



GNSS Android Driver

USER MANUAL

WWW.UNICORE.COM

uDriver

GNSS Android Driver

Copyright© 2009-2025, Unicore Communications, Inc.
Data subject to change without notice.



Foreword

Scope

This manual provides information about the porting guide, configurations, and firmware upgrade methods of Unicore's GNSS driver (uDriver) for Android devices.

Target Readers

This document applies to technicians who possess expertise on GNSS receivers.

Statement

Legal right notice

This manual provides information and details on the products of Unicore Communication, Inc. ("Unicore") referred to herein.

All rights, title and interest to this document and the information such as data, designs, layouts contained in this manual are fully reserved, including but not limited to the copyrights, patents, trademarks and other proprietary rights as relevant governing laws may grant, and such rights may evolve and be approved, registered or granted from the whole information aforesaid or any part(s) of it or any combination of those parts.

Unicore holds the trademarks of "和芯星通", "Unicore", "UNICORECOMM" and other trade name, trademark, icon, logo, brand name and/or service mark of Unicore products or their product serial referred to in this manual (collectively "Unicore Trademarks").

This manual or any part of it, shall not be deemed as, either expressly, implied, by estoppel or any other form, the granting or transferring of Unicore rights and/or interests (including but not limited to the aforementioned trademark rights), in whole or in part.

Disclaimer

The information contained in this manual is provided "as is" and is believed to be true and correct at the time of its publication or revision. This manual does not represent, and in any case, shall not be construed as a commitments or warranty on the part of Unicore with respect to the fitness for a particular purpose/use, the accuracy, reliability and correctness of the information contained herein.

Information, such as product specifications, descriptions, features and user guide in this manual, are subject to change by Unicore at any time without prior notice, which may not be completely consistent with such information of the specific product you purchase.



Should you purchase our product and encounter any inconsistency, please contact us or our local authorized distributor for the most up-to-date version of this manual along with any addenda or corrigenda.

Revision History

Version	Revision History	Date
R1.0	First Release.	Dec. 2024
R1.2.1	Updated uDriver Introduction . Updated the following contents in uDriver Porting Guide : (1) Added the required/optional modules in Modify the Makefile . (2) Specified the required/optional configurations in Configure uDriver . Updated the following contents in uDriver Application Notes : (1) Updated Table 3-1 System Properties . (2) Added I2C Configuration . (3) Updated AGNSS/NTRIP Configuration .	May 2025

uDriver version: v1.2.1



uDriver Introduction

uDriver is an Android open-source driver demo compatible with Unicore GNSS products. By integrating this driver, users can achieve functions such as reading, decoding GNSS data and configuring GNSS receivers on their Android devices.

uDriver is provided as a demo program, and users can adjust its working logic to align with their specific requirements.

It should be noted that the operational stability of uDriver is closely related to the Android system environment and hardware conditions. Before mass production, please conduct thorough testing and verification across all relevant scenarios.



uDriver Porting Guide

This chapter describes the operational steps involved in the porting of uDriver.

2.1 Development Environment

This document is based on the AOSP development environment for the RB5 platform. If other platforms are used, please modify the corresponding files during porting. For porting examples, refer to the content in *unicore_gnss/Documentation/Porting*.

2.2 Disable the Old Driver

If the target system has integrated other GNSS drivers, disable the conflicting driver according to the methods provided by the platform vendor.

2.3 Copy the uDriver Source Code

Copy the *unicore_gnss* folder to the Android *vendor/unicore_gnss* directory.

2.4 Port the Device Files

2.4.1 Modify the Makefile

Find the Makefile in the *device* directory of the source code.

Add modules such as *libunicore.so*, *gnss_service*, *sensors_service*, and *unicore_service*.

Modify the HAL driver version according to *rb5.mk.sample*.

Table 2-1 Modules and Functions

Modules	Functions	Required/Optional
libunicore	Unicore driver library	Required
gnss_service	GNSS HAL driver	Required
sensors_service	SENSOR HAL driver	Optional
unicore_service	Data injection	Optional

```
# HIDL version
#unicore_hidl_gnss_version := 2.1
#unicore_hidl_sensors_version := 2.1

# AIDL version
unicore_aidl_gnss_version := 3
unicore_aidl_sensors_version := 2
```

2.4.2 Modify the ueventd File

Add the GNSS device node. Modify the device name to match the one in use. For specific details, refer to *ueventd.common.rc.sample*.

Note

- The device name in the ueventd file must correspond to that defined in the *unicore_config.h*.
- Ensure that the SEPOLICY is properly configured.

```
/dev/ttyUSB0 0660 gps root #GNSS module communication interface
```

2.4.3 Add SEPOLICY

Integrate the *unicore_gnss/sepolicy* folder into the system's SELinux rules, and ensure the added SEPOLICY is valid. For specific details, refer to *BoardConfigcommon.mk.sample*.

```
BOARD_SEPOLICY_DIRS += vendor/unicore_gnss/sepolicy
```

2.5 Configure uDriver

After the porting operations, uDriver can be compiled successfully and launched normally. Before uDriver can function correctly, configuration steps are needed. Among these, [Configure GNSS Receiver Communication Interface](#) and [Configure GNSS Module Type](#) are required configurations and checks. Others are optional.

2.5.1 Configure GNSS Receiver Communication Interface

(1) Modify GNSS Communication Device Node

First, confirm the device node (e.g., */dev/ttyUSB0*), then modify the macros related to the serial port parameters such as *DEFAULT_GPS_DEVICE*.

```
//Location of the config file: unicon_gnss/libunicore/include/libunicore/unicon_config.h
```

```
#define DEFAULT_GPS_DEVICE    "/dev/ttyUSB0"  
#define WORKING_BAUD         460800
```

(2) Check and Modify SEPOLICY Configuration

Check the secontext of the GNSS communication port and grant uDriver permission to operate the serial port. The following files need to be checked:

- *unicore_gnss/sepolicy/file_contexts*
- *unicore_gnss/sepolicy/device.te*
- *unicore_gnss/sepolicy/hal_gnss_unicore.te*

(3) Check and Modify uevent Configuration

Refer to the [Modify the ueventd File](#) procedures to ensure the GNSS communication port is configured with correct user and permissions.

2.5.2 Configure GNSS Module Type

Set the CMDSET_DEFAULT in the configuration file to match the product type in use. The default configuration is CMDSET_NPL.

```
//Location of the config file: unicon_gnss/libunicore/include/libunicore/unicon_config.h
```

```
enum {  
    CMDSET_UNKNOWN = 0,  
    CMDSET_NPL = 1, //standard-precision products  
    CMDSET_HPL = 2, //high-precision products  
    CMDSET_DEFAULT = CMDSET_NPL, //set to match the product type in use  
};
```

2.5.3 [Optional] Configure the Debug Switch

To print debugging information, enable the GPS_DEBUG macro in the configuration file. Disable it if not needed.

```
//Location of the config file: unicon_gnss/libunicore/include/libunicore/unicon_config.h
```

```
#define GPS_DEBUG
```



2.5.4 [Optional] Configure AGNSS and DGNSS Data Injection

To configure the AGNSS and DGNSS data injection, refer to [AGNSS/NTRIP Configuration](#) in [uDriver Application Notes](#).

2.5.5 [Optional] Configure I2C

To configure the I2C interface, refer to [I2C Configuration](#) in [uDriver Application Notes](#).

2.6 Porting Completed

After the uDriver porting is completed, install UGPSTest APP or other positioning software for testing.

uDriver Application Notes

This chapter describes the uDriver configurations, including system properties, interface configurations, AGNSS/NTRIP configuration, firmware upgrade, etc.

3.1 System Properties

The uDriver configuration items can be modified not only in the configuration file but also via system properties.

System properties have priority over the configuration file, i.e.

- The HAL will prioritize using the settings defined in system properties.
- If no system properties are found, the HAL will use the default settings in the configuration file.

The prefix of the system properties is *persist.vendor.unicore*.

Table 3-1 System Properties

System Properties	Descriptions	Source Code Config Items	Config File	Config Values
gnss_port	GNSS device node	DEFAULT_GNSS_DEVICE	unicore_config.h	/dev/ttyUSB0
gnss_baud	Working baud rate of the firmware	WORKING_BAUD	unicore_config.h	115200, etc.
gnss_baud_default	Factory settings of the firmware baud rate	FACTORY_BAUD	unicore_config.h	115200, etc.
gnss_debug	Status indicator (reserved)	N/A	N/A	N/A
gnss_injection	Assistance data injection	GNSS_INJECTION	unicore_config.h	agnss, etc.
gnss_bl_download_baud	Bootloader download baud rate during upgrade	BOOTLOADER_DOWNLOAD_BAUDRATE	unicore_config.h	115200, etc.

System Properties	Descriptions	Source Code Config Items	Config File	Config Values
gnss_fw_download_baud	Firmware download baud rate during upgrade	FW_DOWNLOAD_BAUDRATE	unicore_config.h	115200, etc.
log_level	Log output level	N/A	N/A	VERBOSE DEBUG INFO WARN ERROR FATAL
log_toconsole	logcat switch	N/A	N/A	true/false
log_tofile	Log file switch	N/A	N/A	true/false
gnss_version	Firmware version of the GNSS receiver	N/A	N/A	Driver configuration
gnss_hal_version	uDriver HAL version	N/A	N/A	Driver configuration

3.2 TCP/IP Interface Configuration

The uDriver HAL listens on the local port by default for debugging interactions and raw data output.

The number of supported clients is defined by the *NR_CLIENTS*, which is not recommended to be modified unless necessary. When the number of connected clients exceeds the limit value, new connections will replace the first client already connected (client0).

The port uses a socket-based connection. Socket clients can transparently transmit commands to the GNSS module. For example,

- Send the command "*\$pdinfo*" to query version information.
- Send the command "*~CONFIG COM1 115200*" (not starting with \$) to configure the GNSS module.

Address: 127.0.0.1

Port: 5744

Table 3-2 Supported Commands

Commands	Parameters	Descriptions
halver	N/A	Query the driver version
rcv.read[option]	"none" "all"	Disable data forwarding Enable data forwarding
putfile <name> <size>	name: "blu.pkg" "fw.pkg" size: file size	Send files (for upgrade only)
upgrade	N/A	Start upgrade
debuglog [option]	"on" "off"	Enable or disable debuglog, for transparent transmission of driver logs

3.3 AGNSS/NTRIP Configuration

The assistance data injection is defined in the *libunicore* directory, and the parameters are described as follows:

```
//Location of the config file: uncore_gnss/libunicore/include/libunicore/unicore_config.h

// Disable data injection.
#define GPS_INJECTION    "none"
// Inject Ntrip data through ttyUSB0 and set the baud rate to 460800.
#define GPS_INJECTION    "ntrip /dev/ttyUSB0 460800"
// Inject Ntrip data through the default port.
#define GPS_INJECTION    "ntrip"
// Inject ephemerides through the default port (see gps_port configuration).
// uncore_service uses http protocol to download ephemerides from rx-networks.cn; https
protocol is not supported.
#define GPS_INJECTION    "agnss"
```

AGNSS and Ntrip accounts are configured in *unicore_gnss/unicore_service/unicore_service.conf*, which will be automatically copied to the Android */vendor/etc/* directory after successful compilation.

3.4 GNSS Module Configuration

If you need to configure the log list or send commands to the GNSS module, follow the steps below:

1. Enable the *GNSS_INIT_HOOK* macro defined in *unicore_config.h*.
2. Input the log list and commands to be sent during the startup of the module in the configuration file: *unicore_gnss/libunicore/unicore_gnss.conf*.



The configuration file will be automatically copied to the Android `/vendor/etc/` directory after successful compilation.

3.5 I2C Configuration

uDriver supports communication with the GNSS module via I2C polling mode. To enable this, follow the configurations below:

(1) Modify the GNSS Communication Interface

```
//Location of the config file: uncore_gnss/libunicore/include/libunicore/unicore_config.h

#define DEFAULT_GNSS_DEVICE      "/dev/i2c-1"
```

(2) If you need to configure the GNSS module's power control or reset via GPIO, follow the steps below:

1. In the `unicore_gnss/libunicore/pwrctl/pwrctl.cpp` file, modify the power control to `pwrctl_gpio.cpp`. In the `pwrctl_gpio.cpp` file, configure the specific GPIO pins.
2. In the `ueventd.common.rc.sample` file, modify the user of the communication interface to root.
3. In the `unicore_gnss/libunicore/libunicore.rc` file, modify the user of the `/data/vendor/unicore` and `log` folder to root.
4. In the rc files of `gnss_service`, `sensors_service`, `unicore_service`, modify the user to root.

3.6 Firmware Upgrade Configuration

uDriver supports the GNSS module for firmware upgrade. The baud rates need to be configured before upgrade.

```
//Location of the config file: uncore_gnss/libunicore/include/libunicore/unicore_config.h

#define BOOTLOADER_DOWNLOAD_BAUDRATE  460800
#define FW_DOWNLOAD_BAUDRATE          460800
```

Firmware Upgrade

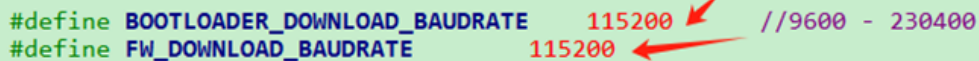
This chapter gives instructions on how to upgrade Unicore's GNSS modules. It takes the standard-precision module UM621 as an example and also applies to high-precision modules.

4.1 Preparations

Before upgrade, prepare the bootloader and firmware. For example:

- Bootloader: FB2S_bootloader_build6435_update_115200.pkg
- Firmware: UM621_R6.0.0.0Build3816_mfg.pkg

Modify the bootloader download baud rate and firmware download baud rate in the file *unicore_gnss\hal\libgps\uc6226_hook.h*, as shown in **Figure 4-1**.



```
#define BOOTLOADER_DOWNLOAD_BAUDRATE 115200 //9600 - 230400
#define FW_DOWNLOAD_BAUDRATE 115200
```

Figure 4-1 Modify the Baud Rates

After the modification, recompile the *gps.default.so* library. Update the */vendor/lib64/hw/gps.default.so* in the Android device.

There are two methods to upgrade the GNSS module:

- Upgrade by UGPSTest APP,
- Upgrade by socket.

Note

Before upgrade, ensure that the GNSS module can communicate with uDriver normally, otherwise uDriver cannot trigger the soft start of the GNSS module.

4.2 Upgrade by UGPSTest APP

[Steps]

To upgrade the GNSS module using the UGPSTest APP, follow the steps below, as shown in **Figure 4-2**.

1. Open the UGPSTest APP.
2. Click "Upgrade", as shown by the red arrow 1.
3. Select the paths of bootloader and firmware respectively, as shown by the red arrows 2 and 3.
4. Click "升级", as shown by the red arrow 4.

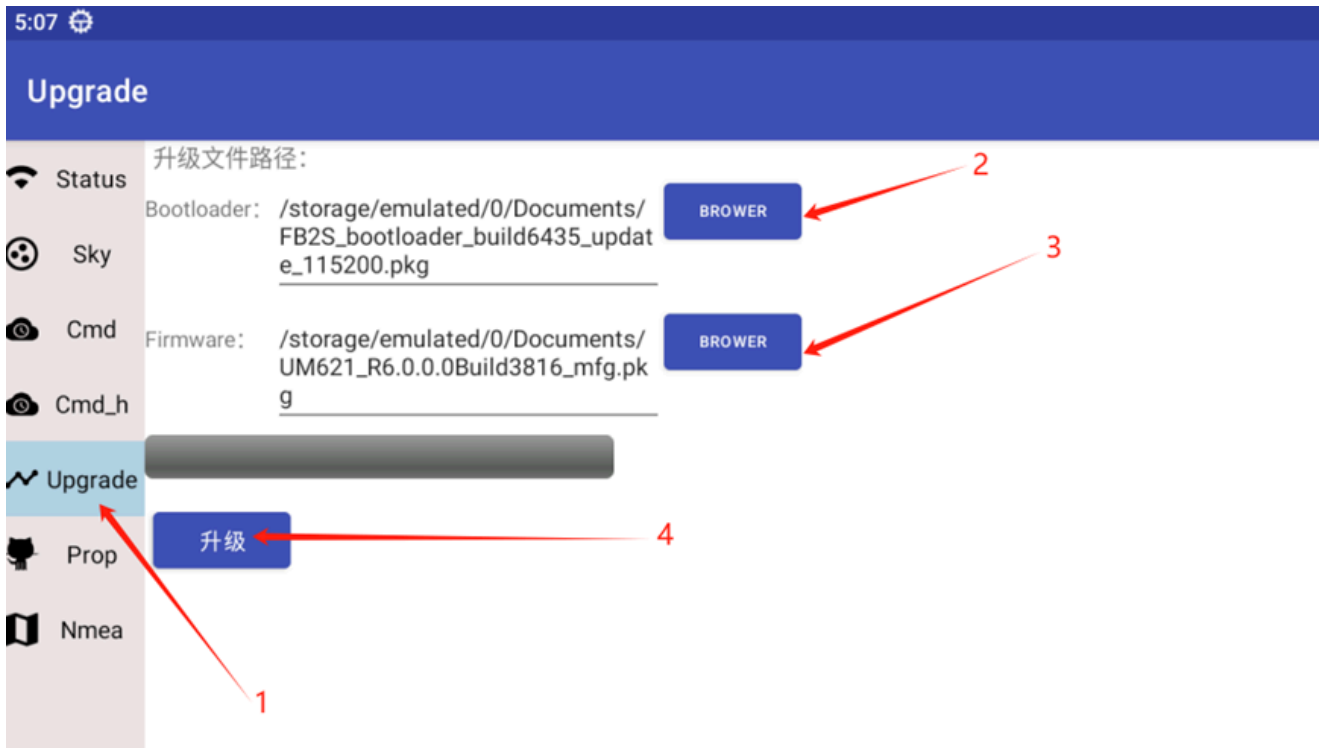


Figure 4-2 Upgrade Steps

When the module is being upgraded, a progress bar is displayed in the window, as shown in Figure 4-3.

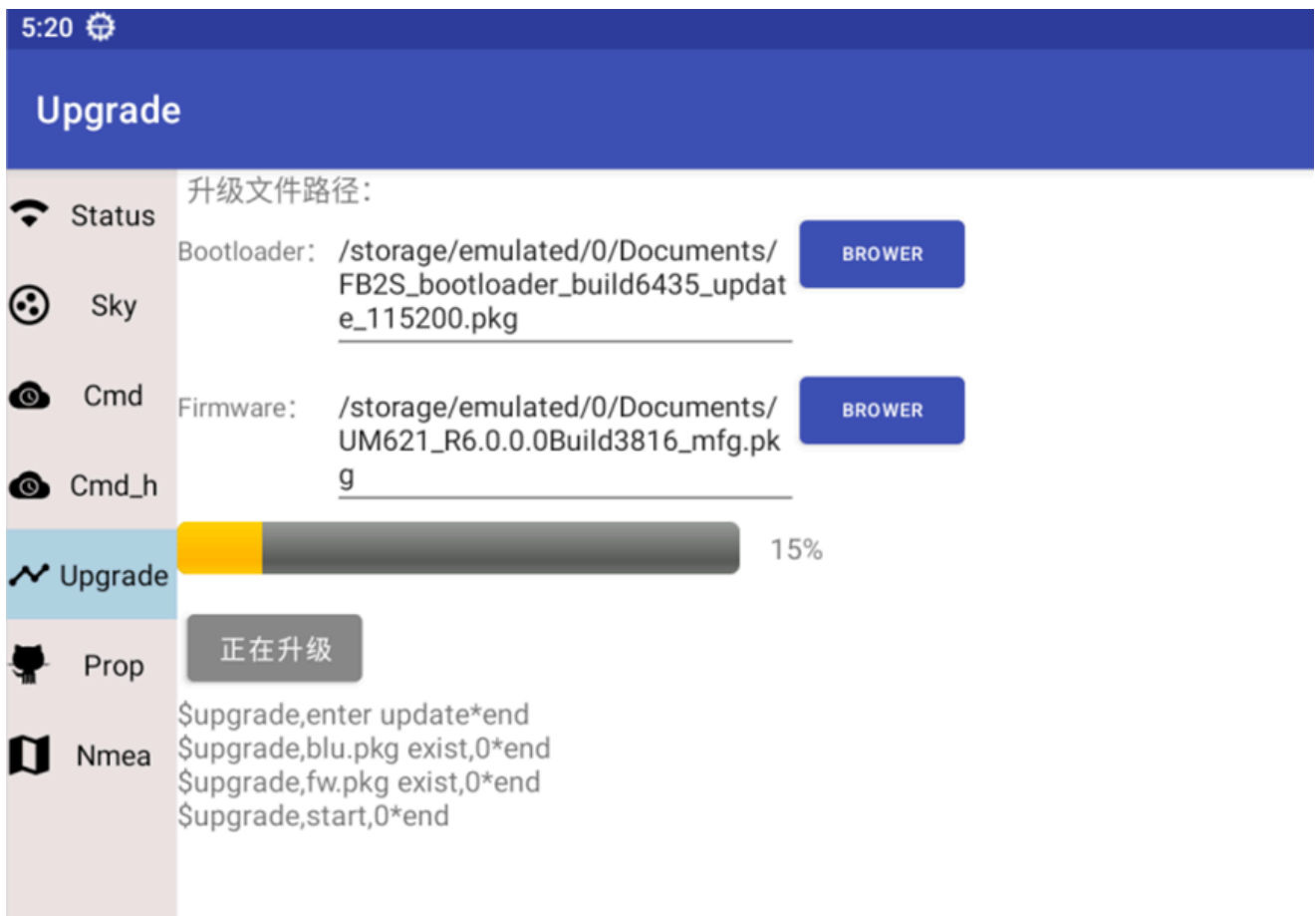


Figure 4-3 Progress Bar

If the upgrade is successful, the window will display "success", as shown in Figure 4-4.

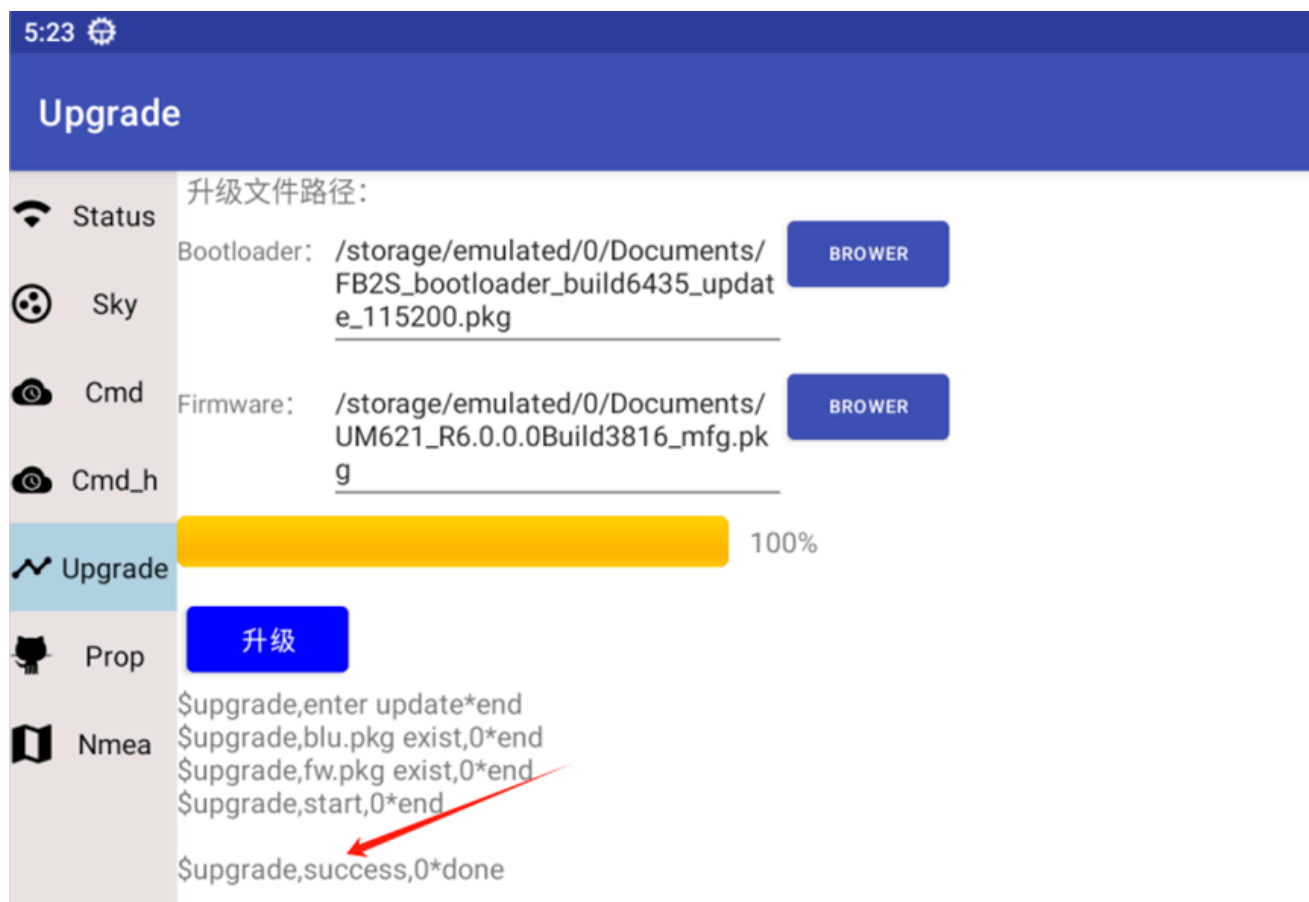


Figure 4-4 Upgrade Success

After a successful upgrade, the module will reboot automatically. After waiting for a period of time, the UGPSTest APP will regain positioning data. (If the baud rate of the firmware does not match that of the serial port configured by uDriver, users need to modify the baud rate of the serial port. Only then will the UGPSTest APP regain positioning data.)

Note

The upgrade logs can be viewed via Logcat or in the `/data/vendor/unicore/log/last_update_log` file.

4.3 Upgrade by Socket

Background: uDriver listens on the local port by default for debugging interactions.

Address: 127.0.0.1

Port: 5744

Table 4-1 Upgrade Commands Supported

Commands	Parameters	Descriptions
putfile <name> <size>	name: "blu.pkg" "fw.pkg" size: the actual size of the bootloader and firmware	Specify the file name and file size of the bootloader and firmware.
upgrade	N/A	Start the upgrade.

[Steps]

To upgrade the GNSS module by means of socket, follow the steps below:

1. Create a local TCP client and connect it to the server of uDriver via the socket port (server address: 127.0.0.1, port: 5744).
2. The client sends "putfile <name> <size>" command to the server.
 - (1) For bootloader, the client first sends "putfile blu.pkg <size>", then sends the bootloader file to the server (This step is required and will trigger a copy operation).
 - (2) For firmware, the client first sends "putfile fw.pkg <size>", then sends the firmware file to the server (This step is required and will trigger a copy operation).
3. The client sends the "upgrade" command to the server to start the upgrade process.
4. After a successful upgrade, disconnect the client.

Note

During the upgrade process, the server will provide timely feedback to the client about the upgrade process, which can be monitored by reading the socket port.

和芯星通科技（北京）有限公司

Unicore Communications, Inc.

北京市海淀区丰贤东路 7 号北斗星通大厦三层
F3, No.7, Fengxian East Road, Haidian, Beijing, P.R.China,
100094

www.unicore.com

Phone: 86-10-69939800

Fax: 86-10-69939888

info@unicorecomm.com



www.unicore.com