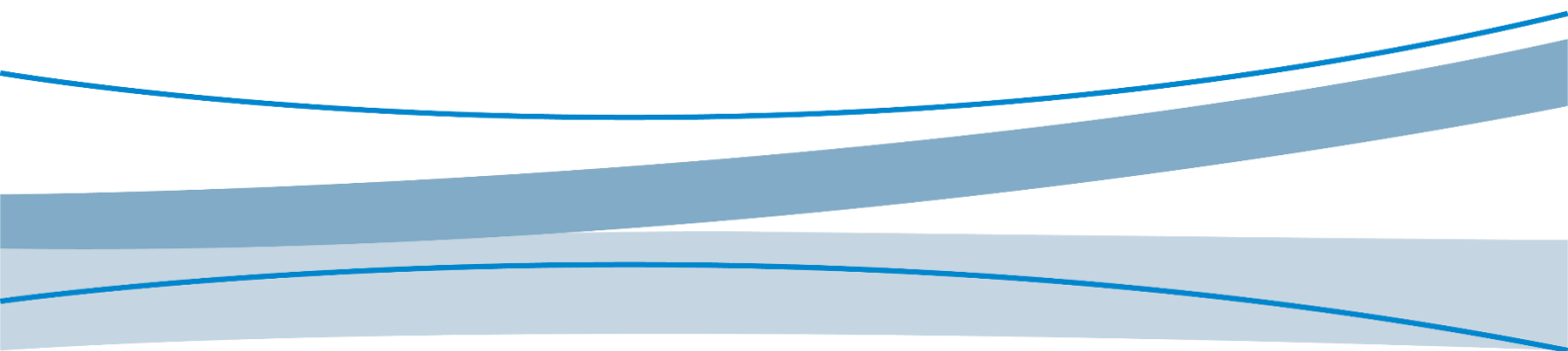




MC66x

RIL Integration Guide_Android

V1.1



Disclaimer

This document serves as a guide for the use of products. Customers must design their products according to the information provided in the document. The Company shall not be liable for any damage caused by failure to comply with relevant operation or design specifications or safety rules. Due to product version upgrade or other reasons, the Company 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.

Copyright Notice

Copyright © 2022 Fibocom Wireless Inc. All rights reserved.

Unless specially authorized by the Company, 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 the Company, 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. The Company 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

FIBOCOM 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

Safety Instructions

Do not operate wireless communication products in areas where the use of radio is not recommended without proper equipment certification. These areas include environments that may generate radio interference, such as flammable and explosive environments, medical devices, aircraft or any other equipment that may be subject to any form of radio interference.

The driver or operator of any vehicle shall not operate wireless communication products while controlling the vehicle. Doing so will reduce the driver's or operator's control and operation of the vehicle, resulting in safety risks.

Wireless communication devices do not guarantee effective connection under any circumstances, such as when the (U) SIM card is invalid or the device is in arrears. In an emergency, please use the emergency call function when the device is turned on, and ensure that the device is located in an area with sufficient signal strength.

Contents

Change History	3
1 About This Document.....	4
2 Android RIL Driver Overview	5
2.1 Functions Supported by RIL Driver	5
2.2 Supported Android Versions.....	5
3 RIL Integration Configuration	7
3.1 Configuring Linux Kernel Serial Port Driver	7
3.2 Configuring Android Startup Script.....	7
3.2.1 Before Android 8	7
3.2.1.1 32-bit device.....	7
3.2.1.2 64-bit device.....	7
3.2.2 After Android 8.....	8
3.2.2.1 32-bit device.....	8
3.2.2.2 64-bit device.....	8
3.2.3 Parameter Description.....	8
3.3 Modifying RILD Permissions	9
3.4 Loading RIL Library File	9
4 RIL Debugging	11
4.1 Capturing RIL LOG.....	11
4.2 Tags in Logs.....	11
5 FAQs	12
5.1 What Can I Do If the Dial Fails?	12
5.2 How to Get the Local IP Address, Primary DNS, and Secondary DNS Information of ECM?	13
5.3 How to Use the PPP Dialing Mode?	14
5.3.1 Switch to the PPP Dialing Mode	14
5.3.2 What Can I Do If PPP Dialing Fails?	15
5.3.3 How to Obtain IP Address and DNS Information of PPP?.....	16
5.4 PPP Dial-up Not Supported by Device	17

5.5 Unable to Use Data Services Normally.....	20
5.6 Unable to Make a Call.....	20
5.7 What Can I Do If Text Messages Cannot Be Sent?	20
5.8 RIL Works Improperly	20
5.9 How to Set APN?	22
5.9.1 Adding APN Manually.....	22
5.9.2 Configuring the apns-conf.xml File.....	22
6 Properties Under Maintenance	24
6.1 Property Setting Methods	24
6.2 ril.fibocom.dialmode	26
6.2.1 Introduction.....	26
6.2.2 Value	26
6.2.3 Example	27
6.3 ril.fibocom.NetifName.....	27
6.3.1 Functions	27
6.3.2 Value	27
6.3.3 Example	28
6.4 ril.fibocom.cid	28
6.4.1 Functions	28
6.4.2 Values	28
6.4.3 Example	28
6.5 ril.fibocom.cgdcont0	29
6.5.1 Functions	29
6.5.2 Values	29
6.5.3 Example	29
6.6 ril.fibocom.usbmode	29
6.6.1 Functions	29
6.6.2 Values	29
6.6.3 Example	30
7 Appendix A References.....	31

Change History

V1.1 (2022-11-29)	The default dialing is changed to ECM and the process of integrating RIL is simplified
V1.0 (2022-05-31)	Initial version

1 About This Document

This document introduces how to integrate the Radio Interface Layer (RIL) driver to the customer's target system and how to modify the configuration file that starts the RIL service.

2 Android RIL Driver Overview

2.1 Functions Supported by RIL Driver

Table 1. Supported functions

Function	Support
SMS	YES
Voice call	YES
Data service	YES
SIM card toolkit	NO

2.2 Supported Android Versions

Fibocom RIL driver currently supports the following Android versions in the following table.

Table 2. Supported Android versions

Function	Support
Android 2.x	NO
Android 3.x	NO
Android 4.x	YES
Android 5.x	YES
Android 6.x	YES
Android 7.x	YES
Android 8.x	YES
Android 9.x	YES
Android 10.x	YES

Android 11.x	YES
--------------	-----

3 RIL Integration Configuration

This chapter introduces the operation steps of integrating the RIL library to the customer's Android system, including the RIL library file structure, configuring the Linux kernel driver, adding the USB device PID/VID supported by the kernel, loading the RIL library file, and modifying the Android system RILD execution permissions and startup scripts.

3.1 Configuring Linux Kernel Serial Port Driver

Please refer to the *Fibocom_MC66x_Application Guide_Linux (Android) Driver Integration_V1.0* document.

3.2 Configuring Android Startup Script

3.2.1 Before Android 8

3.2.1.1 32-bit device

Configure the `init.rc` file to add the `ril-daemon` process.

```
service ril-daemon /system/bin/rild -l /system/lib/libreference-ril.so -- -w 1
class main
socket rild stream 660 root radio
socket rild-debug stream 660 radio system
user root
group radio cache inet misc audio sdcard_rw log
```

3.2.1.2 64-bit device

Configure the `init.rc` file to add the `ril-daemon` process.

```
service ril-daemon /system/bin/rild -l /system/lib64/libreference-ril.so -- -w 1
```

```
class main
socket rild stream 660 root radio
socket rild-debug stream 660 radio system
user root
group radio cache inet misc audio sdcard_rw log
```

3.2.2 After Android 8

3.2.2.1 32-bit device

Configure the **rild.rc** file to add the **ril-daemon** process.

```
service ril-daemon /vendor/bin/hw/rild -l /vendor/lib/libreference-ril.so -- -w 1
    class main
    user root
    group radio cache inet misc audio log readproc wakelock
    capabilities BLOCK_SUSPEND NET_ADMIN NET_RAW
```

3.2.2.2 64-bit device

Configure the **rild.rc** file to add the **ril-daemon** process.

```
service ril-daemon /vendor/bin/hw/rild -l /vendor/lib64/libreference-ril.so -- -w 1
    class main
    user root
    group radio cache inet misc audio log readproc wakelock
    capabilities BLOCK_SUSPEND NET_ADMIN NET_RAW
```

3.2.3 Parameter Description

In the **init.rc**(or **rild.rc**) script, the **-l** and **-w** options work as follows:

- **-l**: Specifies the RIL library load path
- **-w**: Specifying the dialing Mode. 0: ppp Dial mode, 1: ECM dial mode.

3.3 Modifying RILD Permissions

RILD program execution requires the root permission. You can get the permission by removing the call to the switchUser function in the rild.c (<\$ android dir>

/hardware/ril/rild/rild.c) main function, as shown in Figure 1.

```
224     arg_device[sizeof(arg_device)-1] = 0;
225     arg_overrides[1] = "-d";
226     arg_overrides[2] = arg_device;
227     done = 1;
228
229     } while (0);
230
231     if (done) {
232         argv = arg_overrides;
233         argc = 3;
234         i = 1;
235         hasLibArgs = 1;
236         rilLibPath = REFERENCE_RIL_PATH;
237
238         | RLOGD("overriding with %s %s", arg_overrides[1], arg_overrides[2]);
239     }
240 }
241 OpenLib:
242 #endif
243 //switchUser();
244
245 dlHandle = dlopen(rilLibPath, RTLD_NOW);
246
247 if (dlHandle == NULL) {
248     RLOGE("dlopen failed: %s", dlerror());
249     exit(-1);
250 }
251
```

Figure 1. Modifying the main function in RILD



Some Android versions do not have the switchUser function, such as Android 10. In such a situation, skip this section.

3.4 Loading RIL Library File

```
adb root
adb remount
```

For Android 8 (not included) and earlier versions with 32bits:

```
adb push <local-dir>/libreference-ril.so /system/lib
```

For Android 8 (not included) and earlier versions with 64bits:

```
adb push <local-dir>/libreference-ril.so /system/lib64
```

If the customer is using Android 8 or later, 32-bit:

```
adb push <local-dir>/libreference-ril.so /vendor/lib
```

If the customer is using Android 8 or later, 64-bit:

```
adb push <local-dir>/libreference-ril.so /vendor/lib64
```

Then restart the Android device, which is expected to automatically dial successfully.

4 RIL Debugging

4.1 Capturing RIL LOG

In the cmd window of the Windows operating system, enter the following command:

```
adb logcat -b all -v time > radio.txt
```

4.2 Tags in Logs

Process logs are stored in the same file and they can be identified based on tags. The following table shows logs and their respective tags.

Table 3. Tags in the RIL log

Tag	File
GHT-RIL	hardware/ril/reference-ril/reference-ril.c
GHT-AT	hardware/ril/reference-ril/atchannel.c
RILC	hardware/ril/libril/ril.cpp
RILD	hardware/ril/rild/rild.c
RILJ	frameworks/opt/telephony/src/java/com/android/internal/telephony/RIL.java

5 FAQs

5.1 What Can I Do If the Dial Fails?

1. Run the **adb shell** command to enter the Android device, and then use the following command to confirm whether the port is recognized:

```
root@android:/ # ls /dev/ttyUSB*
```

❶ If the device is mounted normally, the following content will be returned:

```
ls /dev/ttyUSB*
```

```
/dev/ttyUSB0
```

```
/dev/ttyUSB1
```

```
/dev/ttyUSB2
```

```
/dev/ttyUSB3
```

Table 4. USB port function description

Device Name	Functions	Remarks
ttyUSB0	Modem	PPP dial-up port.
ttyUSB2	DIAG	Port for getting modem log.
ttyUSB4	AT	Port for the RIL program to send AT commands and receive responses.

2. Check whether the NIC name obtained using the “**dmesg | grep cdc**” command is the same as the NIC name in “**configuring xxx**” in the RADIO log. If they are different, set the following properties in **/system/build.prop**

```
ril.fibocom.NetifName=xx
```

“xx” is the result obtained by command “**dmesg | grep cdc**”, such as “eth1” in the figure below.

```

17219 11-29 16:56:26.498 300 300 D DHCP : ip 10.83.105.230 gw 10.83.105.1 prefixLength 24
17220 11-29 16:56:26.498 300 300 D DHCP : dns1: 211.137.130.2
17221 11-29 16:56:26.498 300 300 D DHCP : server 192.168.1.1, lease 30840 seconds
17222 11-29 16:56:26.498 300 300 D DHCP : configuring eth1
17223 11-29 16:56:26.533 0 0 E selinux : avc: denied { set } for property=net.eth1.dns1 pid=300 uid=0 gid=10
17224 11-29 16:56:26.528 300 300 D GHT_RIL : fibo_bring_up_interface_do_dhcp ret = 0, retry = 1
17225 11-29 16:56:26.529 300 300 D GHT_RIL : fibo_get_ip: 10.83.105.230, if_name:eth1, strlen(ifr.ifr_name):4
17226 11-29 16:56:26.529 300 300 D GHT_RIL : fibo_get_ip: get hwaddr[0]:f0
17227 11-29 16:56:26.529 300 300 D GHT_RIL : fibo_get_ip: get hwaddr[1]:4b

Cmder
[ril.fibocom.version]: [Fibocom_RIL_V10X.06.V1.0.6_004]
[ril.function.dataonly]: [1]
[rild.libargs]: [-d]
[rild.libpath]: [/vendor/lib64/libquectel-ril.so]
[ro.boot.noril]: [true]
[ro.boottime.ril-daemon]: [3885591876]
[ro.ril.ecclist]: [112,911]
rk3399_roc_pc_plus:/ $ su
:/ # dmesg | grep cdc
[ 0.872489] usbcore: registered new interface driver cdc_ether
[ 0.872517] usbcore: registered new interface driver cdc_eem
[ 0.872777] usbcore: registered new interface driver cdc_subset
[ 0.873021] usbcore: registered new interface driver cdc_ncm
[ 0.873080] usbcore: registered new interface driver cdc_mbim
[ 0.950998] usbcore: registered new interface driver cdc_acm
[ 0.951015] cdc_acm: USB Abstract Control Model driver for USB modems and ISDN adapters
[ 0.951107] usbcore: registered new interface driver cdc_wdm
[ 1704.445966] cdc_ether 5-1:1.0 eth1: register 'cdc_ether' at usb-fe380000.usb-1, CDC Ethernet De
vice, f0:4b:b3:b9:eb:e5
:/ #

```

3. Then restart the device. If the network is still not connected, check the module registration network, signal strength, port, APN and DNS by grasping the RIL log

5.2 How to Get the Local IP Address, Primary DNS, and Secondary DNS Information of ECM?

After ECM dial-up is successful, the local IP address, primary DNS, and secondary DNS are written to the net.xx.local-ip, net.xx.dns1, and net.xx.dns2 property items of the Android device ("xx" is the name of the NIC used by the ECM. Each device has a different NIC name, such as eth1 and usb0).

In the case of "eth1", these properties can be obtained by doing the following:

1. Get the local IP address by running the following Android OS commands:

```
getprop net.eth1 // Obtain all information about the eth1 NIC
```

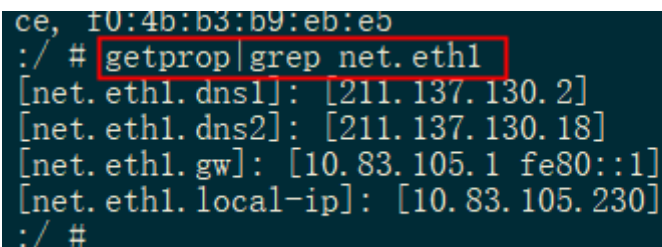
2. The following attributes correspond to IP, gateway, PDNS, and SDNS respectively:

```
net.eth1.local-ip //IP address
```



```
net.eth1.gw           //gateway
net.eth1.dns1         //pdns
net.eth1.dns2         //sdns
```

An example diagram is shown below:



```
ce, f0:4b:b3:b9:eb:e5
:/ # getprop | grep net.eth1
[net.eth1.dns1]: [211.137.130.2]
[net.eth1.dns2]: [211.137.130.18]
[net.eth1.gw]: [10.83.105.1 fe80::1]
[net.eth1.local-ip]: [10.83.105.230]
:/ #
```

If the ECM dial-up uses other NICs (such as usb0 and eth2), change the "eth1" in the above properties to the name of other NICs. For example: net.usb0.local-ip and net.usb0.dns1.

5.3 How to Use the PPP Dialing Mode?

5.3.1 Switch to the PPP Dialing Mode

The MC66x module supports the PPP dialing mode. The PPP dialing mode can be switched as follows:

1. Before dialing, ensure that you have successfully integrated the RIL library by referring to Chapter [3](#).
2. Set the following properties in the /system/build.prop file:

```
ril.fibocom.dialmode=0
ril.fibocom.usbmode=73
```

, For details, see Section [6.1](#).

3. Restarting your Android Device.

5.3.2 What Can I Do If PPP Dialing Fails?

You can perform the following steps to rectify the fault:

1. Run the “getprop|grep ril” command to check whether dialmode and usbmode meet expectations(dialmode is expected to be 0 and usbmode is expected to be 73).

```
rk3399_roc_pc_plus:/ $ getprop|grep ril
[gsm.version.ril-impl]: [Fibocom_RIL_V10X♥]
[init.svc.ril-daemon]: [running]
[persist.ril.current_usbmode]: [32]
[ril.currentapntype]: [default]
[ril.datachannel]: [/dev/ttyUSB6]
[ril.fibocom.dialmode]: [0]
[ril.fibocom.usbmode]: [73]
[ril.fibocom.version]: [Fibocom_RIL_V10X.06.V1.0.6_004]
[ril.function.dataonly]: [1]
[rild.libargs]: [-d]
[rild.libpath]: [/vendor/lib64/libquectel-ril.so]
[ro.boot.noril]: [true]
[ro.boottime.ril-daemon]: [3950684418]
[ro.ril.ecclist]: [112,911]
rk3399_roc_pc_plus:/ $ |
```

2. If dialmode and usbmode are correct, modify the ip-up and ip-down files in directory /system/etc/ppp/.
3. The **ip-up** file is modified as follows:

```
#!/system/bin/sh

/system/bin/setprop "net.interfaces.defaultroute" "ppp0"

/system/bin/setprop "net.ppp0.dns1" "$DNS1"

/system/bin/setprop "net.ppp0.dns2" "$DNS2"

/system/bin/setprop "net.ppp0.local-ip" "$IPLOCAL"

/system/bin/setprop "net.ppp0.remote-ip" "$IPREMOTE"

exit 0
```

The ip-down file is modified as follows:

```
#!/system/bin/sh
```

```
case $1 in
ppp1)
echo 0 > /proc/sys/net/ipv4/ip_forward;
esac
rm /etc/ppp/ppp*.pid
# Use interface name if linkname is not available
NAME=${LINKNAME:-"$1"}
/system/bin/setprop "net.$NAME.local-ip" ""
/system/bin/setprop "net.$NAME.remote-ip" ""
/system/bin/setprop "net.interfaces.defaultroute" ""
/system/bin/setprop "net.ppp0.dns1" ""
/system/bin/setprop "net.ppp0.dns2" ""
/system/bin/setprop "net.ppp0.local-ip" ""
/system/bin/setprop "net.ppp0.remote-ip" ""
```



The **ip-up** and **ip-down** files must be modified with the execution permission 555.

5.3.3 How to Obtain IP Address and DNS Information of PPP?

After the PPP dial-up is successful, the IP address, primary DNS, and secondary DNS are written to the properties `net.ppp0.local-ip`, `net.ppp0.dns1`, `net.ppp0.dns2`. You can get the information by performing the following operations:

1. Execute the following Android OS command:

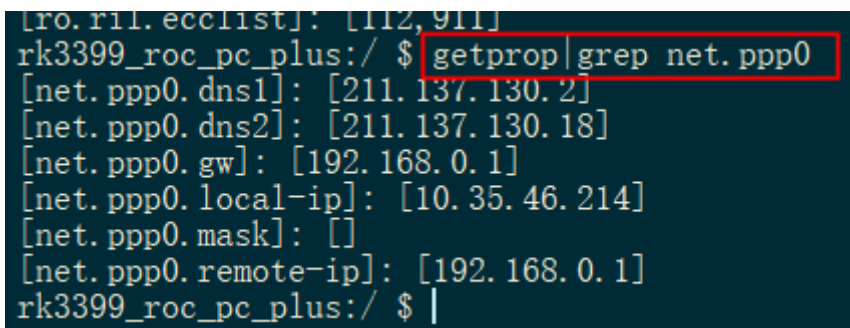
```
getprop net.ppp0
```

2. The following attributes correspond to the IP address, gateway, PDNS, and SDNS:

<code>net.ppp0.local-ip</code>	//IP address
<code>net.ppp0.gw</code>	//gateway
<code>net.ppp0.dns1</code>	//pdns

```
net.ppp0.dns2 //sdns
```

An example diagram is shown below:



```
[ro.ril.eccList]: [112, 911]
rk3399_roc_pc_plus:/ $ getprop | grep net.ppp0
[net.ppp0.dns1]: [211.137.130.2]
[net.ppp0.dns2]: [211.137.130.18]
[net.ppp0.gw]: [192.168.0.1]
[net.ppp0.local-ip]: [10.35.46.214]
[net.ppp0.mask]: []
[net.ppp0.remote-ip]: [192.168.0.1]
rk3399_roc_pc_plus:/ $ |
```

5.4 PPP Dial-up Not Supported by Device

Perform the following steps to compile and install PPP.

- Adding the Kernel Support

Run the **make menuconfig** command in the directory <android_dir>/kernel and select the following components:

Device Drivers --- >

Network device support --- >

<*> PPP(point-to-point protocol) support

[*] PPP multilink support (EXPERIMENTAL)

<*> PPP support for async serial ports

<*> PPP support for sync tty ports

<*> PPP Deflate compression

<*> PPP BSD-Compress compression

The following figure shows the specific steps.

```

Boot options --->
Userspace binary formats --->
Power management options --->
CPU Power Management --->
[*] Networking support --->
[*] Device Drivers --->
    Firmware Drivers --->
[ ] ACPI (Advanced Configuration and Power Interface) Support ----
File systems --->
[ ] Virtualization ----
Kernel hacking --->
Security options --->
-*- Cryptographic API --->
Library routines --->

```

```

-*- Device Tree and Open Firmware support --->
< > Parallel port support ----
[*] Block devices --->
<*> NVM Express block device
    Misc devices --->
        SCSI device support --->
<*> Serial ATA and Parallel ATA drivers (libata) --->
[*] Multiple devices driver support (RAID and LVM) --->
< > Generic Target Core Mod (TCM) and ConfigFS Infrastructure ----
[ ] Fusion MPT device support ----
    IEEE 1394 (FireWire) support --->
[*] Network device support --->
[ ] Open-Channel SSD target support ----
    Input device support --->
        Character devices --->
            I2C support --->
[*] SPI support --->
< > SPMI support ----
< > HSI support ----
    PPS support --->
    PTP clock support --->

```

```

[ ] HIPPI driver support
-*- PHY Device support and infrastructure --->
< > Micrel KS8995MA 5-ports 10/100 managed Ethernet switch
<*> PPP (point-to-point protocol) support
<*>     PPP BSD-Compress compression
<*>     PPP Deflate compression
[*]     PPP filtering
<*>     PPP MPPE compression (encryption)
[*]     PPP multilink support
<*>     PPP over Ethernet
<*>     PPP over L2TP
<*>     PPP on L2TP Access Concentrator
<*>     PPP on PPTP Network Server
<*>     PPP support for async serial ports
<*>     PPP support for sync tty ports
<*>     SLIP (serial line) support
[*]     CSLIP compressed headers
[ ]     Keepalive and linefill
[*]     Six bit SLIP encapsulation
<*>     USB Network Adapters --->
[*]     Wireless LAN --->

```



1. Press **Enter** to enter the corresponding directory. Press **y** to select the corresponding option or press **n** to cancel the selection.
2. The options mentioned before under "PPP (point-to-point) support" must be selected. In this example, all options are selected, and then are compiled into the kernel.

After completing these steps, compile the kernel. Download the generated kernel image file to the board. After the kernel is started, a PPP device node will be generated in the `/dev` directory. For example:

```
rk3399pro:/dev # ls ppp -l
crw-rw---- 1 radio vpn 108,  0 2020-11-29 13:23 ppp
rk3399pro:/dev #
```

- Downloading the PPP

Run the following command to get the PPP-related file:

```
wget -c https://ftp.samba.org/pub/ppp/ppp-2.4.5.tar.gz
```

Run the decompression command to decompress the obtained file and generate the directory **ppp-2.4.5**:

```
tar -zxvf ppp-2.4.5.tar.gz
```

After entering the directory **ppp-2.4.5**, run the following command to perform cross compilation:

```
# ./configure
# make CC=arm-linux-gcc
```



If the "arm-linux-gcc: command not found" error is displayed when the **make CC=arm-linux-gcc** command is executed, the arm-linux-gcc tool may not be installed on Linux or the tool may be configured incorrectly. You can

find the solutions online or contact Fibocom technical support personnel.

Copy the executable programs `pppd`, `chat`, `pppdump`, and `pppstats` generated under `pppd`, `chat`, `pppdump`, and `pppstats` folders in the `ppp-2.4.5` directory to the `/usr/sbin` directory of the development board. Till now, the PPP dial-up is prepared completely.

5.5 Unable to Use Data Services Normally

1. Query the dial-up state. Check whether the Local IP address, primary DNS, and secondary DNS are obtained.
2. If the dial-up fails, please check whether the APN is set in the Android settings. The APN can be set on the Android UI interface through the following path:

Settings > Network & Internet > More > Mobile network > Advanced > Access Point

Names (See section 5.9).

5.6 Unable to Make a Call

Check whether the SIM card supports the voice function. If so, confirm with Fibocom technical support personnel whether the module supports the call function under the current network standard.

5.7 What Can I Do If Text Messages Cannot Be Sent?

Check whether the SIM card supports the SMS function. If so, confirm with Fibocom technical support personnel whether the module supports the SMS function under the current network standard.

5.8 RIL Works Improperly

An Android device may fail to start the RIL due to many reasons. If this happens,

troubleshoot the fault as follows:

1. Check whether the RIL library is correctly loaded.

Enter the Android shell terminal, and then check whether there are contents mentioned in section [3.2](#) in init.rc or rid.rc. If not, add the corresponding content according to the steps in Section 3.2 to load the correct library file.

2. Check whether the RILD is running.

Access Android shell terminal: Enter “getprop | grep init.svc.ril-daemon”; and check the running status of RIL; the expected result is as follows: [init.svc.ril-daemon]: [running]

3. Check whether the USB serial port is enumerated.

Access Android shell terminal, and enter `ls /dev/ttyUSB* -l` to check whether the USB serial port is enumerated. If not, RIL cannot work normally.

5.9 How to Set APN?

If the APN is null, set the APN for dial-up operations in two ways.

5.9.1 Adding APN Manually

Perform the following operations (for debugging only):

- Install the MC66X module on the Android development board.
- Insert a SIM card into the development board.
- Power on the board and wait.
- Set the APN for PPP dial-up or NDIS dial-up:
 - Choose Settings > WiFi and Mobile Connection (more) > Mobile Network > APN.
 - Tap the **Menu** key, select **New APN**, enter the **Name**, **APN**, **MCC** and **MNC**, and tap the **Menu** key to save the setting.

Use the APN, authentication username and password provided by the operator.

- Select the set APN for dial-up, and run the following command to check whether the RIL dial-up is successful:
 1. View usb0 or ppp0 network interface status, and run the following Android OS command.

```
ifconfig
```

2. Get the local IP address after PPP dial-up or NDIS dial-up is successful, and run the following Android OS commands:

```
getprop net.ppp0.local-ip
```

```
getprop net.usb0.local-ip
```

5.9.2 Configuring the apns-conf.xml File

Android device path: /system/etc/apns-conf.xml

When modifying apns-conf.xml file, add information according to the actual information of the SIM card. The main information to be added includes APN, MCC, MNC, user, password, type, protocol, etc.

The following is the details of information added to an APN:

```
<apn carrier="China Mobile"
    apn="cmnet"
    mcc="460"
    mnc="04"
    user=""
    server=""
    password=""
    proxy=""
    port=""
    mmsproxy=""
    mmsport=""
    mmsc=""
    type="default,net,supl"
    preferred="true"
    localized_name="APN_NAME_CMNET"
    protocol="IPV4V6"
    roaming_protocol="IPV4V6"
/>
```

After updating the APN configuration list, telephony.db has been initialized because the Android device has been started up for many times. You must delete the database file and then start the device up again to load the new configuration file.

telephony.db path on the Android device:

/data/data/com.android.providers.telephony/databases/telephony.db

6 Properties Under Maintenance

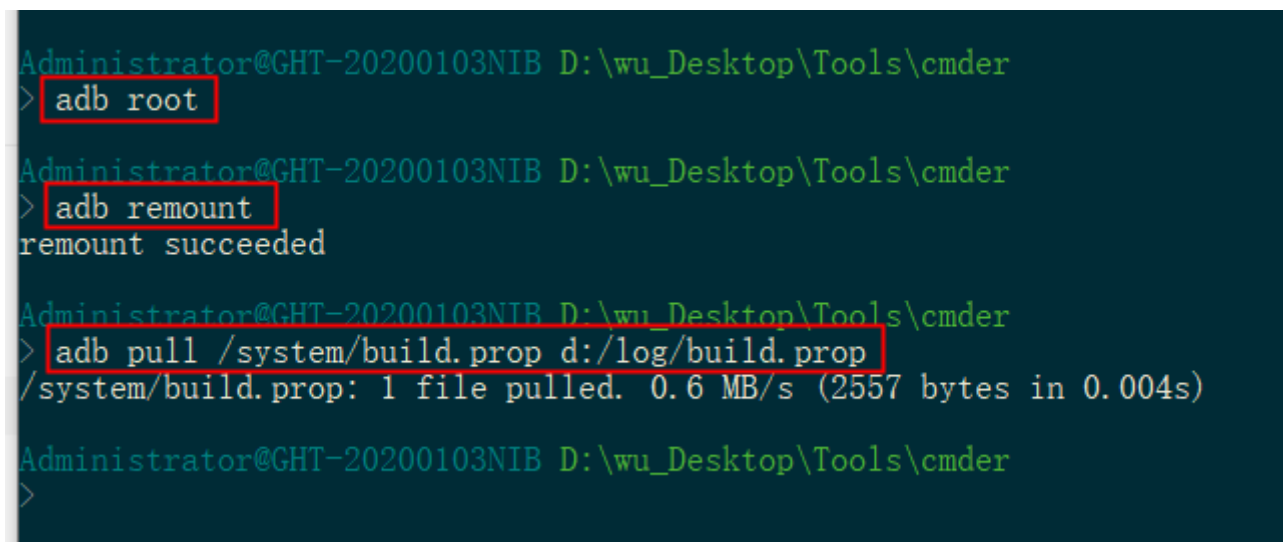
The RIL allows users to modify custom system properties. Customers can customize certain configurations by modifying the values of these properties. The methods and supported properties for modifying these properties are as follows.

6.1 Property Setting Methods

By adding properties and their values to the `/system/build.prop` file, these properties can be saved permanently and remain after the device restarts. The specific steps are as follows:

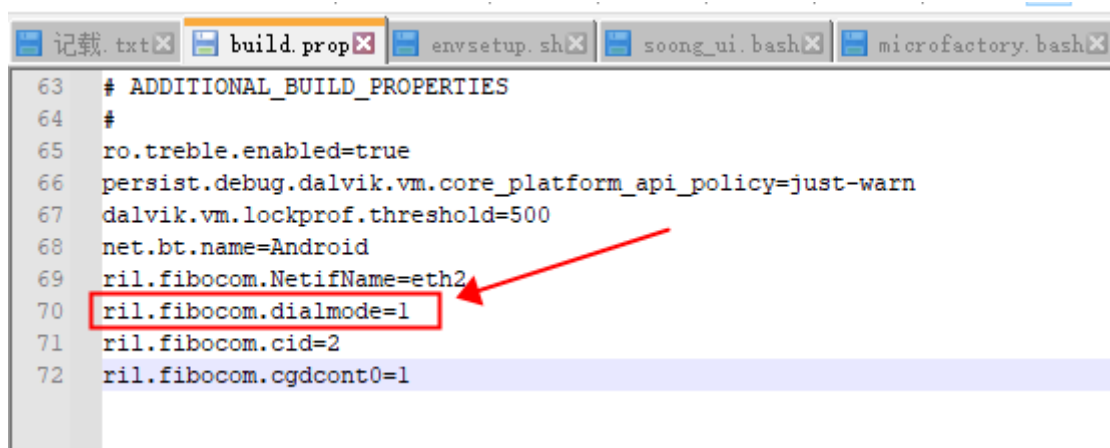
1. Run the `adb pull` command to pull the `/system/build.prop` file from the device.

<code>adb root</code>	① Enter the root mode.
<code>adb remount</code>	② Re-mount the system partition to the read and write mode.
<code>adb pull /system/build.prop <LOCAL_PATH></code>	③ Pull the build.prop file.



```
Administrator@GHT-20200103NIB D:\wu_Desktop\Tools\cmdr
> adb root
Administrator@GHT-20200103NIB D:\wu_Desktop\Tools\cmdr
> adb remount
remount succeeded
Administrator@GHT-20200103NIB D:\wu_Desktop\Tools\cmdr
> adb pull /system/build.prop d:/log/build.prop
/system/build.prop: 1 file pulled. 0.6 MB/s (2557 bytes in 0.004s)
Administrator@GHT-20200103NIB D:\wu_Desktop\Tools\cmdr
>
```

2. Suffix the property (taking `ril.fibocom.dialmode` as an example) to the `build.prop` file. Save the modification.

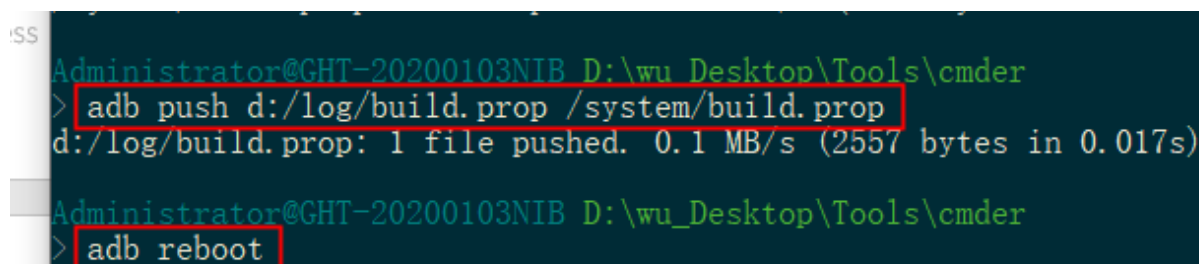


```
63 # ADDITIONAL_BUILD_PROPERTIES
64 #
65 ro.treble.enabled=true
66 persist.debug.dalvik.vm.core_platform_api_policy=just-warn
67 dalvik.vm.lockprof.threshold=500
68 net.bt.name=Android
69 ril.fibocom.NetifName=eth2
70 ril.fibocom.dialmode=1
71 ril.fibocom.cid=2
72 ril.fibocom.cgdcont0=1
```

3. Push the **build.prop** file to the original location in the device and restart the device.

`adb push d:/log/build.prop /system/build.prop` ① Push the build.prop file to the device again.

`adb reboot` ② Restart the device.



```
Administrator@GHT-20200103NIB D:\wu\Desktop\Tools\cmdr
> adb push d:/log/build.prop /system/build.prop
d:/log/build.prop: 1 file pushed. 0.1 MB/s (2557 bytes in 0.017s)
Administrator@GHT-20200103NIB D:\wu\Desktop\Tools\cmdr
> adb reboot
```

4. Run the **adb shell** command to enter the Andriod device. Then, you can run the **getprop|grep ril** command to view the set property. Example:

`adb shell` ① Enter the Andriod device.

`getprop | grep ril` ② Find the property related to RIL.

```
D:\wu\cmdr $ adb shell
rk3399_firefly:/ $ getprop|grep ril
[init.svc.ril-daemon]: [running]
[persist.ril.current_usbmode]: [74]
[ril.fibocom.dialmode]: [1]
[ril.fibocom.version]: [Fibocom_RIL_V7X.06.V1.0.5]
[ril.function.dataonly]: [1]
[rild.libargs]: [-d /dev/ttyACM0]
[rild.libpath]: [/system/lib64/libreference-ril.so]
[ro.boot.noril]: [true]
[ro.ril.ecclist]: [112,911]
rk3399_firefly:/ $
```

The property that we just set ourselves

6.2 ril.fibocom.dialmode

6.2.1 Introduction

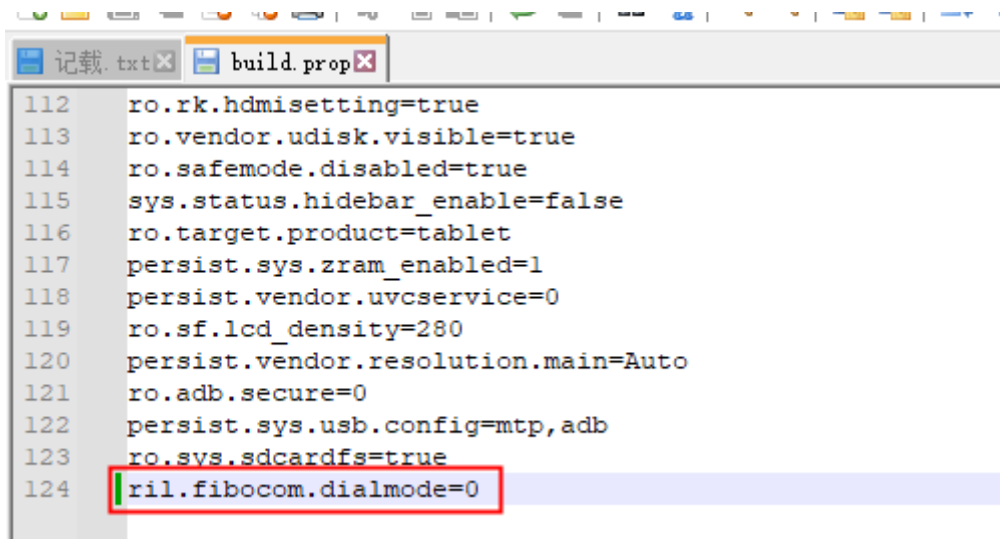
This property specifies whether the dial-up mode used by RIL is ECM, PPP or RNDIS.

6.2.2 Value

Table 5. Value range of dial-up mode

ril.fibocom.dialmode	Dial-up Mode
0 (Default)	PPP
1	ECM
2	RNDIS

6.2.3 Example



```
112 ro.rk.hdmisetting=true
113 ro.vendor.udisk.visible=true
114 ro.safemode.disabled=true
115 sys.status.hidebar_enable=false
116 ro.target.product=tablet
117 persist.sys.zram_enabled=1
118 persist.vendor.uvcservice=0
119 ro.sf.lcd_density=280
120 persist.vendor.resolution.main=Auto
121 ro.adb.secure=0
122 persist.sys.usb.config=mtp,adb
123 ro.sys.sdcardfs=true
124 ril.fibocom.dialmode=0
```

6.3 ril.fibocom.NetifName

6.3.1 Functions

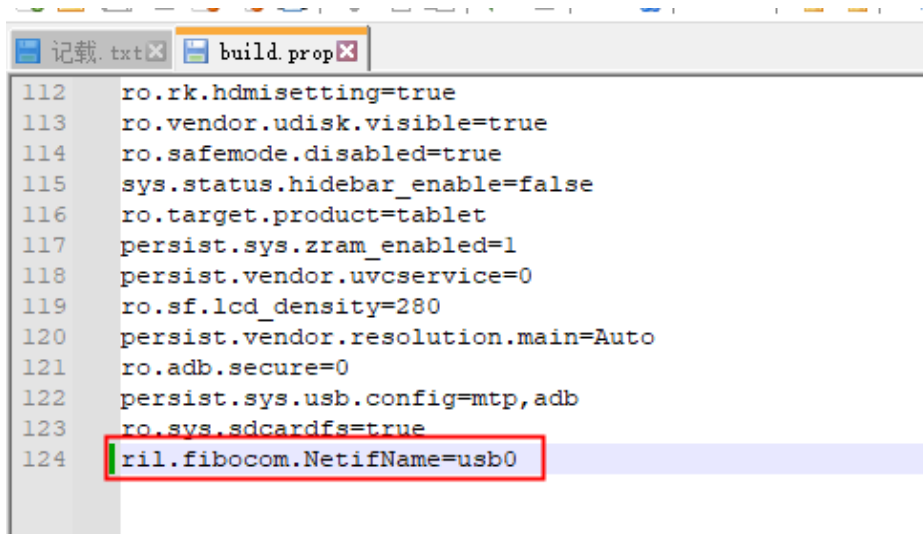
Customize the NIC used for RIL dial-up (commonly used for ECM dial-up).

When RIL initiates ECM dial-up, the order of selecting the NIC is: Specified NetifName > Preset RIL (such as eth1) > usb0. The default NIC "usb0" will be used only if the NIC is not found by the first two methods.

6.3.2 Value

The common values include usb0, eth0 and eth1. The default value is usb0.

6.3.3 Example



```
112 ro.rk.hdmisetting=true
113 ro.vendor.udisk.visible=true
114 ro.safemode.disabled=true
115 sys.status.hidebar_enable=false
116 ro.target.product=tablet
117 persist.sys.zram_enabled=1
118 persist.vendor.uvcservice=0
119 ro.sf.lcd_density=280
120 persist.vendor.resolution.main=Auto
121 ro.adb.secure=0
122 persist.sys.usb.config=mtp,adb
123 ro.sys.sdcardfs=true
124 ril.fibocom.NetifName=usb0
```

6.4 ril.fibocom.cid

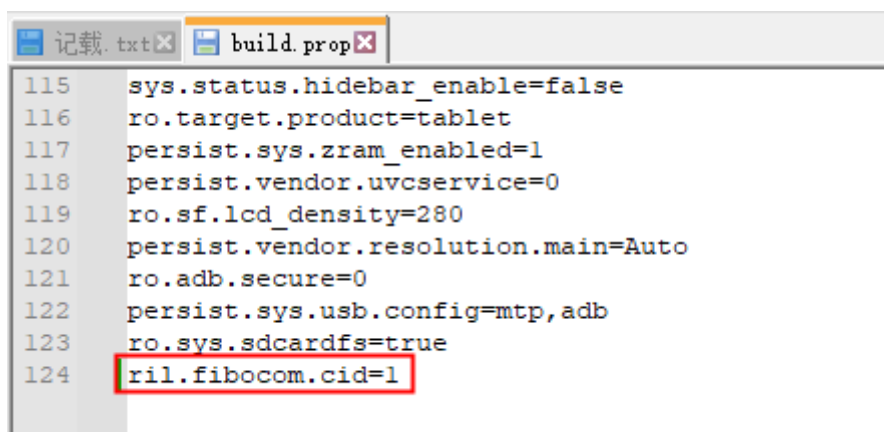
6.4.1 Functions

Customize the CID used by the module for dial-up.

6.4.2 Values

The value ranges from 1 to 7. The default value is 1.

6.4.3 Example



```
115 sys.status.hidebar_enable=false
116 ro.target.product=tablet
117 persist.sys.zram_enabled=1
118 persist.vendor.uvcservice=0
119 ro.sf.lcd_density=280
120 persist.vendor.resolution.main=Auto
121 ro.adb.secure=0
122 persist.sys.usb.config=mtp,adb
123 ro.sys.sdcardfs=true
124 ril.fibocom.cid=1
```

6.5 ril.fibocom.cgdcont0

6.5.1 Functions

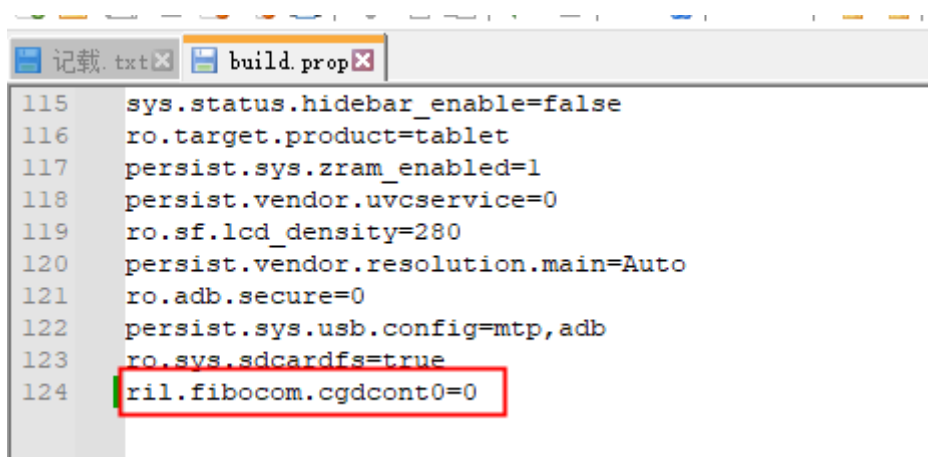
Customize the APN setting of cid0.

6.5.2 Values

0: Not set the APN of cid0 (default).

1: Set the APN of cid0 (default).

6.5.3 Example



```
115 sys.status.hidebar_enable=false
116 ro.target.product=tablet
117 persist.sys.zram_enabled=1
118 persist.vendor.uvcservice=0
119 ro.sf.lcd_density=280
120 persist.vendor.resolution.main=Auto
121 ro.adb.secure=0
122 persist.sys.usb.config=mtp,adb
123 ro.sys.sdcardfs=true
124 ril.fibocom.cgdcont0=0
```

6.6 ril.fibocom.usbmode

6.6.1 Functions

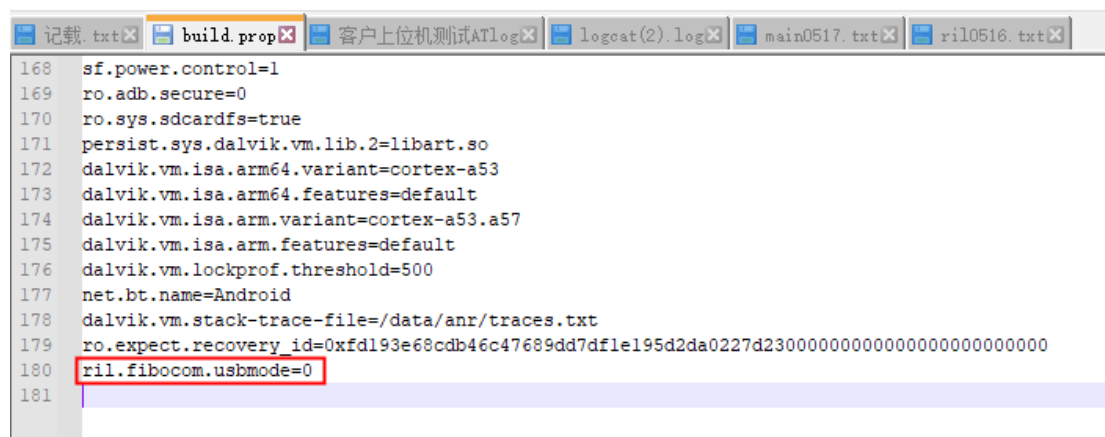
Modify the USB mode to a specified value.

6.6.2 Values

The value range is 0 and 31-72. If this property is not set or is set to 0, the USB mode of the module is not modified. At this time, there is no effect caused.

The default value is 0.

6.6.3 Example



The image shows a screenshot of a text editor window with multiple tabs. The active tab is 'build.prop'. The text in the editor is as follows:

```
168 sf.power.control=1
169 ro.adb.secure=0
170 ro.sys.sdcardfs=true
171 persist.sys.dalvik.vm.lib.2=libart.so
172 dalvik.vm.isa.arm64.variant=cortex-a53
173 dalvik.vm.isa.arm64.features=default
174 dalvik.vm.isa.arm.variant=cortex-a53.a57
175 dalvik.vm.isa.arm.features=default
176 dalvik.vm.lockprof.threshold=500
177 net.bt.name=Android
178 dalvik.vm.stack-trace-file=/data/anr/traces.txt
179 ro.expect.recovery_id=0x9fd193e68cdb46c47689dd7df1e195d2da0227d2300000000000000000000000000000000
180 ril.fibocom.usbmode=0
181
```

The line 'ril.fibocom.usbmode=0' is highlighted with a red rectangular box.

7 Appendix A References

Table 6. Term and Acronyms

Acronyms	Description
GSM	Global System for Mobile Communications
VID	Vendor ID
PID	Product ID
RIL	Radio Interface Layer
RILD	Radio Interface Layer Daemon
APN	Access Point Name