

QMI\_WWAN驱动集成及拨号指南\_Linux（嵌入式）

V2.2

版权声明

版权所有©2021深圳市广和通无线股份有限公司。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

注意

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

商标申明

G:\repos\cbb\zh\static\images\logo.png 为深圳市广和通无线股份有限公司的注册商标，由所有人拥有。

联系方式

公司网址： https://www.Fibocom.com/

总部地址：深圳市南山区西丽街道西丽社区打石一路深圳国际创新谷六栋A座10-14层

总机：+86 755-26733555

目录

[修订记录 2](#_Toc86854961)

[1 引言 3](#_Toc86854962)

[1.1 目的 3](#_Toc86854963)

[1.2 适用范围 3](#_Toc86854964)

[1.3 预置条件 4](#_Toc86854965)

[2 USB端口信息 5](#_Toc86854966)

[3 集成Fibocom模块QMI\_WWAN驱动信息 9](#_Toc86854967)

[3.1 QMI\_WWAN驱动代码结构 9](#_Toc86854968)

[3.2 配置Linux内核编译QMI\_WWAN驱动 9](#_Toc86854969)

[3.2.1 以builtin方式编译 9](#_Toc86854970)

[3.2.2 以.ko方式编译 10](#_Toc86854971)

[3.3 加载QMI\_WWAN驱动 10](#_Toc86854972)

[3.4 检测Fibocom模块设备加载 11](#_Toc86854973)

[4 Fibocom模块拨号 12](#_Toc86854974)

[4.1 QMI拨号流程 12](#_Toc86854975)

[4.2 QMI单路拨号 13](#_Toc86854976)

[4.3 QMI多路拨号 15](#_Toc86854977)

# 修订记录

|  |  |
| --- | --- |
| V2.2（2021-11-02） | 增加适配模块型号，FM101系列，FM130系列和FM160系列  修改ECM、MBIM端口信息描述 |
| V2.1（2021-08-25） | 增加多路拨号描述 |
| V2.0（2021-06-30） | 对外发布文档采用新版本，版本号从三位数修改为两位数 |
| V1.0.0（2021-02-05） | 初始版本 |

# 引言

Fibocom模块在嵌入式Linux上主要映射出Modem/DIAG等串口、标准ECM (or RmNet)网卡等端口。

Modem、DIAG等串口加载的是系统option驱动，ECM网卡端口使用系统cdc\_ether驱动，RmNet网卡则使用GobiNet驱动，基于高通的QMI接口则使用QMI\_WWAN驱动。

在嵌入式Linux上使用我们模块设备时，为了正常加载option、GobiNet 、QMI\_WWAN或者cdc\_ether驱动，需要设置相关Linux内核信息并将我们产品的信息正确添加到系统中。

## 目的

本文主要介绍嵌入式Linux下host侧QMI\_WWAN驱动的编译环境搭建方法，为Fibocom模块在嵌入式Linux上正常使用QMI\_WWAN驱动提供指导。

## 适用范围

适用于测试部/市场导入人员参考文档或指导客户搭建嵌入式Linux Host驱动编译环境使用。

QMI\_WWAN驱动目前适用的产品型号如下表所示：

表1. 适用型号

|  |  |  |
| --- | --- | --- |
| 序号 | 产品型号 | 说明 |
| 1 | NL95X系列 | M.2接口4G通信模组。 |
| 2 | FG150系列 | LGA封装，5G通信模组。 |
| 3 | FM150系列 | M.2接口5G通信模组。 |
| 4 | NL668系列 | NA |
| 5 | FM100系列 | M.2接口4G通信模组。 |
| 6 | FG101系列 | LGA封装，5G通信模组。 |
| 7 | FM101系列 | M.2接口4G通信模组。 |
| 8 | FM130系列 | M.2接口5G通信模组。 |
| 9 | FM160系列 | M.2接口5G通信模组。 |

## 预置条件

* 有嵌入式Linux硬件开发平台；
* 已经下载好对应开发平台Linux镜像开源代码及ubuntu根文件系统镜像；
* 有一台Linux服务器，且可以编译开发板镜像系统；
* 用户具有root操作权限；
* 有一块Fibocom模块。

# USB端口信息

表2. USB端口信息

|  |  |  |
| --- | --- | --- |
| GTUSBMODE: 17 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0104 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | DIAG | Device Diagnostic Interface |
| 1 | Modem | Modem Connector |
| 2 | AT | Device Application Interface |
| 3 | Pipe | Device Pipe |
| 4 | RmNet | Wireless Data Device Ethernet Adapter |
| 5 | ADB | Android Composite ADB Interface |
| GTUSBMODE: 18 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0105 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | DIAG | Device Diagnostic Interface |
| 1 | Modem | Modem Connector |
| 2 | AT | Device Application Interface |
| 3 | Pipe | Device Pipe |
| 4 | ECM | ECM communication class interface |
| 5 | ECM | ECM data class interface |
| 6 | ADB | Android Composite ADB Interface |
| GTUSBMODE: 20 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0107 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | Modem | Modem Connector |
| GTUSBMODE: 21 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0108 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | Modem | Modem Connector |
| 1 | AT | Device Application Interface |
| GTUSBMODE: 22 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0109 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | Modem | Modem Connector |
| 1 | AT | Device Application Interface |
| 2 | RmNet | Wireless Data Device Ethernet Adapter |
| GTUSBMODE: 23 | | |
| Vendor ID: 0x2CB7 Product ID: 0x010A | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | Modem | Modem Connector |
| 1 | AT | Device Application Interface |
| 2 | ECM | ECM communication class interface |
| 3 | ECM | ECM data class interface |
| GTUSBMODE: 28 | | |
| Vendor ID: 0x2CB7 Product ID: 0x010F | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | MBIM | MBIM communication class interface |
| 1 | MBIM | MBIM data class interface |
| GTUSBMODE: 29 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0110 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | MBIM | MBIM communication class interface |
| 1 | MBIM | MBIM data class interface |
| 2 | AT | Device Application Interface |
| 3 | DIAG | Device Diagnostic Interface |
| GTUSBMODE: 30 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0111 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | MBIM | MBIM communication class interface |
| 1 | MBIM | MBIM data class interface |
| 2 | Modem | Modem Connector |
| 3 | DIAG | Device Diagnostic Interface |
| 4 | AT | Device Application Interface |
| (Default) GTUSBMODE: 32 | | |
| Vendor ID: 0x2CB7 Product ID:0x0104 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | DIAG | Device Diagnostic Interface |
| 1 | Modem | Modem Connector |
| 2 | AT | Device Application Interface |
| 3 | Pipe | Device Pipe |
| 4 | RmNet | Remote Network |
| GTUSBMODE: 33 | | |
| Vendor ID: 0x2CB7 Product ID: 0x0105 | | |
| 接口号 | 接口名称 | 接口功能 |
| 0 | DIAG | Device Diagnostic Interface |
| 1 | Modem | Modem Connector |
| 2 | AT | Device Application Interface |
| 3 | Pipe | Device Pipe |
| 4 | ECM | ECM communication class interface |
| 5 | ECM | ECM data class interface |

|  |  |  |  |
| --- | --- | --- | --- |
|  | C:\Users\Administrator\Desktop\图层 4.png |  | ECM/MBIM和ADB端口加载系统驱动，不需要修改Host侧文件。 |

# 集成Fibocom模块QMI\_WWAN驱动信息

QMI\_WWAN和GobiNet都是Linux上高通模块的驱动程序，使用的都是QMI接口， 分界线就是Linux3.4内核，之前版本的Linux内核并没有添加QMI\_WWAN，所以高通开发了GobiNet驱动，包括其4G驱动与4G拨号程序。在Linux3.4内核以后，已经将QMI添加到内核中了，基于原生的QMI\_WWAN驱动，我们开发Fibocom\_QMI\_WWAN\_Driver供客户使用，目前可支持QMI单路与多路拨号。

## QMI\_WWAN驱动代码结构

如下所示，驱动以源代码的形式提供，由用户在自己的系统编译。

qmi\_wwan\_f/

├── qmi\_wwan\_f.c

├── Makefile

├── README

## 配置Linux内核编译QMI\_WWAN驱动

将QMI\_WWAN代码源码文件复制到待编译镜像的kernel/drivers/net/usb（QMI\_WWAN驱动的makefile文件除外）目录下。

### 以builtin方式编译

1. kernel/drivers/net/usb/Makefile中增加以下内容：

obj-y += qmi\_wwan\_f.o

qmi\_wwan\_f -objs: = qmi\_wwan\_f.o

之后每次编译内核都会自动编译QMI\_WWAN驱动。

1. 修改如下路径下各modules.builtin文件：

“kernel/modules.builtin

kernel/drivers/modules.builtin

kernel/drivers/net/modules.builtin

kernel/drivers/net/usb/modules.builtin”

1. 在各文件中增加 “kernel/drivers/net/usb/qmi\_wwan\_f.ko”

之后镜像启动后都会自动加载QMI\_WWAN驱动。

|  |  |  |  |
| --- | --- | --- | --- |
|  | C:\Users\Administrator\Desktop\图层 4.png |  | 此处增加 “kernel/drivers/net/usb/qmi\_wwan\_f.ko”，是提供一种说明，qmi\_wwan\_f已编译进内核，实际kernel/drivers/net/usb目录下并无qmi\_wwan\_f.ko文件。如果找不到modules.builtin文件，请忽略即可。不能修改其他目录下的modules.builtin文件。 |

### 以.ko方式编译

kernel/drivers/net/usb/Makefile中增加以下内容：

obj-m := qmi\_wwan\_f.o

qmi\_wwan\_f-objs := qmi\_wwan\_f.o

之后每次编译内核都会自动编译QMI\_WWAN驱动。

## 加载QMI\_WWAN驱动

1. 若QMI\_WWAN驱动以.ko的方式编译，则qmi\_wwan\_f.ko作为模块加入到系统中，因此会自动进行加载。

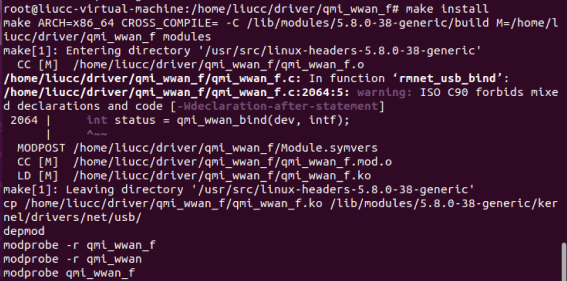


图1. qmi\_wwan\_f驱动加载

1. 如果驱动未进行加载，请将qmi\_wwan\_f.ko复制到系统中，以下有两方法可以手动进行驱动加载：
2. 使用insmod命令加载QMI\_WWAN驱动：

sudo insmod qmi\_wwan\_f.ko

1. 使用modprobe命令加载QMI\_WWAN驱动：

cp -f GobiNet.ko /lib/modules/’uname –r’/kernel/drivers/net/usb/

modprobe GobiNet

1. 使用usb-devices命令检查驱动加载信息：

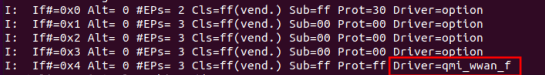


图2. qmi\_wwan\_f驱动加载成功

1. 使用ifconfig -a命令检查网卡信息，如果显示有“wwan0”表示驱动加载成功，如下所示wwan0为新增加的口：

wwan0: flags=193<UP,RUNNING,NOARP> mtu 1500

ether 0a:44:72:90:32:b4 txqueuelen 1000 (以太网)

RX packets 2 bytes 217 (217.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 3 bytes 196 (196.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

|  |  |  |  |
| --- | --- | --- | --- |
|  | C:\Users\Administrator\Desktop\图层 4.png |  | 不同的Linux版本可能会显示不同的名称。 |

## 检测Fibocom模块设备加载

嵌入式Linux Host系统镜像编译完成后，升级到开发板中，使用“ls –l /dev/ttyUSB\*”命令查看模块端口信息，若设备正常挂载，将会有如下内容返回：

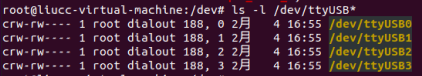


图4. 查看模块端口信息

# Fibocom模块拨号

QMI\_WWAN驱动加载完毕，即可进行fibocom-dial拨号，由于QMI\_WWAN驱动与GobiNet驱动皆对应QMI接口，因此都采用QMI方式进行数据拨号。

## QMI拨号流程

模块开机： 模块的供电稳定后，FG150/FM150/NL952系列模块上电后开机；

模块关机：模块断电。推荐断电和再次上电间隔12秒以上（视断电时电容放电时间而定）建议相邻两次开机间隔大于90秒，相邻两次断电大于300秒。

模块开机

使用lsusb查询模块是否初始化完成。

查询模块是否初始化完成

最少等90秒。仍然没有端口则认为本次开机失败。重新进入模块开机动作；

或者延时5秒后连续给模块发AT，等模块回复OK。最长等90秒。

模块枚举出ttyUSB\*、网口

进入/drivers/qmi\_wwan\_f目录；

执行make install命令；

加载qmi\_wwan\_f驱动；

usb-devices查看驱动加载成功。

加载qmi\_wwan\_f驱动

成功获取IPV4地址说明单路拨号成功；

假如最少连续拨号5次仍然获取不到IPV4地址，复位模块。

如果需要PIN码则先用AT+CPIN按AT手册设置PIN码。

执行./fibocom-dial拨号

验证wwan0是否获取到IP；

假如连续查5次，wwan0都未获得IP，则复位模块；

假如连续查90秒返回值不正确，则复位模块。

ifconfig –a查看wwan0是否获取到ip

查询结果一直连续加载数据包收发则表明成功ping通，如果连续ping超过5次仍然ping不通，复位模块。

ping www.baidu.com查看其是否能ping通网

实网测试是否能正常应用，如果加载不出网页，复位模块。

实网测试

## QMI单路拨号

1. QMI\_WWAN驱动加载完毕，即可进入/fibocom-dial目录（根据个人目录指定），首先执行make命令进行编译，生成fibocom-dial二进制文件，其次执行./fibocom-dial进行数据拨号，成功获取到IPV4地址，如下所示。

root@ght-Lenovo-V130-14IKB:~/lcc/tool/Fibocom-dial\_Linux\_Tool\_V2.0.8/src# ./fibocom-dial

[07-19\_14:28:57:848] Fibocom-dial\_Linux\_Tool\_V2.0.8

[07-19\_14:28:57:848] ./fibocom-dial profile[1] = (null)/(null)/(null)/0, pincode = (null)

[07-19\_14:28:57:849] socket[3] successfuly!

[07-19\_14:28:57:849] Waiting client to connect.....

[07-19\_14:28:57:851] Find /sys/bus/usb/devices/1-1 idVendor=2cb7 idProduct=0104

[07-19\_14:28:57:851] Find /sys/bus/usb/devices/1-1:1.4/net/wwan0

[07-19\_14:28:57:852] Find usbnet\_adapter = wwan0

[07-19\_14:28:57:852] Find /sys/bus/usb/devices/1-1:1.4/usbmisc/cdc-wdm0

[07-19\_14:28:57:852] Find qmichannel = /dev/cdc-wdm0

[07-19\_14:28:57:852] qmichannel(/dev/cdc-wdm0) usbnet\_adapter(wwan0)

[07-19\_14:28:57:852] pcie mode

[07-19\_14:28:57:852] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_14:28:57:852] access /sys/class/net/wwan0/qmap\_mode

[07-19\_14:28:57:853] qmap\_mode = 1, muxid = 0x81, qmap\_netcard = wwan0

[07-19\_14:28:57:853] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_14:28:57:853] access /sys/class/net/wwan0/qmap\_mode

[07-19\_14:28:57:853] qmap\_mode = 1, muxid = 0x81, qmap\_netcard = wwan0

[07-19\_14:28:57:853] qmap\_mode=1

[07-19\_14:28:57:865] cdc\_wdm\_fd = 9

[07-19\_14:28:57:865] write triger\_event: 4098 to qmidevice\_control\_fd

[07-19\_14:28:57:955] QmiWwanGetClientID: QMIType = 1 clientid 14

[07-19\_14:28:57:955] Get clientWDS = 14

[07-19\_14:28:57:987] QmiWwanGetClientID: QMIType = 2 clientid 4

[07-19\_14:28:57:987] Get clientDMS = 4

[07-19\_14:28:58:019] QmiWwanGetClientID: QMIType = 3 clientid 2

[07-19\_14:28:58:019] Get clientNAS = 2

[07-19\_14:28:58:051] QmiWwanGetClientID: QMIType = 11 clientid 2

[07-19\_14:28:58:051] Get clientUIM = 2

[07-19\_14:28:58:083] QmiWwanGetClientID: QMIType = 26 clientid 1

[07-19\_14:28:58:083] Get clientWDA = 1

[07-19\_14:28:58:115] requestBaseBandVersion 89602.1000.00.04.08.21

[07-19\_14:28:58:147] qmap\_settings.rx\_urb\_size = 4096

[07-19\_14:28:58:211] sim\_select = 0

[07-19\_14:28:58:243] curr\_ints\_sim1

[07-19\_14:28:58:243] curr\_ints->CardState is 1

[07-19\_14:28:58:243] curr\_ints->NumApp is 1

[07-19\_14:28:58:243] AppType = 2

[07-19\_14:28:58:243] requestGetSIMStatus SIMStatus: SIM\_READY

[07-19\_14:28:58:275] requestGetICCID DeviceICCID: 898600F0261831632928

[07-19\_14:28:58:307] requestGetIMSI DeviceIMSI: 460026092319783

[07-19\_14:28:58:339] requestGetProfile[1] ///0

[07-19\_14:28:58:371] requestRegistrationState2 MCC: 460, MNC: 0, PS: Detached, DataCap: UNKNOW

[07-19\_14:28:58:371] write signo: 12 to signal\_control\_fd

[07-19\_14:28:58:371] epoll fd = 6, events = 0x0001

[07-19\_14:28:58:371] get signo: 12

[07-19\_14:28:58:403] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED

[07-19\_14:28:58:403] usbnet\_link\_change :link:0

[07-19\_14:28:58:403] enter udhcpc\_stop

[07-19\_14:28:58:403] enter fibo\_set\_driver\_link\_state

[07-19\_14:28:58:404] if\_link\_down wwan0

[07-19\_14:28:58:404] write signo: 10 to signal\_control\_fd

[07-19\_14:28:58:404] epoll fd = 7, events = 0x0000

[07-19\_14:28:58:404] epoll fd = 6, events = 0x0001

[07-19\_14:28:58:404] get signo: 10

[07-19\_14:28:58:404] usbnet\_link\_change :link:0

[07-19\_14:28:59:011] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE

[07-19\_14:28:59:011] epoll fd = 6, events = 0x0000

[07-19\_14:28:59:011] epoll fd = 7, events = 0x0001

[07-19\_14:28:59:011] write signo: 12 to signal\_control\_fd

[07-19\_14:28:59:011] epoll fd = 6, events = 0x0001

[07-19\_14:28:59:011] get signo: 12

[07-19\_14:28:59:075] usbnet\_link\_change :link:1

[07-19\_14:28:59:107] enter fibo\_set\_driver\_link\_state

[07-19\_14:28:59:107] if\_link\_up wwan0

[07-19\_14:28:59:108] IPv4 MTU: 1500

[07-19\_14:28:59:108] IPv4 Address: 10.9.199.226

[07-19\_14:28:59:108] IPv4 Netmask: 30

[07-19\_14:28:59:108] IPv4 Gateway: 10.9.199.225

[07-19\_14:28:59:108] IPv4 DNS1: 211.137.130.2

[07-19\_14:28:59:108] IPv4 DNS2: 211.137.130.4

1. 拨号成功后，输入ifconfig –a命令，查询wwan0是否获取到IP，如下所示，IP地址与上面拨号成功后所获取到的IP地址相同：

wwan0: flags=193<UP,RUNNING,NOARP> mtu 1500

inet 10.9.199.226 netmask 255.255.255.252

inet6 fe80::844:72ff:fe90:32b4 prefixlen 64 scopeid 0x20<link>

ether 0a:44:72:90:32:b4 txqueuelen 1000 (以太网)

RX packets 2 bytes 217 (217.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 3 bytes 196 (196.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

1. ping包测试：

root@ght-Lenovo-V130-14IKB:~/lcc/qmi\_wwan\_f/Fibocom\_QMI\_WWAN\_Driver\_V1.0.3# ping 8.8.8.8

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.

64 bytes from 8.8.8.8: icmp\_seq=1 ttl=52 time=105 ms

64 bytes from 8.8.8.8: icmp\_seq=2 ttl=52 time=75.9 ms

^C

--- 8.8.8.8 ping statistics ---

2 packets transmitted, 2 received, 0% packet loss, time 1002ms

## QMI多路拨号

1. 多路拨号需要在驱动加载前配置完成多PDN，打开源码qmi\_wwan\_f.c文件，将117行的qmap\_mode设置为多路，如下设置为4，表示当前可支持4路拨号（当前最大可支持4路拨号）。

static uint \_\_read\_mostly **qmap\_mode = 4;**

1. 进入qmi\_wwan\_f驱动源码所在的路径下，执行“make install”，安装qmi\_wwan\_f驱动，同理3.2.1安装驱动的操作。
2. 驱动安装完毕，通过“ifconfig –a”命令即可查看当前根据物理网卡wwan0映射出wwan0.1、wwan0.2、wwan0.3、wwan0.4四个虚拟网卡。
3. 进入拨号工具所在路径，执行make命令进行编译，生成fibo\_qmimsg\_server和multi-pdn-manager二进制文件。首先执行“./fibo\_qmimsg\_server &”，启动启动qmi消息服务进程：

root@ght-Lenovo-V130-14IKB:~/lcc/tool/Fibocom-dial\_Linux\_Tool\_V2.0.8/src# **./fibo\_qmimsg\_server &**

Find /sys/bus/usb/devices/1-1 idVendor=2cb7 idProduct=0104

Find /sys/bus/usb/devices/1-1:1.4/usbmisc/cdc-wdm0

Will use cdc-wdm /dev/cdc-wdm0

qmi\_proxy\_init enter

qmi\_proxy\_loop enter thread\_id 140040038012672

send\_qmi\_timeout ret=110

link\_prot 2

ul\_data\_aggregation\_protocol 5

dl\_data\_aggregation\_protocol 5

dl\_data\_aggregation\_max\_datagrams 63

dl\_data\_aggregation\_max\_size 16384

ul\_data\_aggregation\_max\_datagrams 1

ul\_data\_aggregation\_max\_size 4096

qmi\_proxy\_init finished, rx\_urb\_size is 16384

local server: fibo\_qmimsg\_server sockfd = 4

qmi\_start\_server: qmi\_proxy\_server\_fd = 4

1. 执行./multi-pdn-manager进行数据拨号，参考《Fibocom QMI拨号工具使用指南\_Linux》2.4章节，首先用第1路虚拟网卡wwan0.1和第1路profile1进行拨号，如下所示：

root@ght-Lenovo-V130-14IKB:~/lcc/tool/Fibocom-dial\_Linux\_Tool\_V2.0.8/src# **./multi-pdn-manager**

[07-19\_11:29:01:466] Start Fibocom multi-pdn-manager!

dev: /dev/ttyUSB1

rate:115200

sendbuffer:at+gtpcie=3

at+gtpcie=3

+GTPCIE: RC

OK

[07-19\_11:29:01:509] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:29:01:509] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:29:01:511] Fibocom multi-pdn-manager 1.0.4

[07-19\_11:29:01:511] Fibocom manager current qmap\_num is 4

[07-19\_11:29:01:511] Fibocom manager sub-instance1 Disconnected

[07-19\_11:29:01:511] Fibocom manager sub-instance2 Disconnected

[07-19\_11:29:01:511] Fibocom manager sub-instance3 Disconnected

[07-19\_11:29:01:511] Fibocom manager sub-instance4 Disconnected

Please select an action[0-9]

1.show profile setting

2.set profile setting

3.connect to network

4.disconnect to network

5.show connect log

6.reload profile setting

7.save profile setting

8.show max instance number

9.set max instance number

0.exit

**input [0-9]: 3**

**connect visual net interface use:1**

**connect profile use:1**

[07-19\_11:29:04:555] Start connect network use instance 1 profile 1

fibocom-dial -N 4 -n 1 -m 1 -s 1234567890 qwertyuio asdfghjkl 2 -4 -f instance1.txt

Press any key to continus[07-19\_11:29:04:561] exec pid 11717

[07-19\_11:29:04:562] Fibocom-dial\_Linux\_Tool\_V2.0.8

[07-19\_11:29:04:562] fibocom-dial profile[1] = 1234567890/qwertyuio/asdfghjkl/2, pincode = (null)

[07-19\_11:29:04:562] socket[5] successfuly!

[07-19\_11:29:04:562] Waiting client to connect.....

[07-19\_11:29:04:563] Find /sys/bus/usb/devices/1-1 idVendor=2cb7 idProduct=0104

[07-19\_11:29:04:563] Find /sys/bus/usb/devices/1-1:1.4/net/wwan0

[07-19\_11:29:04:563] Find usbnet\_adapter = wwan0

[07-19\_11:29:04:563] Find /sys/bus/usb/devices/1-1:1.4/usbmisc/cdc-wdm0

[07-19\_11:29:04:563] Find qmichannel = /dev/cdc-wdm0

[07-19\_11:29:04:563] qmichannel(/dev/cdc-wdm0) usbnet\_adapter(wwan0)

[07-19\_11:29:04:563] pcie mode

[07-19\_11:29:04:563] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_11:29:04:563] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:29:04:563] qmap\_mode = 4, muxid = 0x81, qmap\_netcard = **wwan0.1**

[07-19\_11:29:04:563] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_11:29:04:563] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:29:04:563] qmap\_mode = 4, muxid = 0x81, qmap\_netcard = wwan0.1

[07-19\_11:29:04:563] connect to fibo\_qmimsg\_server sockfd = 11

[07-19\_11:29:04:563] cdc\_wdm\_fd = 11

[07-19\_11:29:04:563] write triger\_event: 4098 to qmidevice\_control\_fd

[07-19\_11:29:04:621] QmiWwanGetClientID: QMIType = 1 clientid 14

[07-19\_11:29:04:621] Get clientWDS = 14

[07-19\_11:29:04:652] QmiWwanGetClientID: QMIType = 2 clientid 1

[07-19\_11:29:04:652] Get clientDMS = 1

[07-19\_11:29:04:684] QmiWwanGetClientID: QMIType = 3 clientid 2

[07-19\_11:29:04:684] Get clientNAS = 2

[07-19\_11:29:04:716] QmiWwanGetClientID: QMIType = 11 clientid 2

[07-19\_11:29:04:716] Get clientUIM = 2

[07-19\_11:29:04:749] requestBaseBandVersion 89602.1000.00.04.08.21

[07-19\_11:29:04:812] sim\_select = 0

[07-19\_11:29:04:844] curr\_ints\_sim1

[07-19\_11:29:04:844] curr\_ints->CardState is 1

[07-19\_11:29:04:845] curr\_ints->NumApp is 1

[07-19\_11:29:04:845] AppType = 2

[07-19\_11:29:04:845] requestGetSIMStatus SIMStatus: SIM\_READY

[07-19\_11:29:04:875] requestGetICCID DeviceICCID: 898600F0261831632928

[07-19\_11:29:04:908] requestGetIMSI DeviceIMSI: 460026092319783

[07-19\_11:29:04:908] requestSetProfile[1] 1234567890/qwertyuio/asdfghjkl/2

[07-19\_11:29:04:972] requestGetProfile[1] 1234567890/qwertyuio/asdfghjkl/2

[07-19\_11:29:05:004] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE

[07-19\_11:29:05:004] write signo: 12 to signal\_control\_fd

[07-19\_11:29:05:004] epoll fd = 8, events = 0x0001

[07-19\_11:29:05:004] get signo: 12

[07-19\_11:29:05:036] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED

[07-19\_11:29:05:036] usbnet\_link\_change :link:0

[07-19\_11:29:05:036] enter udhcpc\_stop

[07-19\_11:29:05:036] enter fibo\_set\_driver\_link\_state

[07-19\_11:29:05:037] if\_link\_down wwan0.1

[07-19\_11:29:05:037] write signo: 10 to signal\_control\_fd

[07-19\_11:29:05:037] epoll fd = 9, events = 0x0000

[07-19\_11:29:05:037] epoll fd = 8, events = 0x0001

[07-19\_11:29:05:037] get signo: 10

[07-19\_11:29:05:037] usbnet\_link\_change :link:0

[07-19\_11:29:05:069] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE

[07-19\_11:29:05:100] requestSetupDataCall WdsConnectionIPv4Handle: 0x360397e0

[07-19\_11:29:05:132] write triger\_event: 4101 to qmidevice\_control\_fd

[07-19\_11:29:05:164] epoll fd = 9, events = 0x0000

[07-19\_11:29:05:165] epoll fd = 8, events = 0x0000

[07-19\_11:29:05:165] epoll fd = 9, events = 0x0001

[07-19\_11:29:05:165] write signo: 12 to signal\_control\_fd

[07-19\_11:29:05:165] epoll fd = 8, events = 0x0001

[07-19\_11:29:05:165] get signo: 12

[07-19\_11:29:05:229] usbnet\_link\_change :link:1

[07-19\_11:29:05:261] enter fibo\_set\_driver\_link\_state

[07-19\_11:29:05:261] ifconfig wwan0 up

[07-19\_11:29:05:269] if\_link\_up wwan0.1

**[07-19\_11:29:05:270] IPv4 MTU: 1500**

**[07-19\_11:29:05:270] IPv4 Address: 10.84.69.148**

**[07-19\_11:29:05:270] IPv4 Netmask: 29**

**[07-19\_11:29:05:270] IPv4 Gateway: 10.84.69.149**

**[07-19\_11:29:05:271] IPv4 DNS1: 211.137.130.2**

**[07-19\_11:29:05:271] IPv4 DNS2: 211.137.130.4**

**[07-19\_11:29:05:271] if\_link\_up wwan0.1**

1. 从上述拨号log来看，使用第1路虚拟网卡wwan0.1拨号成功，代码段加粗字体可看到成功获取IP，我们再使用第2路虚拟网卡wwan0.2进行拨号，此处采用第2路profile2，拨号过程如下所示：

**input [0-9]: 3**

**connect visual net interface use:2**

**connect profile use:2**

[07-19\_11:49:09:141] Start connect network use instance 2 profile 2

fibocom-dial -N 4 -n 2 -m 2 -s ctnet -4 -f instance2.txt

Press any key to continus[07-19\_11:49:09:147] exec pid 12115

[07-19\_11:49:09:149] Fibocom-dial\_Linux\_Tool\_V2.0.8

[07-19\_11:49:09:149] fibocom-dial profile[2] = ctnet///0, pincode = (null)

[07-19\_11:49:09:150] socket[5] successfuly!

[07-19\_11:49:09:150] Waiting client to connect.....

[07-19\_11:49:09:152] Find /sys/bus/usb/devices/1-1 idVendor=2cb7 idProduct=0104

[07-19\_11:49:09:152] Find /sys/bus/usb/devices/1-1:1.4/net/wwan0

[07-19\_11:49:09:152] Find usbnet\_adapter = wwan0

[07-19\_11:49:09:152] Find /sys/bus/usb/devices/1-1:1.4/usbmisc/cdc-wdm0

[07-19\_11:49:09:152] Find qmichannel = /dev/cdc-wdm0

[07-19\_11:49:09:152] qmichannel(/dev/cdc-wdm0) usbnet\_adapter(wwan0)

[07-19\_11:49:09:152] pcie mode

[07-19\_11:49:09:153] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_11:49:09:153] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:49:09:153] qmap\_mode = 4, muxid = 0x82, qmap\_netcard = **wwan0.2**

[07-19\_11:49:09:153] ioctl(0x89f3, qmap\_settings) failed: Operation not supported, rc=-1

[07-19\_11:49:09:153] access /sys/class/net/wwan0/qmap\_mode

[07-19\_11:49:09:153] qmap\_mode = 4, muxid = 0x82, qmap\_netcard = wwan0.2

[07-19\_11:49:09:154] connect to fibo\_qmimsg\_server sockfd = 11

[07-19\_11:49:09:154] cdc\_wdm\_fd = 11

[07-19\_11:49:09:154] write triger\_event: 4098 to qmidevice\_control\_fd

[07-19\_11:49:09:205] QmiWwanGetClientID: QMIType = 1 clientid 15

[07-19\_11:49:09:205] Get clientWDS = 15

[07-19\_11:49:09:237] QmiWwanGetClientID: QMIType = 2 clientid 2

[07-19\_11:49:09:237] Get clientDMS = 2

[07-19\_11:49:09:268] QmiWwanGetClientID: QMIType = 3 clientid 3

[07-19\_11:49:09:268] Get clientNAS = 3

[07-19\_11:49:09:301] QmiWwanGetClientID: QMIType = 11 clientid 3

[07-19\_11:49:09:301] Get clientUIM = 3

[07-19\_11:49:09:332] requestBaseBandVersion 89602.1000.00.04.08.21

[07-19\_11:49:09:397] sim\_select = 0

[07-19\_11:49:09:429] curr\_ints\_sim1

[07-19\_11:49:09:429] curr\_ints->CardState is 1

[07-19\_11:49:09:429] curr\_ints->NumApp is 1

[07-19\_11:49:09:429] AppType = 2

[07-19\_11:49:09:429] requestGetSIMStatus SIMStatus: SIM\_READY

[07-19\_11:49:09:461] requestGetICCID DeviceICCID: 898600F0261831632928

[07-19\_11:49:09:493] requestGetIMSI DeviceIMSI: 460026092319783

[07-19\_11:49:09:493] requestSetProfile[2] ctnet///0

[07-19\_11:49:09:557] requestGetProfile[2] ctnet///0

[07-19\_11:49:09:589] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE

[07-19\_11:49:09:589] write signo: 12 to signal\_control\_fd

[07-19\_11:49:09:589] epoll fd = 8, events = 0x0001

[07-19\_11:49:09:589] get signo: 12

[07-19\_11:49:09:621] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED

[07-19\_11:49:09:621] usbnet\_link\_change :link:0

[07-19\_11:49:09:621] enter udhcpc\_stop

[07-19\_11:49:09:621] enter fibo\_set\_driver\_link\_state

[07-19\_11:49:09:622] if\_link\_down wwan0.2

[07-19\_11:49:09:622] write signo: 10 to signal\_control\_fd

[07-19\_11:49:09:622] epoll fd = 9, events = 0x0000

[07-19\_11:49:09:622] epoll fd = 8, events = 0x0001

[07-19\_11:49:09:622] get signo: 10

[07-19\_11:49:09:622] usbnet\_link\_change :link:0

[07-19\_11:49:09:653] requestRegistrationState2 MCC: 460, MNC: 0, PS: Attached, DataCap: LTE

[07-19\_11:49:18:839] write signo: 12 to signal\_control\_fd

[07-19\_11:49:18:840] epoll fd = 8, events = 0x0001

[07-19\_11:49:18:840] get signo: 12

[07-19\_11:49:18:874] usbnet\_link\_change :link:1

[07-19\_11:49:18:874] epoll fd = 9, events = 0x0000

[07-19\_11:49:19:131] requestSetupDataCall WdsConnectionIPv4Handle: 0x361b1650

[07-19\_11:49:19:163] write triger\_event: 4101 to qmidevice\_control\_fd

[07-19\_11:49:19:195] epoll fd = 9, events = 0x0000

[07-19\_11:49:19:195] epoll fd = 8, events = 0x0000

[07-19\_11:49:19:195] epoll fd = 9, events = 0x0001

[07-19\_11:49:19:195] write signo: 12 to signal\_control\_fd

[07-19\_11:49:19:195] epoll fd = 8, events = 0x0001

[07-19\_11:49:19:195] get signo: 12

[07-19\_11:49:19:259] usbnet\_link\_change :link:1

[07-19\_11:49:19:291] enter fibo\_set\_driver\_link\_state

[07-19\_11:49:19:291] ifconfig wwan0 up

[07-19\_11:49:19:298] if\_link\_up wwan0.2

**[07-19\_11:49:19:298] IPv4 MTU: 1500**

**[07-19\_11:49:19:299] IPv4 Address: 10.82.57.76**

**[07-19\_11:49:19:299] IPv4 Netmask: 29**

**[07-19\_11:49:19:299] IPv4 Gateway: 10.82.57.77**

**[07-19\_11:49:19:299] IPv4 DNS1: 211.137.130.2**

**[07-19\_11:49:19:299] IPv4 DNS2: 211.137.130.4**

**[07-19\_11:49:19:299] if\_link\_up wwan0.2**

**[07-19\_11:49:21:301] epoll fd = 9, events = 0x0000**

1. 从上述拨号log来看，使用第二路虚拟网卡wwan0.2拨号成功，代码段加粗字体可看到成功获取IP地址。拨号成功后，另开一个窗口，通过“ifconfig –a”查看当前网卡状态，可以看到wwan0.1和wwan0.2均获取到对应的IP地址，如下所示：

wwan0.1: flags=193<UP,RUNNING,NOARP> mtu 1500

**inet 10.84.69.148** netmask 255.255.255.248

ether 0a:44:72:90:32:b4 txqueuelen 1000 (以太网)

RX packets 7 bytes 917 (917.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 7 bytes 488 (488.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wwan0.2: flags=193<UP,RUNNING,NOARP> mtu 1500

**inet 10.82.57.76** netmask 255.255.255.248

ether 0a:44:72:90:32:b4 txqueuelen 1000 (以太网)

RX packets 18 bytes 2474 (2.4 KB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 18 bytes 1272 (1.2 KB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

1. 网卡获取到IP地址后，进行ping测试，如下所示，均可ping通：

root@ght-Lenovo-V130-14IKB:~/lcc/qmi\_wwan\_f/Fibocom\_QMI\_WWAN\_Driver\_V1.0.3# **ping 8.8.8.8 -I wwan0.1**

PING 8.8.8.8 (8.8.8.8) from 10.84.69.148 wwan0.1: 56(84) bytes of data.

64 bytes from 8.8.8.8: icmp\_seq=1 ttl=111 time=140 ms

64 bytes from 8.8.8.8: icmp\_seq=2 ttl=111 time=78.5 ms

64 bytes from 8.8.8.8: icmp\_seq=3 ttl=111 time=86.1 ms

64 bytes from 8.8.8.8: icmp\_seq=4 ttl=111 time=93.4 ms

root@ght-Lenovo-V130-14IKB:~/lcc/qmi\_wwan\_f/Fibocom\_QMI\_WWAN\_Driver\_V1.0.3# **ping 8.8.8.8 -I wwan0.2**

PING 8.8.8.8 (8.8.8.8) from 10.82.57.76 wwan0.2: 56(84) bytes of data.

64 bytes from 8.8.8.8: icmp\_seq=2 ttl=52 time=232 ms

64 bytes from 8.8.8.8: icmp\_seq=3 ttl=52 time=85.5 ms

64 bytes from 8.8.8.8: icmp\_seq=4 ttl=52 time=106 ms