Linux NDIS 使用手册&接口文档

U8300 模块 USB 相关描述

	VID 1C9E	PID 9B05
0	Diagnosis	Diagnostic Interface
1	Modem	Modem Interface
2	Application	Application Interface
3	Pip	Pip Interface
4	NDIS	NDIS Interface
5	ADB	ADB Interface

对于外围操作系统，USB 上面 Endpoint 需要支持 Bulk 12，Interrupt 4。
而 NDIS Interface 4 对上层的功能接口是 Net，而其它 Interface 对上层的功能接口是 serial，所以在 linux 系统中，要防止 USB Serial 驱动也把 NDIS Interface 进行枚举占用。

对于串口功能

1：USB Serial 驱动支持

在 linux 系统中通常使用 usb 转串口的驱动。
驱动添加需要配置 linux 系统内核，实例验证内核 3.3.8 配置方法如下：
cd kernel
make menuconfig
device drivers→usb support→usb serial converter support
选中如下组件：
USB driver for GSM and CDMA modems
选中后保存配置，重新编译内核即可。

2：增加 PID/VID

找到内核源码文件 option.c(一般情况下，路径在..\drivers\usb\serial\option.c)
在源码中找到 option_ids 表，增加 longsung VID (0x1C9E)和 PID(0x9B05)。

3：跳过 NDIS Interface

添加 *static const struct option_blacklist_info longsung_u8300_blacklist*,将 USB 串口驱动中的 NDIS Interface4 跳过，做到不被 USB Serial 驱动枚举，预留给 NDIS 驱动。

添加完成后，重新编译内核，烧录目标设备。
详细如下：
option.c

```
/* Longcheer/Longsung vendor ID; makes whitelabel devices that
 * many other vendors like 4G Systems, Alcatel, ChinaBird,
 * Mobidata, etc sell under their own brand names.
 */
#define LONGCHEER_VENDOR_ID                0x1c9e
//to longsung modem for NDIS
#define LONGSUNG_VENDOR_ID 0x1c9e
#define LONGSUNG_U8300_PRODUCT_ID 0x9b05
… …
static const struct option_blacklist_info zte_mf626_blacklist = {
        .sendsetup = BIT(0) | BIT(1),
        .reserved = BIT(4),
};
//to longsung modem for NDIS
static const struct option_blacklist_info longsung_u8300_blacklist = {
        .reserved = BIT(4),
};
… …
static const struct usb_device_id option_ids[] = {
… …
        { USB_DEVICE(LONGCHEER_VENDOR_ID, ZOOM_PRODUCT_4597) },
//to longsung modem for NDIS
        { USB_DEVICE(LONGSUNG_VENDOR_ID, LONGSUNG_U8300_PRODUCT_ID),
          .driver_info = (kernel_ulong_t)&longsung_u8300_blacklist
        },
… …
}
```

static const struct option_blacklist_info longsung_u8300_blacklist 功能在高版本的内核才支持。
如果对于较早的内核，需要在 option.c(低版本为 usb-serial.c 中的 usb_serial_probe)中的 probe 函数内增加 backlist 进行过滤。

```
… …
static int option_probe(struct usb_serial *serial,
                        const struct usb_device_id *id)
{
… …
        if (serial->dev->descriptor.idVendor == LONGSUNG_VENDOR_ID &&
            serial->dev->descriptor.idProduct == LONGSUNG_U8300_PRODUCT_ID &&
            serial->interface->cur_altsetting->desc.bInterfaceNumber == 4)
        {
```

```
            return -ENODEV;
        }
… …
}
… …
```

4：内核打印

跳过了 Interface 4，并枚举出来了其它 Interface，具体如下所示：

```
[ 1979.740202] usb 1-2: new full-speed USB device number 3 using ohci_hcd
[ 1980.284074] usbcore: registered new interface driver usbserial
[ 1980.284094] USB Serial support registered for generic
[ 1980.284354] usbcore: registered new interface driver usbserial_generic
[ 1980.284356] usbserial: USB Serial Driver core
[ 1980.307639] USB Serial support registered for GSM modem (1-port)
[ 1980.308284] option 1-2:1.0: GSM modem (1-port) converter detected
[ 1980.308856] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB0
[ 1980.308888] option 1-2:1.1: GSM modem (1-port) converter detected
[ 1980.309244] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB1
[ 1980.309383] option 1-2:1.2: GSM modem (1-port) converter detected
[ 1980.309937] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB2
[ 1980.310061] option 1-2:1.3: GSM modem (1-port) converter detected
[ 1980.310620] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB3
[ 1980.310762] option 1-2:1.5: GSM modem (1-port) converter detected
[ 1980.311309] usb 1-2: GSM modem (1-port) converter now attached to ttyUSB4
[ 1980.311454] usbcore: registered new interface driver option
[ 1980.311456] option: v0.7.2:USB Driver for GSM modems
```

对于 NDIS 功能

1：usbnet 驱动支持

NDIS 驱动需要系统的 usbnet 驱动支持，因此需要配置 linux 内核，配置方法如下：

cd kernel

make menuconfig

device drivers→Network device support→USB Network Adapters

选中如下组件

Multi-purpose USB Networking Framework

选中后保存配置，重新编译内核

2：NDIS 驱动编译

用户可以单独编译，也可以将代码放入内核中，一起编译。
a 单独编译
修改 src/Makefile 中的 KDIR 的值为 kernel 的编译路径；
在 ndis_driver 目录下执行 make modules 命令，即可在 src 目录下生成 lc_ether.ko 文件
b 与内核一起编译
将 src 下的代码文件，复制到用户自己的 kernel 代码的 drivers/net/usb 目录下：
在 drivers/net/usb/Makefile 中增加以下内容：

```
lc_ether-obj+=qmi_oper.o qmi_util.o \
                Lc_cdc_ether.o
obj-m+=lc_ether-o
```

之后每次编译内核都会自动编译 NDIS 驱动。

3：驱动加载

modprobe lc_ether
lsmod 查看会有 lc_ether 驱动存在，且使用 ifconfig 命令查看网卡信息，会出现 wan0 interface 存在。

```
wan0        Link encap:Ethernet  HWaddr 00:a0:c6:00:00:00
            UP 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 (0.0 B)  TX bytes:0 (0.0 B)
```

4：修改拨号配置文件

修改 profile.ini 文件，内容如下所示：

```
[profile]
apn=UNINET
usrname=
pwd=
auth=0
ipfamily=4
```

apn，usrname，pwd 这三个参数很简单，
auth 用于拨号的 auth type，

```
1----PAP;
2----CHAP;
3----PAP & CHAP
0----default
```

ipfamily 用于拨号的 ip family:

ipv4==4

ipv6==6

unspec=8,//default ipv4

ipv4ipv6=0x10 //longcheer defined

5：测试

#:~/driver/version3.2/ndis_app$ ./ndis_test

ifr.ifr_name = wan0

ndis open exit.

NDIS_CONNECTING------0X001.

NDIS_CONNECTED-------0X002.

NDIS_DISCONNECTED----0X004.

ipv4 status -------4.

ipv6 status -------4.

ndis dail test!

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

2

connection use: apn:UNINET,usrname:,pwd:,auth:0,ipfamily:6.

Enter extern_ndis_connect ndis_fd=157691912, 239

Enter ndis_qmi_connect 163

extern_ndis_connect success.

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

4

NDIS_CONNECTING------0X001.

NDIS_CONNECTED-------0X002.

NDIS_DISCONNECTED----0X004.

ipv4 status -------4.

ipv6 status -------2.

ndis_get_status success.

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

4

NDIS_CONNECTING------0X001.

NDIS_CONNECTED-------0X002.

NDIS_DISCONNECTED----0X004.

ipv4 status -------4.

ipv6 status -------2.

ndis_get_status success.

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

-1

Enter ndis_close 573

#:~/driver/version3.2/ndis_app$ ifconfig wan0 down

#:~/driver/version3.2/ndis_driver# ifconfig wan0 down

#:~/driver/version3.2/ndis_driver# ifconfig wan0 up

#:~/driver/version3.2/ndis_driver# ping6 fc01:cafe::1

PING fc01:cafe::1(fc01:cafe::1) 56 data bytes

64 bytes from fc01:cafe::1: icmp_seq=1 ttl=64 time=19.7 ms

64 bytes from fc01:cafe::1: icmp_seq=2 ttl=64 time=19.0 ms

64 bytes from fc01:cafe::1: icmp_seq=3 ttl=64 time=18.1 ms

#:~/driver/version3.2/ndis_app$ ./ndis_test

ifr.ifr_name = wan0

ndis open exit.

NDIS_CONNECTING------0X001.

NDIS_CONNECTED-------0X002.

NDIS_DISCONNECTED----0X004.

ipv4 status -------4.

ipv6 status -------4.

ndis dail test!

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

2

connection use: apn:UNINET,usrname:,pwd:,auth:0,ipfamily:4.

Enter extern_ndis_connect ndis_fd=158326792, 239

Enter extern_ndis_connect 260

Enter ndis_qmi_connect 163

extern_ndis_connect success.

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

4

NDIS_CONNECTING------0X001.

NDIS_CONNECTED-------0X002.

NDIS_DISCONNECTED----0X004.

ipv4 status -------2.

ipv6 status -------4.

ndis_get_status success.

1.....get lib version.

2.....connect to internet(extern ndis_qmi_connect, support v4v6 double stack).

3.....disconnect to internet(extern ndis_qmi_disconnect,support v4v6 double stack).

4.....get current status.

5.....ndis_go_active.

6.....get client ID.

7.....connect to internet(ndis_qmi_connect, original qmi dial not support v4v6 double stack).

8.....disconnect to internet(ndis_qmi_disconnect, original qmi dial not support v4v6 double stack).

-1.....exit.

-1

Enter ndis_close 573

#:~/driver/version3.2/ndis_driver# dhclient wan0

There is already a pid file /var/run/dhclient.pid with pid 4483

killed old client process, removed PID file

Internet Systems Consortium DHCP Client V3.1.3

Copyright 2004-2009 Internet Systems Consortium.

All rights reserved.

For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/wan0/00:a0:c6:00:00:00

Sending on LPF/wan0/00:a0:c6:00:00:00

Sending on Socket/fallback

DHCPREQUEST of 172.22.1.100 on wan0 to 255.255.255.255 port 67

DHCPACK of 172.22.1.100 from 172.22.1.101

bound to 172.22.1.100 -- renewal in 2869 seconds.

#:~/driver/version3.2/ndis_driver# ifconfig

wan0 Link encap:Ethernet HWaddr 00:a0:c6:00:00:00

inet addr:172.22.1.100 Bcast:172.22.1.103 Mask:255.255.255.252

inet6 addr: fc01:abab:cdcd:efe0:2a0:c6ff:fe00:0/64 Scope:Global

inet6 addr: fe80::2a0:c6ff:fe00:0/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:1 errors:13 dropped:0 overruns:0 frame:0

TX packets:9 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:320 (320.0 B) TX bytes:3173 (3.1 KB)

#:~/driver/version3.2/ndis_driver# ping 172.22.1.201

PING 172.22.1.201 (172.22.1.201) 56(84) bytes of data.

64 bytes from 172.22.1.201: icmp_seq=1 ttl=64 time=20.5 ms

64 bytes from 172.22.1.201: icmp_seq=2 ttl=64 time=18.1 ms

64 bytes from 172.22.1.201: icmp_seq=3 ttl=64 time=17.9 ms

NDIS 驱动层 API 说明及相关代码

详见文档《Linux NDIS API User Guide.pdf》

Linux NDIS API User Guide.pdf

version3.3.tgz