Get connected Get smart

N58

Linux Integration Guide

Issue 1.0 Date 2020-06-22

Notice

This document provides guide for users to use N58.

This document is intended for system engineers (SEs), development engineers, and test engineers.

THIS GUIDE PROVIDES INSTRUCTIONS FOR CUSTOMERS TO DESIGN THEIR APPLICATIONS. PLEASE FOLLOW THE RULES AND PARAMETERS IN THIS GUIDE TO DESIGN AND COMMISSION. NEOWAY WILL NOT TAKE ANY RESPONSIBILITY OF BODILY HURT OR ASSET LOSS CAUSED BY IMPROPER OPERATIONS.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE DUE TO PRODUCT VERSION UPDATE OR OTHER REASONS.

EVERY EFFORT HAS BEEN MADE IN PREPARATION OF THIS DOCUMENT TO ENSURE ACCURACY OF THE CONTENTS, BUT ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENT DO NOT CONSTITUTE A WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

Neoway provides customers complete technical support. If you have any question, please contact your account manager or email to the following email addresses:

Sales@neoway.com

Support@neoway.com

Website: http://www.neoway.com

Contents

About This Document

Scope

This document is applicable to the N58 series.

Audience

This document is intended for system engineers (SEs), development engineers, and test engineers.

Change History

Issue	Date	Change	Changed By
1.0	2020-04	Initial draft	Wu Guoqing

Conventions

Symbol	Indication
❗	This warning symbol means danger. You are in a situation that could cause fatal device damage or even bodily damage.
⚠	Means reader be careful. In this situation, you might perform an action that could result in module or product damages.
ℹ	Means note or tips for readers to use the module

Related Documents

*Neoway*_N58_*Datasheet*

*Neoway*_N58_*Product_Specifications*

*Neoway*_N58_*HW_User_Guide*

*Neoway*_N58_*AT_Command_Mannual*

*Neoway*_N58_*EVK_User_Guide*

1 Port Mapping

N58 supports dial-up networking via PPP, RNDIC, etc. Variants with different Vender IDs (VIDs) and product IDs (PIDs) support different networking dial-up mode. You can select a mode through **AT+NETSHAREMODE=1/0**. The mapping between the ports and dial-up modes is shown in the following table.

Table 1-1 N58 port mapping

Model	VID	PID	Dial-up Mode	Port Mapping	Function
N58	0x2949	0x7401	PPP/RNDIS	RNDIS(01)	RNDIS NIC port
				MODEM(02)	Private data service port
				NMEA (03)	GPS data output port
				AT(04)	AT Command port
				Diag(05)	Diagnosis port
				CPLOG(06)	Kernel CP log output port
				APLOG(07)	AP log output port
				OPENCON(08)	Log output port for OpenCPU
	0x2949	0x7402	PPP/ECM	ECM(00)	RNDIS NIC port
				MODEM(02)	Private data service port
				NMEA (03)	GPS/BD data output port
				AT(04)	AT Command port
				Diag(05)	Diagnosis port
				CPLOG(06)	Kernel CP log output port
				APLOG(07)	AP log output port
				OPENCON(08)	Log output port for OpenCPU

2 Loading USB-to-Serial Driver

The host communicates with the module through USB ports, the functions of which might vary. For example, AT port is used to issue AT commands and return reponses, Modem port is used for PPP dial-up (via AT commands), and CPLOG port is used for Coolwatcher debugging. For details of the N58 ports, see Table 1-1.

To ensure that the host can identify N58 and enumerate the USB ports as ttyUSB devices in **/dev/**, you need to add the USB-to-serial driver into the host.

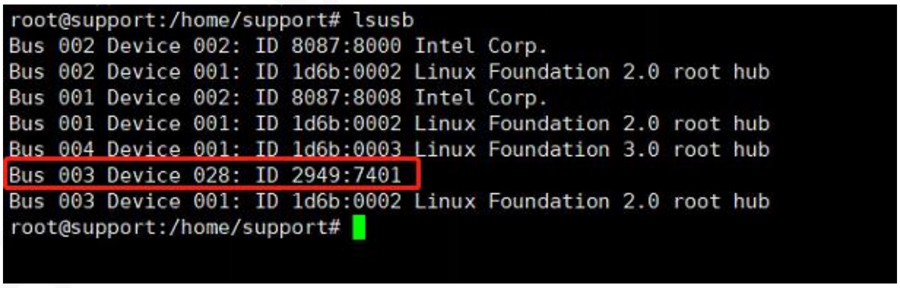
This chapter mainly describes how to check whether the host detects the USB devices of the module and how to load the USB-to-Serial driver, including loading the driver through commands and through rebuilding the kernel after modifying the code.

2.1 Identifying Module

You can check whether the host identifies the N58 module in two manners:

- **lsusb**

  If the host supports **lsusb**, issue **lsusb** to check USB devices.

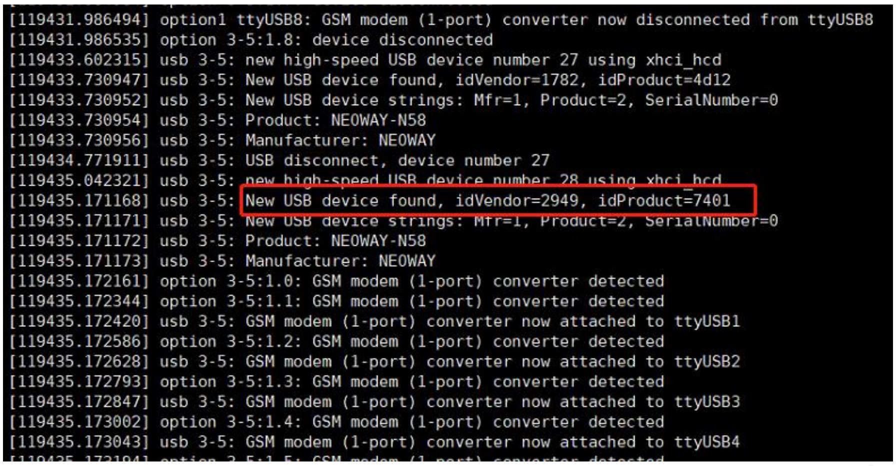


```
root@support:/home/support# lsusb
Bus 002 Device 002: ID 8087:8000 Intel Corp.
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 8087:8008 Intel Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 028: ID 2949:7401
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
root@support:/home/support#
```

  If the host identifies the module, the terminal prints the VID and PID (2949:7401) of the N58 module.

- dmesg log

  If the host does not support **lsusb**, capture the dmesg log to check whether the host identifies N58.



```
[119431.986494] option1 ttyUSB8: GSM modem (1-port) converter now disconnected from ttyUSB8
[119431.986535] option 3-5:1.8: device disconnected
[119433.602315] usb 3-5: new high-speed USB device number 27 using xhci_hcd
[119433.730947] usb 3-5: New USB device found, idVendor=1782, idProduct=4d12
[119433.730952] usb 3-5: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[119433.730954] usb 3-5: Product: NEOWAY-N58
[119433.730956] usb 3-5: Manufacturer: NEOWAY
[119434.771911] usb 3-5: USB disconnect, device number 27
[119435.042321] usb 3-5: new high-speed USB device number 28 using xhci_hcd
[119435.171168] usb 3-5: New USB device found, idVendor=2949, idProduct=7401
[119435.171171] usb 3-5: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[119435.171172] usb 3-5: Product: NEOWAY-N58
[119435.171173] usb 3-5: Manufacturer: NEOWAY
[119435.172161] option 3-5:1.0: GSM modem (1-port) converter detected
[119435.172344] option 3-5:1.1: GSM modem (1-port) converter detected
[119435.172420] usb 3-5: GSM modem (1-port) converter now attached to ttyUSB1
[119435.172586] option 3-5:1.2: GSM modem (1-port) converter detected
[119435.172628] usb 3-5: GSM modem (1-port) converter now attached to ttyUSB2
[119435.172793] option 3-5:1.3: GSM modem (1-port) converter detected
[119435.172847] usb 3-5: GSM modem (1-port) converter now attached to ttyUSB3
[119435.173002] option 3-5:1.4: GSM modem (1-port) converter detected
[119435.173043] usb 3-5: GSM modem (1-port) converter now attached to ttyUSB4
```

If the host does not identify the module, troubleshoot the issue by checking the hardware circuits:

- Whether the N58 is connected to the host.
- Whether the N58 is powered on and its 45th pin outputs 1.8 V level.
- Whether the USB cables are connected properly and whether they are damaged.

2.2 Adding USB-to-Serial Driver

When debugging the N58 module in a Linux OS, you can adopt one of the following two methods to add the USB-to-Serial driver and enumerate the USB devices:

- by executing commands
- by rebuilding kernel.

Ensure that the host detects the module before adding the driver.

2.2.1 By Executing Commands (valid temporarily)

You can add the USB-to-Serial driver through the **modprobe** Linux command. Adopt one of the following two methods to add the driver. Note that the driver added in this method will not work after the host is restarted.

- Add the drivers and enable them in one step.

```
sudo modprobe usbserial vendor=0x2949 product=0x7401
```

- Perform the execution in two steps.

```
 sudo modprobe option   // load the driver
 sudo echo 2949 7401 > /sys/bus/usb-serial/drivers/option1/new_id   // enable the driver
```

After performing the execution, the device nodes ttyUSB0, ttyUSB1, ttyUSB2, and ttyUSB3 are displayed in **/dev**.

2.2.2 By Rebuilding Kernel

You can also add the driver into kernel by modifying **kernel/drivers/usb/serial/option.c** and then compiling it. The driver added though this method is valid all the time even if the host is restarted.

Editing Code

You can decide whether to enumerate all ports.

- If only the PPP dial-up is used, you do not have to filter out ports.

  Open the **option.c** file and Add the VIDs and PIDs of N58 to the data set **option_ids[]**.

```
 static const struct usb_device_id option_ids[] = {
    { USB_DEVICE(0x2949, 0x7401) },
    { USB_DEVICE(0x2949, 0x7402) },
… …
}
```

- If you need the NIC port for dial-up connection, modify the **option.c** file in either of the following way to filter out the NIC port:

  - Blacklist

    Add a blacklist to the **option_ids[]** data set in **option.c** and then add the corresponding NIC port to the blacklist. The port specified in the blacklist will be skipped when the host loads USB-to-serial driver.

    This method is recommended for Linux kernels later than 3.8.

```
static const struct option_blacklist_info neoway_2949_7401_blacklist = {
    .reserved = BIT(1) , //filter out port 1
    };

    static const struct option_blacklist_info neoway_2949_7402_blacklist = {
     .reserved = BIT(0),   // filter out port 0
    };
    static const struct usb_device_id option_ids[] = {
         {USB_DEVICE(0x2949, 0x7401),
              .driver_info = (kernel_ulong_t)&neoway_2949_7401_blacklist },
         {USB_DEVICE(0x2949, 0x7402),
          .driver_info = (kernel_ulong_t)&neoway_2949_7402_blacklist },
    … …
 }
```

  - **option_probe** function

The kernel version of Linux OS earlier than 3.8 does not support blacklist, and you can add an if statement in the **option_probe** function to filter out the NIC port.

For example, filter out RNDIS (port 1), add the following code to the **option_probe** function to.

```
static int option_probe(struct usb_serial *serial,const struct usb_device_id *id)
{
… …
     if (serial->dev->descriptor.idVendor == 0x2949 &&
     serial->dev->descriptor.idProduct == 0x7401 &&
     serial->interface->cur_altsetting->desc.bInterfaceNumber == 1)
     //only filter out port 1
     return -ENODEV;
 /* Store device id so we can use it during attach. */
     usb_set_serial_data(serial, (void *)id);
     return 0;
  }
```

- To filter out other ports, add an if statement in the **option_probe** function.

  For example, add the following code to filter out ports 1, 3, and 5 and reserve the other ports:

```
static int option_probe(struct usb_serial *serial,const struct usb_device_id *id)
{
… …
   if (serial->dev->descriptor.idVendor == 0x2949 &&
   serial->dev->descriptor.idProduct == 0x7401 &&
  (serial->interface->cur_altsetting->desc.bInterfaceNumber == 1 ||
serial->interface->cur_altsetting->desc.bInterfaceNumber == 3 ||
serial->interface->cur_altsetting->desc.bInterfaceNumber == 5))
//filter out ports 1, 3 and 5
  return -ENODEV;
  /* Store device id so we can use it during attach. */
  usb_set_serial_data(serial, (void *)id);
  return 0;
  }
```

Rebuilding Kernel

Follow the steps below to add the USB-to-Serial driver into the Linux kernel.

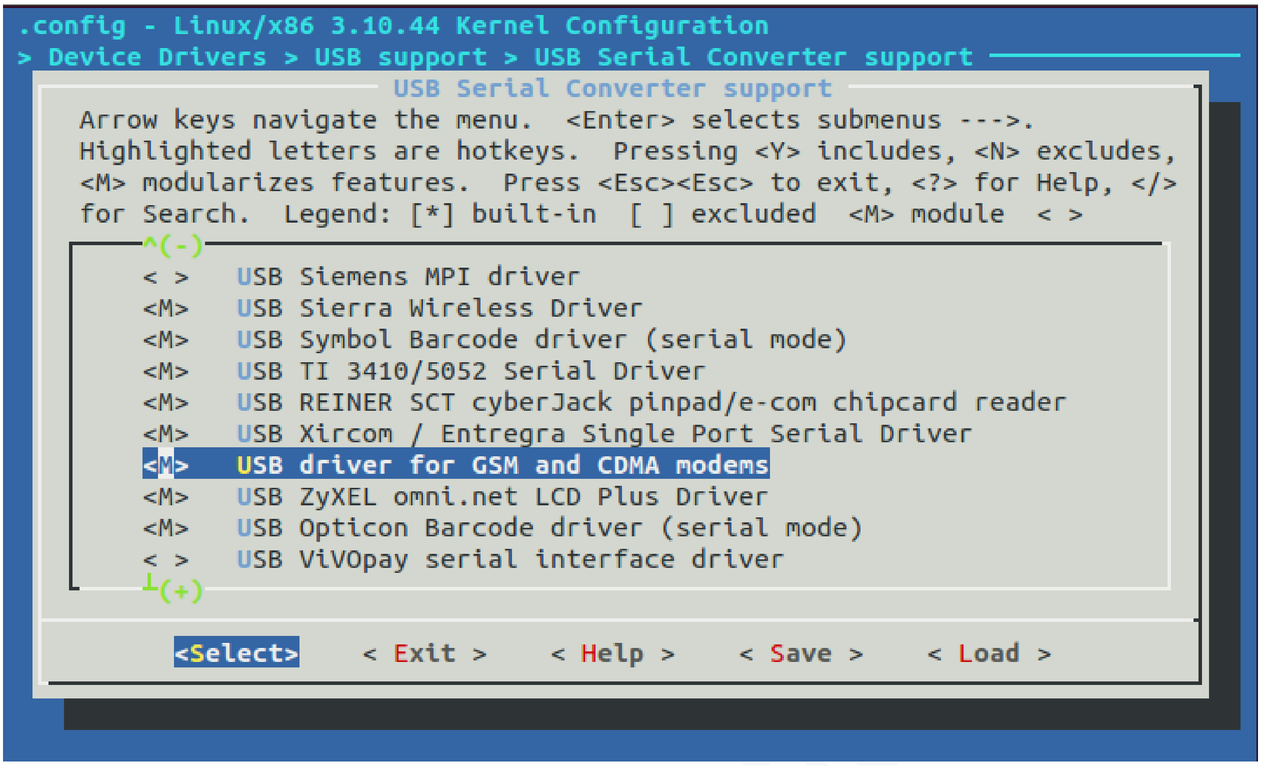
Step 1: Navigate to the Linux kernel directory.

Step 2: Execute **make menuconfig**.

The kernel configuration is displayed.

Step 3: Configure kernel and select **USB driver for GSM and CDMA modems**.

```
Device Drivers  --->
  [*] USB support  --->
     <*>   USB Serial Converter support  --->
     <*>   USB driver for GSM and CDMA modems (CONFIG_USB_SERIAL_OPTION=y)
```



```
.config - Linux/x86 3.10.44 Kernel Configuration
> Device Drivers > USB support > USB Serial Converter support
                    USB Serial Converter support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
      ^(-)
      < >   USB Siemens MPI driver
      <M>   USB Sierra Wireless Driver
      <M>   USB Symbol Barcode driver (serial mode)
      <M>   USB TI 3410/5052 Serial Driver
      <M>   USB REINER SCT cyberJack pinpad/e-com chipcard reader
      <M>   USB Xircom / Entregra Single Port Serial Driver
      <M>   USB driver for GSM and CDMA modems
      <M>   USB ZyXEL omni.net LCD Plus Driver
      <M>   USB Opticon Barcode driver (serial mode)
      < >   USB ViVOpay serial interface driver
      └(+)

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

Step 4: Select **Save** and **Exit**.

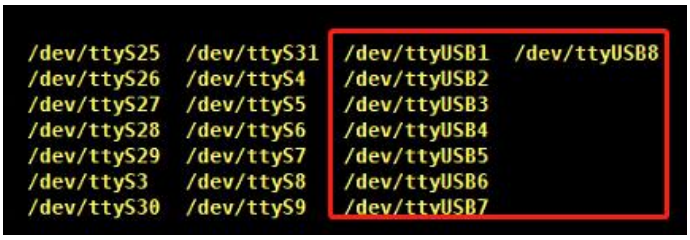
Step 5: Compile the source code to rebuild kernel and then restart the host.

2.3 Verifying Added Driver

After the USB-to-serial driver is added, perform the following steps to verify whether the driver is added successfully:

Step 1: Power up and restart the host and issue **ls /dev/ttyUSB*** to query ttyUSB devices.

If the host enumerates ttyUSB devices in the **/dev** directory, the USB-to-Serial driver has been added successfully.



```
/dev/ttyS25  /dev/ttyS31  /dev/ttyUSB1  /dev/ttyUSB8
/dev/ttyS26  /dev/ttyS4   /dev/ttyUSB2
/dev/ttyS27  /dev/ttyS5   /dev/ttyUSB3
/dev/ttyS28  /dev/ttyS6   /dev/ttyUSB4
/dev/ttyS29  /dev/ttyS7   /dev/ttyUSB5
/dev/ttyS3   /dev/ttyS8   /dev/ttyUSB6
/dev/ttyS30  /dev/ttyS9   /dev/ttyUSB7
```

Step 2: Check the ports mapping to the ttyUSB devices.

Table 2-1 shows the ports mapping of N58 after the driver is loaded by the host.

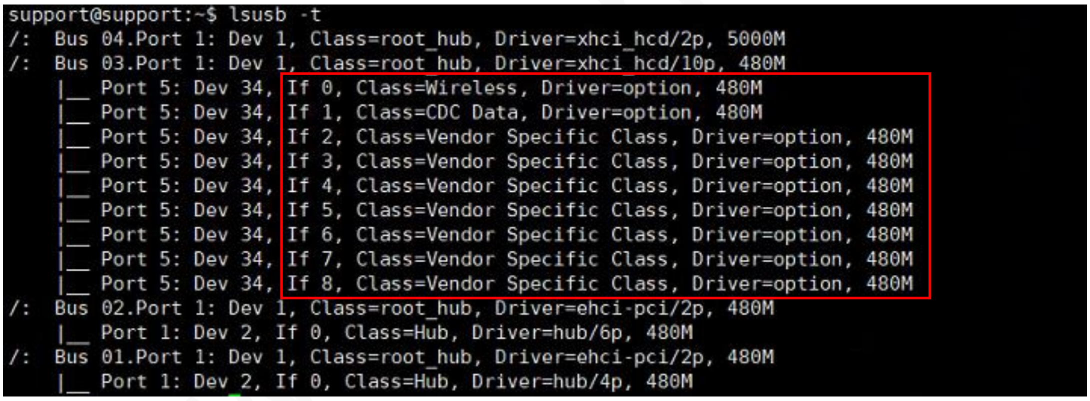
Table 2-1 Ports mapping to the ttyUSB devices

Port Name	Function	Device Node
RNDIS	RNDIS NIC port	ttyUSB0
MODEM	Private data service port	ttyUSB1
NMEA	GPS data output port	ttyUSB2
AT	AT Command port	ttyUSB3
Diag	Diagnosis port	ttyUSB4
CPLOG	Kernel CP log output port	ttyUSB5
APLOG	Application log output port	ttyUSB6
OPENCON	Log output port of OpenCPU	ttyUSB7

Both the modem port and the AT port can be used for AT transmission. To locate the correct ttyUSB device, you can send AT commands through the ttyUSB devices one by one.

Step 3: Check the port mapping.

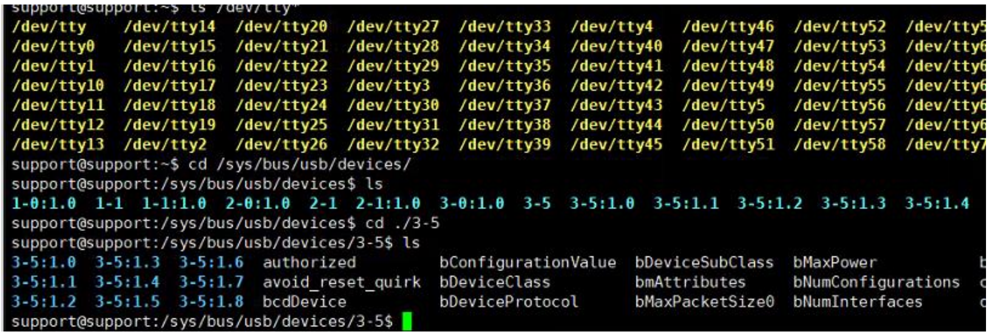
1. Issue **lsusb -t** to check the USB information of the host.



```
support@support:~$ lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/2p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/10p, 480M
    |__ Port 5: Dev 34, If 0, Class=Wireless, Driver=option, 480M
    |__ Port 5: Dev 34, If 1, Class=CDC Data, Driver=option, 480M
    |__ Port 5: Dev 34, If 2, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 3, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 4, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 5, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 6, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 7, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 34, If 8, Class=Vendor Specific Class, Driver=option, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/6p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
```

2. Use the **ls** command to check the port information in the **sysfs** directory.

The information of all USB devices are saved in the 3-5 folders under **/sys/bus/usb/devices/**. For example, 3-5:1.2 corresponds to the modem port; 3-5:1.4 corresponds to the AT port.



```
support@support:~$ ls /dev/tty*
/dev/tty    /dev/tty14  /dev/tty20  /dev/tty27  /dev/tty33  /dev/tty4   /dev/tty46  /dev/tty52  /dev/tty5
/dev/tty0   /dev/tty15  /dev/tty21  /dev/tty28  /dev/tty34  /dev/tty40  /dev/tty47  /dev/tty53  /dev/tty6
/dev/tty1   /dev/tty16  /dev/tty22  /dev/tty29  /dev/tty35  /dev/tty41  /dev/tty48  /dev/tty54  /dev/tty6
/dev/tty10  /dev/tty17  /dev/tty23  /dev/tty3   /dev/tty36  /dev/tty42  /dev/tty49  /dev/tty55  /dev/tty6
/dev/tty11  /dev/tty18  /dev/tty24  /dev/tty30  /dev/tty37  /dev/tty43  /dev/tty5   /dev/tty56  /dev/tty6
/dev/tty12  /dev/tty19  /dev/tty25  /dev/tty31  /dev/tty38  /dev/tty44  /dev/tty50  /dev/tty57  /dev/tty6
/dev/tty13  /dev/tty2   /dev/tty26  /dev/tty32  /dev/tty39  /dev/tty45  /dev/tty51  /dev/tty58  /dev/tty7
support@support:~$ cd /sys/bus/usb/devices/
support@support:/sys/bus/usb/devices$ ls
1-0:1.0  1-1  1-1:1.0  2-0:1.0  2-1  2-1:1.0  3-0:1.0  3-5  3-5:1.0  3-5:1.1  3-5:1.2  3-5:1.3  3-5:1.4
support@support:/sys/bus/usb/devices$ cd ./3-5
support@support:/sys/bus/usb/devices/3-5$ ls
3-5:1.0  3-5:1.3  3-5:1.6  authorized         bConfigurationValue  bDeviceSubClass  bMaxPower           b
3-5:1.1  3-5:1.4  3-5:1.7  avoid_reset_quirk  bDeviceClass         bmAttributes     bNumConfigurations  c
3-5:1.2  3-5:1.5  3-5:1.8  bcdDevice          bDeviceProtocol      bMaxPacketSize0  bNumInterfaces      c
support@support:/sys/bus/usb/devices/3-5$ 
```

3 Debugging Module

The AT function of the module can be debugged through the serial-port tool or echo commands on a Linux OS.

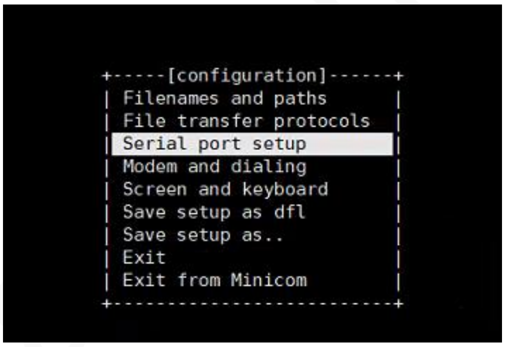
3.1 Through Serial-Port Tool

minicom is one of the most commonly used serial-port tools on Linux. After locating the USB device mapping to the AT port, perform the following operations in the tool to interact with N58:

Step 1: Start minicom and configure the parameters of the serial ports.

1. Run the **minicom –s** command. The configuration UI is displayed.

2. Select **Serial port setup** and press **Enter**.

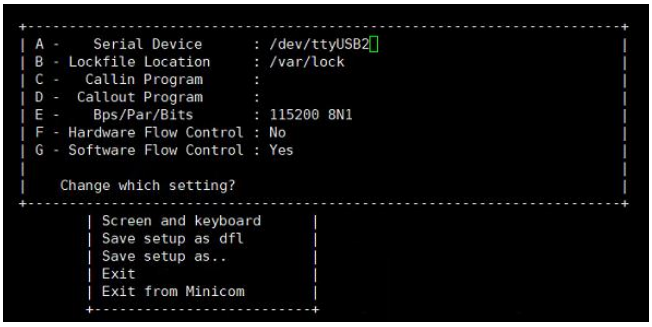


```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

3. In the setup UI, select a letter to set the parameter.

   Generally, you need to specify **Serial Device** only and the rest parameters use the default values.

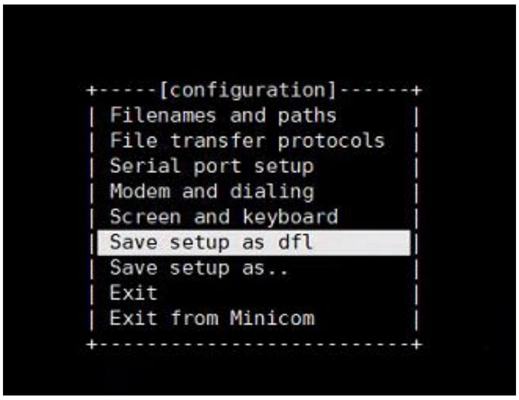
4. Type A, modify the value to **/dev/ttyUSB2**, and then press **Enter**.



```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/ttyUSB2                              |
| B - Lockfile Location     : /var/lock                                 |
| C -   Callin Program      :                                           |
| D -  Callout Program      :                                           |
| E -    Bps/Par/Bits       : 115200 8N1                                |
| F - Hardware Flow Control : No                                        |
| G - Software Flow Control : Yes                                       |
|                                                                       |
|   Change which setting?                                               |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

5. Select **Save setup as dfl**.

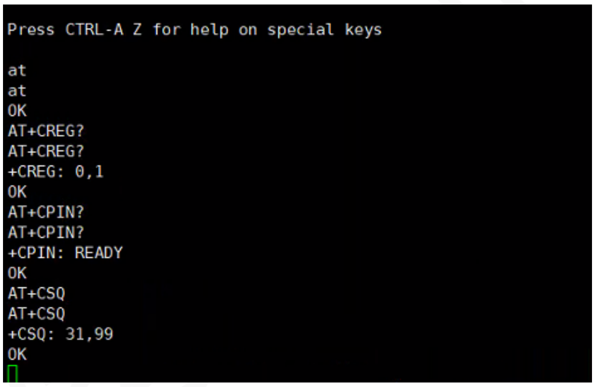
   The configurations will be saved as the default setup. You can skip this step and run minicom commands directly next time.



```
+-----[configuration]------+
| Filenames and paths      |
| File transfer protocols  |
| Serial port setup        |
| Modem and dialing        |
| Screen and keyboard      |
| Save setup as dfl        |
| Save setup as..          |
| Exit                     |
| Exit from Minicom        |
+--------------------------+
```

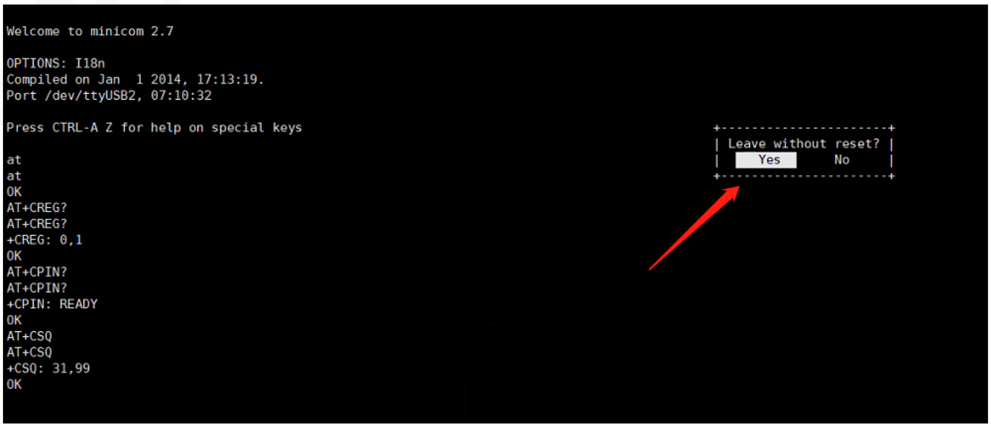
6. Select **Exit**.

Step 2: Send AT commands to interact with the module.



```
Press CTRL-A Z for help on special keys

at
at
OK
AT+CREG?
AT+CREG?
+CREG: 0,1
OK
AT+CPIN?
AT+CPIN?
+CPIN: READY
OK
AT+CSQ
AT+CSQ
+CSQ: 31,99
OK
```

Step 3: Press **Ctrl+A** and then press **Q** to exit from the minicom tool.

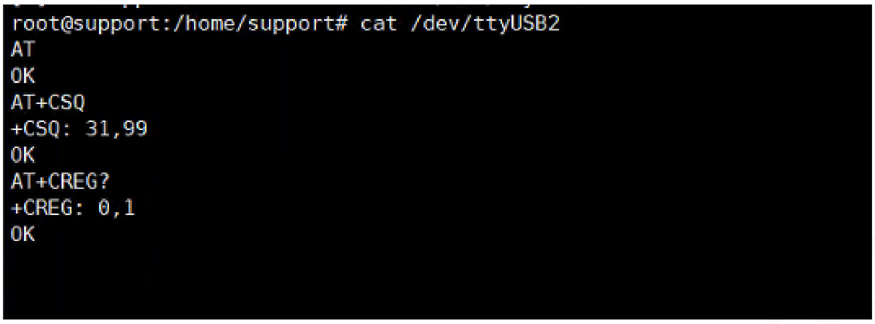


```
Welcome to minicom 2.7

OPTIONS: I18n
Compiled on Jan  1 2014, 17:13:19.
Port /dev/ttyUSB2, 07:10:32

Press CTRL-A Z for help on special keys

at
at
OK
AT+CREG?
AT+CREG?
+CREG: 0,1
OK
AT+CPIN?
AT+CPIN?
+CPIN: READY
OK
AT+CSQ
AT+CSQ
+CSQ: 31,99
OK
```

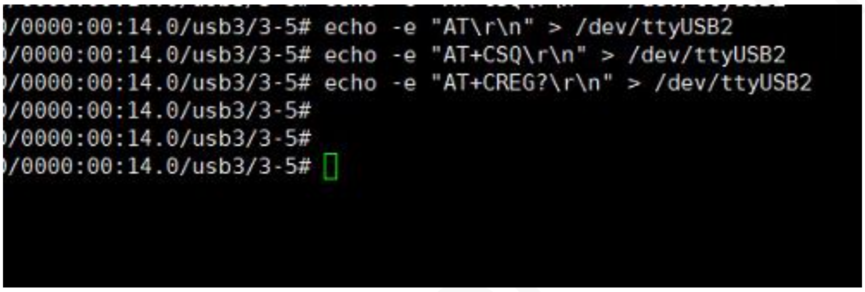
```
+----------------------+
| Leave without reset? |
|    Yes        No     |
+----------------------+
```

3.2 Through echo Commands

echo commands can be used to print character strings in terminals and send AT commands to ttyUSB devices. To facilitate debugging, start two terminals: one for AT command input, and one for AT response display.

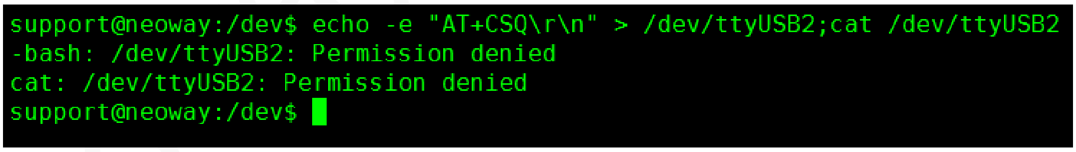


```
root@support:/home/support# cat /dev/ttyUSB2
AT
OK
AT+CSQ
+CSQ: 31,99
OK
AT+CREG?
+CREG: 0,1
OK
```



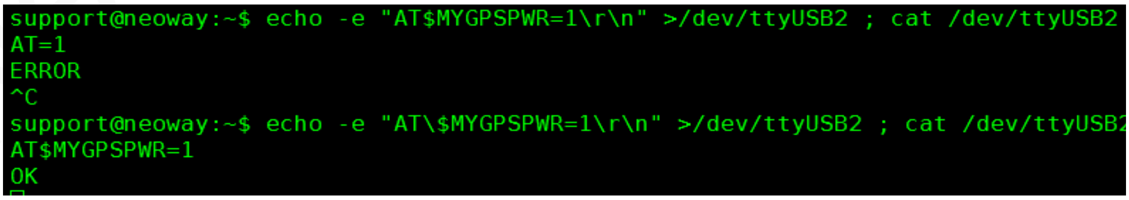
```
)/0000:00:14.0/usb3/3-5# echo -e "AT\r\n" > /dev/ttyUSB2
)/0000:00:14.0/usb3/3-5# echo -e "AT+CSQ\r\n" > /dev/ttyUSB2
)/0000:00:14.0/usb3/3-5# echo -e "AT+CREG?\r\n" > /dev/ttyUSB2
)/0000:00:14.0/usb3/3-5#
)/0000:00:14.0/usb3/3-5#
)/0000:00:14.0/usb3/3-5#
```

- Modem port and AT port can be used to send AT commands. It is recommended to use the AT port. If modem port is used to issue commands to N58, it cannot be used for PPP dial-up, resulting in the failure of network share via PPP.
- If any messages are displayed to ask for permission, modify the permission for the ttyUSB port to root.



```
support@neoway:/dev$ echo -e "AT+CSQ\r\n" > /dev/ttyUSB2;cat /dev/ttyUSB2
-bash: /dev/ttyUSB2: Permission denied
cat: /dev/ttyUSB2: Permission denied
support@neoway:/dev$
```

- If the AT command contains any special character, add \ before it.



```
support@neoway:~$ echo -e "AT$MYGPSPWR=1\r\n" >/dev/ttyUSB2 ; cat /dev/ttyUSB2
AT=1
ERROR
^C
support@neoway:~$ echo -e "AT\$MYGPSPWR=1\r\n" >/dev/ttyUSB2 ; cat /dev/ttyUSB2
AT$MYGPSPWR=1
OK
```

4 PPP Dial-up

All two configuration modes of N58 support dialup networking using PPP. This chapter describes how to access to the Internet using the PPPD script on Linux. The N58 module with VID:0x2949 PID:0x7401 is used as an example.

4.1 Adding USB Driver

Refer to Chapter 2 to add the USB-to-Serial driver and find the corresponding modem port.

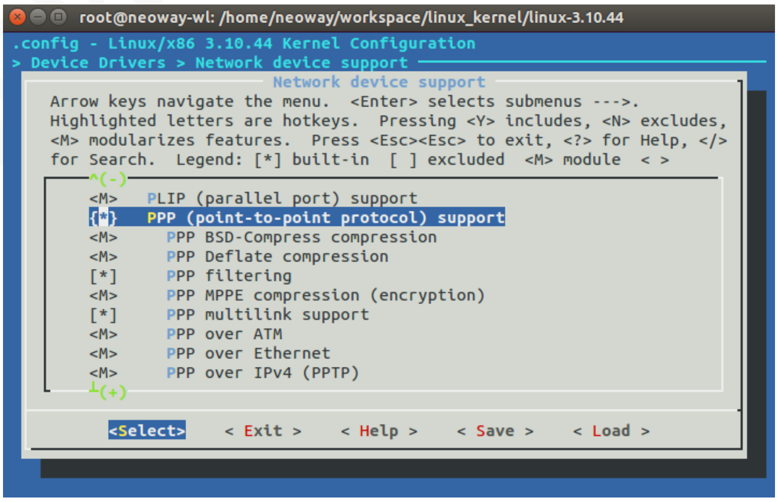
4.2 Adding PPP Support to Kernel

1. Configure PPP in the kernel in the following path.

```
        Device Drivers  --->
            [*]  Network device support (NETDEVICES [=y])  --->
            {*}  PPP(point-to-point protocol)support
```

   Linux generally embeds PPP in its kernel by default.

2. Select **Save** and then **Exit**.
3. Compile the code to rebuild kernel and then burn it to the host.
4. Restart the host.



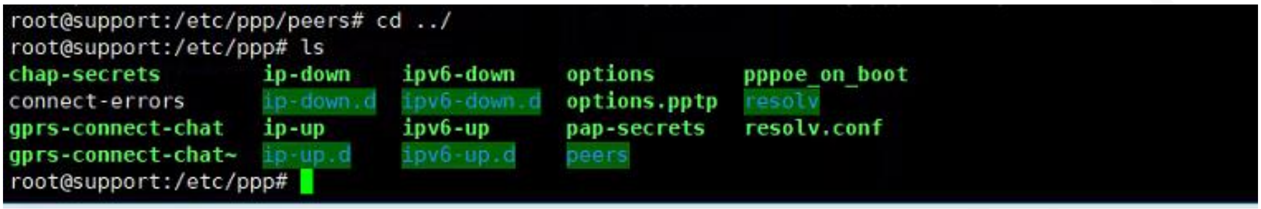
```
root@neoway-wl: /home/neoway/workspace/linux_kernel/linux-3.10.44
.config - Linux/x86 3.10.44 Kernel Configuration
> Device Drivers > Network device support
                        Network device support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
      ^(-)
      <M>   PLIP (parallel port) support
      {*}   PPP (point-to-point protocol) support
      <M>     PPP BSD-Compress compression
      <M>     PPP Deflate compression
      [*]     PPP filtering
      <M>     PPP MPPE compression (encryption)
      [*]     PPP multilink support
      <M>     PPP over ATM
      <M>     PPP over Ethernet
      <M>     PPP over IPv4 (PPTP)
      └(+)

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

4.3 PPPd Dial-up Scripts

Ensure that the Linux OS embeds the pppd and chat programs before dialing up. If the Linux OS does not contain pppd, install kppp, which contains the pppd program. pppd 2.4.5 or pppd 2.4.7 are recommended.

4.3.1 PPPd Scripts

All PPPd scripts and configuration files are stored in the **/etc/ppp** directory.



```
root@support:/etc/ppp/peers# cd ../
root@support:/etc/ppp# ls
chap-secrets        ip-down     ipv6-down     options       pppoe_on_boot
connect-errors      ip-down.d   ipv6-down.d   options.pptp  resolv
gprs-connect-chat   ip-up       ipv6-up       pap-secrets   resolv.conf
gprs-connect-chat~  ip-up.d     ipv6-up.d     peers
root@support:/etc/ppp# 
```

- ip-up: a script that configures settings after a connection is set up.
- ip-down: a script that is used after the connection is disconnected.

The **peers/** directory is used to store custom files. Add the Neoway-pppdial and Neoway-chat-connect folders in this directory.

- Neoway-pppdial: configurations.
- Neoway-chat-connect: AT commands and responses between chat and the module.

4.3.2 PPPd Configuration Files

This section provides example code of the **Neoway-pppdial** and **Neoway-chat-connect** scripts.

Note that you should add ttyUSB based on the actual configurations. In this example, the modem port maps ttyUSB2.

File: **/etc/ppp/peers/Neoway-pppdial**

```
#ttyUSBX mapping modem interface;The China Telecom 2G/3G needs to
#set the username and password to be “ctnet@mycdma.cn” and “vnet.mobi”,
#others,use the default parameters “card” and “card”for username and password.
/dev/ttyUSB2    #modem port of N58
user "card"
password "card"

#The following parameters are recommended to keep the default settings,
#and you can also modify according to your own conditions.
115200
hide-password
noauth
debug
noipdefault
```

```
local
lock
dump
nodetach
remotename cmnet
ipparam cmnet
usepeerdns
connect '/usr/sbin/chat -s -v -f /etc/ppp/peers/gprs-chat'
```

File: **/etc/ppp/peers/Neoway-chat-connect**

```
TIMEOUT 5
ABORT "BUSY"
ABORT "DELAYED"
ABORT "ERROR"
ABORT "NODIALTONE"
ABORT "NOCARRIER"
TIMEOUT 5

''AT
OK AT+CSQ
OK AT+CGDCONT=1,"IP","ctnet"
OK ATD*99#
CONNECT ''
```

Table 4-1 shows the APN information of each carrier in China.

Table 4-1 APN information of carriers

Carrier	Network Modes	APN	Dial-up Number
China Mobile	2G/3G/4G	CMNET	*99#
2G IoT of China Mobile	2G	CMMTM	*99#
4G IoT of China Mobile	4G	CMIOT	*99#
China Unicom	2G/3G/4G	3GNET	*99#
China Telecom	2G/3G	CTNET	*99#
	4G	CTLTE	

For other networks, consult your carriers.

If you use the SIM card with a private network, obtain user and password from carriers when necessary.

4.4 Dial-up Using PPPd Script

1. Issue the following AT commands.

```
AT+CPIN?            //Check if the SIM card is identified
AT+CSQ              //Query the RSSI of current signal.
AT+CGATT?           //Query the attachment status of the data service.
AT$MYSYSINFO        //Query the network mode registered.
```

See the initialization process of N58 in Appendix. For the return values of the commands, see *Neoway_N58_AT_Command_Manual*.
Ensure that the module registers a network.

2. Set the APN, user name, password, authentication correctly.

For details about the commands, see *Neoway_N58_AT_Command_Manual*.
Set PDP context.

```
AT+CGDCONT=1,"IP","CMNET"
OK
```

Query current PDP context.

```
AT+CGDCONT?
+CGDCONT:
1,"IP","CMNET","0.0.0.0",0,0,0,0
OK
```

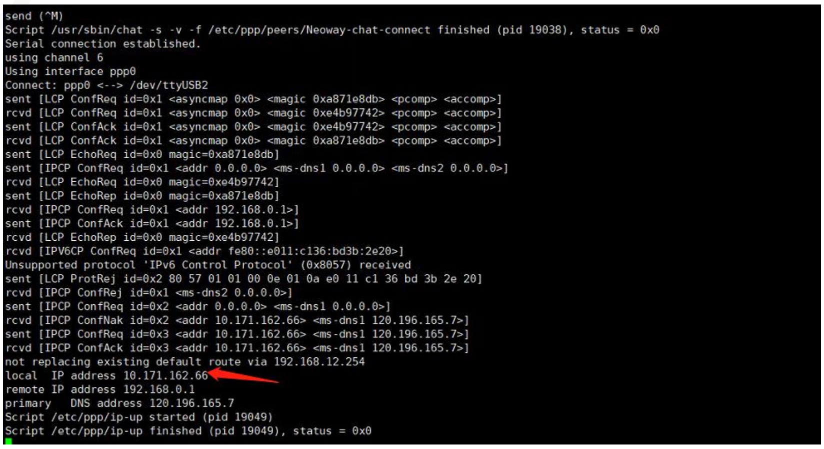
Set authentication.

```
AT+XGAUTH=1,1,”gsm”,”1234”
OK
```

3. Check if APN, user name, and password has been configured correctly in the following scripts.

```
nodetach
user  “gsm”
password “1234”
remotename cmnet
```

4. Execute pppd call Neoway-pppdial.

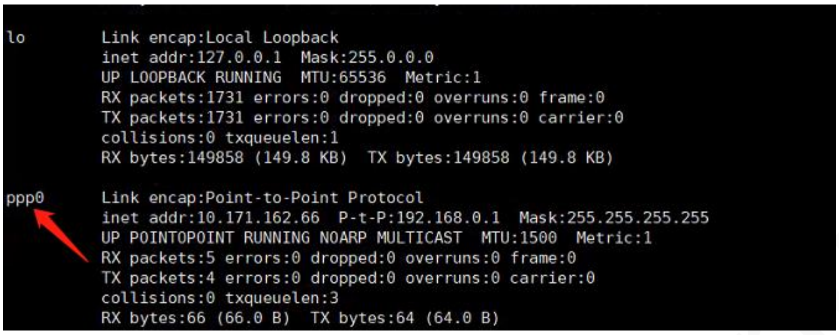


```
send (^M)
Script /usr/sbin/chat -s -v -f /etc/ppp/peers/Neoway-chat-connect finished (pid 19038), status = 0x0
Serial connection established.
using channel 6
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB2
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0xa871e8db> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0xe4b97742> <pcomp> <accomp>]
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0xe4b97742> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0xa871e8db> <pcomp> <accomp>]
sent [LCP EchoReq id=0x0 magic=0xa871e8db]
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [LCP EchoReq id=0x0 magic=0xe4b97742]
sent [LCP EchoRep id=0x0 magic=0xa871e8db]
rcvd [IPCP ConfReq id=0x1 <addr 192.168.0.1>]
sent [IPCP ConfAck id=0x1 <addr 192.168.0.1>]
rcvd [LCP EchoRep id=0x0 magic=0xe4b97742]
rcvd [IPV6CP ConfReq id=0x1 <addr fe80::e011:c136:bd3b:2e20>]
Unsupported protocol 'IPv6 Control Protocol' (0x8057) received
sent [LCP ProtRej id=0x2 80 57 01 01 00 0e 01 0a e0 11 c1 36 bd 3b 2e 20]
rcvd [IPCP ConfRej id=0x1 <ms-dns2 0.0.0.0>]
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0>]
rcvd [IPCP ConfNak id=0x2 <addr 10.171.162.66> <ms-dns1 120.196.165.7>]
sent [IPCP ConfReq id=0x3 <addr 10.171.162.66> <ms-dns1 120.196.165.7>]
rcvd [IPCP ConfAck id=0x3 <addr 10.171.162.66> <ms-dns1 120.196.165.7>]
not replacing existing default route via 192.168.12.254
local  IP address 10.171.162.66
remote IP address 192.168.0.1
primary   DNS address 120.196.165.7
Script /etc/ppp/ip-up started (pid 19049)
Script /etc/ppp/ip-up finished (pid 19049), status = 0x0
```

4.5 Testing and Closing Connection

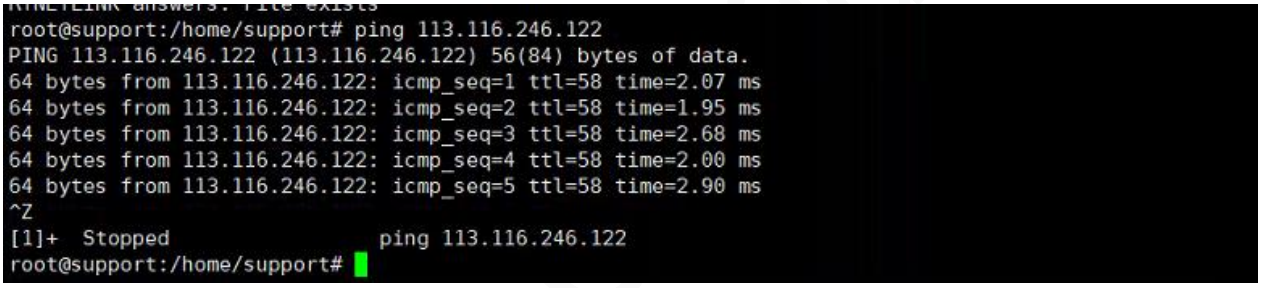
4.5.1 Testing the Connection

Step 1: Input **ifconfig -a** to check the IP address.



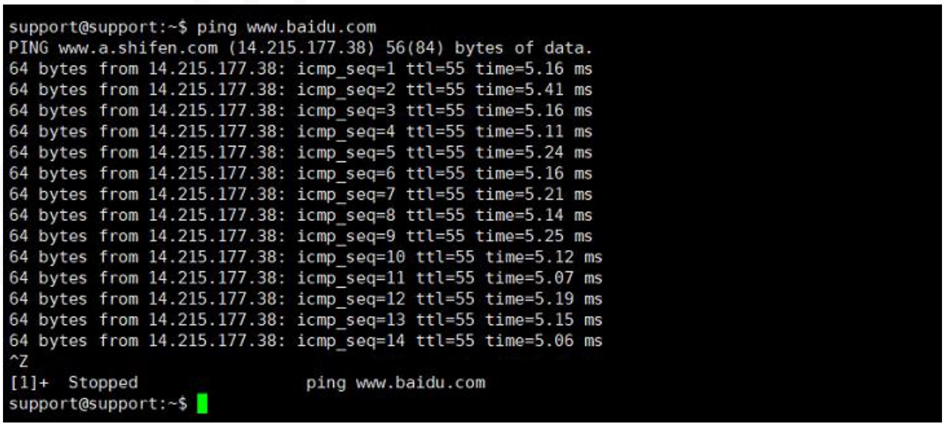
```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1731 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1731 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:149858 (149.8 KB)  TX bytes:149858 (149.8 KB)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.171.162.66  P-t-P:192.168.0.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:66 (66.0 B)  TX bytes:64 (64.0 B)
```

Step 2: Ping a valid IP address to check if the host is connected to the Internet.



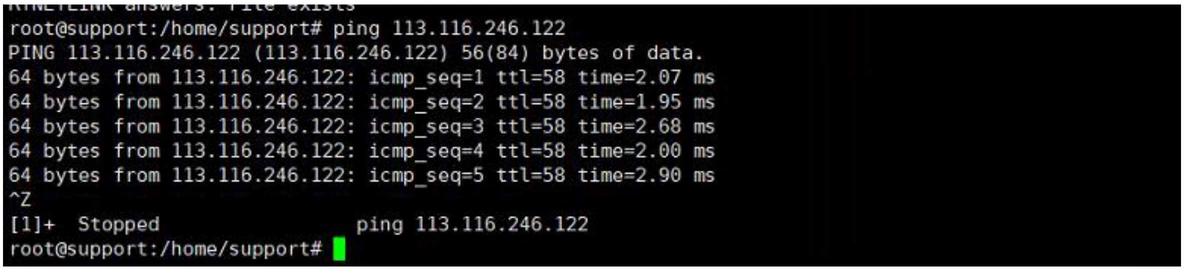
```
RTNETLINK answers: File exists
root@support:/home/support# ping 113.116.246.122
PING 113.116.246.122 (113.116.246.122) 56(84) bytes of data.
64 bytes from 113.116.246.122: icmp_seq=1 ttl=58 time=2.07 ms
64 bytes from 113.116.246.122: icmp_seq=2 ttl=58 time=1.95 ms
64 bytes from 113.116.246.122: icmp_seq=3 ttl=58 time=2.68 ms
64 bytes from 113.116.246.122: icmp_seq=4 ttl=58 time=2.00 ms
64 bytes from 113.116.246.122: icmp_seq=5 ttl=58 time=2.90 ms
^Z
[1]+  Stopped                 ping 113.116.246.122
root@support:/home/support# 
```

Step 3: Ping a valid domain name.



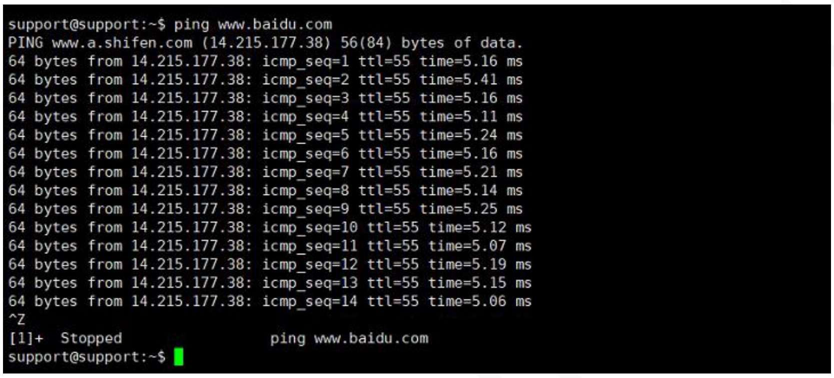
```
support@support:~$ ping www.baidu.com
PING www.a.shifen.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=55 time=5.41 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=55 time=5.11 ms
64 bytes from 14.215.177.38: icmp_seq=5 ttl=55 time=5.24 ms
64 bytes from 14.215.177.38: icmp_seq=6 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=7 ttl=55 time=5.21 ms
64 bytes from 14.215.177.38: icmp_seq=8 ttl=55 time=5.14 ms
64 bytes from 14.215.177.38: icmp_seq=9 ttl=55 time=5.25 ms
64 bytes from 14.215.177.38: icmp_seq=10 ttl=55 time=5.12 ms
64 bytes from 14.215.177.38: icmp_seq=11 ttl=55 time=5.07 ms
64 bytes from 14.215.177.38: icmp_seq=12 ttl=55 time=5.19 ms
64 bytes from 14.215.177.38: icmp_seq=13 ttl=55 time=5.15 ms
64 bytes from 14.215.177.38: icmp_seq=14 ttl=55 time=5.06 ms
^Z
[1]+  Stopped                 ping www.baidu.com
support@support:~$ 
```

- If a website cannot be reached through its IP address, add the default router: **ip route add default dev ppp0**.



```
RTNETLINK answers: File exists
root@support:/home/support# ping 113.116.246.122
PING 113.116.246.122 (113.116.246.122) 56(84) bytes of data.
64 bytes from 113.116.246.122: icmp_seq=1 ttl=58 time=2.07 ms
64 bytes from 113.116.246.122: icmp_seq=2 ttl=58 time=1.95 ms
64 bytes from 113.116.246.122: icmp_seq=3 ttl=58 time=2.68 ms
64 bytes from 113.116.246.122: icmp_seq=4 ttl=58 time=2.00 ms
64 bytes from 113.116.246.122: icmp_seq=5 ttl=58 time=2.90 ms
^Z
[1]+  Stopped                 ping 113.116.246.122
root@support:/home/support# 
```

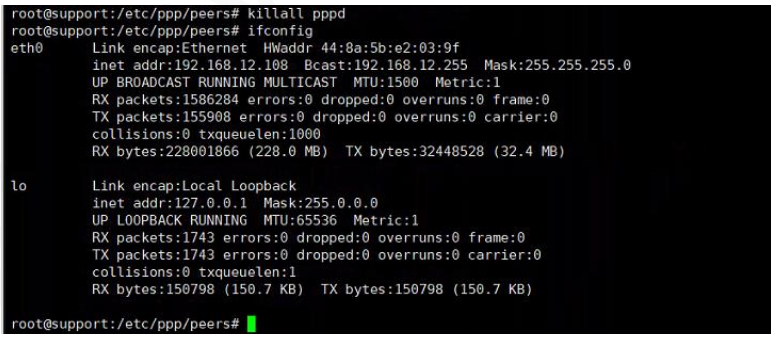
- If a website can be reached through its IP address but cannot be reached through its domain name, configure the DNS manually: system("echo 'nameserver 8.8.8.8' > /etc/resolv.conf").



```
support@support:~$ ping www.baidu.com
PING www.a.shifen.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=55 time=5.41 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=55 time=5.11 ms
64 bytes from 14.215.177.38: icmp_seq=5 ttl=55 time=5.24 ms
64 bytes from 14.215.177.38: icmp_seq=6 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=7 ttl=55 time=5.21 ms
64 bytes from 14.215.177.38: icmp_seq=8 ttl=55 time=5.14 ms
64 bytes from 14.215.177.38: icmp_seq=9 ttl=55 time=5.25 ms
64 bytes from 14.215.177.38: icmp_seq=10 ttl=55 time=5.12 ms
64 bytes from 14.215.177.38: icmp_seq=11 ttl=55 time=5.07 ms
64 bytes from 14.215.177.38: icmp_seq=12 ttl=55 time=5.19 ms
64 bytes from 14.215.177.38: icmp_seq=13 ttl=55 time=5.15 ms
64 bytes from 14.215.177.38: icmp_seq=14 ttl=55 time=5.06 ms
^Z
[1]+  Stopped                 ping www.baidu.com
support@support:~$ 
```

4.5.2 Closing the Connection

You can execute **# killall pppd** to close a PPP dial-up connection.



```
root@support:/etc/ppp/peers# killall pppd
root@support:/etc/ppp/peers# ifconfig
eth0      Link encap:Ethernet  HWaddr 44:8a:5b:e2:03:9f
          inet addr:192.168.12.108  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1586284 errors:0 dropped:0 overruns:0 frame:0
          TX packets:155908 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:228001866 (228.0 MB)  TX bytes:32448528 (32.4 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1743 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1743 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:150798 (150.7 KB)  TX bytes:150798 (150.7 KB)

root@support:/etc/ppp/peers# 
```

4.6 Reconnecting

If the module fails to connect to the network or the connection is closed, stop the PPP dial-up and then dial up to the network again.

5 RNDIS Dial-up

The N58 module with a PID of 0x7401 supports RNDIS dialup.

5.1 Adding VID and PID to Kernel

To use RNDIS dial-up function, filter out RNDIS port.

In **kernel/drivers/usb/serial/option.c**, add VID:0x2949 PID:0x7401 to **option_ids**.

For details, refer to Chapter 2.2.2 By Rebuilding Kernel.

```
static const struct option_blacklist_info neoway_2949_7401_blacklist = {
    .reserved = BIT(1) ,  // filter out port 1
};

static const struct usb_device_id option_ids[] = {
    { USB_DEVICE(0x2949, 0x7401),
         .driver_info = (kernel_ulong_t)&neoway_2949_7401_blacklist },
… …
}
```

5.2 Adding RNDIS Driver to Kernel

Step 1: Navigate to kernel and execute **make menuconfig**.

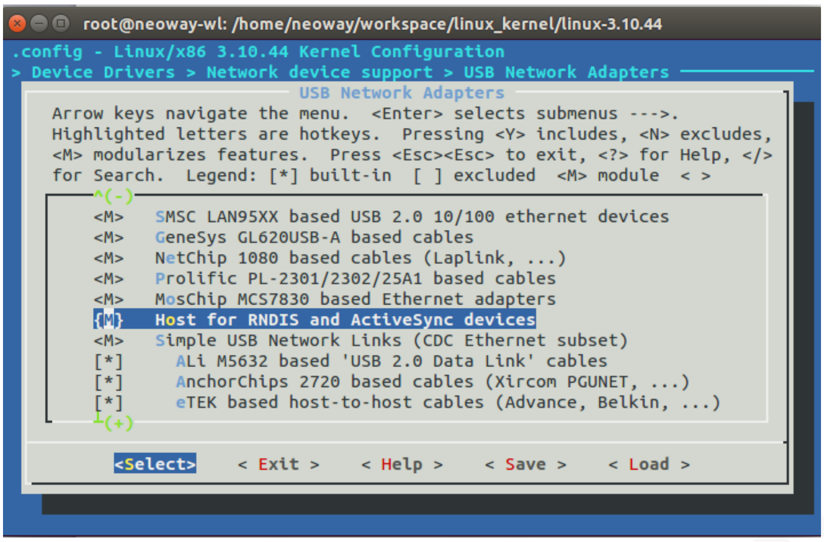
```
Device Drivers  --->
    -*- Network device support  --->
    {M}   USB Network Adapters  --->
        {*}   Multi-purpose USB Networking Framework (CONFIG_USB_NET=m)
{M}   Host for RNDIS and ActiveSync devices
(CONFIG_USB_NET_RNDIS_HOST=m)
```

Step 2: Configure the kernel and select the RNDIS option.

Step 3: Save the configuration and exit.

Step 4: Rebuild kernel and burn it into the host.

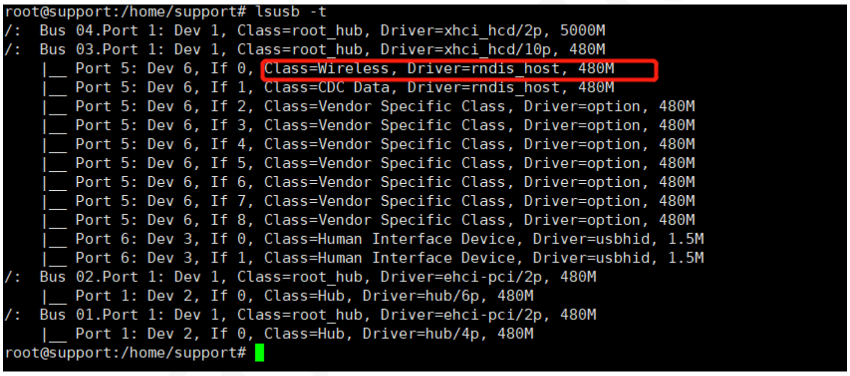
Step 5: Restart the host.



```
root@neoway-wl: /home/neoway/workspace/linux_kernel/linux-3.10.44
.config - Linux/x86 3.10.44 Kernel Configuration
> Device Drivers > Network device support > USB Network Adapters ─────────
                          USB Network Adapters
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
      ^(-)
      <M>   SMSC LAN95XX based USB 2.0 10/100 ethernet devices
      <M>   GeneSys GL620USB-A based cables
      <M>   NetChip 1080 based cables (Laplink, ...)
      <M>   Prolific PL-2301/2302/25A1 based cables
      <M>   MosChip MCS7830 based Ethernet adapters
      {M}   Host for RNDIS and ActiveSync devices
      <M>   Simple USB Network Links (CDC Ethernet subset)
      [*]     ALi M5632 based 'USB 2.0 Data Link' cables
      [*]     AnchorChips 2720 based cables (Xircom PGUNET, ...)
      [*]     eTEK based host-to-host cables (Advance, Belkin, ...)
      └(+)

          <Select>    < Exit >    < Help >    < Save >    < Load >
```

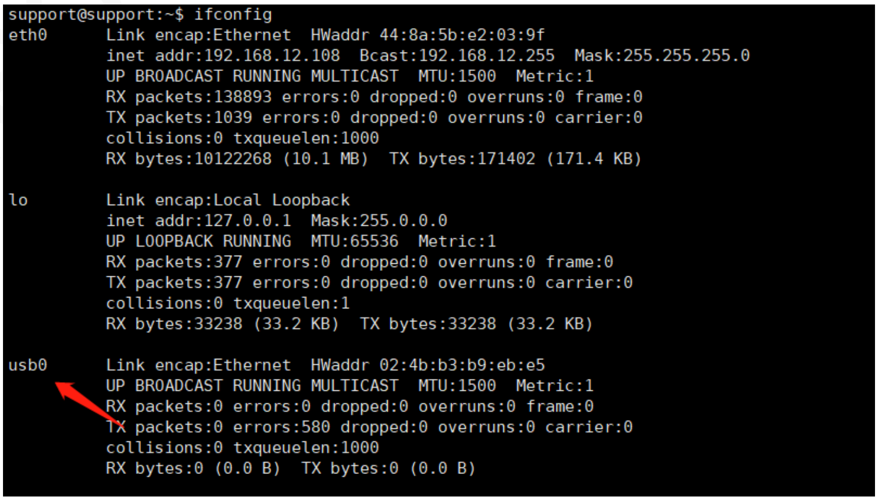
Step 6: Check whether the RNDIS driver is loaded successfully.

- Input **lsusb -t** to check if the device is identified correctly.



```
root@support:/home/support# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/2p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/10p, 480M
    |__ Port 5: Dev 6, If 0, Class=Wireless, Driver=rndis_host, 480M
    |__ Port 5: Dev 6, If 1, Class=CDC Data, Driver=rndis_host, 480M
    |__ Port 5: Dev 6, If 2, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 3, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 4, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 5, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 6, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 7, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 6, If 8, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 6: Dev 3, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
    |__ Port 6: Dev 3, If 1, Class=Human Interface Device, Driver=usbhid, 1.5M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/6p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
root@support:/home/support#
```

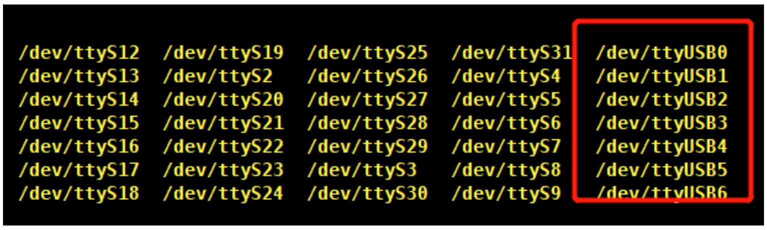
- Input **ifconfig** to check if the RNDIS device is displayed.



```
support@support:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 44:8a:5b:e2:03:9f
          inet addr:192.168.12.108  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:138893 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1039 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10122268 (10.1 MB)  TX bytes:171402 (171.4 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:377 errors:0 dropped:0 overruns:0 frame:0
          TX packets:377 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:33238 (33.2 KB)  TX bytes:33238 (33.2 KB)

usb0      Link encap:Ethernet  HWaddr 02:4b:b3:b9:eb:e5
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:580 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Step 7: Check the ports sequence.

Execute **ls /dev/tty*** to load the device nodes of the USB driver.



```
/dev/ttyS12  /dev/ttyS19  /dev/ttyS25  /dev/ttyS31  /dev/ttyUSB0
/dev/ttyS13  /dev/ttyS2   /dev/ttyS26  /dev/ttyS4   /dev/ttyUSB1
/dev/ttyS14  /dev/ttyS20  /dev/ttyS27  /dev/ttyS5   /dev/ttyUSB2
/dev/ttyS15  /dev/ttyS21  /dev/ttyS28  /dev/ttyS6   /dev/ttyUSB3
/dev/ttyS16  /dev/ttyS22  /dev/ttyS29  /dev/ttyS7   /dev/ttyUSB4
/dev/ttyS17  /dev/ttyS23  /dev/ttyS3   /dev/ttyS8   /dev/ttyUSB5
/dev/ttyS18  /dev/ttyS24  /dev/ttyS30  /dev/ttyS9   /dev/ttyUSB6
```

After RNDIS is loaded as the NIC device, the ports are enumerated starting from modem instead of RNDIS. Table 5-1 shows the new ports mapping of N58 in this situation.

Table 5-1 Ports mapping of N58 (during RNDIS dial-up)

Port Mapping	Function	Device No.
~~RNDIS~~	~~RNDIS NIC port~~	~~ttyUSB0~~
MODEM	Private data service port	ttyUSB0
NMEA	GPS/BD data output port	ttyUSB1
AT	AT Command port	ttyUSB2
Diag	Diagnosis port	ttyUSB3
CPLOG	Kernel CP log output port	ttyUSB4
APLOG	Application log output port	ttyUSB5
OPENCON	Log output for OpenCPU	ttyUSB6

5.3 Dialing Up

For details about the commands, see *Neoway_N58_AT_Command_Manual*.

Step 1: Send the following AT commands to check whether the N58 module is registered to the network. You can refer to the initialization process of N58 in Appendix.

```
AT+CPIN?            //Check if the SIM card is identified
AT+CSQ              //Query the RSSI of current signal.
AT+CGATT?           //Query the attachment status of the data service.
AT$MYSYSINFO        //Query the network mode registered.
```

Step 2: Set the APN, user name, password, authentication correctly

1. Set PDP context.

```
AT+CGDCONT=1,"IP","CMNET"
```

```
OK
```

2. Query current PDP context.

```
AT+CGDCONT?
+CGDCONT:
1,"IP","CMNET","0.0.0.0",0,0,0,0
OK
```

3. Set authentication.

```
AT+XGAUTH=1,1,”gsm”,”1234”
OK
```

Step 3: Send **AT+NETSHAREACT=?** to query the range of the parameters.

```
AT+NETSHAREACT=?
+NETSHAREACT: (0-11),(0-1),(0-1),"apn","user","passwd",(0-3)
```

Step 4: Send **AT+NETSHAREACT?** to query the RNDIS dial-up status.

```
AT+NETSHAREACT?
+NETSHAREACT: 1,0,,,"IPV4",RNDIS         // The connection is established successfully.
AT+NETSHAREACT?
+NETSHAREACT: 0,0,,,"IPV4",RNDIS         // Fail to establish the connection.
```

Step 5: Send **AT+NETSHAREACT=2,1,0** to make a dial-up connection.

```
AT+NETSHAREACT=2,1,0
OK
```

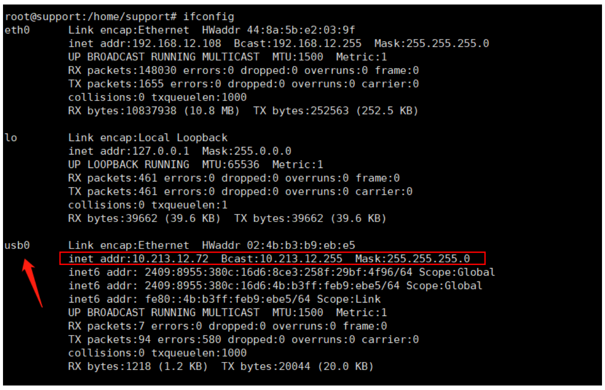
Step 6: Send **AT+NETSHAREACT=2,0,0** to close the connection.

```
AT+NETSHAREACT=2,0,0
OK
```

5.4 Testing and Closing Connection

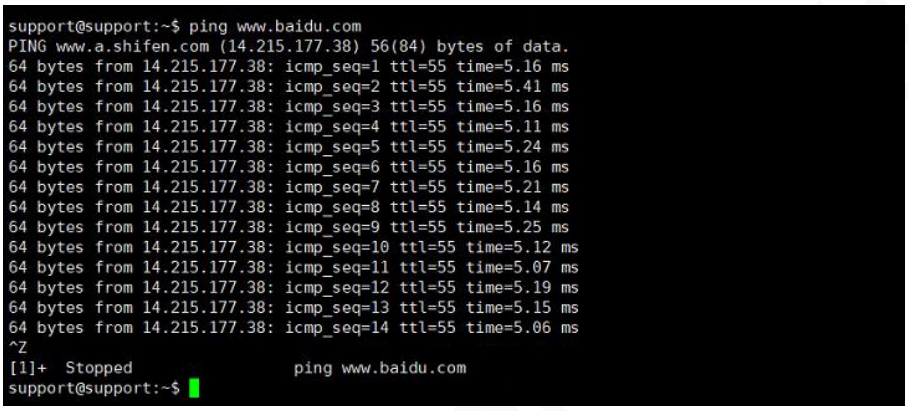
5.4.1 Testing the Connection

1. Input **ifconfig** to check the IP address of usb0.



```
root@support:/home/support# ifconfig
eth0      Link encap:Ethernet  HWaddr 44:8a:5b:e2:03:9f
          inet addr:192.168.12.108  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:148030 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1655 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10837938 (10.8 MB)  TX bytes:252563 (252.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:461 errors:0 dropped:0 overruns:0 frame:0
          TX packets:461 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:39662 (39.6 KB)  TX bytes:39662 (39.6 KB)

usb0      Link encap:Ethernet  HWaddr 02:4b:b3:b9:eb:e5
          inet addr:10.213.12.72  Bcast:10.213.12.255  Mask:255.255.255.0
          inet6 addr: 2409:8955:380c:16d6:8ce3:258f:29bf:4f96/64 Scope:Global
          inet6 addr: 2409:8955:380c:16d6:4b:b3ff:feb9:ebe5/64 Scope:Global
          inet6 addr: fe80::4b:b3ff:feb9:ebe5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:94 errors:580 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1218 (1.2 KB)  TX bytes:20044 (20.0 KB)
```

2. Ping a valid website to check if the host is connected to the Internet.

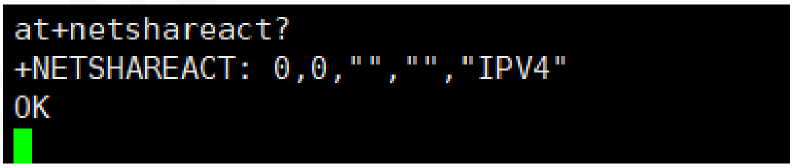


```
support@support:~$ ping www.baidu.com
PING www.a.shifen.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38: icmp_seq=1 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=2 ttl=55 time=5.41 ms
64 bytes from 14.215.177.38: icmp_seq=3 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=4 ttl=55 time=5.11 ms
64 bytes from 14.215.177.38: icmp_seq=5 ttl=55 time=5.24 ms
64 bytes from 14.215.177.38: icmp_seq=6 ttl=55 time=5.16 ms
64 bytes from 14.215.177.38: icmp_seq=7 ttl=55 time=5.21 ms
64 bytes from 14.215.177.38: icmp_seq=8 ttl=55 time=5.14 ms
64 bytes from 14.215.177.38: icmp_seq=9 ttl=55 time=5.25 ms
64 bytes from 14.215.177.38: icmp_seq=10 ttl=55 time=5.12 ms
64 bytes from 14.215.177.38: icmp_seq=11 ttl=55 time=5.07 ms
64 bytes from 14.215.177.38: icmp_seq=12 ttl=55 time=5.19 ms
64 bytes from 14.215.177.38: icmp_seq=13 ttl=55 time=5.15 ms
64 bytes from 14.215.177.38: icmp_seq=14 ttl=55 time=5.06 ms
^Z
[1]+  Stopped                 ping www.baidu.com
support@support:~$
```

For details, refer to section 4.5.1 Testing the Connection.

5.4.2 Closing the Connection

You can send **AT+NETSHAREACT=2,0,0** to close the connection.



```
at+netshareact?
+NETSHAREACT: 0,0,"","","IPV4"
OK
```

5.5 Reconnecting

Send an AT command to check the dial-up connection status. If the connection is closed, dial up to the network again.

```
//No dial-up connection has been set up or the connection is closed.
AT+NETSHAREACT?
+NETSHAREACT: 0,0,,,"IPV4",RNDIS
```

6 ECM Dial-up

The N58 module whose PID is 0x7402 supports ECM dialup.

6.1 Adding VID and PID

To use ECM dial-up function, filter out ECM port.

In **kernel/drivers/usb/serial/option.c**, add VID:0x2949 PID:0x7402 to **option_ids**.

For details, refer to Chapter 2.2.2 By Rebuilding Kernel.

```
static const struct option_blacklist_info neoway_2949_7402_blacklist = {
    .reserved = BIT(0) ,
};

static const struct usb_device_id option_ids[] = {
    { USB_DEVICE(0x2949, 7402),
        .driver_info = (kernel_ulong_t)&neoway_2949_7402_blacklist },
… …
}
```

6.2 Adding ECM Driver to Kernel

Step 1: Navigate to kernel and execute **make menuconfig**.

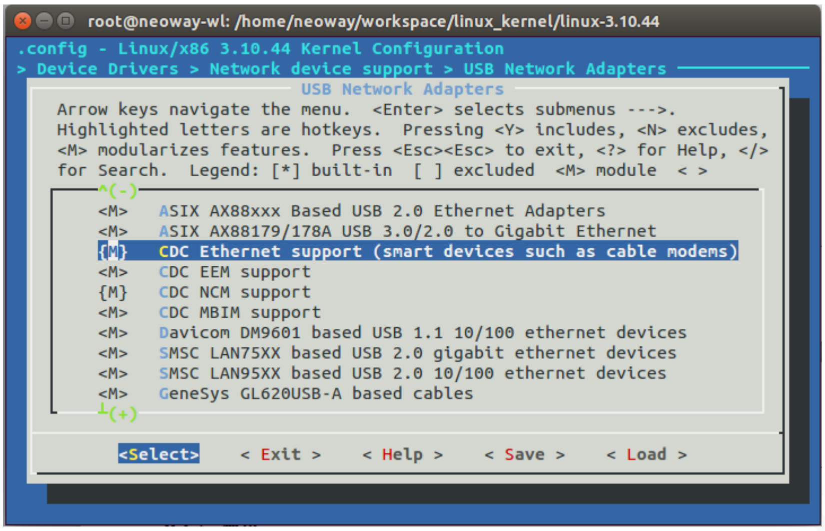
```
Device Drivers  --->
    -*- Network device support  --->
    {M}  USB Network Adapters  --->
         {*}  Multi-purpose USB Networking Framework (CONFIG_USB_NET=m)
            {M}  CDC Ethernet support (smart devices such as cable modems)
                (CONFIG_USB_NET_CDCETHER=m)
```

Step 2: Configure the kernel and select the ECM option.

Step 3: Save the configuration and exit.

Step 4: Rebuild kernel and burn it into the host.

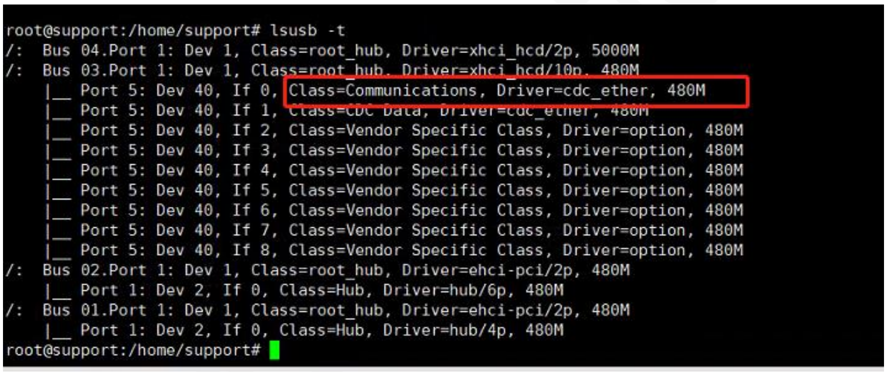
Step 5: Restart the host.



```
root@neoway-wl: /home/neoway/workspace/linux_kernel/linux-3.10.44
.config - Linux/x86 3.10.44 Kernel Configuration
> Device Drivers > Network device support > USB Network Adapters
                          USB Network Adapters
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
      ^(-)
      <M>    ASIX AX88xxx Based USB 2.0 Ethernet Adapters
      <M>    ASIX AX88179/178A USB 3.0/2.0 to Gigabit Ethernet
      {M}    CDC Ethernet support (smart devices such as cable modems)
      <M>    CDC EEM support
      {M}    CDC NCM support
      <M>    CDC MBIM support
      <M>    Davicom DM9601 based USB 1.1 10/100 ethernet devices
      <M>    SMSC LAN75XX based USB 2.0 gigabit ethernet devices
      <M>    SMSC LAN95XX based USB 2.0 10/100 ethernet devices
      <M>    GeneSys GL620USB-A based cables
      ┴(+)

          <Select>    < Exit >    < Help >    < Save >    < Load >
```

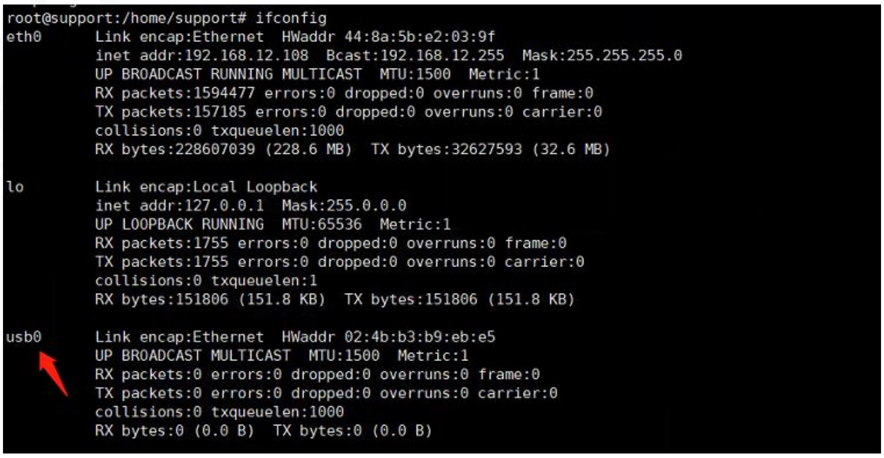
Step 6: Check whether the ECM driver is loaded successfully.

1. Input **lsusb -t** to check if the device is identified correctly.



```
root@support:/home/support# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/2p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/10p, 480M
    |__ Port 5: Dev 40, If 0, Class=Communications, Driver=cdc_ether, 480M
    |__ Port 5: Dev 40, If 1, Class=CDC Data, Driver=cdc_ether, 480M
    |__ Port 5: Dev 40, If 2, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 3, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 4, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 5, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 6, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 7, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 5: Dev 40, If 8, Class=Vendor Specific Class, Driver=option, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/6p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
root@support:/home/support#
```

2. Input **ifconfig** to check if the ECM device is displayed.



```
root@support:/home/support# ifconfig
eth0      Link encap:Ethernet  HWaddr 44:8a:5b:e2:03:9f
          inet addr:192.168.12.108  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1594477 errors:0 dropped:0 overruns:0 frame:0
          TX packets:157185 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:228607039 (228.6 MB)  TX bytes:32627593 (32.6 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1755 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1755 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:151806 (151.8 KB)  TX bytes:151806 (151.8 KB)

usb0      Link encap:Ethernet  HWaddr 02:4b:b3:b9:eb:e5
          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)
```

6.3 Dialing Up

For details about the commands, see *Neoway_N58_AT_Command_Manual*.

Step 1: Send the following AT commands to check whether the N58 module is registered to the network. You can refer to the initialization process of N58 in Appendix.

```
AT+CPIN?          //Check if the SIM card is identified
AT+CSQ            //Query the RSSI of current signal.
AT+CGATT?         //Query the attachment status of the data service.
AT$MYSYSINFO      //Query the network mode registered.
```

Step 2: Set the APN, user name, password, authentication correctly

1. Set PDP context.

```
AT+CGDCONT=1,"IP","CMNET"
OK
```

2. Query current PDP context.

```
AT+CGDCONT?
+CGDCONT:
1,"IP","CMNET","0.0.0.0",0,0,0,0
OK
```

3. Set authentication.

```
AT+XGAUTH=1,1,”gsm”,”1234”
OK
```

Step 3: Send **AT+NETSHAREACT=?** to query the range of the parameters.

```
AT+NETSHAREACT=?
+NETSHAREACT: (0-11),(0-1),(0-1),"apn","user","passwd",(0-3)
```

Step 4: Send **AT+NETSHAREACT?** to query the ECM dial-up status.

```
AT+NETSHAREACT?
+NETSHAREACT: 1,0,,,"IPV4",ECM         // the connection is established successfully.
AT+NETSHAREACT?
+NETSHAREACT: 0,0,,,"IPV4",ECM         // fail to establish the connection.
```

Step 5: Send **AT+NETSHAREACT=2,1,0** to make a dial-up connection.

```
AT+NETSHAREACT=2,1,0
OK
```

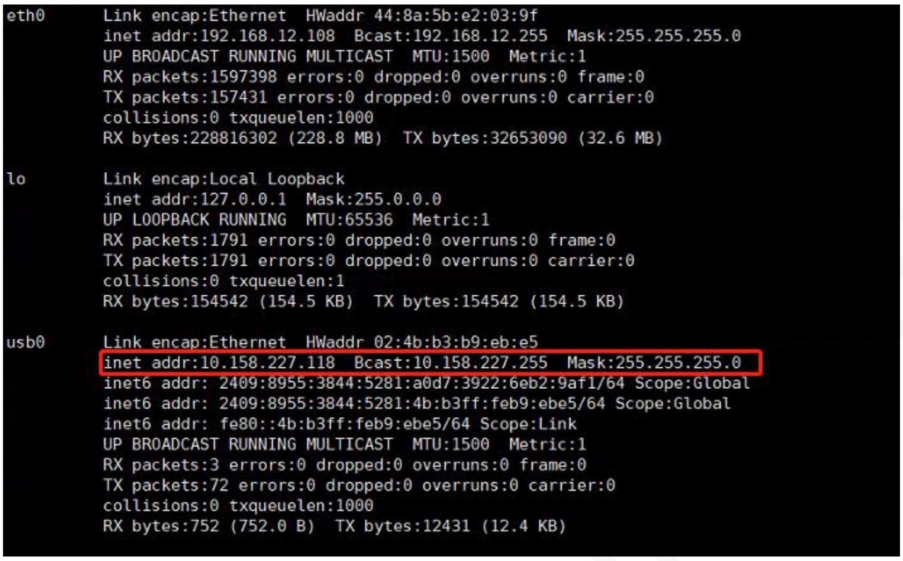
Step 6: Send **AT+NETSHAREACT=2,0,0** to close the connection.

```
AT+NETSHAREACT=2,0,0
OK
```

6.4 Testing and Closing Connection

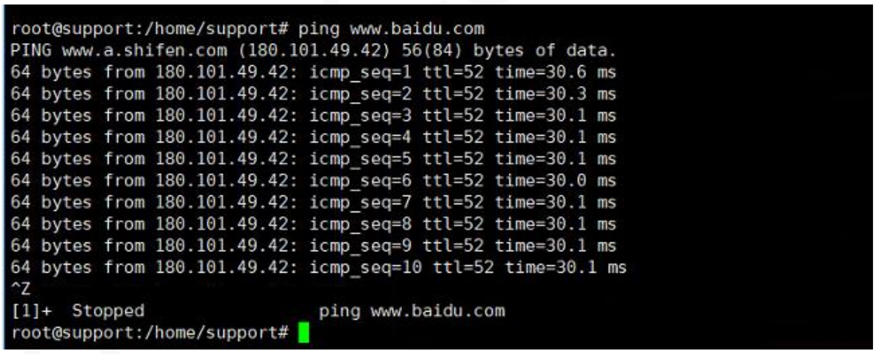
6.4.1 Testing the Connection

1. Input **ifconfig** to check the IP address of usb0.



```
eth0      Link encap:Ethernet  HWaddr 44:8a:5b:e2:03:9f
          inet addr:192.168.12.108  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1597398 errors:0 dropped:0 overruns:0 frame:0
          TX packets:157431 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:228816302 (228.8 MB)  TX bytes:32653090 (32.6 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1791 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1791 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:154542 (154.5 KB)  TX bytes:154542 (154.5 KB)

usb0      Link encap:Ethernet  HWaddr 02:4b:b3:b9:eb:e5
          inet addr:10.158.227.118  Bcast:10.158.227.255  Mask:255.255.255.0
          inet6 addr: 2409:8955:3844:5281:a0d7:3922:6eb2:9af1/64 Scope:Global
          inet6 addr: 2409:8955:3844:5281:4b:b3ff:feb9:ebe5/64 Scope:Global
          inet6 addr: fe80::4b:b3ff:feb9:ebe5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:752 (752.0 B)  TX bytes:12431 (12.4 KB)
```

2. Ping a valid website to check if the host is connected to the Internet.

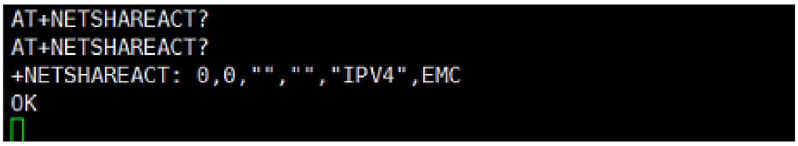


```
root@support:/home/support# ping www.baidu.com
PING www.a.shifen.com (180.101.49.42) 56(84) bytes of data.
64 bytes from 180.101.49.42: icmp_seq=1 ttl=52 time=30.6 ms
64 bytes from 180.101.49.42: icmp_seq=2 ttl=52 time=30.3 ms
64 bytes from 180.101.49.42: icmp_seq=3 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=4 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=5 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=6 ttl=52 time=30.0 ms
64 bytes from 180.101.49.42: icmp_seq=7 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=8 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=9 ttl=52 time=30.1 ms
64 bytes from 180.101.49.42: icmp_seq=10 ttl=52 time=30.1 ms
^Z
[1]+  Stopped                 ping www.baidu.com
root@support:/home/support# 
```

For details, refer to section 4.5.1 Testing the Connection.

6.4.2 Closing the Connection

You can send **AT+NETSHAREACT=2,0,0** to close the connection.



```
AT+NETSHAREACT?
AT+NETSHAREACT?
+NETSHAREACT: 0,0,"","","IPV4",EMC
OK
```

6.5 Reconnecting

Send an AT command to check the dial-up connection status. If the connection is closed, dial up to the network again.

```
    //No dial-up connection has been set up or the connection is closed.
    AT+NETSHAREACT?
    +NETSHAREACT: 0,0,,,"IPV4",ECM
```

7 FAQ

Q: Why cannot I find the USB ports after adding the VID and PID of the module to the kernel and executing **ls /dev/ttyUSB***?

A:

1. Check it the cables are connected properly and correctly.
2. Check if the module is powered up and connected through USB.

   Input **lsusb** or **dmesg** to check the USB device information.
3. Check the VID and PID are added correctly.
4. Check if the modifications are compiled by the system.

You can find the USB ports by inputting **ls /dev/ttyUSB*** if all the above are correct.

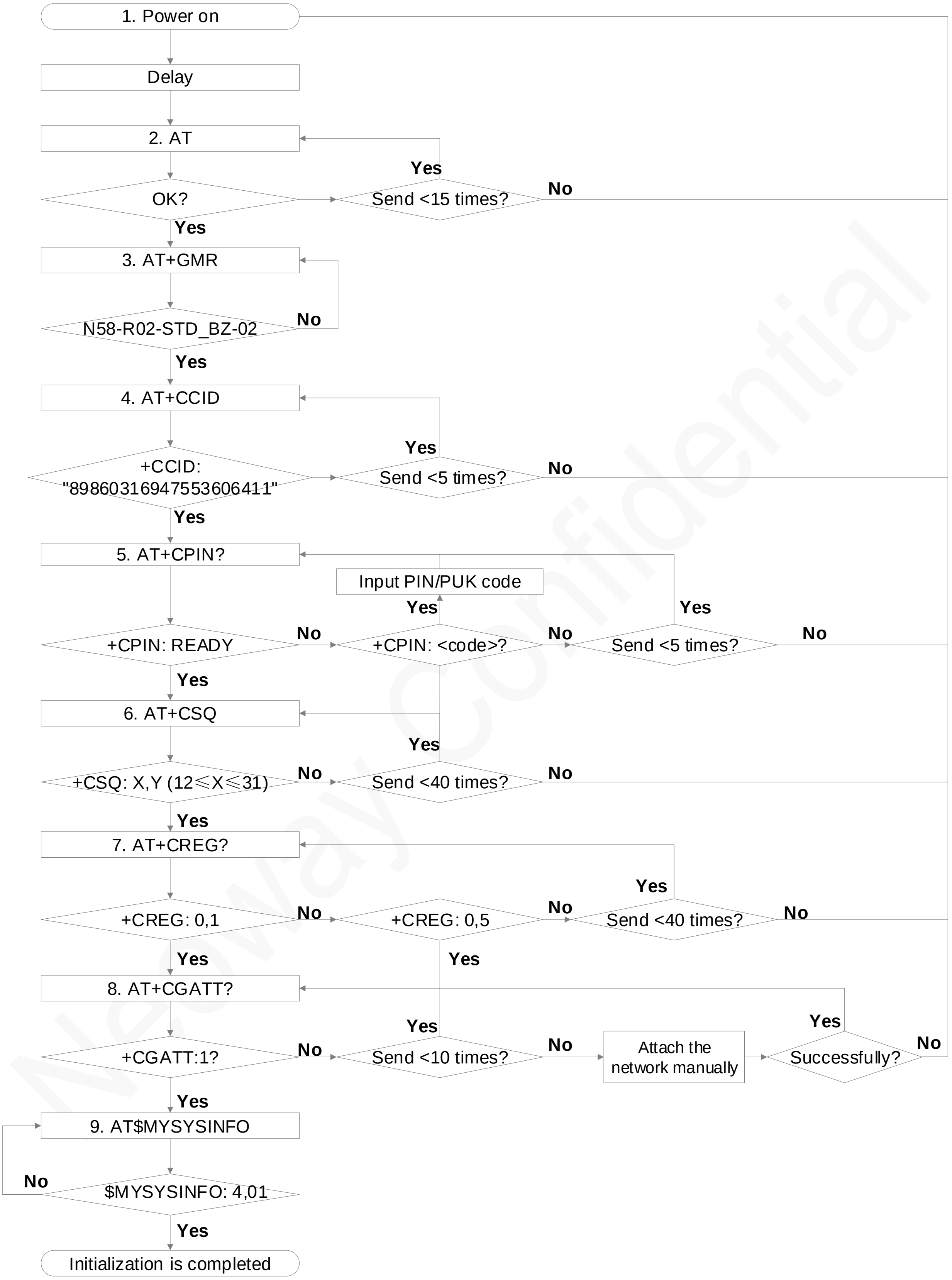
A Appendix

A.1 Initialization Process

Ensure that the module is initialized before setting up a dial-up connection.

```
AT
OK

AT+GMR
+GMR: N58-R02-STD_BZ-02
OK

AT+CCID
+CCID: 898600030450A3163280
OK

AT+CPIN?
+CPIN: READY
OK

AT+CSQ
+CSQ: 25,99
OK

AT+CREG?   // Query network registration status.
+CREG: 0,1
OK

AT+CGATT?  // Query the attach status of data service.
+CGATT: 1
OK

AT$MYSYSINFO   // Query the current network standard.
$MYSYSINFO: 4,01
OK
```

Figure A-1 Initialization process



Wait for 7 seconds after the module is powered on to send AT for baud rate automatic detection.

If UART does not return code after 15 times, check if the baud rate and UART communication settings are correct and if the module is powered on. If everything is OK, restart the module.

ICCID of the SIM card is used to determine whether the module identifies the SIM card and determines the carrier information based on the country code and carrier code.

If **+CPIN: ERROR**, **+CME ERROR**, or **ERROR** is returned to **AT+CPIN?**, the module does not identify the SIM card. You shall restart the module after ensuring the card is valid.

if the PIN code is inputted incorrectly for three times, the PUK code is requested. If the PUK code is inputted incorrectly for 10 times, the SIM card will be locked permanently and unable to be unlocked.

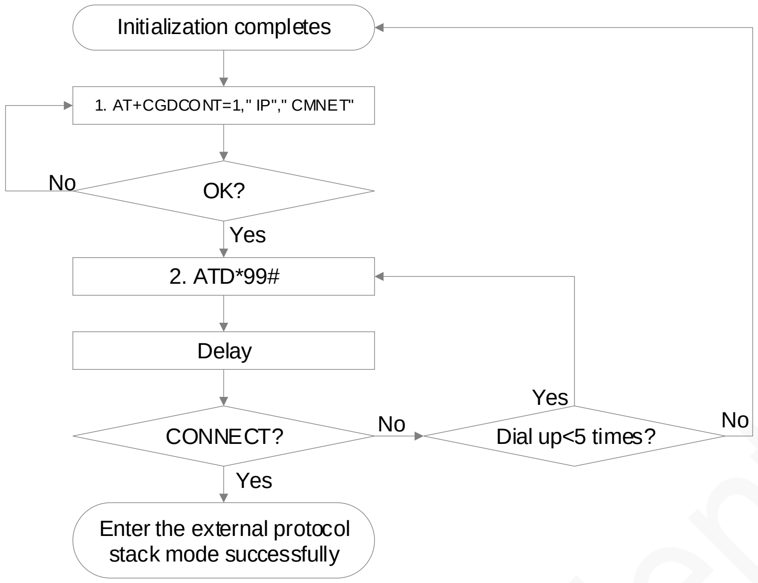
If the module returns **+CSQ:99,99** to **AT+CSQ**, it does not receive any network signal. If the module does not receive any network signal after querying RSSI for 40 times, check the antenna connection, change the location, and restart the module. The return value is in the format of +CSQ: X,Y, wherein, Y indicates the error rate and its value ranges from 0 to 7. The greater the value is, the higher the error rate is. It is valid only for voice calls.

If the module does not return **+CGATT:1** to AT+CGATT? for 10 times, attach the network manually: send **AT+CGATT=0** to de-attach the network; after the module returns OK, send **AT+CGATT=1** to attach the network.

A.2 External Protocol Process

```
AT+CGDCONT=1,"IP","CMNET"
OK
AT+XGAUTH=1,1,"GSM","1234"       Add this command if authentication is required.
OK                               The default user name and password for a China Mobile card are
                                 GSM and 1234; that for a China Telecom or China Unicom card are
                                 CARD and CARD.
                                 If the user name and password are specified, set them to the
                                 specified one.
ATD*99#
CONNECT
```

Figure A-2 External protocol process


Initialization completes
1. AT+CGDCONT=1," IP"," CMNET"
No
OK?
Yes
2. ATD*99#
Delay
Yes
No
No
CONNECT?
Dial up<5 times?
Yes
Enter the external protocol stack mode successfully

If a SIM card with a private network IP address is used and authentication is required, send the following commands: **AT+CGDCONT=1," IP"," CMNET"** and then **AT+XGAUTH=1,1,"GSM","1234"**.

The dial-up command times out if the module does not return any value within 10 seconds.

If dialup fails for 5 times, initialize the module again. Before initialization, check whether the USIM card is valid and whether the module registers to the network successfully.

A.3 Troubleshooting

If the module fails to dial-up to the network, perform the following operations to troubleshoot the issue:

1. Check if the module is initialized and if it registers to a network.
2. Check if the SIM card used is out of service.
3. Check if the APN, user name and password, and authentication are set correctly for the SIM card (note the case difference).
4. Check if the APN, user name and password, and authentication are set correctly in dial-up commands or programs (note the case difference).