Perfect Wireless Experience
完美无线体验

# FIBOCOM NL Series

# Android RIL Adaptation Guide

Version：V1.0.1
Date：2019-10-22

## Applicability type

| No. | Product model | Description |
|---|---|---|
| 1 | NL668-CN | NL668-CN is for China markets |
| 2 | NL668-AM | NL668-AM is for America markets |
| 3 | NL668-EAU | NL668-EAU is for Europe and Asia Pacific markets |
| 4 | NL668-EU | NL668-EU is for Europe markets |
| 5 | NL668-JP | NL668-JP is for Japan markets |
| 6 | NL668-LA | NL668-LA is for Latin America markets |
| 7 | NL678-E | NL678-E is for Europe and Middle East markets |
| 8 | NL652-EU | NL652-EU is for Europe markets |

## Copyright

## 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.

## Versions

| Version | Author | Assessor | Approver | Update Date | Description |
|---|---|---|---|---|---|
| V1.0.0 | Li Shifeng | | | 2019-03-26 | initial version |
| V1.0.1 | Wang Mengying | Long Yiliang | Cheng Kai | 2019-10-22 | add some applicable models |
| | | | | | |
| | | | | | |

# Contents

# 1 Introduction

This document mainly introduces how to integrate the RIL (Radio Interface Layer) driver into customer system, and how to modify the configuration files for RIL service.

# 2 Android RIL driver overview

## 2.1 Supported features

Table1：Supported features

| Function | Support or not |
|---|---|
| SMS | YES |
| VOICE CALL | YES |
| DATA SERVICE | YES |
| SIM TOOL KIT | NO |

## 2.2 Supported android versions

Presently, Fibocom RIL driver supports the following android versions, see Table 2.

Table 2: Supported android versions

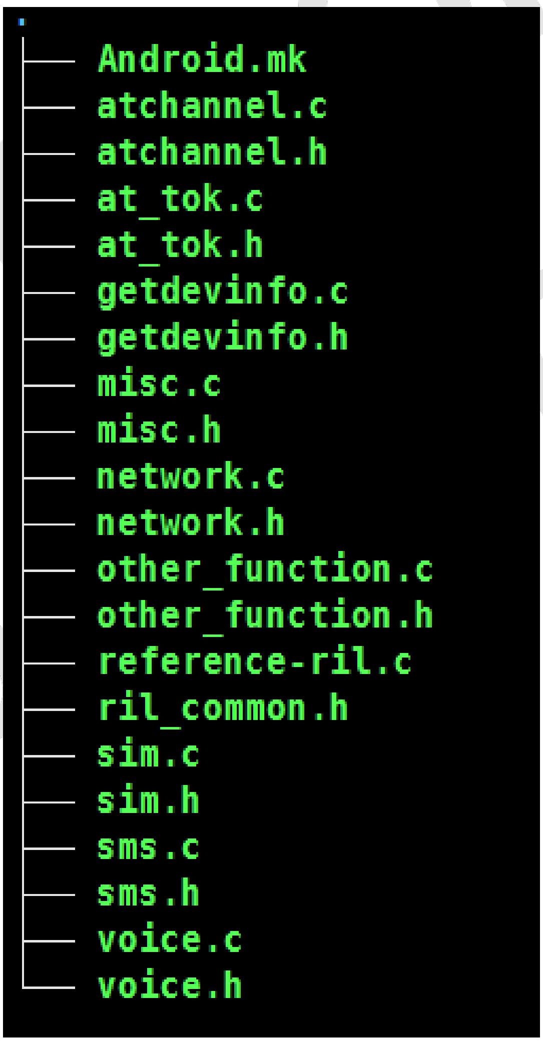
| Versions | Support or not |
|---|---|
| Android 2.x | NO |
| Android 3.x | NO |
| Android 4.x | YES |
| Android 5.x | YES |
| Android 6.x | YES |
| Android 7.x | YES |
| Android 8.x | YES |
| Android 9.x | To be supported later |

# 3 RIL integration

This chapter describes how to integrate the RIL library into customer Android OS, including introduction to RIL library source code, configuring Linux kernel drivers, adding USB devices’ PID/VID, loading RIL library files, as well as modifying the Android start script and execution permissions of RILD.

## 3.1 RIL library source code

Fibocom will provide RIL library file libreference-ril.so and RIL library source code for customers to integrate. The RIL library source code contains 21 files, see Figure 1.



```
.
├── Android.mk
├── atchannel.c
├── atchannel.h
├── at_tok.c
├── at_tok.h
├── getdevinfo.c
├── getdevinfo.h
├── misc.c
├── misc.h
├── network.c
├── network.h
├── other_function.c
├── other_function.h
├── reference-ril.c
├── ril_common.h
├── sim.c
├── sim.h
├── sms.c
├── sms.h
├── voice.c
└── voice.h
```

Figure1: RIL library source code

## 3.2 Configuring serial port driver with Linux kernel

The serial port driver needs to be configured with the Linux kernel, the configuration method as follows:

1) Enter into the kernel directory
2) Make menuconfig
3) Select the following components, then, saved and exited

Device drivers -> usb support - > usb serial converter support

USB driver for GSM and CDMA modems

* About linux kernel configuration method, you should configure according to the configuration rules of your Android OS, the method described in this document is only for reference.

## 3.3 Adding module PID/VID

1) Open the kernel source file option.c ( path is usually drivers/usb/serial/option.c). Find the option_ids array and add module PID and VID to the array. As shown below, take the steps to add VID (0x1508) and PID (0x1001) for NL668 as an example.

```
static const struct usb_device_id option_ids[] = {
……
{ USB_DEVICE(0x1508, 0x1001) },
……
 }
```

2) In the USB serial port driver, filter the NDIS interface. Since the USB serial port and the NDIS are both non-standard CDC devices, you need to prevent the NDIS port from being loaded by the USB serial port driver which causes that NDIS port driver cannot be loaded normally. The current interface number is filtered in the option_probe function, as shown below.

```
static int option_probe(struct usb_serial *serial,const struct usb_device_id *id)
……
if (serial->dev->descriptor.idVendor == cpu_to_le16(0x1508) &&
serial->dev->descriptor.idProduct == cpu_to_le16(0x1001) &&
serial->interface->cur_altsetting->desc.bInterfaceNumber >= 4) {
printk(KERN_INFO “Discover the 4th interface for fibocom\n”);
return –ENODEV;
}
```

**Note：**

If you don’t know NL668 / NL678 series modules’ PID / VID, you can get it by Android OS command "lsusb", you can also consult Fibocom FAE.

## 3.4 NL668 / NL678 module device loading test

As shown below, execute the following command in the adb shell to query USB port.

```
root@android:/ # ls /dev/ttyUSB*
ls /dev/ttyUSB*
/dev/ttyUSB0
/dev/ttyUSB1
/dev/ttyUSB2
/dev/ttyUSB3
```

Table 3: USB Port Function Description

| **Equipment name** | **effect** | **Remarks** |
|---|---|---|
| ttyUSB0 | DIAG | Get the modem log port |
| ttyUSB1 | MODEM | ppp dial-up port |
| ttyUSB2 | AT | AT command port: the RIL program sends an AT command request and receives a response |
| ttyUSB3 | NEMA | GPS port (not used) |

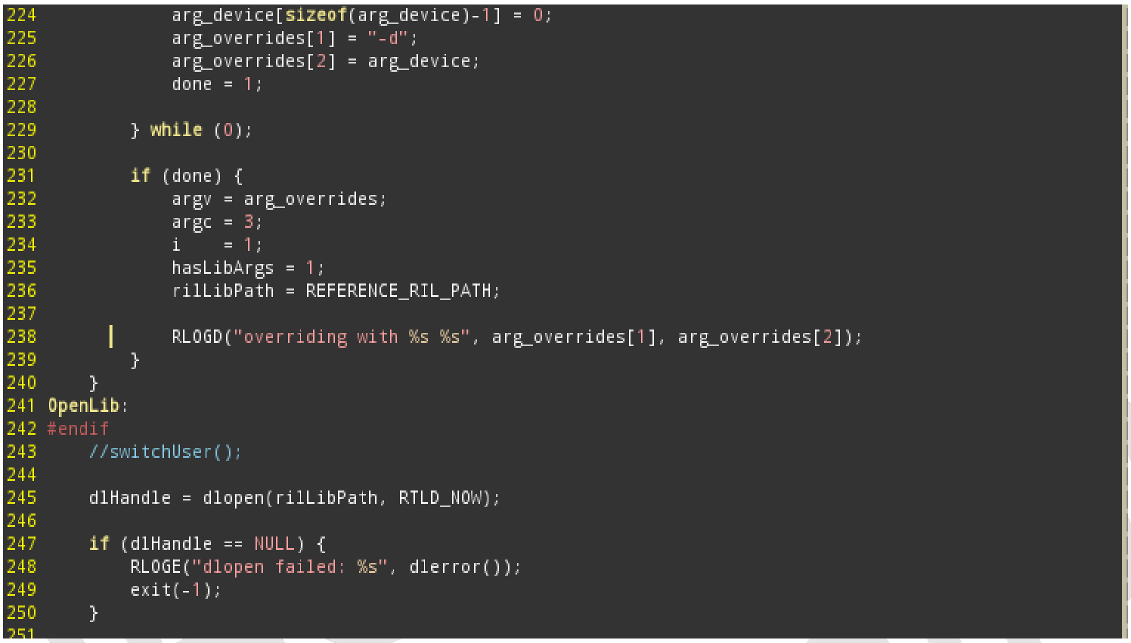
**Note：**

To adapt to RIL, you need to use AT+GTUSBMODE? to check if the USB mode is 17, if not, need set it to 17 with AT+GTUSBMODE=17.

If it is a special version, do not need to switch.

## 3.5 Modify RILD permissions

The RILD program requires root permission to be run. In the main function of the rild.c (<$android dir>/hardware/ril/rild/rild.c), remove the function swichUser to achieve this purpose, as shown in figure 2.



```
            arg_device[sizeof(arg_device)-1] = 0;
            arg_overrides[1] = "-d";
            arg_overrides[2] = arg_device;
            done = 1;

        } while (0);

        if (done) {
            argv = arg_overrides;
            argc = 3;
            i    = 1;
            hasLibArgs = 1;
            rilLibPath = REFERENCE_RIL_PATH;

            RLOGD("overriding with %s %s", arg_overrides[1], arg_overrides[2]);
        }
    }
OpenLib:
#endif
    //switchUser();

    dlHandle = dlopen(rilLibPath, RTLD_NOW);

    if (dlHandle == NULL) {
        RLOGE("dlopen failed: %s", dlerror());
        exit(-1);
    }

```

Figure2 Main function modification in RILD

## 3.6 Configuring Android startup scripts

Modify Android OS startup script init.rc, and add ril-damon process into startup item.

```
service ril-daemon /system/bin/rild -l /system/lib/libreference-ril.so -- -w 0
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
```

- -l specifies the RIL library load path.
- -w specified dial-up mode, 0 means ppp dial-up,1 means ECM dial-up,2 means NDIS dial-up.

You can also add the property item ril.fibocom.dialmode to the Android OS properties file (/system/build.prop).

ril.fibocom.dialmode=2：NDIS dialing

ril.fibocom.dialmode=1：ECM dialing

ril.fibocom.dialmode=0：ppp dialing

**Note：**

If the -w parameter is not configured in the init.rc, the RIL defaults to the ppp dial-up.

If you need to configure the dial-up, you can add the -w parameter to the init.rc, or add the property item ril.fibocom.dialmode to the Android OS properties file(/system/build.prop).

## 3.7 Loading RIL library

### 3.7.1 Manually load the RIL library

Open Android adb shell terminal, copy libreference-ril.so to /system/lib directory.

```
adb reboot
adb remount
adb push <$android dir>/out/target/product/…./system/lib/libreference-ril.so /system/lib
```

After Android6, if your system is 64-bit system, you need to copy libreference-ril.so to /system/lib64.

```
adb push <$android dir>/out/target/product/…./system/lib64/libreference-ril.so /system/lib64
```

### 3.7.2 Automatically package RIL library

Fibocom provides RIL source code, you should copy it to the directory of <Android the dir> /Hardware/ril/Reference-ril. When compiling system firmware, it will compile ril libraries, and automatically packaged in the image of system.img.

The RIL library source code can be compiled separately, as follows:

```
cd   <$android_dir>
source build/envsetup.sh
lunch < Compiled Projects >
mmm hardware/ril/reference-ril/
```

For system with 64-bit, it will generate RIL library libreference-ril.so file directory as bellow.

```
<$android dir>/out/target/product/…./system/lib64/libreference-ril.so
```

For system with 32-bit, it will generate RIL library libreference-ril.so file directory as bellow.

```
<$android dir>/out/target/product/…./system/lib/libreference-ril.so
```

After RIL library libreference-ril.so has been compiled, then, refer to chapter 3. 7.1, manually loaded RIL library file into the system.

# 4 NDIS Dial-up Mode

## 4.1 Loading Gobinet driver

Before NDIS dial-up, you need to load Gobinet NDIS driver, refer to the document 《FIBOCOM NL668 Application Guide_ Linux GobiNet Driver Loading_V1.0.1》.

To confirm whether the GobiNet driver is loaded successfully, use the following command to confirm if there is usb0.

```
android:/ # lsmod
Module                    Size   Used by
GobiNet                  57152   0
android:/ # ifconfig -a
usb0        Link encap:Ethernet   HWaddr e6:e6:3a:93:72:2a
            BROADCAST MULTICAST   MTU:1500   Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:0 TX bytes:0

sit0        Link encap:IPv6-in-IPv4
            NOARP   MTU:1480   Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:0 TX bytes:0

lo          Link encap:Local Loopback
            inet addr:127.0.0.1   Mask:255.0.0.0
            inet6 addr: ::1/128 Scope: Host
            UP LOOPBACK RUNNING   MTU:65536   Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
```

```
          RX bytes:0 TX bytes:0
```

## 4.2 Get address of local ip, primary dns, secondary dns

After NDIS dials successfully, local ip, primary dns and secondary dns will be written to the Android OS property, and named as net.usb0.local-ip, net.usb0.dns1, net.usb0.dns2. Execute the following Android OS commands, you can obtain them.

1) Get the local ip

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

2) Get primary DNS and secondary DNS

```
getprop net.usb0.dns1
getprop net.usb0.dns2
```

**Note:**

NDIS dial-up requires linux kernel supports DHCP protocol (the linux kernel supports it by default) .

# 5 ppp Dial-up Mode

## 5.1 Modify the ip-down, ip-up file

Before ppp dial-up, you should modify the files for ppp dial-up in the Android OS directory of /system/etc/ppp/.

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" ""
```

If the network interface is not ppp0, you need to modify the attribute item net.ppp0.xxx for ppp dial-up, such as net.modem.xxx.

After ppp dial-up successful, the addresses of local ip, primary dns, secondary dns are written to the attributes items, which are named as net.ppp0.local-ip, net.ppp0.dns1 and net.ppp0.dns2.

Execute the following Android OS command, you can obtain them.

1) Get local ip

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

2) Get the primary DNS and secondary DNS

```
getprop net.ppp0.dns1
getprop net.ppp0.dns2
```

**Note：**

The ip-up, ip-down file need to modify execute permission 555 .

# 6 Setting APN and querying dial-up status

If the APN is empty, it is necessary to set dail-up APN. There are two ways.

## 6.1 Manually add APN

Refer to the following steps(only for debug):

- Install the NL668 / NL678 module on the customer' device.
- Insert a SIM card to the device.
- Turn on the power and wait for RILD running.
- Set the APN used for ppp dial-up or NDIS dial-up.
  - Settings -> Wireless & Network Cables (More) -> Mobile Network -> Access Point Name (APN)
  - Press the menu button to select "New APN", enter Name, APN, MCC, MNC and press the menu button to save.

  If the APN carrier provided needs to be authenticated, you should enter username and password which provided by carrier to authenticate APN.

After performing the above operations, you can use the following Android OS command to confirm whether it dials successfully.

1) Execute the following Android OS command, you can query the network interface of usb0 or ppp0.

```
ifconfig
```

2) After ppp or NDIS dials successfully, execute the Android OS command to obtain the local ip.

```
getprop net.ppp0.local-ip
getprop net.usb0.local-ip
```

## 6.2 Configuring apns-conf.xml file

Android devices "apns-conf.xml" path:/system/etc/apns-conf.xml

modify apns-conf.xml file, add according to the actual information of the SIM card. The parameters are apn,mcc,mnc,user,password,type,protocol,etc…

The following is an information added to an APN listed:

```
<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, since the Android device is started many times, telephony.db has been initialized, so you need to delete the database, and then start it again to load the new configuration file.

Android devices “telephony.db” path ：/data/data/com.android.providers.telephony/databases/telephony.db

# 7 Debugging method

1) Open cmd terminal of windows, execute the following command to capture log.

```
adb logcat－b radio  －v time
```

2) Execute the following command, you can capture Android OS log.

```
adb logcat -v time
```

3) Execute the following command, you can capture pppdial-up pppd log.

```
adb logcat -s pppd
```

4) In some special usage scenarios, such as performing tests on many devices or for a long time, you can

save the log in the Android OS by executing the following command.

```
adb shell
logcat －b radio －v time  > <$android_dir>/<filename>
```

5) Execute the following command, you can extract the log which saved in Android OS.

```
adb pull <filename> <$local_directory>
```

# 8 Log tag instructions

The following table lists some tag about RIL log categories, see Table 4.

Table 4 Log tag

| TAG | **file** |
|---|---|
| RLOG-RIL | hardware/ril/reference-ril/reference-ril.c |
| RLOG-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 |

# 9 FAQs

## 9.1 Unable to access network

If your Android device can't access network, you should check the following points.

1. Query dial-up state, whether the local ip, primary DNS and Secondary DNS is available.
2. Check Android device whether APN has be set, which can be set by Android OS UI.

   Refer to the following procedure:

   Settings -> Network& Internet ->More ->Mobile network ->Advanced -> Access Point Names ( see chapter 6 ) .

## 9.2 Can't make a voice call

If your Android device can't make a voice call, first confirm the SIM card supports voice,then contact with Fibocom FAE to confirm if the module supports voice call in the current network environment.

## 9.3 Unable to send short message

If your Android device can't send short message, first confirm the SIM card supports send short message,then contact with Fibocom FAE to confirm whether the module supports SMS features in the current network environment.

## 9.4 Running abnormally with load RIL library

There are many reasons for why the Android devices can't start RIL. If this happens, please refer to the following steps.

1) Check if the RIL library is loaded normally.

Enter into Android adb shell terminal, execute command "cat init.rc | grep ril", then terminal will appear information of "service ril-daemon /vendor/bin/hw/rild -l /system/lib64/libreference-ril.so --w 0", if there is no information, please refer to chapter 3.6 to modify the system startup script of init.rc.

2) Check if RILD is running

On Android adb shell terminal, Execute the command "getprop | grep init.svc.ril-daemon", check the running status of RIL, the expected result is "[init.svc.ril-daemon]: [running]"

3) Check if the USB serial port is enumerated

On Android adb shell terminal, execute command "ls /dev/ttyUSB* -l" to check if the usb serial port is enumerated.

# 10 Appendix A References

Table 5, Related documents

| File name | Remarks |
|---|---|
| 《FIBOCOM Under NL668 Application Guide_Linux GobiNet driver loading》 | Linux GobiNet drivers loading for NL668/NL678 modules. |
| | |

Table 6, Terms and abbreviations

| Abbreviation | Description |
|---|---|
| GSM | Global mobile communication system |
| VID | Vendor ID |
| PID | Product ID |

| **Abbreviation** | **Description** |
|---|---|
| RIL | Radio interface layer |
| RILD | Radio interface layer daemon |
| APN | Access point name |