



SIMCOM_SIM8200_Linux_USB_User_Guide _V1.00.00

LTE Module

Shanghai SIMCom Wireless Solutions Ltd.
Building A, SIM Technology Building, No.633, Jinzhong Road
Changning District 200335
Tel:86-21-31575100/31575200
support@simcom.com
www.simcom.com

Document Title:	SIM8200_Linux_USB_User_Guide
Version:	1.00.00
Date:	2020-05-09
Status:	Release
Document ID:	SIMCOM_SIM8200_Linux_USB_User_Guide _V1.00.00

General Notes

SIMCom offers this information as a service to its customers, to support application and engineering efforts that use the products designed by SIMCom. The information provided is based upon requirements specifically provided to SIMCom by the customers. SIMCom has not undertaken any independent search for additional relevant information, including any information that may be in the customer's possession. Furthermore, system validation of this product designed by SIMCom within a larger electronic system remains the responsibility of the customer or the customer's system integrator. All specifications supplied herein are subject to change.

Copyright

This document contains proprietary technical information which is the property of SIMCom Limited., copying of this document and giving it to others and the using or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights reserved in the event of grant of a patent or the registration of a utility model or design. All specification supplied herein are subject to change without notice at any time.

Copyright © Shanghai SIMCom Wireless Solutions Ltd. 2018

Version History

Version	Date	Chapter	What is new
V1.00.00	2020-05-06		New version

SIMCom Confidential

Contents

Version History.....	2
Contents.....	3
1 Introduction.....	4
1.1. Scope.....	4
1.2. Related Documents	4
2 Device Driver Installation.....	5
2.1. USB Serial Driver	5
2.1.1. Add VID and PID.....	5
2.1.2. Building a Linux Driver Module.....	7
2.2. QMI WWAN Drivers.....	8
2.2.1. Source Code File Modification	8
2.2.2. Building a Linux Driver Module.....	8
2.3. Kernel Compilation Configuration.....	9
2.3.1. Compilation Configuration for USB Serial Driver	9
2.3.2. Compilation Configuration for QMI WWAN Driver	9
2.3.3. Compilation Configuration for RNDIS Driver.....	9
2.3.4. Compilation Configuration for PPP Driver	10
3 Modem Usage	12
3.1. Test AT Commands.....	12
3.2. Use PPP Data connection	12
3.2.1. How Does a PPP Dial-Up Connection Work?.....	12
3.3. Use NDIS Data connection	16
3.4. Use RNDIS Data connection	18
4 Troubleshooting	20
4.1. How to check whether the correct USB serial driver exists in the kernel?.....	20
4.2. What can I do if the port number does not start from ttyUSBO?.....	20
4.3. How to check whether the correct QMI WWAN driver integration in the kernel?.....	20
5 Appendix A Abbreviations	22
Table 1: Terms and Abbreviations	22

1 Introduction

1.1. Scope

This user guide serves the following purpose:

- Short introductions how to customize the USB driver for Simcom SIM8200 module in Linux OS
- Describes how software developers can use Linux devices for typical use cases.

1.2. Related Documents

1: SIM8200 ATC.

2 Device Driver Installation

In order to recognize the modem, you must add VID and PID in Linux driver option, after the operating system recognizes the modem, devices named /dev/ttyUSBx are created, for example:

- dev/ttyUSB0 diag port for output developing messages
- dev/ttyUSB1 NMEA port for GPS NMEA data output
- dev/ttyUSB2 AT port for AT commands
- dev/ttyUSB3 Modem port for ppp-dial
- dev/ttyUSB4 audio port

2.1. USB Serial Driver

2.1.1. Add VID and PID

In order to recognize the module, customers should add module VID and PID information as below:

1. If the Linux kernel version higher than 3.2 and lower than 4.4.131:

If the following macro definitions are not contained in the usb.h file under linux-x.x.x/include/linux, add them

```
#define USB_DEVICE_INTERFACE_CLASS(vend, prod, cl) \
    .match_flags = USB_DEVICE_ID_MATCH_DEVICE | \
        USB_DEVICE_ID_MATCH_INT_CLASS, \
    .idVendor = (vend), \
    .idProduct = (prod), \
    .bInterfaceClass = (cl)
```

Add the following statements:

File: [linux-x.x.x]/drivers/usb/serial/option.c

```
#define SIMCOM_VENDOR_ID          0x1e0e
static const struct option_blacklist_info simcom_pid9001_blacklist = {
    .reserved = BIT(5) | BIT(6),
};

static const struct option_blacklist_info simcom_pid9011_blacklist = {
```

```

    .reserved = BIT(7),
};

static const struct usb_device_id option_ids[] = {
    .....
    { USB_DEVICE(SIMCOM_VENDOR_ID, 0x9001),
        .driver_info = (kernel_ulong_t)&simcom_pid9001_blacklist },
    { USB_DEVICE_INTERFACE_CLASS(SIMCOM_VENDOR_ID, 0x9011, 0xff), /* Simcom
SIM8200 RNDIS mode */
        .driver_info = (kernel_ulong_t)&simcom_pid9011_blacklist },

```

NOTE

If the following statements are contained in the option.c file under linux-x.x.x/drivers/usb/serial/, remove them,as they will conflict with SIM8200's USB driver.

```

    { USB_DEVICE(ALINK_VENDOR_ID, SIMCOM_PRODUCT_SIM7100E),
        .driver_info = (kernel_ulong_t)&simcom_sim7100e_blacklist },
    { USB_DEVICE_INTERFACE_CLASS(0x1e0e, 0x9011, 0xff),

```

2. If the Linux kernel version higher than 4.4.132:

Add the following statements:

File: [linux-x.x.x]/drivers/usb/serial/option.c

```

#define SIMCOM_VENDOR_ID          0x1e0e

static const struct usb_device_id option_ids[] = {
    .....
    { USB_DEVICE(SIMCOM_VENDOR_ID, 0x9001),
        .driver_info = RSVD(5) | RSVD(6) },
    { USB_DEVICE_INTERFACE_CLASS(SIMCOM_VENDOR_ID, 0x9011, 0xff), /* Simcom
SIM8200 RNDIS mode ,Reserved the interface for ADB */
        .driver_info = RSVD(7) },

```

NOTE

If the following statements are contained in the option.c file under linux-x.x.x/drivers/usb/serial/, remove them, as they will conflict with SIM8200's USB driver.

```
{ USB_DEVICE(ALINK_VENDOR_ID, SIMCOM_PRODUCT_SIM7100E),  
    .driver_info = RSVD(5) | RSVD(6) },  
{ USB_DEVICE_INTERFACE_CLASS(0x1e0e, 0x9011, 0xff),  
    .driver_info = RSVD(7) },
```

2.1.2. Building a Linux Driver Module

How to compile and install a kernel module in Linux. follow the steps below will guide you along in compiling and install your option driver On Ubuntu operating system.

Step 1: Enter to kernel directory.

```
cd <your kernel directory>
```

Step 2: Build the driver.

```
sudo make -C /lib/modules/`uname -r`/build M=`pwd`/drivers/usb/serial obj-  
m=option.o modules
```

Step 3: Load the driver and reboot.

```
sudo cp drivers/usb/serial/option.ko /lib/modules/`uname -  
r`/kernel/drivers/usb/serial  
sudo depmod  
sudo reboot
```

2.2. QMI WWAN Drivers

2.2.1. Source Code File Modification

NOTE

If the following statements are contained in the qmi_wwan.c file under linux-x.x.x/drivers/net/usb/, remove them, as they will conflict with SIM8200's QMI WWAN driver.

```
{QMI_QUIRK_SET_DTR(0x1e0e, 0x9001, 5)}  
{QMI_FIXED_INTF(0x1e0e, 0x9001, 5)},
```

Simcom provides the source file qmi_wwan_simcom.c, which only be used for SIM8200. Contact customer engineer of simcom to get the file.

Please add the following statements to build qmi_wwan_simcom.c and keep the following order.

File: linux-x.x.x/drivers/net/usb/Makefile.

```
obj-$(CONFIG_USB_NET_QMI_WWAN) += qmi_wwan_simcom.o  
obj-$(CONFIG_USB_NET_QMI_WWAN) += qmi_wwan.o
```

2.2.2. Building a Linux Driver Module

How to compile and install a kernel module in Linux, follow the steps below will guide you along in compiling and install your option driver On Ubuntu operating system.

Step 1: Enter to kernel directory.

```
cd <your kernel directory>
```

Step 2: Build the driver.

```
sudo make -C /lib/modules/`uname -r`/build M=`pwd`/drivers/net/usb obj-  
m=qmi_wwan.o modules  
sudo make -C /lib/modules/`uname -r`/build M=`pwd`/drivers/net/usb obj-  
m=qmi_wwan.o modules
```

Step 3: Load the driver and reboot.

```
sudo cp drivers/net/usb/qmi_wwan_simcom.ko /lib/modules/`uname -r`/kernel/drivers/net/usb
sudo cp drivers/net/usb/qmi_wwan.ko /lib/modules/`uname -r`/kernel/drivers/net/usb
sudo depmod
sudo reboot
```

2.3. Kernel Compilation Configuration

2.3.1. Compilation Configuration for USB Serial Driver

Configuration	Configuration (N/Y)
CONFIG_USB_SERIAL	Y
CONFIG_USB_SERIAL_WWAN	Y
CONFIG_USB_SERIAL_OPTION	Y

2.3.2. Compilation Configuration for QMI WWAN Driver

Configuration	Configuration (N/Y)
CONFIG_USB_SERIAL	Y
CONFIG_USB_SERIAL_WWAN	Y
CONFIG_USB_SERIAL_OPTION	Y
CONFIG_USB_USBNET	Y
CONFIG_USB_WDM	Y

2.3.3. Compilation Configuration for RNDIS Driver

Configuration	Configuration (N/Y)
CONFIG_USB_SERIAL	Y
CONFIG_USB_SERIAL_WWAN	Y
CONFIG_USB_SERIAL_OPTION	Y

CONFIG_USB_UBNET	Y		
CONFIG_USB_NET_CDCETHER	Y		

2.3.4. Compilation Configuration for PPP Driver

Configuration	Configuration (N/Y)		
CONFIG_USB_SERIAL	Y		
CONFIG_USB_SERIAL_WWAN	Y		
CONFIG_USB_SERIAL_OPTION	Y		
CONFIG_PPP	Y		
CONFIG_PPP_FILTER	Y		
CONFIG_PPP_MULTILINK	Y		
CONFIG_PPP_BSDCOMP	Y		
CONFIG_PPP_ASYNC	Y		
CONFIG_PPP_SYNC_TTY	Y		
CONFIG_PPP_DEFLATE	Y		

SIMCom Confidential

3 Modem Usage

This chapter mainly introduces several commonly used HSUSB tethering methods and their general processes.

3.1. Test AT Commands

```
#cat /dev/ttyUSB2 &
#echo -e "at\r\nn">/dev/ttyUSB2
#
OK
```

NOTE

Check the driver to make sure that Simcom VID and PID information modification has been added.

3.2. Use PPP Data connection

3.2.1. How Does a PPP Dial-Up Connection Work?

You will need the right software and a couple of pieces of information before you start. First, check the pppd. If the programs do not exist, you can download the source code from <https://ppp.samba.org/download.html> and port them to your embedded development environment. Next you must write configuration file for pppd.

3.2.1.1. Chat Scription

```
#named simcom-connect-chat and place in /etc/ppp/peers
ABORT "BUSY"
ABORT "NO CARRIER"
ABORT "NO DIALTONE"
```

```

ABORT "ERROR"
ABORT "NO ANSWER"
TIMEOUT 30
"" AT
OK ATE0
OK ATI;+CSUB;+CSQ;+CPIN?;+COPS?;+CGREG?;&D2
# Insert the APN provided by your network operator, default apn is 3gnet
OK AT+CGDCONT=1,"IP","3gnet",,0,0
OK ATD*99#
CONNECT

```

```

#named simcom-disconnect-chat and place in /etc/ppp/peers
ABORT "ERROR"
ABORT "NO DIALTONE"
SAY "\nSending break to the modem\n"
"" ++
"" ++
"" ++
SAY "\nGoodbay\n"

```

3.2.1.2. Configure dialing and AT port

```

# named simcom-pppd and place in /etc/ppp/peers
/dev/ttyUSB3 115200
#Insert the username and password for authentication, default user and password
are test
user "test" password "test"
# The chat script, customize your APN in this file
connect 'chat -s -v -f /etc/ppp/peers/simcom-connect-chat'
# The close script
disconnect 'chat -s -v -f /etc/ppp/peers/simcom-disconnect-chat'
# Hide password in debug messages
hide-password

```

```
# The phone is not required to authenticate
noauth

# Debug info from pppd
debug

# If you want to use the HSDPA link as your gateway
defaultroute

# pppd must not propose any IP address to the peer
noipdefault

# No ppp compression
novj

novjccomp

noccp

ipcp-accept-local

ipcp-accept-remote

local

# For sanity, keep a lock on the serial line
lock

modem

dump

nodetach

# Hardware flow control
nocrtscts

remotename 3gppp

ipparam 3gppp

ipcp-max-failure 30

# Ask the peer for up to 2 DNS server addresses
usepeerdns
```

3.2.1.3. Dial-Up Connection

```
# pppd call simcom-pppd &
```

When you see the output below,it shows that dial-up succeeded.

```
Connect: ppp0 <--> /dev/ttyUSB3
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x5107d141> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x9a5c1936>
<pcomp> <accomp>]
sent [LCP ConfAck id=0x0 <asyncmap 0x0> <auth chap MD5> <magic 0x9a5c1936>
<pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x5107d141> <pcomp> <accomp>]
sent [LCP EchoReq id=0x0 magic=0x5107d141]
rcvd [LCP DiscReq id=0x1 magic=0x9a5c1936]
rcvd [CHAP Challenge id=0x1 <dd93b9f04d75e2bbba3786f6d24df3d7>, name =
"UMTS_CHAP_SRVR"]
sent [CHAP Response id=0x1 <498d4d7cf3b59dacfc07a45ce6eb7e26>, name = "test"]
rcvd [LCP EchoRep id=0x0 magic=0x9a5c1936 51 07 d1 41]
rcvd [CHAP Success id=0x1 ""]
CHAP authentication succeeded
CHAP authentication succeeded
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP ConfReq id=0x0]
sent [IPCP ConfNak id=0x0 <addr 0.0.0.0>]
rcvd [IPCP ConfNak id=0x1 <addr 10.51.68.23> <ms-dns1 222.66.251.8> <ms-dns2
116.236.159.8>]
sent [IPCP ConfReq id=0x2 <addr 10.51.68.23> <ms-dns1 222.66.251.8> <ms-dns2
116.236.159.8>]
rcvd [IPCP ConfReq id=0x1]
sent [IPCP ConfAck id=0x1]
rcvd [IPCP ConfAck id=0x2 <addr 10.51.68.23> <ms-dns1 222.66.251.8> <ms-dns2
116.236.159.8>]
Could not determine remote IP address: defaulting to 10.64.64.64
local IP address 10.51.68.23
remote IP address 10.64.64.64
primary DNS address 222.66.251.8
secondary DNS address 116.236.159.8
Script /etc/ppp/ip-up started (pid 6616)
Script /etc/ppp/ip-up finished (pid 6616), status = 0x0
```

Now PPP call is set up successfully. Please use following commands to check IP/DNS/Route.

```
# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.216.159.39  P-t-P:10.64.64.64  Mask:255.255.255.255
                  UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
                  RX packets:9 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:3
                  RX bytes:362 (362.0 B)  TX bytes:316 (316.0 B)

# cat /etc/resolv.conf
nameserver 221.180.132.108

# route -n
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref  Use Iface
0.0.0.0         0.0.0.0        0.0.0.0        U      0      0      0  ppp0
10.64.64.64    0.0.0.0        255.255.255.255 UH    0      0      0  ppp0

# ping baidu.com
PING baidu.com (220.181.57.216) 56(84) bytes of data.
64 bytes from 220.181.57.216: icmp_seq=1 ttl=50 time=84.0 ms
64 bytes from 220.181.57.216: icmp_seq=2 ttl=50 time=34.2 ms
```

Following commands can be used to terminate PPPD process to disconnect a PPP call:

```
# killall pppd
```

3.3. Use NDIS Data connection

Make sure the <PID> has been switched to 9001.

Please use the source code in the simcom-cm.rar package provided by our company. Adjust the Makefile file on the target host, and then compile them into the executable file or the library file if needed.

1) The Makefile in the source code applies to the Linux OS environment. After decompressing it in one Linux directory, enter the simcom-cm folder, and then execute the 'make' command to compile. If the

source code is compiled successfully, the simcom-cm execution program will be generated. Connect the module to the Linux OS host through a USB cable. Please make sure the driver is installed successfully. Enter the directory which locates the simcom-cm execution program under root authority, and then execute ‘./simcom-cm’.

```
# sudo ./simcom-cm
[05-07_01:27:02:351] Find qmichannel = /dev/cdc-wdm0
[05-07_01:27:02:351] Find usbnet_adapter = wwan0
[05-07_01:27:02:362] cdc_wdm_fd = 7
[05-07_01:27:02:400] Get clientWDS = 15
[05-07_01:27:02:408] Get clientDMS = 1
[05-07_01:27:02:419] Get clientNAS = 2
[05-07_01:27:02:428] Get clientUIM = 1
[05-07_01:27:02:437] Get clientWDA = 1
[05-07_01:27:02:445] requestBaseBandVersion MPSS.HI.1.0.c8-00770-SDX55_ALL_PACK-
1 1 [Apr 08 2020 22:00:00]
[05-07_01:27:02:455] requestSetEthMode QMUXResult = 0x1, QMUXError = 0x11
[05-07_01:27:02:460] requestGetSIMStatus SIMStatus: SIM_READY
[05-07_01:27:02:468] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached,
DataCap: LTE
[05-07_01:27:02:477] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[05-07_01:27:02:496] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached,
DataCap: LTE
[05-07_01:27:02:521] requestSetupDataCall WdsConnectionIPv4Handle: 0x98df87e0
[05-07_01:27:02:533] requestQueryDataCall IPv4ConnectionStatus: CONNECTED
[05-07_01:27:02:540] ifconfig wwan0 up
[05-07_01:27:02:543] Fail to access /usr/share/udhcpc/default.script, errno: 2
(No such file or directory)
[05-07_01:27:02:543] busybox udhcpc -f -n -q -t 5 -i wwan0
[05-07_01:27:02:622] udhcpc (v1.21.1) started
[05-07_01:27:02:672] Sending discover...
[05-07_01:27:02:708] Sending select for 10.147.148.239...
[05-07_01:27:02:744] Lease of 10.147.148.239 obtained, lease time 7200
[05-07_01:27:02:750] /etc/udhcpc/default.script: Resetting default routes
SIOCDELRT: No such process
[05-07_01:27:02:755] /etc/udhcpc/default.script: Adding DNS 219.148.204.66
```

```
[05-07_01:27:02:755] /etc/udhcpc/default.script: Adding DNS 219.149.6.99
```

Please use following commands to check IP/DNS/Route.

```
# ifconfig wwan0
wwan0      Link encap:Ethernet HWaddr 0a:f7:70:0c:c3:3b
           inet addr:10.147.148.239 Bcast:10.147.148.255 Mask:255.255.255.224
             inet6 addr: fe80::8f7:70ff:fe0c:c33b/64 Scope:Link
                   UP BROADCAST RUNNING NOARP MULTICAST MTU:1500 Metric:1
                   RX packets:5 errors:0 dropped:0 overruns:0 frame:0
                   TX packets:45 errors:0 dropped:0 overruns:0 carrier:0
                   collisions:0 txqueuelen:1000
                   RX bytes:1019 (1.0 KB) TX bytes:6659 (6.6 KB)

# ping baidu.com
PING baidu.com (123.125.114.144) 56(84) bytes of data.
64 bytes from 123.125.114.144: icmp_seq=1 ttl=56 time=114 ms
64 bytes from 123.125.114.144: icmp_seq=2 ttl=56 time=58.6 ms
64 bytes from 123.125.114.144: icmp_seq=3 ttl=56 time=45.1 ms
```

3.4. Use RNDIS Data connection

```
# cat /dev/ttyUSB2 &
# echo -e "AT+CUSBCFG=usbid,1e0e,9011\r\n">/dev/ttyUSB2
#
# OK

Wait for the system to reset to rndis mode, make sure the <PID> has been
switched to 9011.

# echo -e "AT+NETACT=1\r\n">/dev/ttyUSB2
#
# OK
```

Please use following commands to check IP/DNS/Route.

```
# ifconfig usb0
usb0      Link encap:Ethernet HWaddr ae:68:46:d6:b2:80
```

```
inet addr:192.168.0.100 Bcast:192.168.0.255 Mask:255.255.255.0
inet6 addr: fe80::ac68:46ff:fed6:b280/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:45 errors:0 dropped:0 overruns:0 frame:0
      TX packets:104 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:4237 (4.2 KB) TX bytes:13148 (13.1 KB)
```

```
# ping baidu.com
PING baidu.com (220.181.38.148) 56(84) bytes of data.
64 bytes from 220.181.38.148: icmp_seq=1 ttl=50 time=94.8 ms
64 bytes from 220.181.38.148: icmp_seq=2 ttl=50 time=135 ms
64 bytes from 220.181.38.148: icmp_seq=3 ttl=50 time=61.9 ms
```

SIMCom Confidential

4 Troubleshooting

4.1. How to check whether the correct USB serial driver exists in the kernel?

If Linux does not create devices, check for the kernel module:

```
# lsmod | grep option
```

If entries aren't found, load the kernel module with root privileges:

```
# modprobe option
```

Check dmesg output to see that the radio was detected:

```
# dmesg | grep option
```

Check dmesg output to see that the radio was detected:

```
# dmesg | grep option
[ 16.672003] usbcore: registered new interface driver option
[ 16.672105] option 2-1.2:1.0: GSM modem (1-port) converter detected
[ 16.672216] option 2-1.2:1.1: GSM modem (1-port) converter detected
[ 16.672292] option 2-1.2:1.2: GSM modem (1-port) converter detected
[ 16.672365] option 2-1.2:1.3: GSM modem (1-port) converter detected
[ 16.672438] option 2-1.2:1.4: GSM modem (1-port) converter detected
```

If this returns an error response, the kernel module is not on your system. You will need to build the driver

4.2. What can I do if the port number does not start from ttyUSB0?

Check the ttyUSB port usage. Check whether the ttyUSB port is released when the module is disconnected.

4.3. How to check whether the correct QMI WWAN driver integration in the kernel?

```
[88843.080420] option 1-5:1.0: GSM modem (1-port) converter detected
[88843.080671] usb 1-5: GSM modem (1-port) converter now attached to ttyUSB0
```

```
[88843.080891] option 1-5:1.1: GSM modem (1-port) converter detected
[88843.081086] usb 1-5: GSM modem (1-port) converter now attached to ttyUSB1
[88843.081332] option 1-5:1.2: GSM modem (1-port) converter detected
[88843.082825] usb 1-5: GSM modem (1-port) converter now attached to ttyUSB2
[88843.083040] option 1-5:1.3: GSM modem (1-port) converter detected
[88843.083213] usb 1-5: GSM modem (1-port) converter now attached to ttyUSB3
[88843.083445] option 1-5:1.4: GSM modem (1-port) converter detected
[88843.083541] usb 1-5: GSM modem (1-port) converter now attached to ttyUSB4
[88843.648962] usbcore: registered new interface driver cdc_wdm
[88843.650716] qmi_wwan_simcom 1-5:1.5: cdc-wdm0: USB WDM device
[88843.650718] qmi_wwan_simcom 1-5:1.5: SIMCom 8200 work on RawIP mode
[88843.651108] qmi_wwan_simcom 1-5:1.5 wwan0: register 'qmi_wwan_simcom' at usb-0000:00:14.0-5, WWAN/QMI device, aa:39:fd:18:de:95
[88843.651173] usbcore: registered new interface driver qmi_wwan_simcom
```

Check interface wwan0

```
wwan0      Link encap:Ethernet  HWaddr aa:39:fd:18:de:95
           inet6 addr: fe80::a839:fdff:fe18:de95/64  Scope:Link
                     UP BROADCAST RUNNING NOARP 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 (0.0 B)  TX bytes:0 (0.0 B)
```

5 Appendix A Abbreviations

Table 1: Terms and Abbreviations

Abbreviation	Description
USB	Universal Serial Bus
VID	Vendor ID
PID	Product ID
PPP	Point-to-Point Protocol
IPCP	IP Control Protocol
IP	Internet Protocol
NMEA	National Marine Electronics Association
DNS	Domain Name Server
NDIS	Network Driver Interface Specification
RNDIS	Remote Network Driver Interface Specification