Ubuntu Development Guide

Initiated by	suchao.wang	**Job Title**		**Release Date**	2024/08/27
Reviewed by		**Job Title**		**Revision**	V1.0
Approved by		**Job Title**		**Release Status**	Release

Revision History

Date	Revision	Description	Creator
2024-8-27	V1.0	First release	suchao.wang

Directory

1 Introduction

1.1 Instructions

This document mainly display in the intelligent network closed basic operation command, to facilitate customers to quickly understand the basic operations of a gateway. The following is a basic introduction the function of each section.

1.2 System features

System software is divided into three parts, respectively for the Bootloader (Uboot), the Linux kernel and rootfs (busybox). UBoot mainly to guide the kernel boot, supporting the NFS mount, NAND Flash start; linux kernel is the bottom layer of the entire operating system, responsible for the entire hardware driver, as well as providing a variety of system required core functions; Rootfs is a collection of system files.

2 Linux system is introduced

2.1 Directory Notes

/ dev device node directory
/ media multimedia directory
The/proc directory system configuration
/ sys system configuration directory
/var temporary directory
/ bin ordinary user command directory
Configuration file is/etc directory
Dynamic library/lib directory
With/media/MNT directory
/ sbin root user commands
Temporary directory/TMP
/ www Web directory
/home user data directory

/lost+found Delete files temporary storage directory
/opt config directory
/srv cgi command directory
/usr normal user directory

3 Serial port and network configuration

3.1 Serial port

The hardware serial port of this product is unified named ttyAP0,ttyAP1, etc., according to the different type of equipment, the number of serial ports is not the same, use ls /dev/ttyAP* to view the current serial port information on this device. Because some serial ports support RS232/RS485 and other modes, please parameterize the hardware manual before use, and set the jumper to the corresponding position.

ls /dev/ttyAP*
/dev/ttyAP0 /dev/ttyAP1 /dev/ttyAP2 /dev/ttyAP3 /dev/ttyAP4

3.1.1 Simple test if the serial port is working

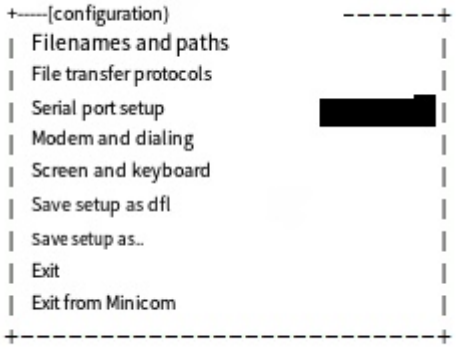
Connect the hardware com1 to com2. Make sure that both com1 and com2 are in RS232 mode or RS485 mode.
Start the first ssh. Type it at the command line
#cat /dev/ttyAP0

Start the second ssh. Type it at the command line
#echo “abcd” > /dev/ttyAP1

At this point in the first window you should see "abcd".

3.1.2 minicom operation command

#minicom -s
Through the up and down arrow keys, select the "Serial port setup" to enter configuration interface.



Enter in the configuration screen

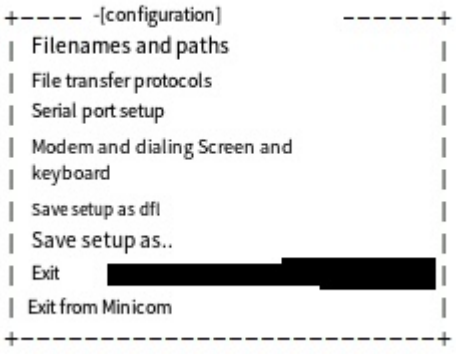
Press A to access the serial port Name configuration options. End of the keyboard editing and press enter.

Press E to configure baud rate parameters.

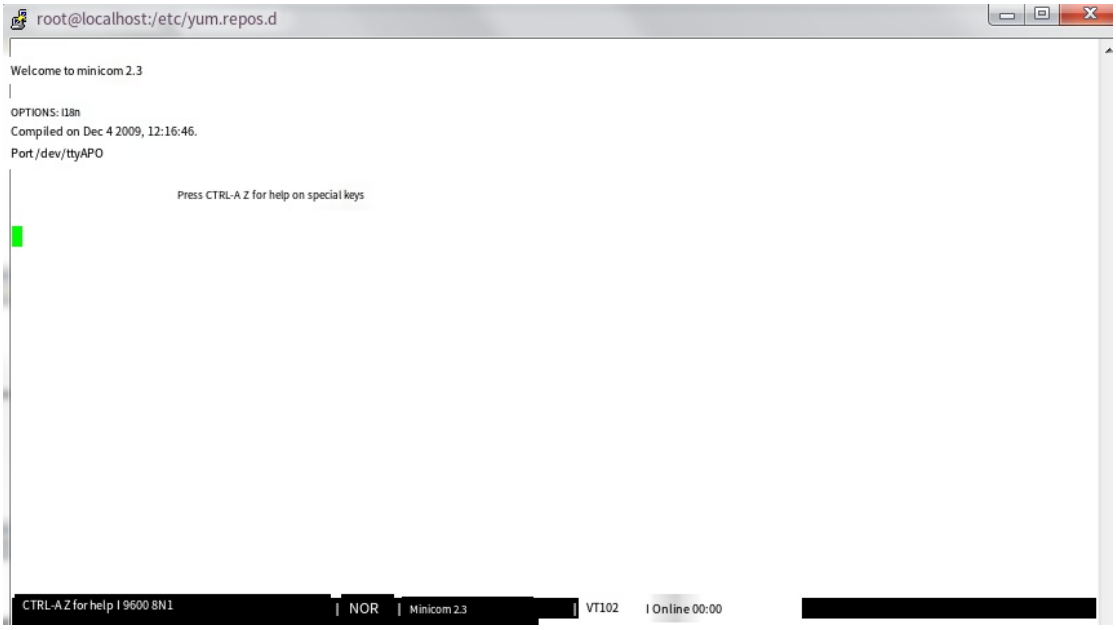
Press the F key opening or closing the hardware flow control.

Press G open or close the software flow control.

Press ESC to exit the configuration screen.



Configuration is completed, select "Exit" button, the end of the configuration, into the communication mode.



3.2 Fixed network card IP configuration

3.2.1 View the current network information

root@xxxx:~# **ifconfig -a**

eth0 Link encap:Ethernet HWaddr 54:4A:16:8F:71:98
inet addr:192.168.0.253 Bcast:0.0.0.0 Mask:255.255.255.0
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 (0.0B) TX bytes:0 (0.0B)
          Interrupt:56

eth1      Link encap:Ethernet  HWaddr 54:4A:16:8F:71:9A
          inet addr:172.21.67.37 Bcast:0.0.0.0 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:102657 errors:0 dropped:3992 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes: (MiB) 14.4 TX 15166631 bytes: 5614 KiB (5.4)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  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:0
          RX bytes:0 (0.0B) TX bytes:0 (0.0B)
```

3.2.2 Temporarily configure the network card IP

```
root@xxxx:~# ifconfig eth0 192.168.1.252 netmask 255.255.255.0
root@xxxx:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 54:4A:16:8F:71:98
          Inet addr: 192.168.1.252 Bcast: 192.168.1.255 Mask: 255.255.255.0
          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 (0.0B) TX bytes:0 (0.0B)
          Interrupt:56
```

3.2.3 Temporary use of DHCP to obtain network card IP

```
root@xxxx:~# udhcpc  -i eth1
udhcpc (v1.22.1) started
Sending discover...
Sending the select for 172.21.67.37...
Lease of 172.21.67.37 obtained, lease time 1800
```

The/etc/udhcpc. D / 50 default: Adding DNS 172.21.66.40
The/etc/udhcpc. D / 50 default: Adding DNS 172.21.66.83
root@xxxx:~# **ifconfig eth1**
eth1 Link encap:Ethernet HWaddr 54:4A:16:8F:71:9A
inet addr:172.21.67.37 Bcast:0.0.0.0 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:113053 errors:0 dropped:4449 overruns:0 frame:0
TX packets:33 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:16752245 (15.9 MiB) TX bytes:6700 (6.5 KiB)

3.2.4 Persistent configuration fixed IP

Change eth0 to a static IP address, and modify the related network card name file in the directory /etc/systemd/network/10-eth.network
vi /etc/systemd/network/10-eth.network

Configure eth0 to static IP:
[Match]
Name=eth0
KernelCommandLine=! root=/dev/nfs

[Network]
Address = 192.168.1.9/24
Gateway = 192.168.1.1

3.2.5 Persistence config dynamic IP

Change eth0 to a static IP address, and modify the related network card name file in the directory /etc/systemd/network/10-eth.network
#vi /etc/systemd/network/10-eth.network

[Match]
Name=eth0
KernelCommandLine=! root=/dev/nfs

[Network]
DHCP=yes

3.3 WIFI configuration

3.3.1 Check to see if the wireless card has been correctly identified

The interface name of the wireless network card is generally wlan0

root@xxxx:~# **ifconfig -a**

```
eth0      Link encap:Ethernet  HWaddr 54:4A:16:8F:71:98
          inet addr:192.168.0.253 Bcast:0.0.0.0 Mask:255.255.255.0
          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 (0.0B) TX bytes:0 (0.0B)
          Interrupt:56

eth1      Link encap:Ethernet  HWaddr 54:4A:16:8F:71:9A
          inet addr:172.21.67.37 Bcast:0.0.0.0 Mask:255.255.255.0
          inet6 addr: fe80::564a:16ff:fe8f:719a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:207 errors:0 dropped:11 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32497 (31.7 KiB) TX bytes:1332 (1.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  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:0
          RX bytes:0 (0.0B) TX bytes:0 (0.0B)

wlan0     Link encap:Ethernet  HWaddr 00:0E:8E:6C:16:B3
          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 (0.0B) TX bytes:0 (0.0B)
```

3.3.2 Enable the wireless network card

```
root@xxxx:~# ifconfig wlan0
wlan0        Link encap:Ethernet    HWaddr 00:0E:8E:6C:16:B3
             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 (0.0B) TX bytes:0 (0.0B)

root@xxxx:~# ifconfig wlan0 up
root@xxxx:~#    ifconfig wlan0
wlan0        Link encap:Ethernet    HWaddr 00:0E:8E:6C:16:B3
             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 (0.0B) TX bytes:0 (0.0B)
```

3.3.3 Scan for wireless networks that are searchable

```
root@xxxx:~# iwlist wlan0 scan
wlan0        Scan completed :
             Cell 01 - Address: 1C:AF:F7:C0:3D:E1
                       Channel:1
                       Frequency:2.412 GHz (Channel 1)
                       Quality=43/70    Signal level=-67 dBm
                       Encryption key:off
                       ESSID:"MOTT"
                       Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
                                 24 Mb/s; 36 Mb/s; 54 Mb/s
                       Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 48 Mb/s
                       Mode:Master
                       Extra:tsf=000000049fe1d8eb
                       Extra: Last beacon: 60ms ago
                       IE: Unknown: 00044D4F5454
                       IE: Unknown: 010882848B962430486C
                       IE: Unknown: 030101
                       IE: Unknown: 2A0100
                       IE: Unknown: 2F0100
                       IE: Unknown: 32040C121860
```

IE: Unknown:
2D1A6C181BFF0000000000000000000000000000000000000000000000
IE: Unknown:
3D160100130000000000000000000000000000000000000000
IE: Unknown: DD090010180201F0010000
IE: Unknown:
DD1E00904C336C181BFF0000000000000000000000000000000000000000000000
IE: Unknown:
DD1A00904C34010013000000000000000000000000000000000000
Cell 02 - Address: C8:3A:35:05:3E:80
Channel:2
Frequency:2.417 GHz (Channel 2)
Quality=63/70 Signal level=-47 dBm
Encryption key:on
ESSID:"WebAccess"
Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
24 Mb/s; 36 Mb/s; 54 Mb/s
Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 48 Mb/s
Mode:Master
Extra:tsf=0000000000a986e3
Extra: Last beacon: 60ms ago
IE: Unknown: 0009576562416363657373
IE: Unknown: 010882840B162430486C
IE: Unknown: 030102
IE: Unknown: 2A0100
IE: Unknown: 2F0100
IE: IEEE 802.11i/WPA2 Version 1
Group Cipher : CCMP
Pairwise Ciphers (1) : CCMP
Authentication Suites (1) : PSK
IE: Unknown: 32040C121860
IE: Unknown:
2D1AFE181BFFFF000001000000000000000000000000000000000000
IE: Unknown:
3D16020D0000000000000000000000000000000000000000000000
IE: Unknown: DD090010180205F02C0000
IE: WPA Version 1
Group Cipher : CCMP
Pairwise Ciphers (1) : CCMP
Authentication Suites (1) : PSK
IE: Unknown:
DD180050F2020101000003A4000027A4000042435E0062322F00

If you want to see if a specific SSID has been searched, you can use the grep command to query the results.

root@xxxx:~# **iwlist wlan0 scan | grep WebAccess**

3.3.4 Modify the configuration file (WPA2 mode)

root@xxxx:~# **vi /etc/wpa_supplicant.conf**

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

# Only WPA-PSK is used. Any valid cipher combination is accepted.
network={
        ssid="WebAccess"
        scan_ssid=1
        proto=WPA2 WPA
        key_mgmt=WPA-PSK
        pairwise=CCMP TKIP
        group=CCMP TKIP WEP104 WEP40
        psk="password"
        priority=2
}
```

3.3.5 Modify the configuration file (Open mode)

root@xxxx:~# **vi /etc/wpa_supplicant.conf**

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

# Only WPA-PSK is used. Any valid cipher combination is accepted.
network={
        ssid="Advantech"
        scan_ssid=1
        key_mgmt=NONE
        priority=1
}
```

3.3.6 Connect to AP

root@xxxx:~# **wlan.sh up**
Successfully initialized wpa_supplicant
OK
udhcpc (v1.22.1) started
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending select for 192.168.10.36...
Lease of 192.168.10.36 obtained, lease time 86400
RTNETLINK answers: File exists
/etc/udhcpc.d/50default: Adding DNS 192.168.10.1
/etc/udhcpc.d/50default: Adding DNS 0.0.0.0
OK

root@xxxx:~# **iwconfig wlan0**
wlan0 IEEE 802.11bgn ESSID:"WebAccess"
Mode:Managed Frequency:2.417 GHz Access Point: C8:3A:35:05:3E:80
Bit Rate=1 Mb/s Tx-Power=20 dBm
Retry long limit:7 RTS thr:off Fragment thr:off
Encryption key:off
Power Management:off
Link Quality=55/70 Signal level=-55 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:1 Invalid misc:8 Missed beacon:0

3.3.7 Save SSID password encryptively

wpa_passphrase command parameters are user and password.
root@xxxx:~#**wpa_passphrase max 1234567890**
network={
ssid="max"
#psk="1234567890"
psk=4b2bc7cbb3710e9ea43f09e8d57e8bdb3b2a2127af44960d73216c3612f6baba
}
Copy psk= encryption password to wpa_supplicant.conf.
The final file looks like this:

```
network={
                ssid="max" // Fill in the username of the wireless network
                key_mgmt=WPA-PSK
                proto=WPA
                pairwise=TKIP
                group=TKIP
psk=4b2bc7cbb3710e9ea43f09e8d57e8bdb3b2a2127af44960d73216c3612f6baba
}
```

3.4 Cellular device Use

3.4.1 Query modemscan for module information

When using this program, please ensure that the module serial port is not in use, so that you can correctly identify whether each serial port can return the AT command.

```
# modemscan
Current con tty: /dev/pts/0
Current cmd tty: ttyO0
==========tty used info==========
1347      /bin/tinylogin   /dev/ttyO0
1347      /bin/tinylogin   /dev/ttyO0
1347      /bin/tinylogin   /dev/ttyO0
1348      /bin/tinylogin   /dev/tty1
1348      /bin/tinylogin   /dev/tty1
1348      /bin/tinylogin   /dev/tty1
 1347 root         0:00 /sbin/getty 115200 ttyO0
 1348 root         0:00 /sbin/getty 38400 tty1
 1412 root         0:00 sh -c ps aux | grep tty
 1414 root         0:00 grep tty
----------tty used info----------

Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 0 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 1 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1. 2 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 3 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 4 /] driver: qmi_wwan
set_operation_quectel
quectel_pre_init,vendor=2c7c,proecut=0125
Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 0 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 1 /] driver: option
Scan_interface: [/ sys/bus/usb/devices / / 2-1. 2 /] driver: option
```

Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 3 /] driver: option

Scan_interface: [/ sys/bus/usb/devices / / 2-1 to 1. 4 /] driver: qmi_wwan

set_operation_quectel

quectel_pre_init,vendor=2c7c,proecut=0125

========USB Driver Info========

[0]type=1, driver=option, node=/dev/ttyUSB0

[1]type=1, driver=option, node=/dev/ttyUSB1

[2]type=4, driver=option, node=/dev/ttyUSB2

[3]type=4, driver=option, node=/dev/ttyUSB3

[4]type=2, driver=qmi_wwan, node=wwan0

========USB Interface device Info========

QMI count=1,interface=4

QMI interface=wwan0

PPP count=2,

ppp interface[0]=/dev/ttyUSB2

ppp interface[1]=/dev/ttyUSB3

[gps status:0]

[gps status:1]

[0]type=1, driver=option, node=/dev/ttyUSB0

[1]type=1, driver=option, node=/dev/ttyUSB1

[2]type=4, driver=option, node=/dev/ttyUSB2

[3]type=4, driver=option, node=/dev/ttyUSB3

[4]type=2, driver=qmi_wwan, node=wwan0

========Cellular device Info========

find modem usbid [2c7c:0125]

[0] [/dev/ttyUSB0] tty port

[1] [/dev/ttyUSB1] tty port

[2] [/dev/ttyUSB2] at port

[3] [/dev/ttyUSB3] at port

[4] [wwan0] qmi net

Modem info:

[version: Quectel EC20F Revision: EC20CEFAR02A04M4G]

[modem imei=862815030700775]

[sim status: READY]

[imsi :460115864165295]

[csq= 30,ber=99]

[operator: mode= 0,format=2,oper="46011",act=7]

[operator: mode= 0,format=0,oper="CHN-CT",act=7]

========XML device Info========

X7d728 get_modem_version: / dev/ttyUSB2, 0

Please copy the xml data to /home/sysuser/project/CellularDeviceInfo.acr or /home/root/project/CellularDeviceInfo.acr

<Device deviceID="2c7c:0125" GPSType="embedded" GPSInterface="1" ATPortCount="2" ATPortInterface="2,3" DialType="ppp" deviceName=" Quectel EC20F Revision: EC20CEFAR02A04M4G QMI(wwan0) " />

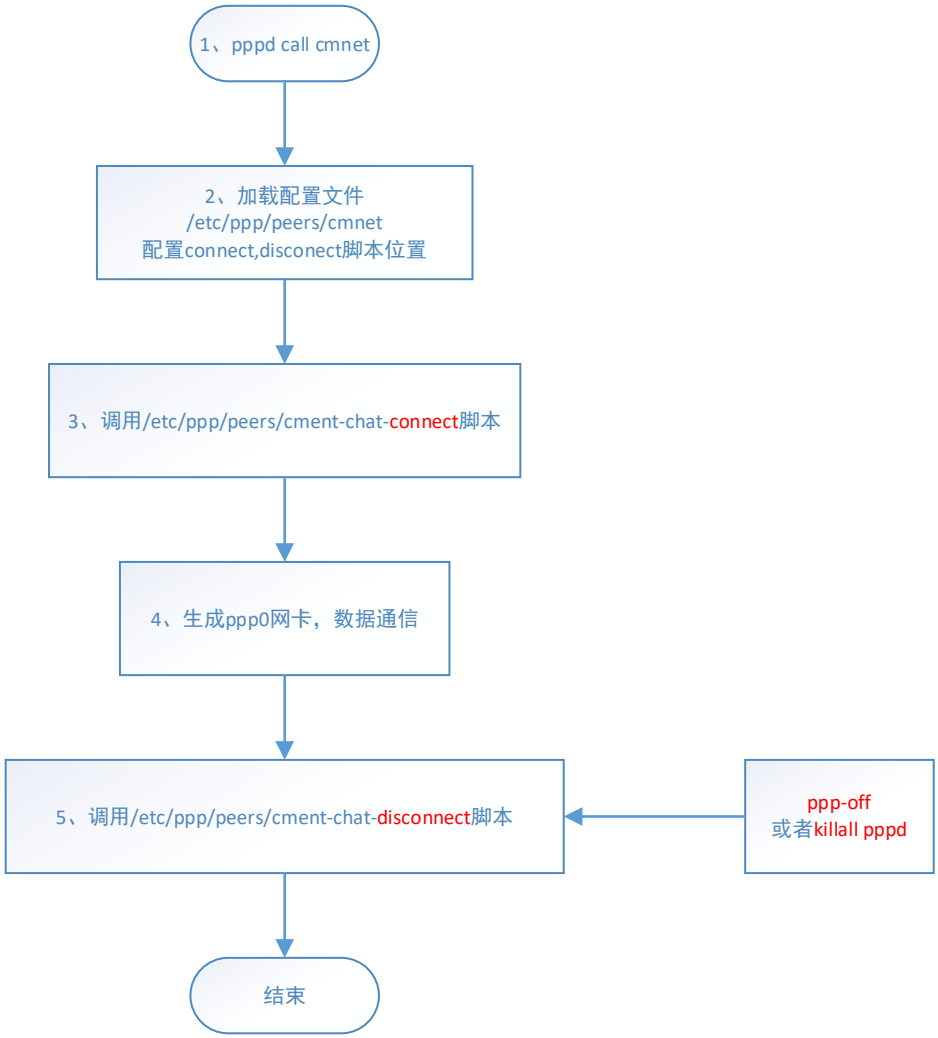
xml file format content parsing see 3.5.3 configuration file CellularDeviceInfo.acr instructions3.5.3Profile CellularDeviceInfo.acr instructions

3.4.2 PPP Dialing

The pppd program of this platform is used for dialing.

3.4.2.1pppd dialing process

PPPD dialing process is as follows:



1、 Invoke pppd from the command line call cmnet/dev/ttyUSB2

2、 PPPD will load/etc/PPP/peers/cmnet configuration parameters, main initialization/dev/ttyXXX serial port name, the connect script, disconnect the script.

3、 Call/etc/PPP/peers/cmnet - chat - connect script, to dial out.

4、 Generate ppp0 network card for communication

5、 Call the /etc/ppp/peers/cmnet-chat-disconnect script to disconnect.

According to the previous process, we can according to your own need to modify the connect script, inside set command or parameters you need.

3.4.2.2Check the communication module

Confirm that the module has been installed correctly and confirm the serial port number that can be used according to the hardware manual. You can also use the modemscan command to search for available modems first.

lsusb

Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Bus 002 Device 002: **ID 2c7c:0125 Quectel Wireless Solutions Co., Ltd. EC25 LTE modem**

001: Bus 002 Device ID 1 d6b using the color picker: 0002 Linux Foundation 2.0 root hub

root@xxxx:~# **dmesg | grep tty**

[0.000000] Kernel command line: console=ttyO0,115200n8 root=/dev/mmcblk0p2 rorootfstype =ext3 rootwait ip=none

[1.553842] serial8250.0: ttyS0 at MMIO 0 x1000000 (irq = 161, base_baud = 921600) is a XR16850

[1.554793] serial8250.0: ttyS1 at MMIO 0 x1000801 (irq = 160, base_baud = 921600) is a XR16850

[1.555659] serial8250.0: ttyS2 at MMIO 0 x1001201 (irq = 250, base_baud = 921600) is a XR16850

[1.556892] 44e09000.serial: ttyO0 at MMIO 0x44e09000 (irq = 88, base_baud = 3000000) is a OMAP UART0

[2.233388] console [ttyO0] enabled

[2.238325] 48022000.serial: ttyO1 at MMIO 0x48022000 (irq = 89, base_baud = 3000000) is a OMAP UART1

[2.249277] 481 a6000. Serial: ttyO3 at MMIO 0 x481a6000 (irq = 60, base_baud = 3000000) is a OMAP UART3

[2.536298] userial_init: registered 4 ttyGS * devices

2-1: [25.741860] usb GSM modem (1 - port) converter now attached to ttyUSB0

2-1: [25.757201] usb GSM modem (1 - port) converter now attached to ttyUSB1

[25.772418] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB2

[25.787615] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB3

2-1: [25.802911] usb GSM modem (1 - port) converter now attached to ttyUSB4

2-1: [25.818109] usb GSM modem (1 - port) converter now attached to ttyUSB5

[25.833328] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB6

[25.848539] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB7

[25.863781] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB8

2-1: [25.879082] usb GSM modem (1 - port) converter now attached to ttyUSB9

[25.894403] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB10

[25.909913] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB11

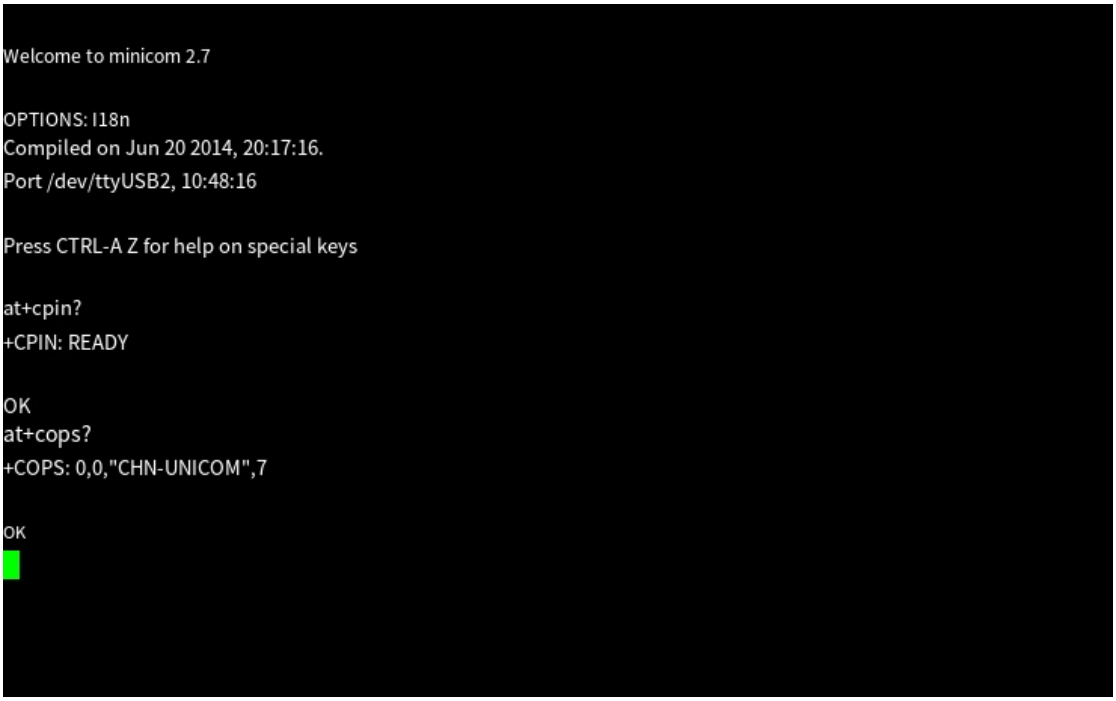
[25.925279] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB12

[25.940661] usb 2-1: GSM modem (1-port) converter now attached to ttyUSB13

3.4.2.3Verify the basic status of the module

According to 3.4.1 track module information query modemscan section method, can be detected AT the port for/dev/ttyUSB2, / dev/ttyUSB3. 3.4.1Query modemscan for module informationThrough the two port to check the module state. Take /dev/ttyUSB2 as an example

#minicom -D /dev/ttyUSB2



at+cpin?
+CPIN: READY
Returning READY indicates that SIM card recognition is normal.

at+cops?
+ COPS: 0, 0, "CHN - UNICOM", 7
Return the above information to indicate that the registered connection carrier is normal.

If the SIM card is not normal, please re-plug the SIM card, restart the device, and then check.
If there is no registration to the operator, please check the state of the antenna, 5 to 4 g module antenna plug in all.

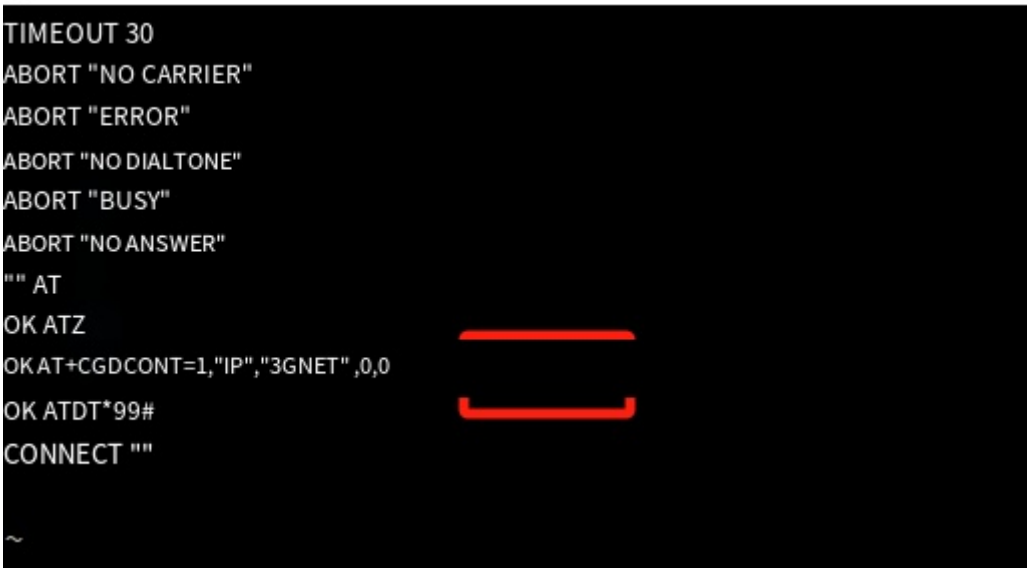
3.4.2.4Example of PPP dialing

If the above two steps check normal, communication module identification is normal, SIM card and operator status is normal. You can dial directly. The following uses the default script as an example, to demonstrate.
pppd dial directly, virtual serial port name can refer to 3.4Cellular3.4Cellular device Use

3.4.2.4.1 Modifying APN

If the carrier APN information needs to be set separately, it can be modified directly in the Dial connect script.

#vi /etc/ppp/peers/default-chat-connect



Please change the 3GNET in the above document to the actual APN of the customer.

3.4.2.4.2 Set a username password

Refer to this section if the customer needs to set a username and password, if not, ignore this section.

The ppp dial-up username and password are stored in /etc/ppp/chap-secrets, /etc/ppp/pap-secrets, /etc/ppp/peers/default.

/etc/ppp/chap-secrets is where the chap password is stored

/etc/ppp/pap-secrets holds PAP passwords.

The /etc/ppp/peers/default main configuration file uses the name parameter to specify which password to use in the password file.

#vi /etc/ppp/chap-secrets

#/etc/ppp/chap-secrets

#client server secret IP address

"cmnet" * "cmnet" * 4i

#vi /etc/ppp/pap-secrets

#/etc/ppp/pap-secrets
#client server secret IP address
"cmnet" * "cmnet" * 4i

#vi /etc/ppp/peers/default
debug
#nodetach
/dev/ttyUSB1
115200
nocrtscts
lock
usepeerdns
noauth
noipdefault
novj
novjccomp
noccp
defaultroute
#lcp-echo-failure 5
#lcp-echo-interval 30
persist
ipcp-accept-local
ipcp-accept-remote
connect '/usr/sbin/chat -s -v -f /etc/ppp/peers/default-chat-connect'
disconnect '/usr/sbin/chat -s -v -f/etc/ppp/peers/default-chat-disconnect'
name cmnet
auth

Add the name parameter to the above file

3.4.2.4.3 Command dial

root@xxxx:~# **pppd call default /dev/ttyUSB1 &**
pppd is the program name
call action parameters
default is the call script name
/dev/ttyUSB1 is the serial port parameter in the replacement script
And on behalf of the program is running after the meeting

3.4.2.4.4 Use script dialing

Use script dialing that is already configured
root@xxxx:~# **wan.sh**
Usage: wan.sh unicom|cmnet|telecom|other [devicename]
root@xxxx:~# **wan.sh default /dev/ttyUSB1**
Usage: wan.sh unicom|cmnet|telecom|other [devicename]

3.4.2.4.5 Dialing effect

A normal dial-up tip as shown below
root@xxxx:~# **wan.sh default /dev/ttyUSB1**
killall: pppd: no process killed
The timeout set 30 seconds # began to **show chat - connect script execution**
abort on (NO CARRIER)
abort on (ERROR)
abort on (NO DIALTONE)
abort on (BUSY)
abort on (NO ANSWER)
send (AT^M)
expect (OK)
AT^M^M
OK
-- got it

send (ATZ^M)
expect (OK)
^M
ATZ^M^M
OK
-- got it

send (AT+CGDCONT=1,"IP","3GNET",,0,0^M)
expect (OK)
^M
The AT + CGDCONT = 1, "IP", "3 gnet,", 0, 0 ^ ^ M M
OK
-- got it

send (ATDT*99#^M)

expect (CONNECT)
^M
ATDT*99#^M^M
CONNECT
-- got it

send (^M)
Script /usr/sbin/chat -s -v -f /etc/ppp/peers/default-chat-connect finished (pid 2441), status = 0x0 **# Prompt for chat-connect** script to execute successfully
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB1 **# Generate ppp0 network card, start protocol negotiation with the operator below**
rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x79049dfc> <pcomp> <accomp>]
Warning - secret file /etc/ppp/pap-secrets has world and/or group access
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x28feffa7> <pcomp> <accomp>]
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x79049dfc> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x28feffa7> <pcomp> <accomp>]
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
Sent [IPCP ConfReq id = 0 x1 < addr 0.0.0.0 > < ms - dns1 0.0.0.0 > < ms - dns2 0.0.0.0 >]
rcvd [IPCP ConfReq id=0x1]
sent [IPCP ConfNak id=0x1 <addr 0.0.0.0>]
RCVD [IPCP ConfNak id = 0 x1 < addr 10.53.206.231 > < ms - dns1 123.123.123.123 > < ms - dns2 123.123.123.124 >]
Sent [IPCP ConfReq id = 0 x2 < addr 10.53.206.231 > < ms - dns1 123.123.123.123 > < ms - dns2 123.123.123.124 >]
RCVD [IPCP ConfReq id = 0 x2 < addr 10.53.206.231 >]
Sent [IPCP ConfAck id = 0 x2 < addr 10.53.206.231 >]
RCVD [IPCP ConfAck id = 0 x2 < addr 10.53.206.231 > < ms - dns1 123.123.123.123 > < ms - dns2 123.123.123.124 >]
The not replacing existing default route via 172.21.67.1
The local IP address 10.53.206.231
Remote IP address 10.53.206.231
Primary DNS address 123.123.123.123
Secondary DNS address 123.123.123.124
Script /etc/ppp/ip-up started (pid 2476)
Script /etc/ppp/ip-up finished (pid 2476), status = 0x0

3.4.2.5The PPP configuration file parsing

3.4.2.5.1 The default configuration file argument parsing

root@xxxx:/etc/ppp/peers# ls default*
default default-chat-connect default-chat-disconnect

Check the default script content
root@xxxx:/etc/ppp/peers# cat default
debug # Enable this parameter to output information in syslog
#nodetach # does not run in the background, ctrl+c will interrupt the pppd program
/ dev/ttyUSB1 # this parameter as the default use serial port name, the command line if there is this parameter, then according to the command line parameters
115200 # Serial port baud rate
nocrtscts
lock
Usepeerdns # use DNS server information
noauth # Disable the use of auth authorization, this parameter must be enabled
noipdefault
novj
novjccomp
noccp
Defaultroute # use server provides the default route, if the local have default routes, will add to fail
#lcp-echo-failure 5
#lcp-echo-interval 30
persist
#ipcp-accept-local
#ipcp-accept-remote
connect '/usr/sbin/chat -s -v -f /etc/ppp/peers/default-chat-connect'
disconnect '/usr/sbin/chat -s -v -f/etc/ppp/peers/default-chat-disconnect'

3.4.2.5.2 connect script parameters parsed

Check the chat connect script, this script sends the AT command before the modem dialing, and configure the relevant. If the APN Settings, etc., are configured in this script. If you want to query or set the configuration command, you can also add in this script.
root@xxxx:/etc/ppp/peers# cat default-chat-connect

30 # set the TIMEOUT TIMEOUT, if more than time to exit the dial, the default value for 45 seconds
ABORT "NO CARRIER" # abort dialing if "NO CARRIER" is returned
ABORT "ERROR"
ABORT "NO DIALTONE"
ABORT "BUSY"
ABORT "NO ANSWER"
"" AT # Send the AT command to confirm that the modem is working, and the "" in front of it means no preset result is required
OK ATZ # if a command returns on OK, then send ATZ command
OK AT+CGDCONT=1,"IP","3GNET",,0,0 # This parameter is set APN
OK ATDT * 99 # # call center number, the official start of the dial.
CONNECT ""

Commonly used commands:

ATZ restore original Settings
The **AT + CGDCONT** Settings APN
The values that
□ < *cid* > : 1-4, the PDP to set up the environment index values. Other PDP related command can through this index value
To invoke the saved Settings
□ < *PDP_type* > : string value, said package exchange protocol type.
The value meaning
IP IPv4
IPV6 IPV6
IPV4V6 IPv4 / v6
The PPP end-to-end protocol
□ < *APN* > : string value, said connection GGSN or extranet access point of the domain name.

Name of the operator	Access point	Username Password	Dial-up number	Notes
China Mobile	cmnet cmwap	There is no	* * * * 1 # 99	2.5 G2.75 G (GPRS)
China Mobile	cmnet cmwap	There is no	* 98 * 1 #	3G(TD-SCDMA)
China Unicom	3gnet	There is no	* 99 #	3G(WCDMA)
China Telecom	There is no	card card	# 777	CDMA200

< *PDP_addr* > : string value, says MS's address.
<*d_comp*> : Numeric value that controls the compression of PDP data.

Value taken	meaning
0	Without compression

1	Compress

Note: Without <d_comp> is equivalent to <d_comp> being 0.

< *h_comp* > : a numeric value, control the compression of the PDP head.

Value taken	Meaning
0	No compression
1	The compression

Note: do not take < h_comp > is equivalent to the < h_comp > 0.

3.4.2.5.3 Disconnect the script argument parsing

Chat - disconnect the script, the script for the end connection to send AT commands

root@xxxx:/etc/ppp/peers# cat default-chat-disconnect

ABORT "ERROR"

ABORT "NO DIALTONE"

SAY "\nSending break to the modem\n"

'' "\K"

'' "+++ATH"

SAY "\nGoodbye\n"

3.4.2.5.4 Common carrier connect scripts

3.4.2.5.4.1 China Mobile

root@xxxx:/etc/ppp/peers# cat cmnet-chat-connect

TIMEOUT 30

ABORT "NO CARRIER"

ABORT "ERROR"

ABORT "NO DIALTONE"

ABORT "BUSY"

ABORT "NO ANSWER"

"" AT

#OK AT+COPS=2

#OK AT+URAT=1,2

#OK AT+COPS=0

OK ATZ

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

OK ATDT*99***1#

CONNECT ""

3.4.2.5.4.2 China Telecom

root@xxxx:/etc/ppp/peers# cat telecom-chat-connect
TIMEOUT 60
ABORT "NO CARRIER"
ABORT "ERROR"
ABORT "NO DIALTONE"
ABORT "BUSY"
ABORT "NO ANSWER"
"" AT
OK ATZ
OK ATDT#777
CONNECT ""

3.4.2.5.4.3 China Unicom

root@xxxx:/etc/ppp/peers# cat unicom-chat-connect
TIMEOUT 30
ABORT "NO CARRIER"
ABORT "ERROR"
ABORT "NO DIALTONE"
ABORT "BUSY"
ABORT "NO ANSWER"
"" AT
OK ATZ
OK AT+CGDCONT=1,"IP","3GNET",,0,0
OK ATDT*99#
CONNECT ""

3.4.2.5.4.4 Custom connect scripts

This use case takes the unicom-chat-connect script as an example
root@xxxx:/etc/ppp/peers# cat unicom-chat-connect
TIMEOUT 30
ABORT "NO CARRIER"
ABORT "ERROR"
ABORT "NO DIALTONE"
ABORT "BUSY"
ABORT "NO ANSWER"
"" AT
OK ATZ
OK AT+COPS? # Add custom command to query if registered with carrier

OK AT+CNUM; +CSQ # Add multiple commands in one line with ";" ", the second command does not need AT to start
OK AT+CGDCONT=1,"IP","3GNET",,0,0
OK ATDT*99#
CONNECT ""

3.4.3 QMI dial

3.4.3.1Verify that the QMI driver was installed successfully

After the QMI driver is successfully installed, the WWAN0 network card will be generated, and if it is a 5G module, multiple network cards for WWAN* may be generated. Before using QMI dialing, make sure the network card is already working properly.

 # ifconfig -a

```
eth0      Link encap:Ethernet  HWaddr F8:2E:0C:8A:87:00
          inet addr:192.168.172.110 Bcast:0.0.0.0 Mask:255.255.255.0
          inet6 addr: fe80::fa2e:cff:fe8a:8700/64 Scope:Link
          inet6 addr: 2408:8207:1833:3240:fa2e:cff:fe8a:8700/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19942 errors:0 dropped:36 overruns:0 frame:0
          TX packets:909 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1812163 (1.7 MiB) TX bytes:115942 (113.2 KiB)
          Interrupt:173

eth1      Link encap:Ethernet  HWaddr F8:2E:0C:8A:87:02
          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 (0.0B) TX bytes:0 (0.0B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:608 (608.0B) TX bytes:608 (608.0B)
```

```
sit0      Link encap:IPv6-in-IPv4
          NOARP  MTU:1480  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:1
          RX bytes:0 (0.0B) TX bytes:0 (0.0B)

wwan0     Link encap:Ethernet  HWaddr 76:97:99:36:E0:81
          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 (0.0B) TX bytes:0 (0.0B)
```

3.4.3.2QMI dial

```
# quectel-CM
[04-26_10:42:38:959] Quectel_QConnectManager_Linux_V1.6.0.24
[04-26_10:42:38:963] Find /sys/bus/usb/devices/2-1 idVendor=0x2c7c idProduct=0x125, bus=0x002, dev=0x002
[04-26_10:42:38:966] Auto find qmichannel = /dev/cdc-wdm0
[04-26_10:42:38:967] Auto find usbnet_adapter = wwan0
[04-26_10:42:38:968] netcard driver = qmi_wwan, driver version = 22-Aug-2005
[04-26_10:42:38:970] Modem works in QMI mode
[04-26_10:42:38:983] cdc_wdm_fd = 7
[04-26_10:42:39:081] Get clientWDS = 19
[04-26_10:42:39:113] Get clientDMS = 1
[04-26_10:42:39:144] Get clientNAS = 3
[04-26_10:42:39:177] Get clientUIM = 1
[04-26_10:42:39:209] Get clientWDA = 1
[04-26_10:42:39:241] requestBaseBandVersion EC20CEFDR02A09M4G 1 [Nov 30 2016 04:00:00]
[04-26_10:42:39:370] requestGetSIMStatus SIMStatus: SIM_READY
[04-26_10:42:39:400] requestGetProfile[1] 3gnet///0
[04-26_10:42:39:433] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE
[04-26_10:42:39:464] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[04-26_10:42:39:465] ifconfig wwan0 0.0.0.0
[04-26_10:42:39:481] ifconfig wwan0 down
[04-26_10:42:39:560] requestSetupDataCall WdsConnectionIPv4Handle: 0x879bcde0
[04-26_10:42:39:689] ifconfig wwan0 up
```

[04-26_10:42:39:705] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.36.1
udhcpc: broadcasting discover
udhcpc: broadcasting discover
udhcpc: broadcasting discover
udhcpc: broadcasting discover
udhcpc: broadcasting discover
udhcpc: no lease, failing
[04-26_10:42:55:227] File:ql_raw_ip_mode_check Line:136 udhcpc fail to get ip address, try next:
[04-26_10:42:55:228] ifconfig wwan0 down
[04-26_10:42:55:266] echo Y > /sys/class/net/wwan0/qmi/raw_ip
[04-26_10:42:55:267] ifconfig wwan0 up
[04-26_10:42:55:282] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.36.1
udhcpc: broadcasting discover
udhcpc: broadcasting select for 10.61.130.213, server 10.61.130.214
udhcpc: lease of 10.61.130.213 obtained from 10.61.130.214, lease time 7200
[04-26_10:42:55:650] /etc/udhcpc.d/50default: Adding DNS 123.123.123.123
[04-26_10:42:55:650] /etc/udhcpc.d/50default: Adding DNS 123.123.123.124
[04-26_10:42:55:662] dns:/var/run/resolv.conf

Note log messages during QMI dialing,
The bottom line, SIM card recognition is normal, must be **SIM_READY** state.
[04-26_10:42:39:370] requestGetSIMStatus **SIMStatus: SIM_READY**
[04-26_10:42:39:400] requestGetProfile[1] 3gnet///0
[04-26_10:42:39:433] requestRegistrationState2 MCC: 460, MNC: 1, PS: Attached, DataCap: LTE

3.4.3.3QMI set APN, etc

#quectel-CM --help
[04-26_14:47:28:160] Quectel_QConnectManager_Linux_V1.6.0.24
[04-26_14:47:28:162] Usage: ./quectel-CM [options]
[04-26_14:47:28:163] -s [apn [user password auth]] Set apn/user/password/auth get from your network provider. auth: 1~pap, 2~chap
[04-26_14:47:28:164] -p pincode Verify sim card pin if sim card is locked
[04-26_14:47:28:166] -p [quectel-][qmi|mbim]-proxy Request to use proxy

[04-26_14:47:28:167] -f logfilename Save log message of this program to file
[04-26_14:47:28:168] -u usbmonlog filename Save usbmon log to file
[04-26_14:47:28:169] -i interface Specify which network interface to setup data call when multi-modems exits
[04-26_14:47:28:169] -4 Setup IPv4 data call (default)
[04-26_14:47:28:170] -6 Setup IPv6 data call
[04-26_14:47:28:170] -n pdn Specify which pdn to setup data call (default 1 for QMI, 0 for MBIM)
[04-26_14:47:28:171] -k pdn Specify which pdn to hangup data call (by send SIGINT to 'quectel-CM -n pdn')
[04-26_14:47:28:171] -m iface-idx Bind QMI data call to wwan0_<iface idx> when QMAP used. E.g '-n 7 -m 1' bind pdn-7 data call to wwan0_1
[04-26_14:47:28:171] -b Enable network interface bridge function (default 0)
[04-26_14:47:28:172] -v Verbose log mode, for debug purpose.
[04-26_14:47:28:172] [Examples]
[04-26_14:47:28:173] Example 1: ./quectel-CM
[04-26_14:47:28:173] Example 2: ./quectel-CM -s 3gnet
[04-26_14:47:28:173] Example 3: ./quectel-CM -s 3gnet carl 1234 1 -p 1234 -f gobinet_log.txt

Setting up APN
quectel-CM -s 3gnet &
Set up APN and username password
#**quectel-CM -s 3gnet carl 1234 1 &**

Since the QMI dialer will not run in the background by default, it will run in the background if required. Please add &
quectel-CM &

3.4.4 Debug a list of commonly used AT commands

Since AT command each modem manufacturer has its own command list, here only a few of the more common commands are listed. For your reference, generally use minicom software to open the corresponding virtual serial port of the modem for operation. After opening the serial port, first send the AT command, see if there is OK return, to confirm whether the serial port is configurable with serial port.

Use minicom to read the module information

minicom -D /dev/ttyUSB0

Read module information using microcom

#microcom -t 15000 -s 115200 /dev/ttyUSB0

Welcome to minicom 2.7

OPTIONS: I18n

Compiled on Jun 20 2014, 20:17:16.

Port /dev/ttyUSB0, 09:45:28

Press CTRL-A Z for help on special keys

at

OK

3.4.4.1Query SIM card status

at+cpin?

+CPIN: READY

OK

Other reply parameters

ERROR : MT is not found sim card

READY: MT is not pending for any password

SIM PIN: MT is waiting for SIM PIN to be given

S**IM PUK:** MT is waiting for SIM PUK to be given

SIM PIN2: MT is waiting for SIM PIN2 to be given

SIM PUK2: MT is waiting for SIM PUK2 to be given

PH-NET PIN: MT is waiting for network personalization password to be given

PH-NET PUK: MT is waiting for network personalization unblocking passwordto be given

PH-NETSUB PIN: MT is waiting for network subset personalization password to be given

PH-NETSUB PUK: MT is waiting for network subset personalization unblocking password to be given

PH-SP PIN: MT is waiting for service provider personalization password to be given

PH-SP PUK: MT is waiting for service provider personalization unblocking

password to be given
PH-CORP PIN: MT is waiting for corporate personalization password to be given
PH-CORP PUK: MT is waiting for corporate personalization unblocking password to be given

3.4.4.2 Check for connected carrier information

at+cops?
+COPS: 0,0,"CHINA-UNICOM",7

OK
+COPS: <mode>[,<format>[,<oper>][,<Act>]]
<mode>
0 Automatic mode. **<oper>** field is ignored
1 Manual operator selection. **<oper>** field shall be present and <**Act**> optionally
2 Manually deregister from network
3 Set only **<format>** (for **AT+COPS?** Read Command), and do not attempt registration/deregistration (**<oper>** and **<Act>** fields are ignored). This value is invalid in the response of Read Command.
4 Manual/automatic selection. **<oper>** field shall be presented. If manual selection fails, automatic mode (**<mode>**=0) is entered
<format>
0 Long format alphanumeric **<oper>** which can be up to 16 characters long
1 Short format alphanumeric **<oper>**
2 Numeric **<oper>**. GSM location area identification number
<Act>
Access technology selected. Values 3, 4, 5 and 6 occur only in the response of Read Command while MS is in data service state and is not intended for the **AT+COPS** Write Command.
0 GSM
2 UTRAN
3 GSM W/EGPRS
4 UTRAN W/HSDPA
5 UTRAN W/HSUPA
6 UTRAN W/HSDPA and HSUPA
7 E-UTRAN
100 CDMA

Not connected to carrier base station return information as follows:
at+cops?

+COPS: 0

OK

3.4.4.3Looking up phone numbers

at+cnum
+CNUM: "","+8618600100000",145

OK

This command returns normally. The mobile phone number must be written in the sim card, otherwise this command will return an ERROR

[+CNUM: [<alpha>],<number>,<type>]
<alpha>
Optional alphanumeric string associated with **<number>**.
<number>
String type phone number of format specified by **<type>**
<type>
Type of address of octet in integer format (Refer to *3GPP TS 24.008* ***subclause 10.5.4.7*** for details). Usually, it has three kinds of values:
129 Unknown type
145 International type (contains the character "+")
161 National type

3.4.4.4Query Signal Strength

at+csq
+CSQ: 21,99

OK

+CSQ: <rssi>,<ber>

<rssi>
0 -113dBm or less
1 -111dBm

2... 30 -109dBm... -53dBm
31 -51dBm or greater
99 Not known or not detectable
100 -116dBm or less
101 -115dBm
102... 190 -114dBm... -26dBm
191 -25dBm or greater
199 Not known or not detectable
100~199 Extended to be used in TD-SCDMA indicating received signal code power (RSCP)

<ber>

Channel bit error rate (in percent)

0... 7 As RXQUAL values in the table in *3GPP TS 45.008* ***subclause 8.2.4***
99 Not known or not detectable

3.4.4.5IMEI

microcom -t 5000 /dev/ttyUSB3
at+cgsn
862815030700775

OK

3.4.5 Zte ME3760 module configuration

The ZTE ME3460 module does not use the ppp dialing method and needs to use a dedicated command for dialing.

root@xxxx:~# **AutoDialup4G**
netcard [eth2] not exist
Usage:
AutoDialup4G com_port_name netcard
example: AutoDialup4G /dev/ttyUSB0 eth2
Configure this program parameter as needed
root@xxxx:~# **AutoDialup4G /dev/ttyUSB1 eth4**
com port:/dev/ttyUSB1,netcard:eth4
[0]AT
[0]AT OK

This program automatically sends the following command:
"AT"
"AT+ZGACT?"

"The AT ^ SYSCONFIG = 17,0,1,1"
"AT+CFUN=1"
"AT^SYSINFO"
"AT + CGACT = 1, 1"
"AT+ZGACT?"
"AT + ZGACT = 1, 1"

3.4.6 4G module reset

Confirm BUS number:
lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 002: ID 2c7c:0125
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Verify that the USB bus number of device 2c7c:0125 is 2 by using the lsbub command

Check the reset script parameters
/usr/bin/minipcie_reset.sh

usage : minipcie_reset.sh PowerReset/ModuleReset BusNum

Reset module
#/usr/bin/minipcie_reset.sh PowerReset 2
crw-rw---- 1 root dialout 188, 0 Jul 22 14:43 /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 1 Jul 22 14:43 /dev/ttyUSB1
crw-rw---- 1 root dialout 188, 2 Jul 22 14:43 /dev/ttyUSB2
crw-rw---- 1 root dialout 188, 3 Jul 22 14:43 /dev/ttyUSB3
Modem initial Success....

3.4.7 dual sim handover

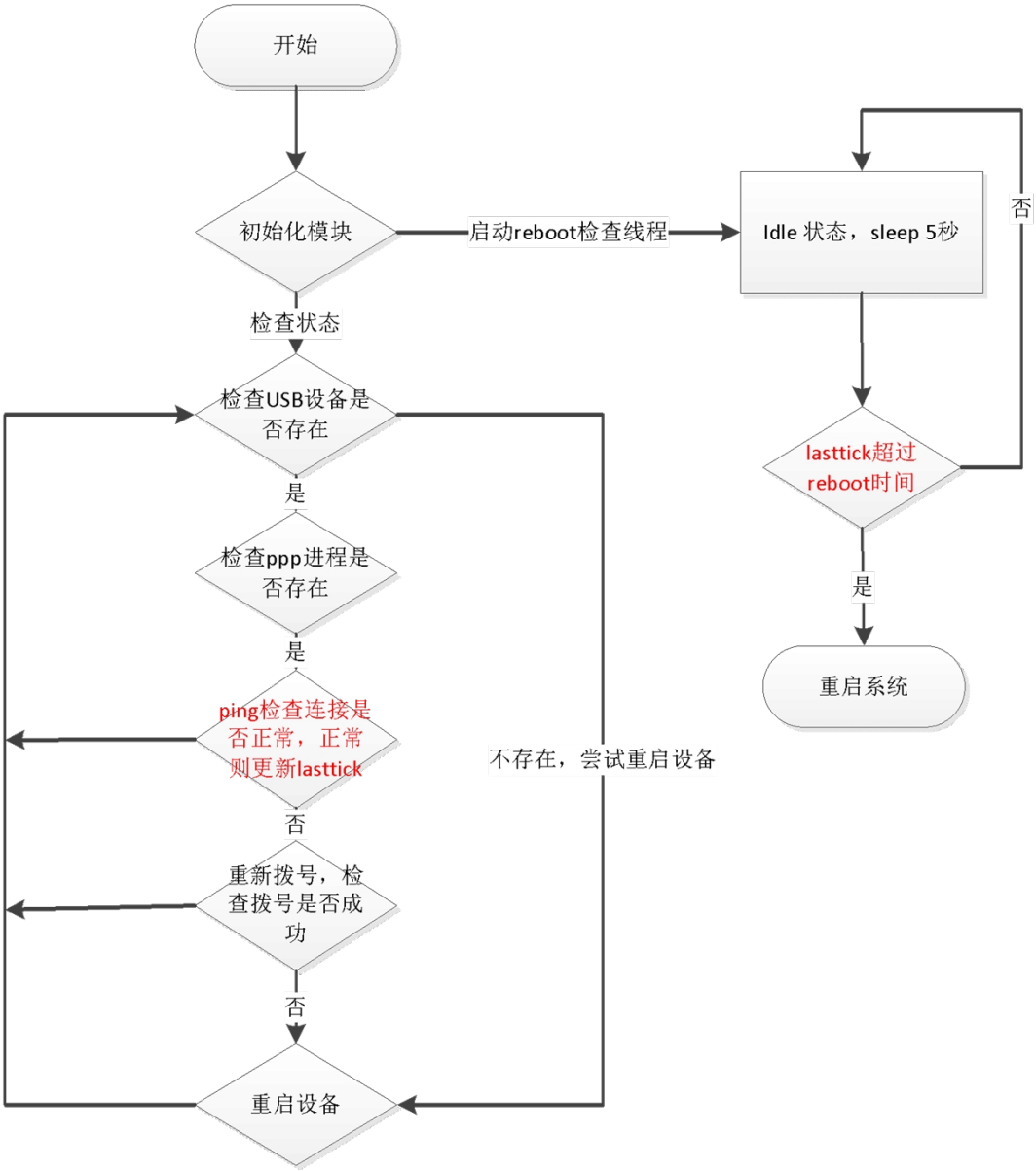
This method only supports dual SIM devices.
/usr/bin/sim_switch.sh
usage: /usr/bin/sim_switch.sh [1/2]
parameters and options:
[1 -> SMI1]
[2 -> SIM2]

#/usr/bin/sim_switch.sh 1

board name : ecu1051
switch to SIM1
crw-rw---- 1 root dialout 188, 0 Jul 22 15:01 /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 1 Jul 22 15:01 /dev/ttyUSB1
crw-rw---- 1 root dialout 188, 2 Jul 22 15:01 /dev/ttyUSB2
crw-rw---- 1 root dialout 188, 3 Jul 22 15:01 /dev/ttyUSB3
Modem initial Success....

3.5 AdvWrielessCheckd instructions

3.5.1 Process Instructions



Currently supports ppp0,lan fixed network card check function.

3.5.2 How to Use

3.5.2.1Select the correct profile

Equipment information unified integration in/home/root/project/CellularDeviceInfo acr. No additional configuration is required.

3.5.2.2Start the test program

```
root@ecu1251:~# AdvWirelessCheckd
AdvWirelessCheck Aug  2 2019 build 15:21:28
open libDCTag.so failed
ERROR:libDCTag.so: cannot open shared object file: No such file or directory
open libwatchprocess.so failed
ERROR:libwatchprocess.so: cannot open shared object file: No such file or directory
TAGLINK_PATH:/home/root

Config file: /home/root/project/SystemSetting.acr

Dual  sim:0
open libDCTag.so failed
ERROR:libDCTag.so: cannot open shared object file: No such file or directory
load DCTag failed

connProcessThread,/home/root/bin/awc_3g.so++++
Load module /home/root/bin/awc_3g.so, netcard ppp0

ERROR: PPP link is not active on ppp0
killall: pppd: no process killed
switchTag=(null),switchType=None
smsOnly=0

ppp0,cb=80
ppp0,restartSystemSeconds=0
ppp0,dialUpCommand=wan.sh
ppp0,dialDownCommand=/etc/ppp/ppp-off; /usr/bin/killall pppd
ppp0,processName=pppd
ppp0,ttyCommPort=Not Init
```

ppp0,ttyConfigPort=Not Init
ppp0,ttyCommPortNo=3
ppp0,ttyConfigPortNo=2
ppp0,usbName=
ppp0,usbLable=Android_Android
ppp0,usbID=2c7c:0125
ppp0,usbBus=0
ppp0,operator=auto
ppp0,netmode=1
ppp0,checkmode=0
ppp0,sim1,operator=
ppp0,sim1,netmode=0
ppp0,sim2,operator=
ppp0,sim2,netmode=0
return:1

rebootCheckThread:last connect tick:15324
ERROR: PPP link is not active on ppp0
killall: pppd: no process killed
return:1,dual sim:0

setMode+++
setMode:file[/home/root/project/urat] not found
ppp0,cb=80
ppp0,restartSystemSeconds=0
ppp0,dialUpCommand=wan.sh
ppp0,dialDownCommand=/etc/ppp/ppp-off; /usr/bin/killall pppd
ppp0,processName=pppd
ppp0,ttyCommPort=/dev/ttyUSB3 # This serial port is used for dialing and must be identified normally
ppp0,ttyConfigPort=/dev/ttyUSB2
ppp0,ttyCommPortNo=3
ppp0,ttyConfigPortNo=2
ppp0,usbName=
ppp0,usbLable=Android_Android
ppp0,usbID=2c7c:0125
ppp0,usbBus=2
ppp0,operator=auto
ppp0,netmode=1
ppp0,checkmode=0
ppp0,sim1,operator=
ppp0,sim1,netmode=0
ppp0,sim2,operator=

ppp0,sim2,netmode=0
n3g_init return:1

killall: GPSManager: no process killed
ERROR: PPP link is not active on ppp0
killall: pppd: no process killed
check_carrier:sim card found!
get_mno_info: scan operator name failed! buf = +COPS: 0,0,"JD Mobile",7

OK

The following information marked in red is the normal information read before dialing and is used for dialing

set_cops_mode: tty = /dev/ttyUSB3
get_mobile_mno: csq = 27
get_mno_info: operator code = 46001
get_mno_info: tty = /dev/ttyUSB3,mcc = 460,mnc = 01
find_provider_apn: found apn = 3gnet for mcc = 460, mnc = 01.
checkProvider: find apn = 3gnet, netmode = 1
checkProvider:mcc = 460, mnc = 01,ret = 0, wan.sh default
check_carrier:checkProvider return = 1
get_mno_info: operator code = 46001
get_mno_info: tty = /dev/ttyUSB3,mcc = 460,mnc = 01
switch to 3
connProcessThread,awc_3g.so:checkProcess failed
switch to 5
switch to 6
connProcessThread,awc_3g.so:redialUp[1/4]
ERROR: PPP link is not active on ppp0
killall: pppd: no process killed
rebootCheckThread:last connect tick:15324
killall: pppd: no process killed
rebootCheckThread:last connect tick:15324
rebootCheckThread:last connect tick:15324
switch to 0
switch to 0
rebootCheckThread:last connect tick:15324
rebootCheckThread:last connect tick:15324

3.5.2.3Boot by default

~# vi /etc/rc.local

#! /bin/sh -e
#
rc.local
#
This script is executed at the end of each multiuser runlevel.
Make sure that the script will "exit 0" on success or any other
value on error.
#
In order to enable or disable this script just change the execution
bits.
#
By default this script does nothing.
export LD_LIBRARY_PATH=$ LD_LIBRARY_PATH :/home/root/lib/
/home/root/bin/AdvWirelessCheckd -d
exit 0

3.5.3 Profile CellularDeviceInfo.acr instructions

~# cat /home/root/project/CellularDeviceInfo.acr

```
<? xml version="1.0" encoding="utf-8"? >

<CellularDeviceInfo name="" description="">

   <Device deviceID="2cb7:0001" GPSType="none" GPSInterface="0" ATPortCount="2" ATPortInterface="2,4" DialType="ppp" deviceName="CU101-GL(UNICOM)" />

   <Device deviceID="12d1:15c1" GPSType="none" GPSInterface="0" ATPortCount="3" ATPortInterface="4,2,5" DialType="ppp" deviceName="ME909S(Huawei)" />

   <Device deviceID="12d1:1c25" GPSType="none" GPSInterface="0" ATPortCount="3" ATPortInterface="4,2,5" DialType="ppp" deviceName="MU709S(Huawei)" />

   <Device deviceID="12d1:1573" GPSType="none" GPSInterface="0" ATPortCount="3" ATPortInterface="4,2,6" DialType="ppp" deviceName="MU609(Huawei)" />

   <Device deviceID=" 19d1:0199 "GPSType="none" GPSInterface="0" ATPortCount="2" ATPortInterface="0,2" DialType="none" deviceName="ME3760(ZTE)" />

   <Device deviceID="19d2:1476" GPSType="none" GPSInterface="0" ATPortCount="2" ATPortInterface="1,2" DialType="ppp" deviceName="ME3630(ZTE)" />

   <Device deviceID="05c6:90b3" GPSType="embedded" GPSInterface="2" ATPortCount="1" ATPortInterface="3" DialType="none" deviceName="MDG100 RNDIS(usb0) " />

   <Device deviceID="1546:01a7" GPSType="independ" GPSInterface="1" ATPortCount="0" ATPortInterface="0" DialType="none" deviceName="EWM-G108" />

   <Device deviceID="2c7c:0296" GPSType="embedded" GPSInterface="1" ATPortCount="2" ATPortInterface="2,3" DialType="ppp" deviceName="BG96(Quectel)" />

</CellularDeviceInfo>
```

This file lists information about all devices currently supported
deviceID="2c7c:0296" modem module usbid.
GPSType="embedded" whether gps is supported. The value is none,embedded, independ in total 3 parameters.
GPSInterface="1" usb device interface number.

ATPortCount="2" The number of virtual serial ports that can send at commands, at least 1 is required.

ATPortInterface="2,3" can send at command usb interface number, the first one is used to query module information, the second one is used for ppp dial, like Huawei series, please put the dedicated ppp dial in the second position.

DialType="ppp" dial type. ppp is currently supported, with none2 parameters

deviceName="BG96(Quectel)" module name, for display only

This information can be generated automatically with modescan, see 3.4Cellular3.4Cellular device Use

3.5.4 Configuration file [SystemSetting.acr] instructions

System has a demo in the configuration file reference for new modules/home/root/project/cellular/demo/SystemSetting acr

3.5.4.1Cellular Configuration parameters <GPRS>

```
<tDeviceConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <LAN>
    <GPRS othername2="telecom" isMutliSim="false" is4G="false" othername="unicom" connection="true" othername1="cmnet">
      <ModuleName>Auto</ModuleName>
      <APN />
      <UserName />
      <Password />
      <PINNumber />
      <PhoneNumber />
      <PingInterval>60</PingInterval>
      <RetryCount>0</RetryCount>
      <NetworkMode>4G</NetworkMode>
      <Operator>auto</Operator>
      <deviceName>EC20CEFA-512-STD(Quectel)</deviceName>
      <Lable>Android_Android</Lable>
      <CommunicationPort>3</CommunicationPort>
      <ConfigPort>2</ConfigPort>
      <USBDeviceID>2c7c:0125</USBDeviceID>
      <SMSPort>2</SMSPort>
      <USBDeviceName />
```

```
        <ConnectionCheckType>0</ConnectionCheckType>
        <MaxSilenceTime>1</MaxSilenceTime>
        <RebootOnFailureTime>0</RebootOnFailureTime>
        <NetworkInterface>ppp0</NetworkInterface>
        <DNSList_ipv4 isAutomatically="true" />
        <DNSList_ipv6 isAutomatically="true" />
        <EnableSim>false</EnableSim>
         <MutliSim switchType="None" masterSIM="1">^M
          <Sims name="sim1" enable="true">^M
            <Operator>auto</Operator>^M
            <NetworkMode />^M
            <APN />^M
            <UserName />^M
            <Password />^M
            <PhoneNumber />^M
            <AuthMethod />^M
            <isAuthentication>false</isAuthentication>^M
          </Sims>^M
          <Sims name="sim2" enable="true">^M
            <Operator>auto</Operator>^M
            <NetworkMode />^M
            <APN />^M
            <UserName />^M
            <Password />^M
            <PhoneNumber />^M
            <AuthMethod />^M
            <isAuthentication>false</isAuthentication>^M
          </Sims>^M
        </MutliSim>
        <AuthMethod />
        <isAuthentication>false</isAuthentication>
      </GPRS>
    </LAN>
</tDeviceConfig>
```

See 3.5.4.4.2-3.5.4.4.4 parameters explanation for the meaning of Celullar configuration parameters.3.5.4.4.23.5.4.4.4

3.5.4.2Fixed network card configuration parameters < ChildLAN >

```
<tDeviceConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <LAN>
    <ChildLAN name="eth0" isDHCPv6="true" isDHCP="true">^M
      <DNSList_ipv4 isAutomatically="true" />^M
      <DNSList_ipv6 isAutomatically="true" />^M
      <ConnectionCheckType>0</ConnectionCheckType>^M
      <PingInterval>60</PingInterval>^M
      <PingURL>www.badiu.com</PingURL>^M
      <MaxSilenceTime>1</MaxSilenceTime>^M
      <RebootOnFailureTime>0</RebootOnFailureTime>^M
    </ChildLAN>
  </LAN>
</tDeviceConfig>
```

Configure name= "ethx" network card name in ChildLAN node to configure different network card information. Refer to 3.2 Configuring Fixed network card IP for the LAN configuration file After ensuring that the network card can be properly enabled using the #3.2Fixed network card IP configurationifup ethX command. Then enable this function in this configuration file.

3.5.4.3WIFI Config < WiFi >

```
<tDeviceConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <LAN>
    <WiFi isDHCPv6="true" isDHCP="true" BSSID="" enableBSSID="false" NetworkCard="wlan0" Enable="true" Security="Open" PassWord="eXqGgy55tYmGuv1Jx8ZRGg==">
      <DNSList_ipv4 isAutomatically="true" />
      <DNSList_ipv6 isAutomatically="true" />
      <ConnectionCheckType>0</ConnectionCheckType>
      <WiFiAPMode isDHCPv6="false" isDHCP="true" enable="false" max_num_sta="0">
        <DNSList_ipv4 isAutomatically="false" />
        <DNSList_ipv6 isAutomatically="false" />
        <ConnectionCheckType>0</ConnectionCheckType>
      </WiFiAPMode>
```

```
        </WiFi>
      </LAN>
</tDeviceConfig>
```

In ChildLAN node configuration WiFi to enable the checks, WiFi configuration file please refer to 3.3 WiFi configuration. 3.3WIFI configurationEnsure that use # wlan. Sh up command can be normal after dialing. In this configuration file to enable this feature again.

3.5.4.4Description of Configuration parameters

3.5.4.4.1 ConnectionCheckType

<ConnectionCheckType>0</ConnectionCheckType>

This parameter to check connection.
0: disable, do not check connection
1: ping, ping ways confirm the connection address is normal
2: traffic, check the connection status by checking whether there is any change in the data received and sent by the network card, this method is not suitable for WIFI

3.5.4.4.2 PingInterval

< PingInterval > < 60 / PingInterval > this parameter as the time interval between two ping, unit is in seconds, the default for 60 seconds.
Valid when <ConnectionCheckType>1</ConnectionCheckType>.

3.5.4.4.3 PingURL

<PingURL>www.baidu.com</PingURL> This parameter is the ping host address, which can be a domain name or an IP address. You can support up to three, when there is more than one, as long as one of the ping means that the network is available, communication is normal.
Valid when <ConnectionCheckType>1</ConnectionCheckType>.

3.5.4.4.4 RebootOnFailureTime

<RebootOnFailureTime>0</RebootOnFailureTime>
The default is 0, this parameter is not enabled. This feature is enabled for 0<hour<24. Unit for hours, can use decimal.

When the value is less than 0.2, the time is calculated as the minimum value of 0.2 and the maximum value of 24.

For example, a value of 0.2. When the timer starts after the first ping fails, 12 minutes later, it still cannot ping. It reboots the entire system

3.5.4.5DUAL SIM< MutliSim >

This parameter is only valid on the dual SIM card equipment.

```
<MutliSim switchType="None" masterSIM="1">^M
        <Sims name="sim1" enable="true">^M
          <Operator>auto</Operator>^M
          <NetworkMode />^M
          <APN />^M
          <UserName />^M
          <Password />^M
          <PhoneNumber />^M
          <AuthMethod />^M
          <isAuthentication>false</isAuthentication>^M
        </Sims>^M
        <Sims name="sim2" enable="true">^M
          <Operator>auto</Operator>^M
          <NetworkMode />^M
          <APN />^M
          <UserName />^M
          <Password />^M
          <PhoneNumber />^M
          <AuthMethod />^M
          <isAuthentication>false</isAuthentication>^M
        </Sims>^M
      </MutliSim>
```

3.5.4.5.1 Use SIM cards first

masterSIM="1"

1: SIM1 is preferred when both SIM slots have SIM cards.

2: When both SIM slots have SIM cards, use SIM2 first.

If two SIM card slot only one SIM card, this parameter is null and void.

3.5.4.5.2 sim card switching mode

switchType="None"
None: Automatic mode, it will traverse two slots when booting, if
ConnectCheck: Check whether the SIM card is switched by connect. ConnectionCheckType must be 1 or 2 when using this method.

When the connection check failed last in redial, dial-up success is not to switch from the SIM card. If the dial it fails to switch from the SIM card.

3.6 Routing Table Configuration

3.6.1 View the current routing table

root@xxxx:~# **route -n**
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0172.21.67.1 0.0.0.0UG 1 0 0 eth1
172.21.67.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1

3.6.2 Add routing table

root@xxxx:~# **route add default eth0**
root@xxxx:~# **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 eth0
0.0.0.0 172.21.67.1 0.0.0.0UG 1 0 0 eth1
172.21.67.0 0.0.0.0 255.255.255.0U 0 0 0 eth1

3.6.3 Delete the routing table

root@xxxx:~# **route del default eth0**

```
root@xxxx:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0172.21.67.1 0.0.0.0UG 1 0 0 eth1
172.21.67.0 0.0.0.0 255.255.255.0U 0 0 0 eth1
root@xxxx:~#
```

3.6.4 Add gateway

```
root@xxxx:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.21.67.0 0.0.0.0 255.255.255.0U 0 0 0 eth1
Root @ XXXX: ~ # route add default gw 172.21.67.1 dev eth1
root@xxxx:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0172.21.67.1 0.0.0.0UG 0 0 0 eth1
172.21.67.0 0.0.0.0 255.255.255.0U 0 0 0 eth1
root@xxxx:~#
```

3.7 Firewall configuration

3.7.1 Check the current status of your firewall

```
root@xxxx:~# iptables -L -n -v
Chain INPUT (policy DROP 4848 packets, 402K bytes)
 pkts bytes target     prot opt in     out     source               destination
    0 0 ACCEPT TCP - 8 0.0.0.0/0 eth0 * 172.0.0.0 / TCP DPT: 345
Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0 0 DROP icmp -- * * 0.0.0.0/0 0.0.0.0/0 state INVALID
```

3.8 DNS configuration

The DNS information configuration file is /etc/resolv.conf. The contents are as follows:

cat /etc/resolv.conf
nameserver 8.8.8.8

There are four main keywords for resolv.conf, which are:
nameserver # Defines the IP address of the DNS server
domain # defines the local domain name
search # Defines a search list for the domain name
sortlist # Sort the returned domain names
Generally just setting the nameserver parameter is enough.

Because this file needs to be read and written frequently when the network card obtains the IP address, the default soft link in this system is /var/run/resolv.conf.
ll /etc/resolv.conf
lrwxrwxrwx 1 root root 20 Jun 23 23:54 /etc/resolv.conf -> /var/run/resolv.conf

If you need to use your default DNS when booting, there are three methods for reference, you can choose the method suitable for your own actual needs.

3.8.1 /etc/rc.local

#vi /etc/resolv.conf
#! /bin/sh -e
#
rc.local
#
This script is executed at the end of each multiuser runlevel.
Make sure that the script will "exit 0" on success or any other
value on error.
#
In order to enable or disable this script just change the execution
bits.
#
By default this script does nothing.

echo "nameserver 114.114.114.114" >> /etc/resolv.conf
exit 0

4 System Settings

4.1 sramutil

4.1.1 Help

sramutil
fram dev name : /dev/sram
fram size: 32 k[32768 bytes]
Usage:
./sramutil r ADDRESS [WIDTH] ; read value with file
./sramutil w ADDRESS VALUE [WIDTH] ; write value with file
./sramutil mr ADDRESS [WIDTH] ; read value with mmap
./sramutil mw ADDRESS VALUE [WIDTH] ; write value with mmap
./sramutil hexshow ADDRESS length ; show value with file
./sramutil dump dump.bin ; dump sram to a file
WIDTH 8/16/32... default is 32

4.1.2 Operate via file read/write mode

Currently only support by file to read and write operation, the address for 4 byte alignment, length is in multiples of four.
./sramutil r ADDRESS [WIDTH] ; read value with file
./sramutil w ADDRESS VALUE [WIDTH] ; write value with file
WIDTH 8/16/32... default is 8

Read the position 0 x0 content. 4 bytes.
sramutil r 0x0
fram dev name : /dev/sram
fram size: 32 k[32768 bytes]
Read address = 0x0,value = 0x3020100[50462976],width = 32,return = 4

4.2 The RTC clock

4.2.1 RTC clock command

hwclock
To use:
Hwclock -f/dev/rtc1 / / display the current time of RTC
Hwclock-s -f /dev/rtc1 / / synchronizes the time of the current RTC to Linux system time
hwclock -w -f /dev/rtc1/ / synchronizes Linux system time to the time of RTC
hwclock-f /dev/rtc1-localtime //RTC time is localtime
Hwclock -f/dev/rtc1 - utc / / RTC time to utc time
Hwclock -- hctosys -f/dev/rtc1 / / adjust the hardware clock is consistent with the system clock
hwclock -- systohc-f /dev/rtc1/ / adjusts the system clock to the hardware clock.

Note: now will the unified platform RTC clock soft links to/dev/RTC. The clock information can also be accessed directly by operating this node.

4.3 ntp time correction

4.3.1 ntp client

Because, ntpdate synchronization time, will cause the time jumps, on some programs and services that rely on time. Such as sleep, timer, etc. Moreover, ntpd service can correct the cpu tick while correcting the time. The ideal is to use ntpdate to force time synchronization at boot time and ntpd to do so at other times.

Note that ntpd has a self-protection setting: if the local time is too different from the source time, ntpd does not run. So the newly set time server must first update the time with the ntpdate command, and then start the ntpd service. After the ntpd service runs, it will synchronize with the source server every 64 seconds, and adjust its own time gradually according to the error value measured during each synchronization through complex calculation. As the error decreases, it will gradually increase the synchronization interval. This adjustment process will be repeated for each beat.

ntpd

-x

Normally, the time is slewed if the offset is less than the step threshold, **which is 128 ms by default**, and stepped if above the threshold. **This option sets the threshold to 600 s**, which is well within the accuracy window to set the clock manually. Note: Since the slew rate of typical Unix kernels is

limited to **0.5 ms/s,** each second of adjustment requires an amortization interval of 2000 s. Thus, an adjustment as much as 600 s will take almost 14 days to complete. This option can be used with the **-**g a**nd** -q options. Se**e the t**inker command for other options. Note: The kernel time discipline is disabled with this option.

-g

Normally, **ntpd** exits with a message to the system log if the offset exceeds the panic threshold, which is 1000 s by default. This option allows the time to be set to any value without restriction; however, this can happen only once. If the threshold is exceeded after that, **ntpd** will exit with a message to the system log. This option can be used with the **-q** and **-x** options. See the **tinker** command for other options.

General method of use:

A), use ntpdate for calibration on startup

#ntpdate -t 3 -u -s edu.ntp.org.cn

B), boot, and then use the NTPD for correction

#/usr/sbin/ntpd

4.3.1.1The ntpdate school

#ntpdate -t 3 -u -s edu.ntp.org.cn

-s specifies the use of the log operation syslog facility instead of using standard output.

-t TimeOut Specifies the time to wait for a response. The value of the given TimeOut is rounded to a multiple of 0.2 seconds. The default value is 1 second.

-u specifies using unprivileged port to send data packets.

4.3.1.2ntpd timing

/etc/ntp.conf configuration file,server address configuration see the red marked part below.

root@xxxx:~# cat /etc/ntp.conf

/etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

```
# Enable this if you want statistics to be logged.
statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable


# You do need to talk to an NTP server or two (or three).
server time.windows.com
server 127.127.1.0

# Access control configuration; see /usr/share/doc/ntp-doc/html/accopt.html for
# details.    The web page <http://support.ntp.org/bin/view/Support/AccessRestrictions>
# might also be helpful.
#
# Note that "restrict" applies to both servers and clients, so a configuration
# that might be intended to block requests from certain clients could also end
# up blocking replies from your own upstream servers.

# By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
# restrict ::1

# Clients from this (example!) subnet have unlimited access, but only if
# cryptographically authenticated.
# restrict 172.21.67.0 mask 255.255.255.0 nomodify

# If you want to provide time to your local subnet, change the next line.
# (Again, the address is an example only.)
#broadcast 192.168.123.255

# If you want to listen to time broadcasts on your local subnet, de-comment the
# next lines.    Please do this only if you trust everybody on the network!
#disable auth
#broadcastclient
```

4.3.2 ntp server

root@xxxx:~# cat /etc/ntp.conf
/etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

Enable this if you want statistics to be logged.
statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

You do need to talk to an NTP server or two (or three).

server 127.127.1.0

Access control configuration; see /usr/share/doc/ntp-doc/html/accopt.html for
details. The web page <http://support.ntp.org/bin/view/Support/AccessRestrictions>
might also be helpful.
#
Note that "restrict" applies to both servers and clients, so a configuration
that might be intended to block requests from certain clients could also end
up blocking replies from your own upstream servers.

By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery

Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
restrict ::1

Clients from this (example!) subnet have unlimited access, but only if
cryptographically authenticated.
restrict 172.21.67.0 mask 255.255.255.0 nomodify

If you want to provide time to your local subnet, change the next line.
(Again, the address is an example only.)
#broadcast 192.168.123.255

If you want to listen to time broadcasts on your local subnet, de-comment the
next lines. Please do this only if you trust everybody on the network!
#disable auth
#broadcastclient

Start method:
#/usr/sbin/ntpd

4.3.3 Time service only to specified network segments

root@xxxx:~# cat /etc/ntp.conf
/etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

Enable this if you want statistics to be logged.
statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

You do need to talk to an NTP server or two (or three).
server time.windows.com
server 127.127.1.0

Access control configuration; see /usr/share/doc/ntp-doc/html/accopt.html for
details. The web page <http://support.ntp.org/bin/view/Support/AccessRestrictions>
might also be helpful.
#
Note that "restrict" applies to both servers and clients, so a configuration
that might be intended to block requests from certain clients could also end
up blocking replies from your own upstream servers.

```
# By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
# restrict ::1

# Clients from this (example!) subnet have unlimited access, but only if
# cryptographically authenticated.
restrict 172.21.67.0 mask 255.255.255.0 nomodify

# If you want to provide time to your local subnet, change the next line.
# (Again, the address is an example only.)
#broadcast 192.168.123.255

# If you want to listen to time broadcasts on your local subnet, de-comment the
# next lines.  Please do this only if you trust everybody on the network!
#disable auth
#broadcastclient
```

4.3.4 ntpd related commands

Query time difference with server

```
# ntpq -p
root@xxxx:~# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 192.168.1.1 LOCAL(0) 6 u 55 64 37 0.518-0.021 149923.
*LOCAL(0).LOCL.5 l 33 64 77 0.000 0.000 0.004
```

Note: offset column is a time lag and server, if the time difference is too big, please use ntpdate command to update

4.4 USB device View

```
root@xxxx:~# lsusb
002: Bus 001 Device ID 0424:2512 Standard Microsystems Corp., a USB 2.0 Hub
```

001: Bus 001 Device ID 1 d6b using the color picker: 0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 003: ID 148f:5370 Ralink Technology, Corp. RT5370 Wireless Adapter
root@xxxx:~# **lsusb -t**
/: Bus 02.Port 1: Dev 1, Class=root_hub, Driver=musb-hdrc/1p, 480M
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=musb-hdrc/1p, 480M
|__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/2p, 480M
|__ Port 2: Dev 3, If 0, Class=Vendor Specific Class, Driver=rt2800usb, 480M
root@xxxx:~#

4.5 Boot self start configuration method

Add the program that needs to be started to /etc/rc.local.
root@xxxx:~# **vi /etc/rc.local**

#! /bin/sh -e
#
rc.local
#
This script is executed at the end of each multiuser runlevel.
Make sure that the script will "exit 0" on success or any other
value on error.
#
In order to enable or disable this script just change the execution
bits.
#
By default this script does nothing.
/home/sysuser/start.sh
exit 0
~
Note: You must ensure that rc.local is executable
root@xxxx:~# ll /etc/rc.local
-rwxr-xr-x 1 root root 306 Nov 2 13:37 /etc/rc.local
If convenience, through methods such as FTP will be used to save the rc. The local files uploaded to the system, it is possible that lost can execute permissions, this time through the chmod + x a/etc/rc. The local command add executable permissions.

4.6 Driver installation view lsmod

View current installed module information

```
root@xxxx:~# lsmod
Module                  Size  Used by
boardio                27695  0
biokernbase             5963  1 boardio
gpioinfo                5514  1
ipv6                  268782  12
option                 26392  0
usb_wwan                5240  1 option
ext4                  331096  0
jbd2                   55796  1 ext4
```

Installing modules

```
root@xxxx:~# insmod /home/sysuser/driver/boardio.ko
root@xxxx:~# lsmod
Module                  Size  Used by
boardio                27695  0
biokernbase             5963  1 boardio
gpioinfo                5514  1
ipv6                  268782  12
option                 26392  0
usb_wwan                5240  1 option
ext4                  331096  0
jbd2                   55796  1 ext4
root@xxxx:~#
```

Uninstall the installed module

```
root@xxxx:~# rmmod boardio
```

5 Programming development

5.1 Cross compilation Instructions

Due to equipment platform for ARM platform, develop machine mostly X86 platform, so the application needs to use cross-compilation cross-compilation toolchain, execution on the target.

For example:

```
arm-linux-gnueabihf-g++ -I.. /inc diread.cpp    -o diread -L .. /lib/
arm-linux-gnueabihf-gcc -I.. /inc    wdttest.c    -o wdttest -L .. /lib/
```

If the development platform is more, it is recommended to use CROSS_COMPILE environment variables to specify prefix cross-compilation toolchain. For example:

export CROSS_COMPILE=arm-linux-gnueabihf-

Makeflie file compiling parts as follows:

```
all:
    $(CROSS_COMPILE)gcc wdttest.c -Wall -o wdttest -lBoardResource -I. -L.
```

Also cross compiler by different versions, limits may be different, advice - L - L specifies the related parameters in the source file library files, to compatible with different versions of GCC restrictions.

Attached code description:

```
├ — — atscript          use microcom command for virtual serial port communication sample script
├── boardresouce Resources SDK and development samples
├── comtest serial port programming code sample
├── fastcgi_example fastcgi programming code sample
The sample ├ — — openVPN openVPN configuration parameters
├── readme.txt
├── sramutil SRAM read and write code samples
├── tcptest TCP programming code samples
└── udptest UDP programming code examples
```

5.2 On-board resources programming (BoardResource SDK)

5.2.1 Watchdog

a), the use of watchdog mainly has the following steps

- Initialize Watchdog (WDT_Init)
- Enable Watch dog (WDT_Enable)
- Feed the dog (WDT_Strobe)
- Disable guard dog (WDT_Disable)
- Release the watchdog resources (WDT_DeInit)

b)API introduction

➢ BR_RESULT **WDT_Init** (BR_HANDLE * handle)

Initialize the Watchdog.

This function must be called before any other watchdog functions.

Parameters

[out] handle Handle of the Watchdog.

Returns

result, BR_SUCCESS if successful.

➢ BR_RESULT **WDT_Enable** (BR_HANDLE handle,
unsigned int spanSeconds
)

Enable the Watchdog.

Parameters

[in] handle Handle of the Watchdog.

[in] spanSeconds time span of the Watchdog.range from 1 to 3600 seconds

Returns

result, BR_SUCCESS if successful.

➢ BR_RESULT **WDT_Strobe** (BR_HANDLE handle)

Strobe the Watchdog.

after enabling the Watchdog using WDT_Enable, the application must continuously call WDT_Strobe to trigger the Watchdog.

Parameters

[in] handle Handle of the Watchdog.

Returns

result, BR_SUCCESS if successful.

➢ BR_RESULT **WDT_Disable** (BR_HANDLE handle)

Disable the Watchdog.

Parameters

[in] handle Handle of the Watchdog.

Returns

result, BR_SUCCESS if successful.

➢ BR_RESULT **WDT_DeInit** (BR_HANDLE handle)

De-initialize the Watchdog.

Parameters

[in] handle Handle of the Watchdog.

Returns

result, BR_SUCCESS if successful.

c), code instance

```
/**********************************************************************
**********
 > File Name: wdttest.c
 **********************************************************************
********* /

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <syslog.h>
#include <string.h>
#include "board_resource.h"

#define TIMEOUT 10

int main(int argc, char *argv[])
{
        BR_HANDLE wdt_fd = 0;
        int timeout = TIMEOUT;
        BR_RESULT ret = BR_SUCCESS;
        //init handle
        printf("init handle\n");
        ret = WDT_Init(&wdt_fd);
        if (ret ! = BR_SUCCESS)
        {
                printf("open device fail[%d]\n", ret);
                return 0;
        }
        //enable watch dog
        ret = WDT_Enable(wdt_fd,timeout);
        if (ret ! = BR_SUCCESS)
        {
                printf("enable wdt fail[%d]\n", ret);
```

```
        return 0;
    }

    printf("press Ctrl+C in %d second,the wdt will reboot system\n",timeout);
    while(timeout--){
        //strobe dog
        ret = WDT_Strobe(wdt_fd);
        if (ret == BR_SUCCESS)
        {
            printf("strobe wdt success[%d]\n",timeout);
        }
        sleep(1);
    }

    sleep(5);

    //disable dog
    printf("disable wdt\n");
    ret = WDT_Disable(wdt_fd);
    if (ret ! = BR_SUCCESS)
    {
        printf("disable wdt fail[%d]\n", ret);
        return 0;
    }
    //uninit handle
    WDT_DeInit(wdt_fd);
    printf("test over\n");
    return 0;
}
```

Makefile:

```
all:
    $(CROSS_COMPILE)gcc wdttest.c -Wall -o wdttest -lBoardResource -I. -L.
```

5.2.2 PLED

a), programmable LED use mainly has the following steps

- Initialize LED(LED_Init)
- Turn on the lights (LED_On)
- Turn lights off (LED_On)
- Release resources (LED_DeInit)

b)API introduction

Available LED type, according to the panel label to correspond to LED.

LED_TYPE_RUN RUN LED

LED_TYPE_ERROR ERROR LED

LED_TYPE_PROGRAM PROG LED

LED_TYPE_P1 LED 1

LED_TYPE_P2 LED 2

LED_TYPE_P3 LED 3

LED_TYPE_P4 LED 4

- BR_RESULT **LED_Init** (LEDType type,
  BR_HANDLE * handle
)

Initialize the LED device.

This function must be called before any other LED functions.

Parameters

[in] type type of the LED device.

[out] handle Handle of the LED device.

Returns

result, BR_SUCCESS if successful.

- BR_RESULT **LED_On** (BR_HANDLE handle)

Light the LED.

Parameters

[in] handle Handle of the LED device.

Returns

result, BR_SUCCESS if successful.

- BR_RESULT **LED_Off** (BR_HANDLE handle)

Turn off the LED.

Parameters

[in] handle Handle of the LED device.

Returns

result, BR_SUCCESS if successful.

➢ BR_RESULT **LED_DeInit** (BR_HANDLE handle)

De-initialize the LED device.

Parameters

[in] handle Handle of the LED device.

Returns

result, BR_SUCCESS if successful.

c) Code examples

```
/*************************************************
****************************
 > File Name: ledtest.c
 ***********************************************
*************************/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include "board_resource.h"

int main(int argc, char *argv[])
{
    int fd =0;
    BR_RESULT ret = BR_SUCCESS;
    ret = LED_Init(LED_TYPE_RUN, &fd);
    if (ret ! = BR_SUCCESS)
    {
        printf("open LED_TYPE_RUN[P1] fail[%d]\n", ret);
        return -1;
    }
    LED_On(fd);
    sleep(1);
    LED_Off(fd);
    LED_DeInit(fd);

    return 0;
}
```

```
Makefile

all:
    $(CROSS_COMPILE)gcc ledtest.c -Wall -o ledtest -lBoardResource -I.  -L.
```

5.2.3 DIO(ECU1251 only)

a), DIO node use mainly has the following steps

- Initialize resources (DIO_Init)
- Get the DO count (Get_DOCount)
- Getting the DO value (Do_Read)
- Set the DO value (Do_Write)
- Get the number of DI (Get_DICount)
- Get the DI value (Di_Read)
- Release resource (DIO_DeInit)

b), code instance

```
/**********************************************************************
**********
 > File Name: diotest.c
 **********************************************************************
*********/

#include <stdio.h>
#include <unistd.h>
#include "board_resource.h"

int main(int argc, char *argv[])
{
        int fd;
        BR_RESULT ret = BR_SUCCESS;
        ret = DIO_Init(&fd);
        if (ret ! = BR_SUCCESS)
                printf("open device fail[%d]\n", ret);

        int DI = Get_DICount();
        int DO = Get_DOCount();
        printf("DI:%d,DO:%d\n",DI,DO);
        unsigned int value;

        int i = 0;
```

```
        while(1)
        {
                if(DI)
                {
                        Di_Read(fd,0,DI,&value);
                        printf("DI:%08x\n",value);
                }
                if(DO)
                {
                        for(i=0; i< DO; i++)
                        {
                                Do_Read(fd,i,1,&value);
                                value = ~value;
                                Do_Write(fd,i,1,value);
                                Do_Read(fd,i,1,&value);
                                printf("DO[%d]:%04x\n",i,value);
                                sleep(1);
                        }
                }

                sleep(3);
        }

        return 0;
}
```

Makefile

```
all:
        $(CROSS_COMPILE)gcc diotest.c -Wall -o diotest -lBoardResource -I.    -L.
```

5.3 IO resources (ADAM3600 ECU1370)

1， To view module information:

root@xxxx:~# mdlsearch

IO Module: 0, ADAM-3600

IO Module: 0, ADAM-3600,ver:01010183,ai: 8,ao: 0,di: 8,do: 4

2. Programming code:

diread.cpp

```
/*****************************************************************************
*************************
*
* Linux Example:
*       diread.cpp
*
* Example Category:
*       DI
*
* Description:
*       This example demonstrates how to use DI function.
*
* Instructions for Running:
*       1. Set the 'deviceNumber' for opening the device.
*       2. Set the 'startChannel' as the first channel for scan analog samples
*       3. Set the 'channelCount' to decide how many sequential channels to scan analog samples.
*
* I/O Connections Overview:
*       Please refer to your hardware reference manual.
*
*****************************************************************************
************************/
#include <stdlib.h>
#include <stdio.h>
#include "compatibility.h"
#include "bdaqcl.h"
using namespace Automation::BDaq;
//----------------------------------------------------------------------------
// Configure the following three parameters before running the example
//----------------------------------------------------------------------------
#define             deviceNumber        0
#define         channelCountMax 8
#define         MAX_MODULE_COUNT  5



inline void waitAnyKey()
{
    do{SLEEP(1); } while(! kbhit());
}
int main(int argc, char* argv[])
{
      ErrorCode             ret = Success;
```

```
BDaqDevice *device = NULL;
BDaqDio  *dio = NULL;
long        moduleNumber = 0;
long            startChannel = 0;
long        channelCount = 8;

//Open device
ret = BDaqDevice::Open(deviceNumber, ModeWrite, device);

do
{
        //Get dio module
        ret = device->GetModule(0, dio);
        CHK_RESULT(ret);
        long rngCode[channelCountMax] = { DI_NORMAL_MODE };

        ret = BDaqDevice::Open(deviceNumber, ModeWrite, device);
        CHK_RESULT(ret);

        ret     =     dio->DiSetFuncCode(moduleNumber,     startChannel,     channelCount,
&rngCode[startChannel]);
        CHK_RESULT(ret);
        memset(rngCode, 0, channelCountMax*sizeof(long));
        SLEEP(1);
        ret     =     dio->DiGetFuncCode(moduleNumber,     startChannel,     channelCount,
&rngCode[startChannel]);
        CHK_RESULT(ret);
        for (long i = startChannel; i < startChannel + channelCount; ++i)
        {
            printf("Channel %ld function code: %lx\n", i,rngCode[i - startChannel]);
        }

        printf("Acquisition is in progress, any key to quit! \n\n");
        BYTE            dioData[ 1 ] = {0xff};

        do
        {
            //Read di value
            ret = dio->DiRead(moduleNumber,startChannel, channelCount, dioData);
            CHK_RESULT(ret);
            printf("dio value: %2x\n", dioData[0]);
```

```
                SLEEP(1);
            } while(! kbhit());
        }while(false);

        //Close device
        if(device ! = NULL)
        {
            device->Close();
        }
        // If something wrong in this execution, print the error code on screen for tracking.
        if(BioFailed(ret))
        {
            printf("Some error occurred. And the last error code is Ox%X.\n", ret);
            waitAnyKey(); // wait any key to quit!
        }
        return 0;
}
```

Makefile:

```
    $(CROSS_COMPILE)g++ -I.. /inc –L .. /lib/ diread.cpp    -o diread
```

5.4 A serial port programming,

5.4.1 The basic steps of programming

Serial port operation in Linux is mainly achieved by setting struct termios. The steps are generally as follows:

a) Open a device node such as /dev/ttyAP0
b) Get struct node termios information, set the baud rate of serial port parameters, such as, and then save the struct termios information.
c) Directly call the read and write functions to read and write data
d) Shut down the device node.

5.4.2 Parameter Configuration method

a) Open the device node
   fd = open(Dev, O_RDWR | O_NOCTTY);
b) Set baud rate

```
cfsetispeed(&opt, B9600);
cfsetospeed(&opt, B9600);
```

c) Set data bits

```
opt.c_cflag &= ~CSIZE;
        switch ( databits )
        {
        case 5:
                opt.c_cflag |= CS5;
                break;
        case 6:
                opt.c_cflag |= CS6;
                break;
        case 7:
                opt.c_cflag |= CS7;
                break;
        case 8:
                opt.c_cflag |= CS8;
                break;
        default:
                printf( "Unsupported data size\n" );
                return -1;
        }
```

d) Set the stop bit

```
switch (stopbits)
        {
        case 1:
                opt.c_cflag &= ~CSTOPB;
                break;

        case 2:
                opt.c_cflag |= CSTOPB;
                break;

        default:
                printf("Unsupported stop bits\n");
                return -1;
        }
```

e) Set the parity bit

```
switch (parity)
        {
        case 'n':
        case 'N':
                opt.c_cflag &= ~PARENB;
```

```
            opt.c_iflag &= ~INPCK;
            break;

        case 'o':
        case 'O':
            opt.c_cflag |= (PARODD | PARENB);
            opt.c_iflag |= INPCK;
            break;

        case 'e':
        case 'E':
            opt.c_cflag |= PARENB;
            opt.c_cflag &= ~PARODD;
            opt.c_iflag |= INPCK;
            break;

        default:
            printf("Unsupported parity\n");
            return -1;
        }
```

Makefile

```
all:
    $(CROSS_COMPILE)gcc comtest.c -Wall -o comtest
```

5.4.3 Details of other parameters

In general, one of the most basic set serial port including baud rate Settings, parity and stop bits. Serial port Settings is mainly set struct termios structure members of the value, as shown below:

```
struct termio
{
    unsigned short c_iflag; / * input mode marked * /
    unsigned short c_oflag; Mark * / / * output mode
    unsigned short c_cflag; / * control mode marked * /
    unsigned short c_lflag; Mark * / / * local mode
    unsigned char c_line; /* line discipline */
    unsigned char c_cc[NCC]; /* control characters */
};
```

In this structure is the most important in c_cflag, through its assignment, the user can set the baud rate, character size, data bits, stop bits, parity bit and hardware flow control. A sign of another c_iflag and c_cc is also more commonly used. In the main to detail the three members.

c_cflag supports constant names

A bitmask CBAUD baud rate

B0 0 baud rate (give up DTR)

B1800 1800 baud rate

B2400 2400 baud rate

B4800 4800 baud rate

B9600 9600 baud rate

B19200 19200 baud rate

B38400 38400 baud rate

B57600 57600 baud rate

B115200 115200 baud rate

EXTA external clock rate

EXTB external clock rate

Bitmask for CSIZE data bits

CS5 5 data bits

CS6 6 data bits

CS7 seven data bits

CS8 gives eight data bits

CSTOPB 2 stop bits (1 stop bit if not set)

CREAD receive enable

The PARENB check bit enables

PARODD uses parity without parity

Hang when HUPCL finally closes (ditches DTR)

CLOCAL Local connection (does not change port owner)

LOBLK piece of job control output

Can CNET_CTSRTS hardware flow control

In the name of the constant c_iflag support

INPCK parity can make

IGNPAR ignore parity error

PARMRK parity error mask

ISTRIP remove the parity bit

Start IXON export hardware flow control

IXOFF starts ingress software flow control

IXANY allows the character to restart the flow control

IGNBRK ignore interruption

BRKINT SIGINT signal when the interrupt occurs

INLCR map the NL to CR

IGNCR ignores CR

ICRNL maps CR to NL

Situation is mapped to a low IUCLC will be high
IMAXBEL reply ECHO when the input is too long

Names of constants supported by **c_cc**
VINTR interrupt control, corresponding key is CTRL+C
VQUIT exit the operation, the corresponding key CRTL + Z
VERASE delete operation, the corresponding key to Backspace (BS)
VKILL delete rows, the corresponding key CTRL + U
VEOF is located in the file at the end, the corresponding key CTRL + D
VEOL is at the end of the line and the key is Carriage return (CR).
VEOL2 is located in the second Line of the tail, the corresponding key for the Line feed (LF)
VMIN specifies the minimum number of characters read
VTIME specifies read the waiting time of each character

A serial port control function
tcgetattr gets attributes (termios structure)
Tcsetattr set properties (termios structure)
Cfgetispeed input speed
Cfgetospeed output speed
Cfsetispeed set the input speed
Cfsetospeed set the output speed
Tcdrain waiting all output is transmitted
Tcflow hang transmitted or received
Tcflush clearing the pending input and/or output
Tcsendbreak BREAK character

5.5 Network programming

5.5.1 TCP communication

5.5.1.1 TCP client

Based on the TCP (connection-oriented) socket programming, divided into client and server.
The client process is as follows:
(1) create a socket (socket)
(2) send a connection request to the server (connect)
(3) and server communicate (send/recv)
(4) close the socket

```
/*************************************************************************
**********
 > File Name: tcpc.c
 *************************************************************************
*********/
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <fcntl.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <arpa/inet.h>

int main(int argc, char **argv)
{
        int index = 0;
        char buf[1024];
        int sockfd;
        struct sockaddr_in dest_addr;
        if (argc ! = 3)
        {
                printf("useage:./tcpc ipaddress port\n ");
                printf("\teg:./tcpc 127.0.0.5555 \n");
                return -1;
        }
        int destport = atoi(argv[2]);
        if (-1 == (sockfd = socket(AF_INET, SOCK_STREAM, 0)))
        {
                perror("error in create socket\n");
                exit(0);
        }
        memset(&dest_addr,0,sizeof(dest_addr));
        dest_addr.sin_family = AF_INET;
        dest_addr.sin_port = htons(destport);
        dest_addr.sin_addr.s_addr = inet_addr(argv[1]);
        //connect
        if (-1 == connect(sockfd, (struct sockaddr*) &dest_addr,
                                sizeof(struct sockaddr)))
        {
                perror("connect error\n");
```

```
        exit(0);
    }


    while (1)
    {
        sprintf(buf, "%s %d", "tcp send data",index++);
        int n_send_len;
        n_send_len = send(sockfd, buf, strlen(buf), MSG_NOSIGNAL);

        if(n_send_len < 0)
        {
            perror("socket send");
            break;
        }
        printf("send:[%d]%s\n", n_send_len, buf);

        int nread = recv(sockfd, buf, sizeof(buf), 0);
        if (nread > 0)
        {
            printf("receive:[%d]%s\n", nread, buf);
        }
        if (nread < 0)
        {
            break;
        }
        sleep(1);
    }
    printf("exit program\n");
    shutdown(sockfd, 0);
    close(sockfd);
    return 0;
}
```

Makefile

```
all:
    $(CROSS_COMPILE)gcc tcpc.c -Wall -o tcpc
```

5.5.1.2TCP server

The process on the server side is as follows:

Creating a socket

(2) bind the socket to a local address and port (bind)

Put the socket in listen mode to prepare it for client requests

4.Wait for client requests to arrive. When the request arrives, accept the connection request and return a new socket (accept) corresponding to this connection.

Communicate with the client using the returned socket (send/recv)

(6) Return and wait for another client request.

Close the socket.

```
/*************************************************************************
**********
  > File Name: tcps.c
 *************************************************************************
*********/
#include <sys/socket.h>
#include <unistd.h>// for close function
#include <string.h> // for bzero function
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <arpa/inet.h>


#define SERV_PORT 5555
#define BACKLOG 10 //the counts of connect can keep in wait queen
#define MAXBUFSIZE 200


int main(int argc, char **argv)
{
        char buf[MAXBUFSIZE]; //receive buf
        int sockfd, sockfd_client = 0;
        socklen_t sin_size;
        struct sockaddr_in serv_addr, client_sockaddr; //server ip info
        int serverport;
        if (argc == 2)
        {
                serverport = atoi(argv[1]);
        }
        else
```

```
        {
                serverport = SERV_PORT;
        }
        if (-1 == (sockfd = socket(AF_INET, SOCK_STREAM, 0)))
        {
                perror("error in create socket\n");
                exit(0);
        }
        //set the sockaddr_in struct
        memset(&serv_addr,0,sizeof(serv_addr));
        serv_addr.sin_family = AF_INET;
        serv_addr.sin_port = htons(serverport); //server listening port
        serv_addr.sin_addr.s_addr = INADDR_ANY; //here is the specia in listening tcp connect
//bind , the ip and port information is aready in the sockaddr
        if (-1 == bind(sockfd, (struct sockaddr*) &serv_addr,
                                sizeof(struct sockaddr)))
        {
                perror("bind error\n");
                exit(0);
        }
        printf("bind seccessful\n");

        if (-1 == listen(sockfd, BACKLOG))
        {
                perror("lisenning");
                exit(1);
        }
        printf("the server is listenning... \n");
        //accept
        if (-1 == (sockfd_client = accept(sockfd,
                                (struct sockaddr*) &client_sockaddr, &sin_size)))
        {
                perror("accept");
                exit(1);
        }
        printf("accept connect from ip:%s
port:%d\n",inet_ntoa(client_sockaddr.sin_addr),ntohs(client_sockaddr.sin_port));
        while (1)
        {
                memset(buf,0,sizeof(buf));
                int recvbytes; //the number of bytes receive from socket
                recvbytes = recv(sockfd_client, buf, MAXBUFSIZE, 0);
                if (-1 == recvbytes)
```

```
        {
            perror("receive");
            exit(1);
        }
        printf("%d bytes receive from connect:%s\n", recvbytes, buf);
        if(recvbytes > 0) {
            recvbytes = send(sockfd_client, buf, recvbytes, MSG_NOSIGNAL);
        }else{
            recvbytes = send(sockfd_client, "heartbeat", strlen("heartbeat"), MSG_NOSIGNAL);
        }
        if(recvbytes < 0)
            break;
    }
    printf("eixt program\n");
    shutdown(sockfd_client,0);
    close(sockfd_client);
    shutdown(sockfd,0);
    close(sockfd);
    return 0;
}
```

Makefile:

```
all:
    $(CROSS_COMPILE)gcc tcps.c -Wall -o tcps
```

5.5.2 UDP communication

5.5.2.1UDP client

Client: (sender)

1) Create a socket
2) Send data to the server (sendto)
3) Close the socket

```
/ *
  * File:      udpc.c
  * UDP client
  *
  * Main implementation: Send one text message per second
  * /
```

```
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<stdio.h>
#include<stdlib.h>
#include<errno.h>
#include<netdb.h>
#include<stdarg.h>
#include<string.h>

#define SERVER_PORT 5555
#define BUFFER_SIZE 1024

int main(int argc, char **argv)
{
        /* server address */
        struct sockaddr_in server_addr;

        if (argc ! = 3)
        {
                printf("useage:./udpc ipaddress port\n ");
                printf("\teg:./udpc 127.0.0.5555 \n");
                return -1;
        }

        int destport = atoi(argv[2]);

        bzero(&server_addr, sizeof(server_addr));
        server_addr.sin_family = AF_INET;
        server_addr.sin_addr.s_addr = inet_addr(argv[1]);
        server_addr.sin_port = htons(destport);

        /* Create socket */
        int client_socket_fd = socket(AF_INET, SOCK_DGRAM, 0);
        if (client_socket_fd < 0)
        {
                perror("Create Socket Failed:");
                exit(1);
        }

        int index=0;
```

```
    char buffer[BUFFER_SIZE];
    bzero(buffer, BUFFER_SIZE);

    while (1)
    {
        sprintf(buffer, "%s %d", "udp send data",index++);
        int n_send_len;
        n_send_len = sendto(client_socket_fd, buffer, strlen(buffer), 0,
                (struct sockaddr*) &server_addr, sizeof(server_addr));

        if(n_send_len < 0)
        {
            perror("socket send");
            break;
        }
        printf("send:[%d]%s\n", n_send_len, buffer);

        struct sockaddr_in client_addr;
        size_t client_addr_length = 0;
        int nread = recvfrom(client_socket_fd, buffer, BUFFER_SIZE, 0,
                        (struct sockaddr*) &client_addr, &client_addr_length);
        if ( nread == -1)
        {
            perror("Receive Data Failed:");
            exit(1);
        }
        printf("recv:[%d]%s\n", nread, buffer);
        sleep(1);
    }

    close(client_socket_fd);
    return 0;
}
```

Makefile:

```
all:
    $(CROSS_COMPILE)gcc udpc.c -Wall -o udpc
```

5.5.2.2UDP server

Server: (receiver)
1) Create the socket
2) bind the socket to a local address and port (bind)
3) communicate with the client using the returned socket (recvfrom).
4) close the socket

```
/*************************************************************************
**********
 > File Name: server.c
 *************************************************************************
*********/
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<stdio.h>
#include<stdlib.h>
#include<errno.h>
#include<netdb.h>
#include<stdarg.h>
#include<string.h>

#define SERVER_PORT 5555
#define BUFFER_SIZE 1024

int main(int argc, char **argv)
{
        /* Create UDP socket */
        struct sockaddr_in server_addr;
        int serverport;
        if (argc == 2)
        {
                serverport = atoi(argv[1]);
        }
        else
        {
                serverport = SERVER_PORT;
        }
```

```
bzero(&server_addr, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
server_addr.sin_port = htons(serverport);

/* Create socket */
int server_socket_fd = socket(AF_INET, SOCK_DGRAM, 0);
if (server_socket_fd == -1)
{
        perror("Create Socket Failed:");
        exit(1);
}

/* Binding socket interface */
if (-1 == (bind(server_socket_fd, (struct sockaddr*) &server_addr,
                        sizeof(server_addr))))
{
        perror("Server Bind Failed:");
        exit(1);
}
printf("bind port %u success\n",ntohs(server_addr.sin_port = htons(serverport)));

char buffer[BUFFER_SIZE];
/* data transfer */
while (1)
{
        /* Define an address to capture the client address */
        struct sockaddr_in client_addr;
        socklen_t client_addr_length = sizeof(client_addr);

        /* Receive data */
        bzero(buffer, BUFFER_SIZE);
        int nread = recvfrom(server_socket_fd, buffer, BUFFER_SIZE, 0,
                        (struct sockaddr*) &client_addr, &client_addr_length);
        if ( nread == -1)
        {
                perror("Receive Data Failed:");
                exit(1);
        }
        printf("from                                                                    ip:%s
port:%d,[%d]%s\n",inet_ntoa(client_addr.sin_addr),ntohs(client_addr.sin_port),nread,buffer);
        if (sendto(server_socket_fd, buffer, strlen(buffer), 0,
                        (struct sockaddr*) &client_addr, sizeof(client_addr)) < 0)
```

```
            {
                perror("Send Failed:");
                exit(1);
            }
        }
        close(server_socket_fd);
        return 0;
}
```

Makefile

```
all:
        $(CROSS_COMPILE)gcc udps.c -Wall -o udps
```

5.6 SRAM programming

5.6.1 sram write operations

The SRAM operation method is similar to the file operation, using lseek to locate the location to be operated, and then performing the read and write operation.
The write operation code is as follows:

```
int write_with_file(int address, char buff[], int len)
{
   int fd = open(devname, O_RDWR);
   int nret = 0;
   if (fd == -1)
{
       perror("open");
       return -1;
}

   nret = lseek(fd, address, SEEK_SET);
   if (nret < 0)
{
       close(fd);
       return 0;
}
   nret = write(fd, buff, len);
   close(fd);
   return nret;
}
```

5.6.2 sram read operation

The read operation code is as follows:

```
int read_with_file(int address, char buff[], int len)
{
   int fd = open(devname, O_RDWR);
   int nret = 0;
   if (fd == -1)
{
      perror("open");
      return -1;
}

   nret = lseek(fd, address, SEEK_SET);
   if (nret < 0)
{
      close(fd);
      return 0;
}
   nret = read(fd, buff, len);
   close(fd);
   return nret;
}
```