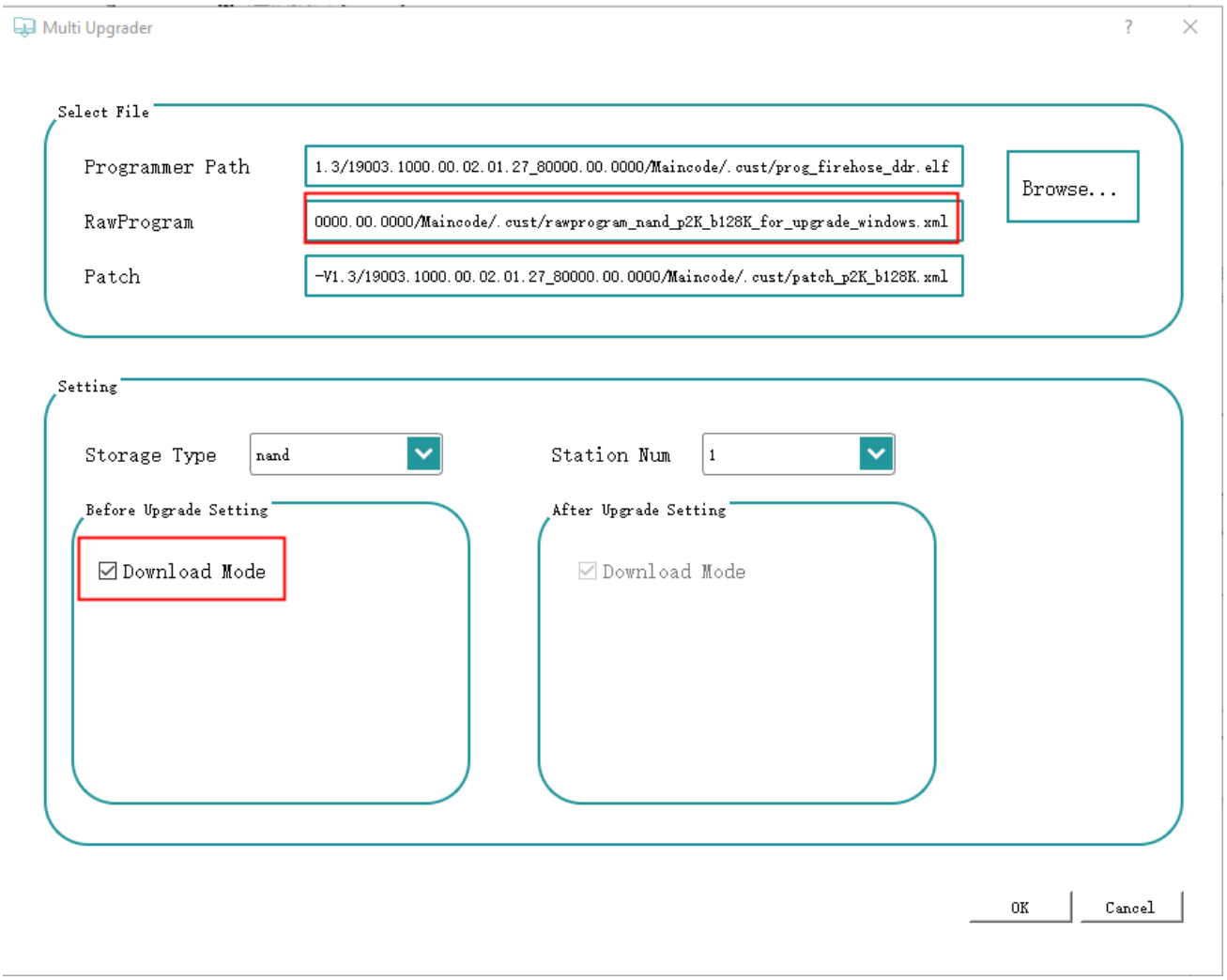

Figure 6. Multi Upgrader configuration

3. Click ▶ to start downloading. After the download progress reaches 100%, power off the module and restart it.

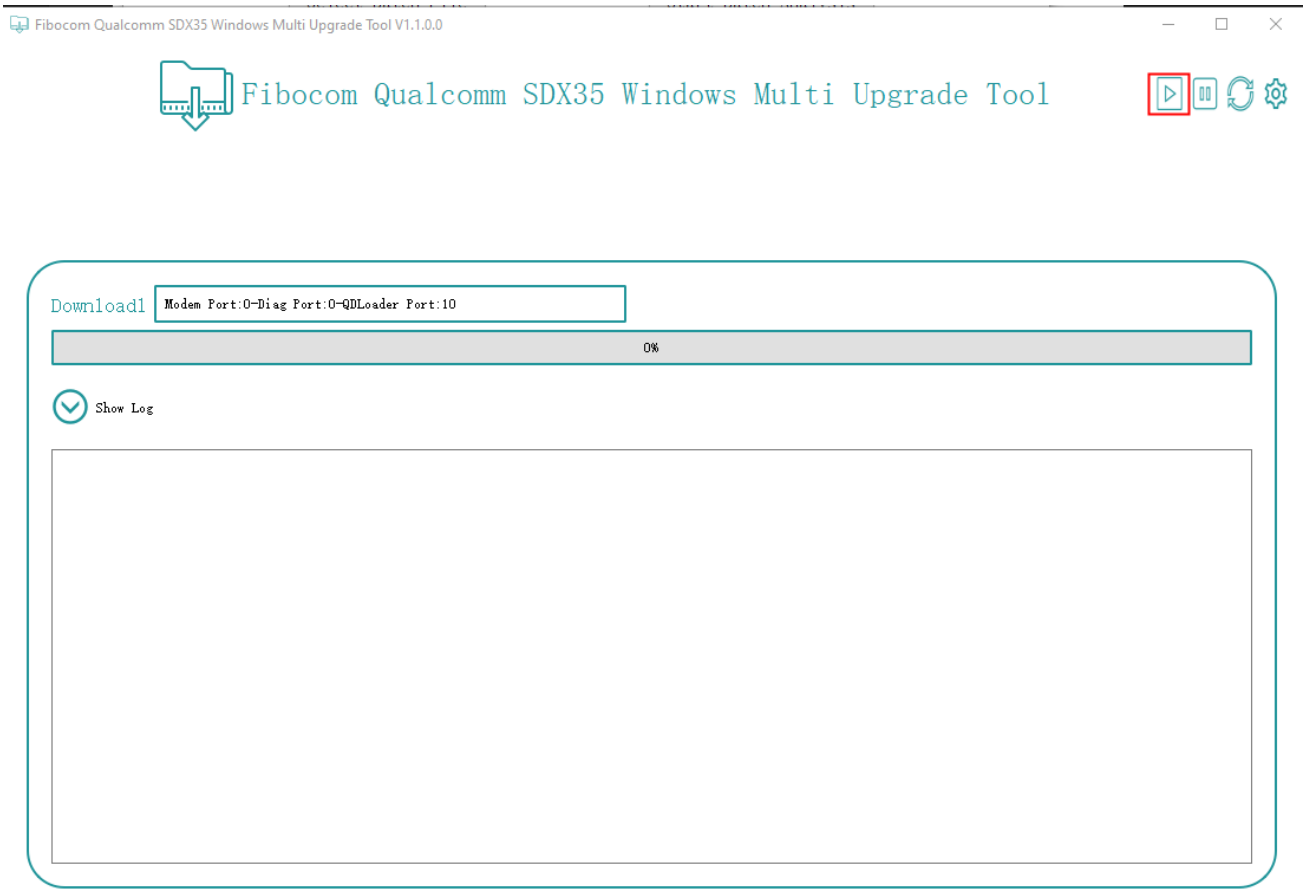

Figure 7. Start to download

Due to COM port conflicts upon enabling, disable the serial port tool before download. For more details about the tools, refer to the documents in the toolkit.