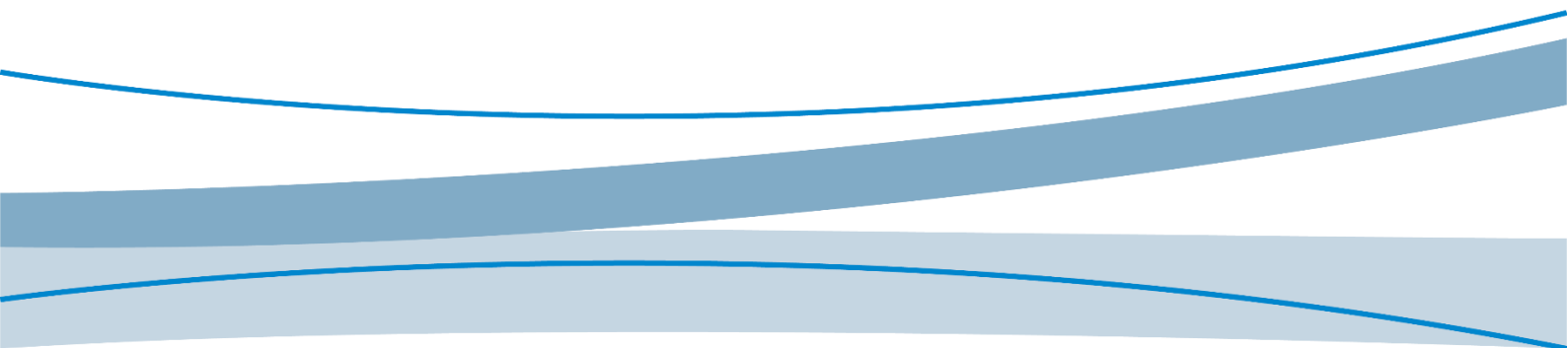




Fibocom Upgrade_Tool Upgrade Guide _Linux&Android

V1.3



Copyright

Copyright ©2022 Fibocom Wireless Inc. All rights reserved.

Without the prior written permission of the copyright holder, any company or individual is prohibited to excerpt, copy any part of or the entire document, or transmit the document in any form.

Notice

The document is subject to update from time to time owing to the product version upgrade or other reasons. Unless otherwise specified, the document only serves as the user guide. All the statements, information and suggestions contained in the document do not constitute any explicit or implicit guarantee.

Trademark

 The trademark is registered and owned by Fibocom Wireless Inc.

Contact

Website: <https://www.fibocom.com/en/>

Address: Floor 10, Building A, Shenzhen International Innovation Valley, First Stone Road, Xili Community, Xili Street, Nanshan District, Shenzhen

Tel: +86 755-26733555

Contents

Change History	3
1 Forward	4
1.1 Applicable Models.....	4
2 Compilation.....	5
2.1 Upgrade tool package Introduction	5
2.2 Compile Upgrade Tool	5
2.2.1 Linux Upgrade Tool	5
2.2.2 Android Upgrade Tool	6
3 Upgrade.....	8
3.1 Local Upgrade	8
3.2 Remote Upgrade	8
3.3 Viewing Upgrade Result.....	9
4 Upgrade Parameter Description.....	10
5 Restore NV.....	11
5.1 MDM9205 TX Modules NV Restoration.....	11
5.2 MDM9X07 LE Modules NV Restoration.....	11

Change History

V1.3 (2022-03-27)	Support rawprogram_nand_XX.xml without erase. Add -r support mdm9x07 Le series NV auto restore function 9205 TX restore NV instructions After the upgrade is successful, delete the upgrade log.
V1.2 (2021-12-27)	Add PCIe Port Upgrade Support specified firmware package path for upgrade
V1.1 (2021-11-01)	Added an applicable model Modify section Appendix A
V1.0 (2021-09-14)	Initial version

1 Forward

Upgrade_ tool is used for Linux and Android hosts to upgrade the firmware of Qualcomm platform modules.

1.1 Applicable Models

Num	Modules	Description
1	FM150/ FG150	SDX55 LE Platform
2	FM160	SDX65 LE Platform
3	FG10x/FM10x	SDX12 LE Platform
4	NL668/ MC116/LC116	MDM9x07 LE Platform
5	MA510/ MC109/MC100E	MDM9205 TX Platform

2 Compilation

2.1 Upgrade tool package Introduction

- code: upgrade tool source code
- Doc: Guidance document
- code/Makefile: Compiling configuration file for Linux Environment
- code/Android.mk: Compiling configuration file for Android Environment
- doc: Chinese and English guidance documents



```
ght@ght-pc:~/CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3
$ tree
.
├── code
│   ├── Android.mk
│   ├── fibo_firehose.c
│   ├── firehose_protocol.c
│   ├── hostdl_packet.h
│   ├── Makefile
│   ├── md5.c
│   ├── md5.h
│   ├── README
│   ├── sahara_protocol.c
│   ├── sahara_protocol.h
│   ├── stream_download_protocol.c
│   ├── upgrade_tool
│   ├── usb2tcp.c
│   ├── usb_linux.c
│   └── usb_linux.h
└── doc
    ├── Fibocom Upgrade_Tool Upgrade Guide_Linux&Android_V1.3.pdf
    └── Fibocom Upgrade_Tool升级指南_Linux&Android_V1.3.pdf
2 directories, 17 files
```

2.2 Compile Upgrade Tool

2.2.1 Linux Upgrade Tool

- Configure cross compilation tool

GCC is used by default. If arm GCC is required, set CROSS_COMPILE variable in Makefile.

As blow:

```

ght@ght-pc: ~/CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3/code
#arm gcc tool
#CROSS_COMPILE = ~/ARM_Linux_GCC/bin/arm-none-linux-gnueabi-

CC = ${CROSS_COMPILE}gcc

CFLAGS = -g -Wall
CFLAGS += -Wno-format-overflow
LD_LIBRARY = -lpthread -ldl

INCLUDE =
SOURCES = \
    fibo_firehose.c \
    firehose_protocol.c \
    sahara_protocol.c \
    usb_linux.c \
    md5.c \
    usb2tcp.c \
    stream_download_protocol.c

linux: clean
    ${CC} ${CFLAGS} -s ${SOURCES} -o upgrade_tool ${LD_LIBRARY}

debug: clean
    ${CC} ${CFLAGS} -DFH_DEBUG -s ${SOURCES} -o upgrade_tool ${LD_LIBRARY}

clean:
    rm -f upgrade_tool

~
~
"Makefile" [dos] 29L, 557C                                     29,0-1    All

```

- Compile

Put the upgrade tool code on the Linux host, then in the code directory, execute make. The upgrade_tool will be generated if the compilation is OK.

As shown in the figure below.

```

ght@ght-pc:~/CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3/code
$ make
rm -f upgrade_tool
gcc -g -Wall -Wno-format-overflow -s fibo_firehose.c firehose_protocol.c sahara_protocol.c usb_linux.c md5.c usb2tcp.c stream_download_protocol.c -o upgrade_tool -lpthread -ldl
ght@ght-pc:~/CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3/code
$ ls
Android.mk      hostdl_packet.h  md5.h           sahara_protocol.h  usb2tcp.c
fibo_firehose.c Makefile         README          stream_download_protocol.c  usb_linux.c
firehose_protocol.c md5.c           sahara_protocol.c upgrade_tool        usb_linux.h
ght@ght-pc:~/CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3/code

```

2.2.2 Android Upgrade Tool

1. Put the upgrade tool code into the Android code directory.
2. Run *source build/envsetup.sh*
3. Run *lunch*, then selects the build option.
4. Run *mmm CBB_Linux_Upgrade_Tool_V1.0*

5. If the compilation is successful, the upgrade_tool will be generated

The path of upgrade_tool will be displayed in the compilation log.

e.g.

out/target/product/msm8953_64/system/bin/upgrade_tool

```
ght@ubuntu-230:~/android
$ source build/envsetup.sh
=====BSP_SELECT_ID=====
including device/qcom/common/vendorsetup.sh
including vendor/qcom/proprietary/common/vendorsetup.sh
including sdk/bash_completion/adb.bash
ght@ubuntu-230:~/android
$ lunch

You're building on Linux

Lunch menu... pick a combo:
 1. msm8953_64-userdebug
 2. msm8953_64-user

Which would you like? [aosp_arm-eng] 2

=====
PLATFORM_VERSION_CODENAME=REL
PLATFORM_VERSION=7.1.2
TARGET_PRODUCT=msm8953_64
TARGET_BUILD_VARIANT=user
TARGET_BUILD_TYPE=release
TARGET_BUILD_APPS=
TARGET_ARCH=arm64
TARGET_ARCH_VARIANT=armv8-a
TARGET_CPU_VARIANT=generic
TARGET_2ND_ARCH=arm
TARGET_2ND_ARCH_VARIANT=armv7-a-neon
TARGET_2ND_CPU_VARIANT=cortex-a53
HOST_ARCH=x86_64
HOST_2ND_ARCH=x86
HOST_OS=linux
HOST_OS_EXTRA=Linux-4.4.0-31-generic-x86_64-with-Ubuntu-14.04-trusty
HOST_CROSS_OS=windows
HOST_CROSS_ARCH=x86
HOST_CROSS_2ND_ARCH=x86_64
HOST_BUILD_TYPE=release
BUILD_ID=N2G47H
OUT_DIR=out
=====
ght@ubuntu-230:~/android
$ mmm CBB_Linux_Host_Upgrade_Tool_Android_Linux_V1.3/code/

=====
PLATFORM_VERSION_CODENAME=REL
PLATFORM_VERSION=7.1.2
TARGET_PRODUCT=msm8953_64
TARGET_BUILD_VARIANT=user
TARGET_BUILD_TYPE=release
TARGET_BUILD_APPS=
TARGET_ARCH=arm64
TARGET_ARCH_VARIANT=armv8-a
TARGET_CPU_VARIANT=generic
TARGET_2ND_ARCH=arm
TARGET_2ND_ARCH_VARIANT=armv7-a-neon
TARGET_2ND_CPU_VARIANT=cortex-a53
HOST_ARCH=x86_64
HOST_2ND_ARCH=x86
HOST_OS=linux
HOST_OS_EXTRA=Linux-4.4.0-31-generic-x86_64-with-Ubuntu-14.04-trusty
HOST_CROSS_OS=windows
HOST_CROSS_ARCH=x86
HOST_CROSS_2ND_ARCH=x86_64
HOST_BUILD_TYPE=release
BUILD_ID=N2G47H
OUT_DIR=out
=====
fatal: Not a git repository (or any parent up to mount point /fwork3)
Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).
make: Entering directory `/fwork3/ght/android'
Running kati to generate build-ba5cb121c5911a0d1757241257c87132.ninja...
No need to regenerate ninja file
Starting build with ninja
ninja: Entering directory `.'
[100% 14/14] Install: out/target/product/msm8953_64/system/bin/upgrade_tool
make: Leaving directory `/fwork3/ght/android'

#### make completed successfully (3 seconds) ####
```


3 Upgrade

3.1 Local Upgrade

1. Check whether the USB connection of the module is normal.

If the module is in normal mode, send `at+disk=0,0,0` to unlock diag port first.

2. Copy the unzipped firmware package and `upgrade_tool` to the host.
3. Enter the directory where the `upgrade_tool` is located, and then execute the upgrade command. If there is only one module, it is usually unnecessary to specify the upgrade port

`./upgrade_tool -f <firmware image dir>`

e.g.

```
./upgrade_tool -f 19010.1000.00.02.73.15/Maincode
```

4. Specify upgrade port to upgrade

- UART and USB port: `-p /dev/ttyUSBx`

```
./upgrade_tool -f <firmware image dir> -p /dev/ttyUSB0
```

- PCIE port: `-M /dev/mhi_DUN`

```
./upgrade_tool -f <firmware image dir> -p /dev/mhi_DUN
```

3.2 Remote Upgrade

If the device host does not have enough storage space, `upgrade_Tool` can upgrade firmware with remote host through network.

Operation steps:

1. Put the upgrade_tool on the device host
2. Put the firmware package and upgrade_tool on the remote host
3. Run upgrade_tool on the device host to start the USB TCP service

```
./upgrade_tool -p 9008
```

4. Check the IP address of the device host. You can ping the device host through the remote host

5. Execute the upgrade tool on the remote host

```
./upgrade_tool -f <firmware image dir> -p <The client IP:9008> <-r 1>
```

3.3 Viewing Upgrade Result

The following log will be printed after the upgrade succeeds:

```
Upgrade module successfully
```

If the upgrade fails, the corresponding log will be printed according to different failure reasons. If it is inconvenient for the customer to view the result through log, the upgrade_tool supports saving upgrade result. The filpdate_result.txt file will be generated in the same directory of the tool. The upgrade result will be saved in the file according to the specific upgrade conditions. The result of successful upgrade is OK, the result of failed upgrade is ERROR

4 Upgrade Parameter Description

No.	Parameter	Necessary	Description
1	-f <firmware image dir>	Yes	Upgrade firmware image dir
2	-r <0/1>	No	This parameter is only applicable to nl668 projects. When -r 1 is set, NV will be restored automatically after upgrade. If -r 1 is not set, you need to manually execute at command to restore NV after upgrade
3	-s </sys/bus/usb/devices/xx>	No	When multiple modules are connected to the host, this parameter is used to specify the module that need to be upgraded. When only one module is connected to the host, this parameter does not need to be set.
4	-p <port>	No	Upgrade port <-p /dev/ttyUSBX> or TCP USB Remote Upgrade <-p 9008 and -p IP:9008>
5	-z <0/1>	No	Send Zero-length package.Defaults is 0. No special instructions, no need to set parameters.
6	-e <0/1>	No	Erase ALL partitions before upgrading. Defaults is 0. No special instructions, no need to set parameters.
7	-l <log dir>	No	set log dir and save the upgrade failed log.

5 Restore NV

5.1 MDM9205 TX Modules NV Restoration

MDM9205 TX modules can not restore NV automatically. After upgrade, you need to execute at command to restore NV.

The operation steps:

1. Send at command: `at+efserrfatal`
2. Send at command: `at+cfun=15`
3. After the modules restart, check whether IMEI and SN are normal.

5.2 MDM9X07 LE Modules NV Restoration

MDM9X07 LE modules with parameter `<-r 1>` to upgrade firmware, and the module can automatically restore NV.

If `<-r 1>` is not set when upgrading firmware, you can execute the at command to restore NV.

The operation steps:

1. Send at command: `at+efserrfatal`
 2. Send at command: `at+cfun=15`
 3. The modules will restart and restore nv.
 4. Wait for the module to bringup and check whether IMEI and SN.
-