Fibocom

Perfect Wireless Experience
完美无线体验

Linux System Driver Integration and Application Guidance

Document version: V2.1.0

Update date: 26.03.2016

Applicability Table

No.	Product model	Description

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.

Version Record

Version	Update Date	Remarks
V1.0.0	10-17-2015	Initial version
V2.0.0	11-24-2015	Optimize document name and title; Optimize document structure and content; Optimize document figure;
V2.1.0	2016-03-26	Optimize document for amazon.

Contents

1 Introduction

1.1 Purpose

- This article is the guidance for driver integration development activities for L8 series 4G module devices based on Linux systems. This document is mainly for driver developers for product developers based on the above systems.

1.2 Scope

The document applies to the following:

- Linux2.6.22 and higher version.

2 Instructions for Linux system

2.1 Linux Kernel Device Driver Architecture

Linux kernel will load the USB driver according to the USB device interfaces reported by the 4G module. After the correct driver is loaded, the module can begin to work.

Linux kernel driver architecture of the Linux system is shown in Figure 2-1:

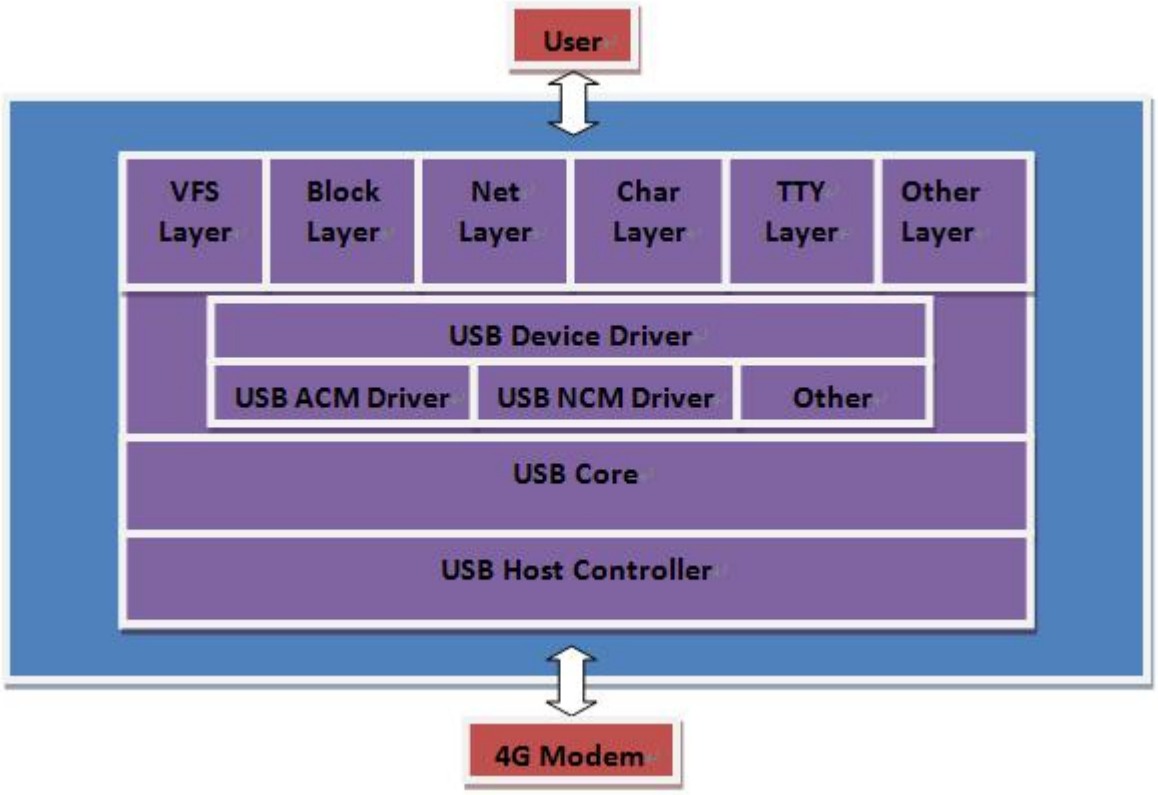


Figure 2-1 Driver Architecture

As is shown in Figure 2-1, driver modules related to 4G devices in the USB driver architecture of Linux system are: USB ACM driver modules.

Attention:

ACM Driver: USB ACM driver supports modem ports, AT ports and so on; the source code (cdc-acm.c) of the ACM driver is built-in in the Linux kernel.

2.2 Linux ACM Driver Integration

2.2.1 ACM Driver Porting

1. Modification for driver code: As shown in Figure 2-2, add the codes in the red box to array "static const struct usb_device_idacm_ids[ ]" in file "drivers/usb/class/cdc-acm.c".

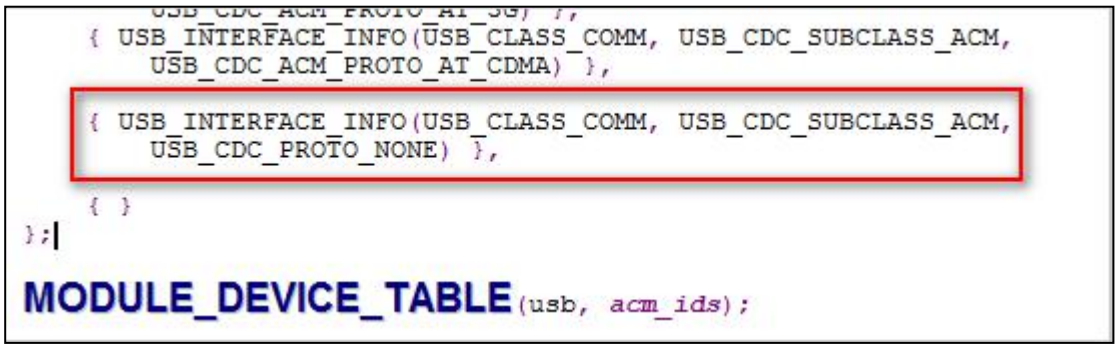


Figure 2-2 Codes of acm_ids

Codes in Figure 2-2 are as follows:

```
{ USB_INTERFACE_INFO(USB_CLASS_COMM, USB_CDC_SUBCLASS_ACM,
    USB_CDC_PROTO_NONE) },
```

2. Modify the compiled configuration of kernel (config files under the kernel directory) and ensure the following configuration items have been selected:

1) Related configuration items of PPP dial-up:

CONFIG_PPP=y

CONFIG_PPP_MULTILINK=y

CONFIG_PPP_FILTER=y

CONFIG_PPP_ASYNC=y

CONFIG_PPP_SYNC_TTY=y

CONFIG_PPP_DEFLATE=y

CONFIG_PPP_BSDCOMP=y

2) Related configuration items of USB ACM:

CONFIG_USB_ANNOUNCE_NEW_DEVICES=y (if such option exists, it's suggested to configure; if not, please ignore)

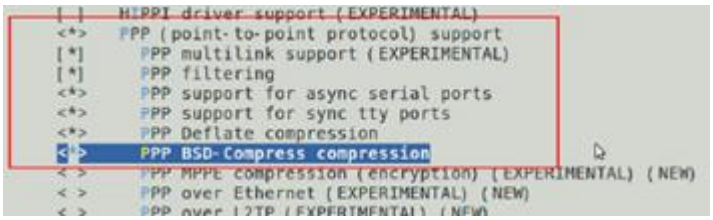
CONFIG_USB_ACM=y

2.2.2 Detailed Configuration Setup

1. Open the Terminal tool, enter the kernel directory (it is assumed to be "/linux-3.0.8/ /home/ght"), and execute the <configuration> make command (it's assumed to use standard menuconfig).

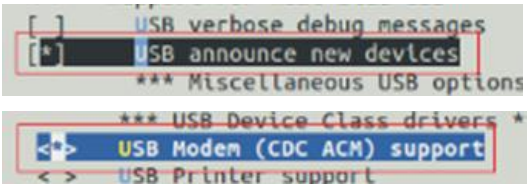
2. Complete configurations of PPP dial-up as the following guidelines:

   Enter **"Device Drivers"→"Network device support"** menu and select all items in the red border



3. Complete configurations of ACM driver as the following guidelines:

   Enter "**Device Drivers"**→"**USB support**" menu and select USB announce new devices and USB Modem (CDC ACM) support items:



4. After the configuration, exit the configuration interface step by step by selecting "<Exit>". And then select "<Yes>" and exit the save interface.

5. After completing configurations, run the make command to edit the modified kernel.

2.3 ACM Driver Configuration Confirmation

When the system starts up, execute the dmesg command and check the kernel messages. The information as shown in the red box in Figure 2-3 indicate that the ACM driver in the system has been successfully configured.

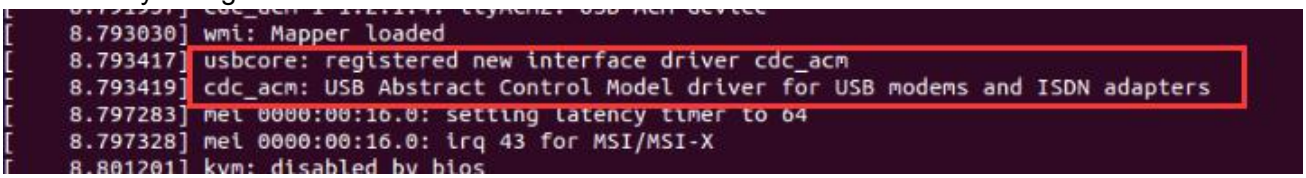


Figure 2-3 ACM Configuration

After the system is started and the 4G module is powered up, execute the dmesg command to check the kernel messages; the information as shown in the red box of in Figure 2-4 indicate that ACM driver has been successfully loaded, Execute ls -al /dev/ttyACM* command to inquire ttyACM0,ttyACM1 and ttyACM2.

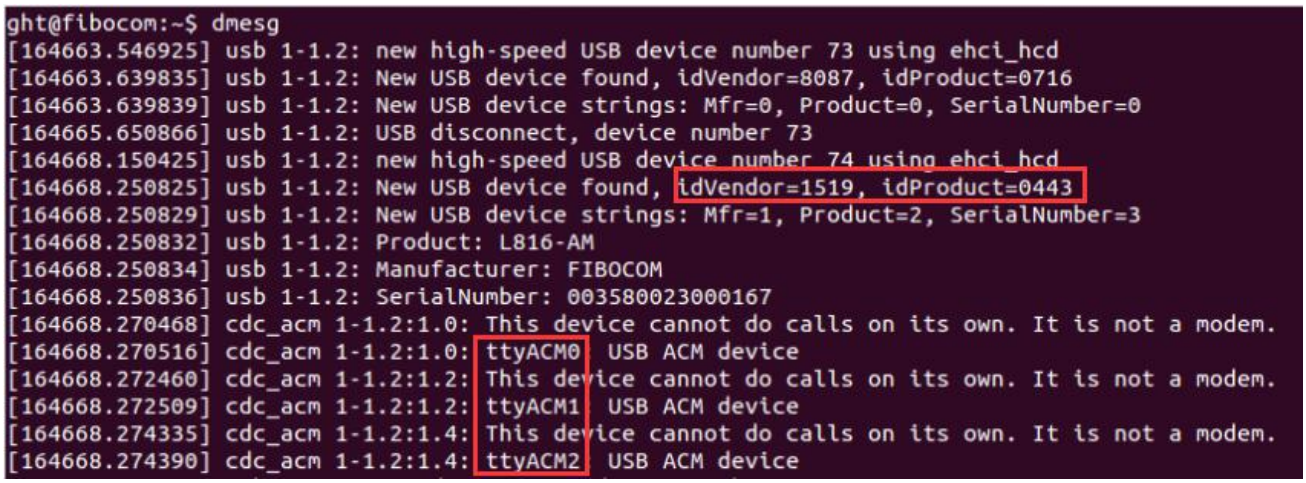
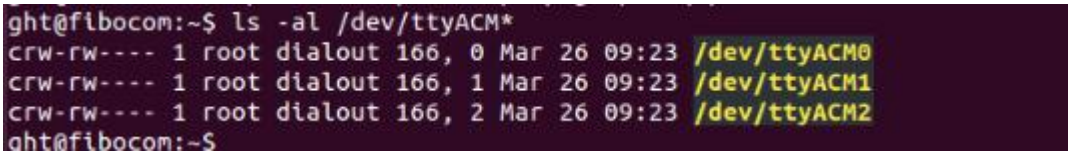


Figure 2-4 Driver Loading

Execute ls –al /dev/ttyACM* command to inquire ttyACM0, ttyACM1 and ttyACM2.



2.4 Port Form Description

No.	Port name	Port form	Remarks
1	ttyACM0	Modem Port	For PPP data traffic, or for sending and receiving AT command under the non-data mode
2	ttyACM1	Trace Port	For capturing module debug information
3	ttyACM2	At Port	For AT communications, namely, sending and receiving AT commands

2.5 Port Testing

2.5.1 Command Line Testing

1. Open the terminal.
2. Execute echo -e "ATE0\r\n"> /dev/ttyACM2 **(**Execute this command before any other command or the cat command will be abnormal.**)**
3. Execute cat /dev/ttyACM2 & to read the result.
4. Execute echo -e "at+cgmr\r\n"> /dev/ttyACM2 to inquire the software version.
5. Executer echo -e "at+cgdcont=1,\"ip\",\"3gnet\"\r\n"> /dev/ttyACM2 to set up APN.

If AT command contains double quotation marks, ESC"\"should be added, please refer to Step 5 for the format.

2.5.2 Program Testing

The C program below can be used to test send and receive of AT commands. The program opens the /dev/ttyACM2 device node, and calls the write and read function to send AT commands and receive the reply.

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <termios.h>
#define ATPORT "/dev/ttyACM2"
#define BUFSIZE 1000
#define BAUDRATE B115200
int open_port(char *port)
{
    struct termios options;
    int fd;
    fd = open(port, O_RDWR | O_NOCTTY | O_NDELAY);
    if (fd == -1) {
        printf("%s: Unable to open the port - \r\n",__func__);
    } else {
        fcntl(fd, F_SETFL, FNDELAY);
        tcgetattr(fd, &options );
        cfsetispeed(&options, BAUDRATE );
        cfsetospeed(&options, BAUDRATE );
        options.c_cflag |= ( CLOCAL | CREAD);
        options.c_cflag&= ~(CSIZE | PARENB | CSTOPB | CSIZE);
```

```
            options.c_cflag |= CS8;
            options.c_cflag&= ~CRTSCTS;
            options.c_lflag&= ~(ICANON | ECHO | ECHOE | ISIG);
            options.c_iflag&= ~(IXON | IXOFF | IXANY | ICRNL | INLCR | IGNCR);
            options.c_oflag&= ~OPOST;
            if ( tcsetattr( fd, TCSANOW, &options ) == -1 ) {
                printf ("Error with tcsetattr = %s\r\n", strerror ( errno ) );
            } else {
                printf ( "Open port succeed\r\n");
            }
        }
        return (fd);
}
int main()
{
        int fd = open_port(ATPORT);
        char at_cmd_ch[50]="AT+CGMR\r\n";
        char buf[BUFSIZE];
        memset(buf,0,BUFSIZE);
        printf("AtSend: %s\r\n", at_cmd_ch);
        write(fd, at_cmd_ch , strlen(at_cmd_ch));
        sleep(1);
        read(fd, buf, BUFSIZE );
        printf("AtRecevie: %s\r\n", buf);
        close(fd);
        return 0;
}
```

Save the above code in the TestPort.c text, execute the o - TestPortTestPort.c GCC command to compile the TestPort program, and then execute the compiled program to see the returned results.

```
ght@fibocom:~$ gcc -o TestPort TestPort.c
ght@fibocom:~$ ./TestPort
Open port succeed
AtSend: AT+CGMR

AtRecevie:
+CGMR: "L810_V5G.0C.01.02_TEST02"

OK
```

Because the 4G module needs time to process after sending out the "AT+CGMR" command, it is necessary to delay at least 500ms before reading. Sleep (1) in the demo code is only for reference.

Attention:

sleep(1) means delaying 1 second.

2.6 Connect Internet via PPP Dial-up

In application scenarios where NCM driver can't be supported, it's necessary to use PPP dial-up.

There are a total of three script files for PPP dial-up: chat-wcdma-connect, chat-wcdma-disconnect and wcdma, and the content for scripts is as shown in 2.10.

1. Put the above three script files in the /etc/ppp/peers/ directory, and use Chmod 777 XXX command to give the file read and execute permissions. Input the following in the command line:

   PPPD call <dial-up script >

   For example, if file name of the dial-up script is "wcdma", the command is as follows:

   pppd call wcdma

2. After successful dial-up, execute the ifconfig command to inquire IP address.

Figure 2-5 shows the query results after executing ifconfig command after successful ppp dial-up.

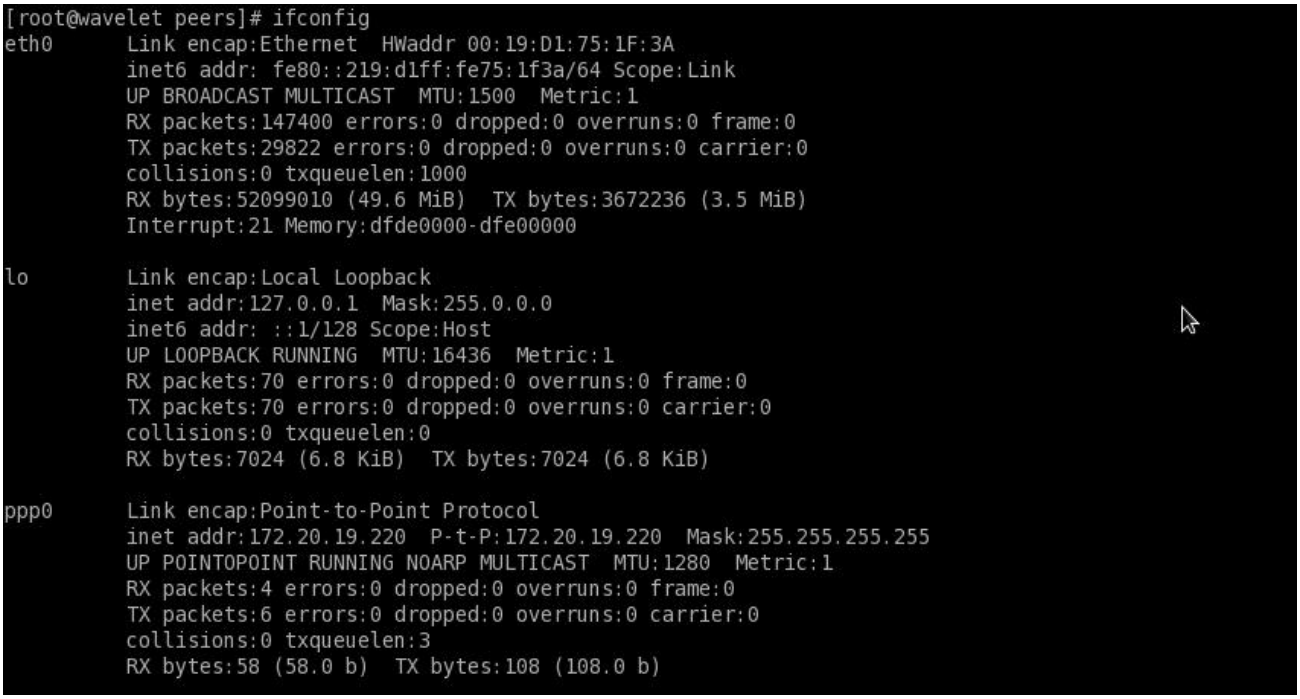


Figure 2-5 ifconfig query

Attention:

Precondition for dial-up:

- A valid SIM card is inserted.
- Module is powered up and running.
- Module can register network.

2.7 PPP Dial-up Script Description

Example of wcdma script:

```
nodetach
lock
/dev/ttyACM0
115200
crtscts
debug
#logfile /data/logfile
modem
hide-password
usepeerdns
noauth
noipdefault
novj
novjccomp
noccp
defaultroute
ipcp-accept-local
ipcp-accept-remote
connect 'chat -s -v -f /etc/ppp/peers/chat-wcdma-connect'
disconnect 'chat -s -v -f /etc/ppp/peers/chat-wcdma-disconnect'
```

Attention:

/dev/ttyACM0 assigns the port for dial-up; if it's necessary to use ACM2 port for dial-up, just modify ttyACM0 into ttyACM2.

Example of chat-wcdma-connect script:

```
" AT
OK "
ABORT 'NO CARRIER'
ABORT 'ERROR'
ABORT 'NO DIALTONE'
ABORT 'BUSY'
ABORT 'NO ANSWER'
" AT
OK ATZ
OK AT+GTRAT?
OK AT+CMEE=2
OK AT+CSQ
OK AT+CPIN?
OK AT+COPS?
OK AT+CGACT=0,1
OK AT+CGDCONT=1,\"IP\",\"cmnet\",,0,0
OK ATDT*99#
CONNECT "
```

Attention:

AT+CGDCONT=1,\"IP\",\"cmnet\",,0,0 (cmnet represents APN for China Mobile and APN for China Unicom is 3gnet)

Example of chat-wcdma-disconnect script:

```
ABORT   OK
ABORT   BUSY
ABORT   DELAYED
ABORT   "NO ANSWER"
ABORT   "NO CARRIER"
ABORT   "NO DIALTONE"
ABORT   VOICE
ABORT   ERROR
ABORT   RINGING
TIMEOUT   12
""   \K
""   \K
""   \K
""   +++ATH
""   +++ATH
""   +++ATH
""   ATZ
SAY "\nGoodbay\n"
```