GOSUNCN 高新兴物联
Welink your smart

LTE Module

Linux 系统下 ECM 功能
指导手册

Version:V1.1
Date:2018-09-03

insight WeLink

Website: ww.gosuncnwelink.com
E-mail: welink@gosuncn.com

版本历史

版本	日期	备注
V1.0	2017-05-01	初始版本
V1.1	2018-09-05	完善文档

目录

1. Linux PC 系统 ECM 拨号操作步骤

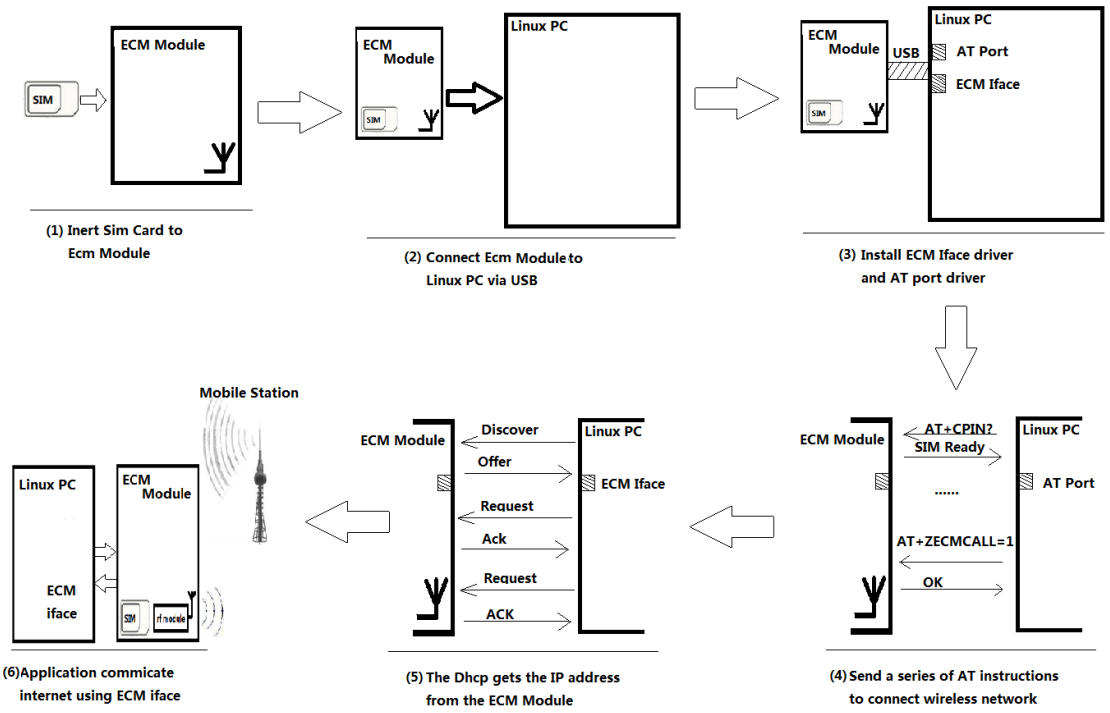


图 1-1 ECM 拨号接入 Internet 网用户操作步骤

如图 1-1 所示，ECM 拨号接入 internet 网络，用户需要 6 个操作步骤。分别是：①插入 SIM 卡；②连接 ECM Module；③安装 ECM 网卡驱动和 AT 口驱动；④发送拨号 AT 命令；⑤使能 DHCP 获取 IP 地址；⑥ECM 上网。

第一步：插入 SIM 卡

用户插入一张有效的 SIM 卡，不欠费，SIM 卡大小与 ECM Module 卡槽匹配。

第二步：连接 ECM Module

将 ECM Module 通过 USB 连接到 LINUX PC，通电。

第三步：安装 ECM 网卡驱动和 AT 口驱动

Linux PC 枚举出 ECM 网卡设备和 AT 端口设备后，需要先安装设备驱动。设备驱动的安装，用户可以阅读第 2 章“LINUX ECM 网卡驱动安装”和“LINUX PC AT 驱动安装”，参考文档操作。

第四步：发送拨号 AT 命令

AT 命令是应用于 ECM Module 与 Linux PC 应用之间的连接与通信的指令，AT 命令通过 AT 口发送。
AT 口是一种串行数据通讯接口，Linux PC 安装完 AT 口驱动之后，用户可以通过 ls /dev/ttyUSB*查看所

有的 ttyUSB 设备，AT 口默认的设备结点是/dev/ttyUSB1。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ls /dev/ttyUSB*
/dev/ttyUSB0  /dev/ttyUSB1  /dev/ttyUSB2
```

图 1-2 查看 ttyUSB 设备

AT 口可以发送拨号命令和断网命令，ECM Module 可以接受 AT 拨号命令后建立无线网络连接，也可以接受断网 AT 命令后断开无线网络连接。

拨号 AT 命令格式：AT+ZECMCALL=1，ECM Module 响应“OK”表示成功。

断网 AT 命令格式：AT+ZECMCALL=0，ECM Module 响应“OK”表示成功。

用户可以使用串口工具 Minicom 发送拨号 AT 命令或者断网 AT 命令，详细内容可以参考第 2.3 节“ECM 拨号 AT 命令”及“如何获取和使用 Minicom 开具”；

用户也可以运行高新兴物联发布的 ECM DEMO 程序进行拨号和断网操作，详细内容可以参考第 2.3 节“如何使用 ECM DEMO 代码”。

关于 AT 命令，用户若想了解什么是 AT 命令，有哪些 AT 命令与连网断网相关，用户可以参考第 2.3 节“ECM 拨号 AT 命令”和 “ECM 拨号相关 AT 命令”。

第五步：使能 DHCP 获取 IP 地址

该操作步骤目的是使能 DHCP 功能，如果 DHCP 功能默认是打开的，则没有必要进行该操作。

ECM 网卡是网络数据通讯接口，Linux PC 安装完网卡驱动之后，用户可以通过 ifconfig-s 查看所有的网络设备。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ifconfig -s
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
ens33      1500 0     21798      0      0 0         22950      0      0      0 BMRU
ens35u1i3  1500 0         6      0      0 0             5      0      0      0 BMU
lo        65536 0      3611      0      0 0          3611      0      0      0 LRU
```

图 1-3 ifconfig-s 查询命令结果图

ECM Module 插入 Linux PC 之后，多出了“ens35u1i3”网卡，那么该网卡就是 ECM Module 网卡。用户可以再次使用 ifconfig –a 查看网卡“ens35u1i3”的状态。

注*网卡名称根据不同的 Linux 系统不同的版本，加载的名称也是不相同的。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ifconfig -a ens35u1i3
ens35u1i3 Link encap:Ethernet  HWaddr 4a:e1:ed:0b:1e:ff
          inet addr:10.84.126.232  Bcast:10.84.126.239  Mask:255.255.255.240
          inet6 addr: fe80::39ec:138e:1fe4:9d20/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:44 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1750 (1.7 KB)  TX bytes:7562 (7.5 KB)
```

图 1-4 ifconfig-a 查询命令结果有 IP 地址示意图

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ifconfig -a ens35u1i3
ens35u1i3 Link encap:Ethernet  HWaddr 6a:e7:f4:8b:3a:0c
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:420 (420.0 B)  TX bytes:698 (698.0 B)
```

图 1-5 ifconfig-a 查询命令结果无 IP 地址示意图

如果出现图 1-4 所示的结果，“inet addr”字段已经有 IP 地址，说明 DHCP 服务已经启动，用户可以进一步使用 ping 命令查看是否可以 ping 一下外部主机，看看网络是否接通。

如果出现图 1-5 所示的结果，ECM 网卡未出现“inet addr”字段，说明 IP 地址未获取，需要查看 DHCP 服务是否启动。

以 ubuntu16.04 系统为例，DHCP 服务被封装到了 NetworkManager 的服务中，用户可以先使用“sudo service --status-all”列出所有 Linux PC 的服务。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo service --status-all
 [ + ]  acpid
 [ - ]  alsa-utils
 [ - ]  anacron
 [ + ]  apparmor
 [ + ]  apport
 [ + ]  avahi-daemon
 [ - ]  bluetooth
 [ - ]  bootmisc.sh
 [ - ]  brltty
 [ - ]  mountnfs-bootclean.sh
 [ - ]  mountnfs.sh
 [ - ]  network-manager
 [ + ]  networking
 [ + ]  ondemand
 [ - ]  plymouth
 [ - ]  plymouth-log
 [ - ]  pppd-dns
 [ + ]  procps
 [ - ]  rc.local
 [ + ]  resolvconf
 [ - ]  rsync
 [ + ]  rsyslog
 [ - ]  saned
 [ - ]  sendsigs
 [ + ]  speech-dispatcher
 [ + ]  thermald
 [ + ]  udev
```

图 1-6 service –status-all 命令列出所有服务结果图

如图 1-6 所示，对于 network-manager 服务，前边的减号表示此服务未运行。此时用户需要手动执行 sudo service NetworkManager start 开启 NetworkManager 服务，并再次查询服务 NetworkManager 的状态。

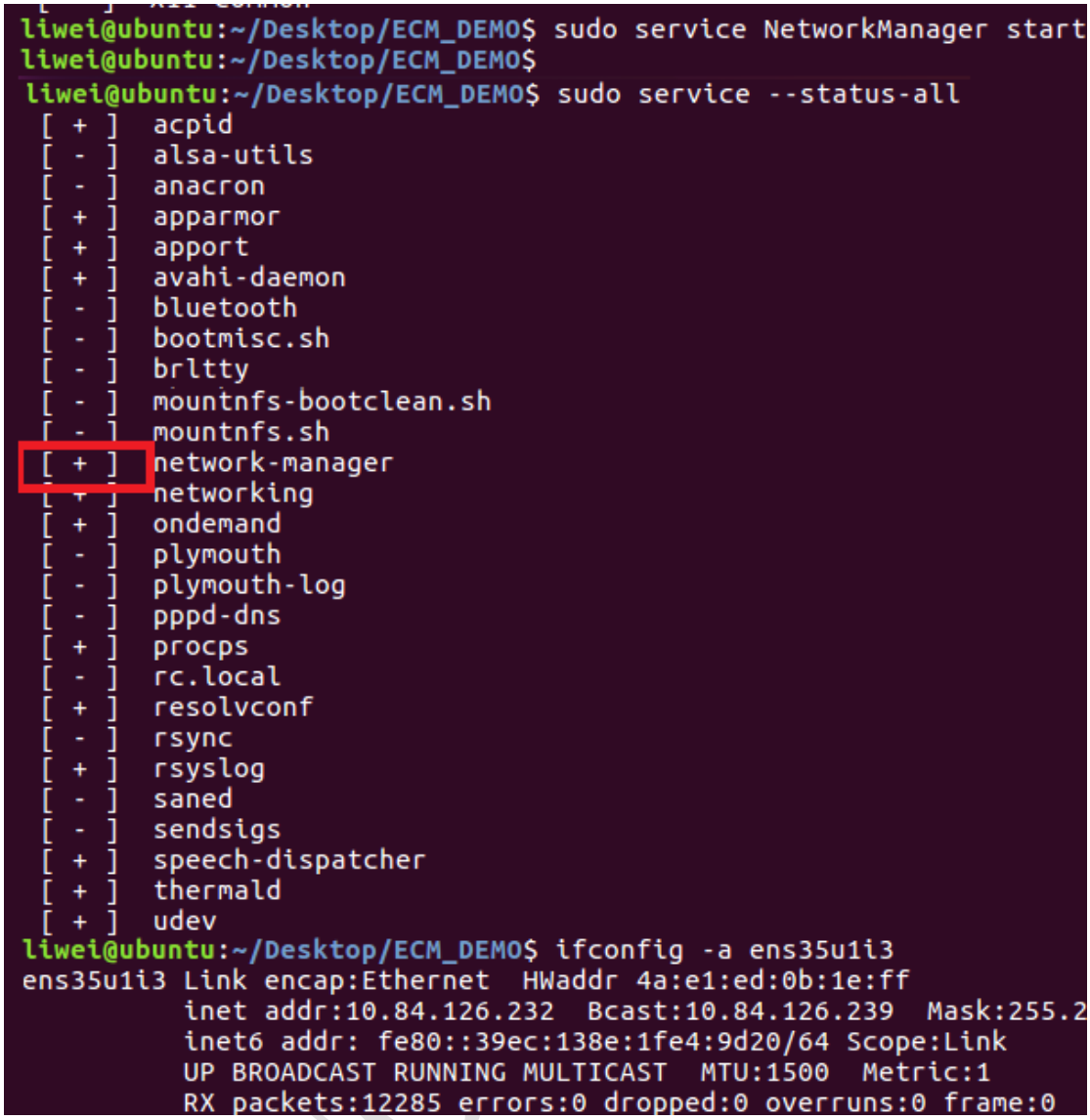


图 1-7 使能 NetworkManager 并查询服务状态示意图

如图 1-7 所示，开启 NetworkManger 服务后，network-manager 服务显示已经变成了加号，查询 ECM 网卡状态，也已经获取到了 IP 地址。

ubuntu12.04 系统可以输入 busybox 命令进行 IP 地址和 DNS 服务器地址的获取，如下图 1-8 所示。

不同版本的 LinuxOS，DHCP 存在的名称和配置方式有所不同，用户可以根据其 OS 版本支持情况定制化开启 DHCP 服务。感兴趣的用户也可以参考第 2.4 节对 DHCP 做进一步了解。

```
root@yhj-Lenovo:~/Desktop/dxy# busybox udhcpc -i usb0
udhcpc (v1.18.5) started
Setting IP address 0.0.0.0 on usb0
Sending discover...
Sending select for 10.208.161.173...
Lease of 10.208.161.173 obtained, lease time 43200
Setting IP address 10.208.161.173 on usb0
Deleting routers
SIOCDELRT: No such process
Adding router 10.208.161.174
Recreating /etc/resolv.conf
 Adding DNS server 61.134.1.6
 Adding DNS server 218.30.19.40
```

图 1-8 busybox 命令获取 IP 和 DNS

第六步：ECM 上网

判断网络是否可以使用，用户可以手动 ping 下外部主机，如果可以 ping 通，说明联网成功。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ping www.baidu.com
PING www.a.shifen.com (183.232.231.172) 56(84) bytes of data.
64 bytes from 183.232.231.172: icmp_seq=1 ttl=53 time=211 ms
64 bytes from 183.232.231.172: icmp_seq=2 ttl=53 time=112 ms
```

图 1-9 ping 通外网示意图

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ping www.baidu.com
```

图 1-10 无法 ping 通外网示意图

联网成功，ping 百度会有回包；如果联网有问题，ping 外部主机无法 ping 通，或者有找不到外部主机的提示。

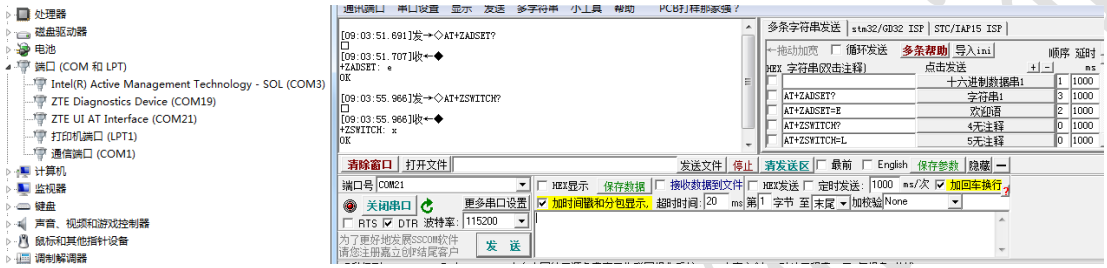
最后，联网成功，用户可以打开浏览器，或者运行 Linux 网络程序和外部主机交互数据。

2. LINUX 系统下使用 ECMCALL

2.1. Linux PC 系统驱动安装

ECM Module 网卡，实际上也是 USB CDC ECM，用于 ECM Module 和 Linux PC 之间传输以太网数据包。

注意：如果按照 2.1 的方法没有返回网卡信息，请确认模块是否处于 ECM 模式，方法为在 windows 系统中通过串口发送 AT+ZADSET?和 AT+ZSWITCH?，具体见 2.3 ECM 拨号相关 AT 命令



2.1.1. Linux PC ECM 网卡驱动调试

对于 Linux PC，ECM 网卡驱动一般是内核自带的，CDC ECM 设备结点是一个虚拟以太网卡，包含标准网卡需要的 MAC 地址和 IP 地址，用户不必在 Linux PC 系统上安装 ECM 网卡驱动。

不论是 Linux PC 系统，还是嵌入式 Linux 系统，用户都可以使用 dmesg 命令查看网卡加载的 log 信息：

```
[ 8096.482555] option 3-4:1.0: GSM modem (1-port) converter detected
[ 8096.482749] usb 3-4: GSM modem (1-port) converter now attached to ttyUSB0
[ 8096.482944] option 3-4:1.1: GSM modem (1-port) converter detected
[ 8096.483135] usb 3-4: GSM modem (1-port) converter now attached to ttyUSB1
[ 8096.483337] option 3-4:1.2: GSM modem (1-port) converter detected
[ 8096.483553] usb 3-4: GSM modem (1-port) converter now attached to ttyUSB2
[ 8096.487617] cdc_ether 3-4:1.3 usb0: register 'cdc_ether' at usb-0000:00:14.0-
4, CDC Ethernet Device, aa:5c:41:8c:7b:4b
[ 8240.386714] type=1400 audit(1535593969.689:127): apparmor="DENIED" operation=
"capable" profile="/usr/sbin/cupsd" pid=957 comm="cupsd" capability=36  capname=
"block_suspend"
```

图 2-1 dmesg 查看 cdc ecm 加载过程的 log 片段

可以看出，cdc_ether 加载后，该虚拟网卡的接口是 usb0，对应的 mac 地址是 aa:5c:41:8c:7b:4b。用户可以使用 ifconfig -a 查看该网卡的状态：

```
root@yhj-Lenovo:~# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 44:39:c4:91:f8:bb
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:5650 (5.6 KB)
          Interrupt:20 Memory:f7f00000-f7f20000

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 (0.0 B)  TX bytes:0 (0.0 B)

usb0      Link encap:Ethernet  HWaddr aa:5c:41:8c:7b:4b
          inet6 addr: fe80::a85c:41ff:fe8c:7b4b/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:9163 errors:1 dropped:0 overruns:0 frame:0
          TX packets:7032 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8900814 (8.9 MB)  TX bytes:732607 (732.6 KB)
```

图 2-2 ifconfig-a 查看所有网卡信息

2.1.2. Linux 串口(AT 口)驱动调试

ECM Module 插入 Linux PC 系统后，可以使用 lsusb 查看所有 USB 驱动的绑定状态。

```
root@yhj-Lenovo:~/Desktop# lsusb
Bus 001 Device 002: ID 8087:8008 Intel Corp.
Bus 002 Device 002: ID 8087:8000 Intel Corp.
Bus 003 Device 002: ID 04b3:3025 IBM Corp.
Bus 003 Device 017: ID 046d:c05a Logitech, Inc. Optical Mouse M90
Bus 003 Device 024: ID 19d2:1476 ZTE WCDMA Technologies MSM
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
```

图 2-3 lsusb 查看驱动加载图示

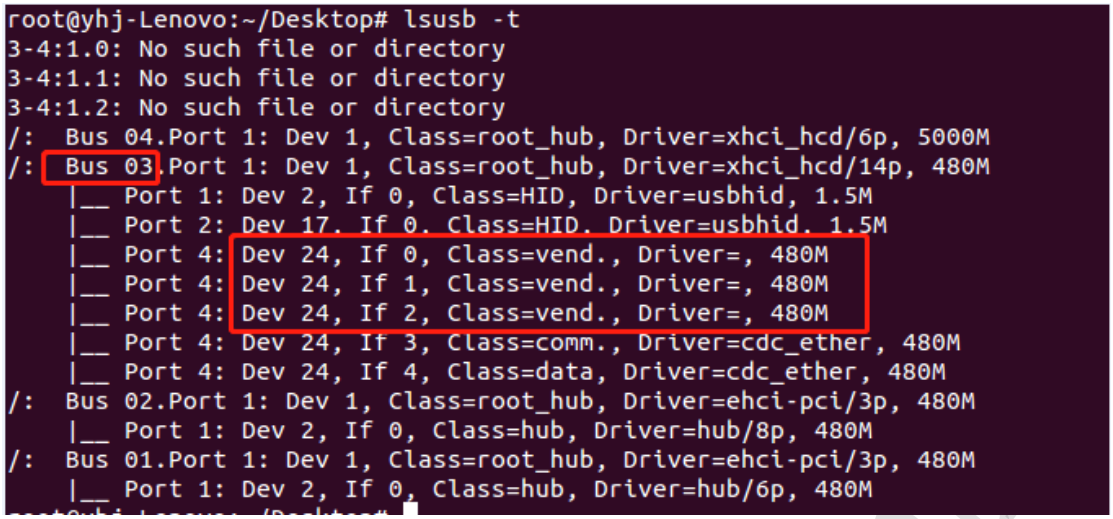


图 2-4 lsusb -t 查看驱动加载图示

以 ubuntu16.04 为例，通过 lsusb 可以发现 Bus=003,Device=024 对应的设备名是“ZTE WCDMA Technologies MSM”为 ECM Module 设备，其 PID=0x1476 VID=0x19d2。再使用 lsusb –t 查看驱动加载细节，发现 Driver 为空。If 0,If 1,If2 均未绑定设备驱动。

如果简单的加载调试 AT 驱动，用户可以通过以下步骤来绑定 If 0 ~ If 2 的驱动。

注意：如下操作需要在 root 权限下操作

①创建文件 load_option.sh，内容填写如下

```
#!/bin/bash
modprobe option
chmod 777 /sys/bus/usb-serial/drivers/options1/new_id
echo 0x19d2 0x1476 > /sys/bus/usb-serial/drivers/option1/new_id
```

②保存为 load_option.sh，并给 load_option.sh 添加执行权限

```
sudo chmod +x ./load_option.sh
```

③创建文件 unload_option.sh，内容填写如下

```
#!/bin/bash
rmmod option
```

④保存为 unload_option.sh，并给 unload_option.sh 添加执行权限

```
sudo chmod +x ./unload_option.sh
```

⑤执行 load_option.sh，查看状态

```
root@yhj-Lenovo:~/Desktop/dxy# sudo ./load_option.sh
root@yhj-Lenovo:~/Desktop/dxy# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/14p, 480M
    |__ Port 1: Dev 2, If 0, Class=HID, Driver=usbhid, 1.5M
    |__ Port 2: Dev 17, If 0, Class=HID, Driver=usbhid, 1.5M
    |__ Port 4: Dev 24, If 0, Class=vend., Driver=option, 480M
    |__ Port 4: Dev 24, If 1, Class=vend., Driver=option, 480M
    |__ Port 4: Dev 24, If 2, Class=vend., Driver=option, 480M
    |__ Port 4: Dev 24, If 3, Class=comm., Driver=cdc_ether, 480M
    |__ Port 4: Dev 24, If 4, Class=data, Driver=cdc_ether, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/3p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/8p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/3p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/6p, 480M
```

图 2-5 运行脚本绑定驱动并查看设备

⑥执行 unload_option.sh，卸载驱动，并查看状态

```
root@yhj-Lenovo:~/Desktop/dxy# sudo ./unload_option.sh
root@yhj-Lenovo:~/Desktop/dxy# lsusb -t
3-4:1.0: No such file or directory
3-4:1.1: No such file or directory
3-4:1.2: No such file or directory
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/14p, 480M
    |__ Port 1: Dev 2, If 0, Class=HID, Driver=usbhid, 1.5M
    |__ Port 2: Dev 17, If 0, Class=HID, Driver=usbhid, 1.5M
    |__ Port 4: Dev 24, If 0, Class=vend., Driver=, 480M
    |__ Port 4: Dev 24, If 1, Class=vend., Driver=, 480M
    |__ Port 4: Dev 24, If 2, Class=vend., Driver=, 480M
    |__ Port 4: Dev 24, If 3, Class=comm., Driver=cdc_ether, 480M
    |__ Port 4: Dev 24, If 4, Class=data, Driver=cdc_ether, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/3p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/8p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/3p, 480M
    |__ Port 1: Dev 2, If 0, Class=hub, Driver=hub/6p, 480M
```

图 2-6 运行脚本卸载驱动并查看设备

2.2. 嵌入式 Linux 驱动配置

2.2.1. 如何配置内核 make menuconfig

下载 linux 内核源文件，官网：www.kernel.org,可以下载新旧不同的版本。
本例以内核 Kernel V4.17.14 2018-08-09 [tarball]为列，先下载该 kernel 源码包。
下载源码包：linux-4.17.14.tar.xz

- **解压源码包：**

```
xz -d linux-4.17.14.tar.xz
tar -xvf linux-4.17.14.tar
```

```
linux-4.17.14/virt/kvm/arm/vgic/vgic.c
linux-4.17.14/virt/kvm/arm/vgic/vgic.h
linux-4.17.14/virt/kvm/async_pf.c
linux-4.17.14/virt/kvm/async_pf.h
linux-4.17.14/virt/kvm/coalesced_mmio.c
linux-4.17.14/virt/kvm/coalesced_mmio.h
linux-4.17.14/virt/kvm/eventfd.c
linux-4.17.14/virt/kvm/irqchip.c
linux-4.17.14/virt/kvm/kvm_main.c
linux-4.17.14/virt/kvm/vfio.c
linux-4.17.14/virt/kvm/vfio.h
linux-4.17.14/virt/lib/
linux-4.17.14/virt/lib/Kconfig
linux-4.17.14/virt/lib/Makefile
linux-4.17.14/virt/lib/irqbypass.c
liwei@ubuntu:~/kernel_source_package$ ls
linux-4.17.14  linux-4.17.14.tar
liwei@ubuntu:~/kernel_source_package$ cd linux-4.17.14/
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ ls
arch      CREDITS        firmware  ipc      lib          mm        scripts   usr
block     crypto         fs        Kbuild   LICENSES     net       security  virt
certs     Documentation  include   Kconfig  MAINTAINERS  README    sound
COPYING   drivers        init      kernel   Makefile     samples   tools
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$
```

- 安装 libncurses5-dev 库

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ sudo apt-get install libncur
ses5-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libtinfo-dev
Suggested packages:
  ncurses-doc
The following NEW packages will be installed:
  libncurses5-dev libtinfo-dev
0 upgraded, 2 newly installed, 0 to remove and 40 not upgraded.
Need to get 253 kB of archives.
After this operation, 1,190 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

- 安装/更新 sh 库

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ sudo apt-get install libshel
l-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  libshell-perl
0 upgraded, 1 newly installed, 0 to remove and 40 not upgraded.
Need to get 9,560 B of archives.
After this operation, 24.6 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial/universe i386 libshell-perl all
 0.73-1 [9,560 B]
Fetched 9,560 B in 1s (9,336 B/s)
Selecting previously unselected package libshell-perl.
(Reading database ... 179062 files and directories currently installed.)
Preparing to unpack .../libshell-perl_0.73-1_all.deb ...
Unpacking libshell-perl (0.73-1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up libshell-perl (0.73-1) ...
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$
```

- 安装/更新 bison 库

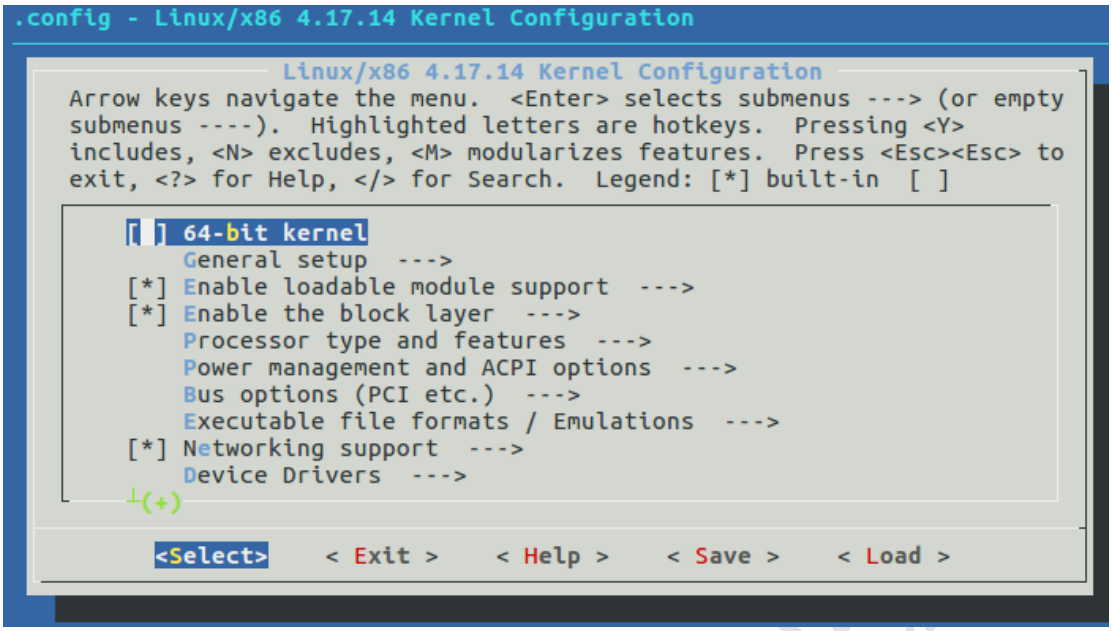
```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ sudo apt-get install libbiso
n-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bison libsigsegv2 m4
Suggested packages:
  bison-doc
The following NEW packages will be installed:
  bison libbison-dev libsigsegv2 m4
0 upgraded, 4 newly installed, 0 to remove and 40 not upgraded.
Need to get 803 kB of archives.
After this operation, 2,221 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

- 安装 flex 包

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ sudo apt-get install flex
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libfl-dev
The following NEW packages will be installed:
  flex libfl-dev
0 upgraded, 2 newly installed, 0 to remove and 47 not upgraded.
Need to get 296 kB of archives.
After this operation, 891 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us.archive.ubuntu.com/ubuntu xenial/main i386 libfl-dev i386 2.6.0-
11 [12.5 kB]
0% [1 libfl-dev 1,007 B/12.5 kB 8%]
```

- 执行 make menuconfig

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ make menuconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/basic/bin2c
  HOSTCC  scripts/kconfig/mconf.o
  YACC    scripts/kconfig/zconf.tab.c
  LEX     scripts/kconfig/zconf.lex.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
```



…

- 配置完成后，<SAVE>，查看生成的.config 文件：

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ vi .config
```

```
#
# Automatically generated file; DO NOT EDIT.
# Linux/x86 4.17.14 Kernel Configuration
#
# CONFIG_64BIT is not set
CONFIG_X86_32=y
CONFIG_X86=y
CONFIG_INSTRUCTION_DECODER=y
CONFIG_OUTPUT_FORMAT="elf32-i386"
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/i386_defconfig"
CONFIG_LOCKDEP_SUPPORT=y
CONFIG_STACKTRACE_SUPPORT=y
CONFIG_MMU=y
CONFIG_ARCH_MMAP_RND_BITS_MIN=8
CONFIG_ARCH_MMAP_RND_BITS_MAX=16
CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MIN=8
CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MAX=16
CONFIG_NEED_DMA_MAP_STATE=y
CONFIG_NEED_SG_DMA_LENGTH=y
CONFIG_GENERIC_ISA_DMA=y
CONFIG_GENERIC_BUG=y
CONFIG_GENERIC_HWEIGHT=y
CONFIG_ARCH_MAY_HAVE_PC_FDC=y
```

- 编译 Kernel：Make bzImage

```
liwei@ubuntu:~/kernel_source_package/linux-4.17.14$ make bzImage
  SYSTBL  arch/x86/include/generated/asm/syscalls_32.h
  HYPERCALLS arch/x86/include/generated/asm/xen-hypercalls.h
  SYSHDR  arch/x86/include/generated/uapi/asm/unistd_32.h
  SYSHDR  arch/x86/include/generated/uapi/asm/unistd_64.h
  SYSHDR  arch/x86/include/generated/uapi/asm/unistd_x32.h
  HOSTCC  scripts/basic/bin2c
  HOSTCC  arch/x86/tools/relocs_32.o
  HOSTCC  arch/x86/tools/relocs_64.o
```

```
  AS      arch/x86/lib/retpoline.o
  AS      arch/x86/lib/rwsem.o
  CC      arch/x86/lib/string_32.o
  CC      arch/x86/lib/strstr_32.o
  CC      arch/x86/lib/usercopy.o
  CC      arch/x86/lib/usercopy_32.o
  AR      arch/x86/lib/lib.a
  EXPORTS arch/x86/lib/lib-ksyms.o
  AR      arch/x86/lib/built-in.a
  AR      virt/lib/built-in.a
  AR      virt/built-in.a
  GEN     .version
  CHK     include/generated/compile.h
  AR      built-in.a
  LD      vmlinux.o
  MODPOST vmlinux.o
  KSYM    .tmp_kallsyms1.o
  KSYM    .tmp_kallsyms2.o
  LD      vmlinux
```

```
  HOSTCC  arch/x86/boot/tools/build
  BUILD   arch/x86/boot/bzImage
Setup is 17468 bytes (padded to 17920 bytes).
System is 7519 kB
CRC 72286652
Kernel: arch/x86/boot/bzImage is ready  (#1)
```

2.2.2. 串口驱动和网卡驱动添加

在内核中添加 USB 串口驱动和 USB 网卡驱动，可以选择将其直接编入内核，或者编译为模块待内核启动时加载，总之，要确保 Linux 内核启动完成后，这两个驱动是运行在内核当中的。

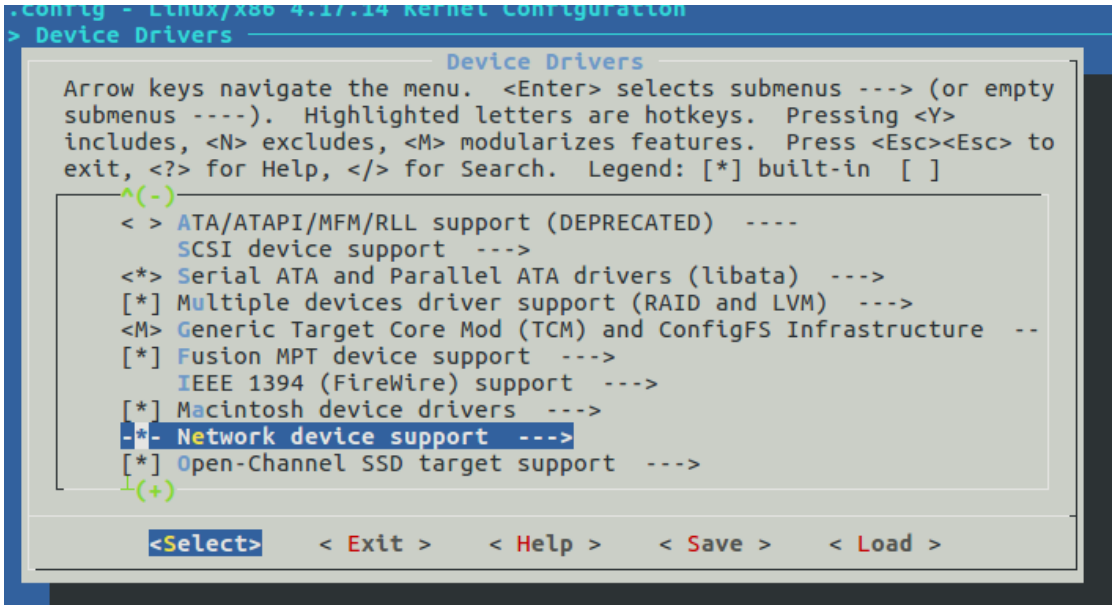
通常，配置内核是通过指令 make menuconfig，进入 kernel 目录下，执行该指令后：

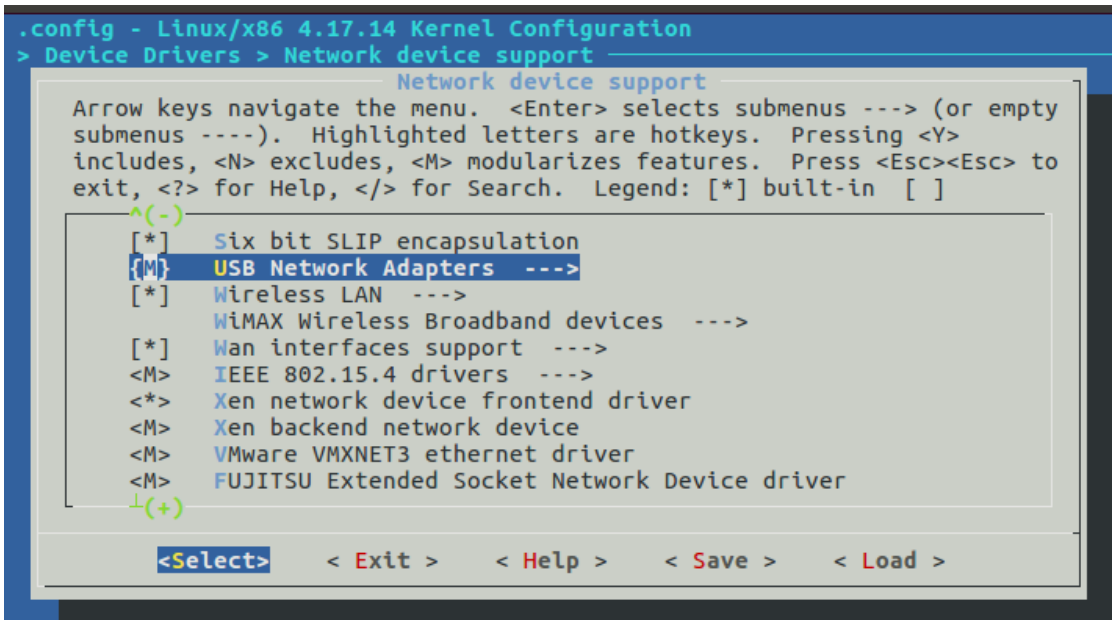
- 添加 USB 串口驱动：

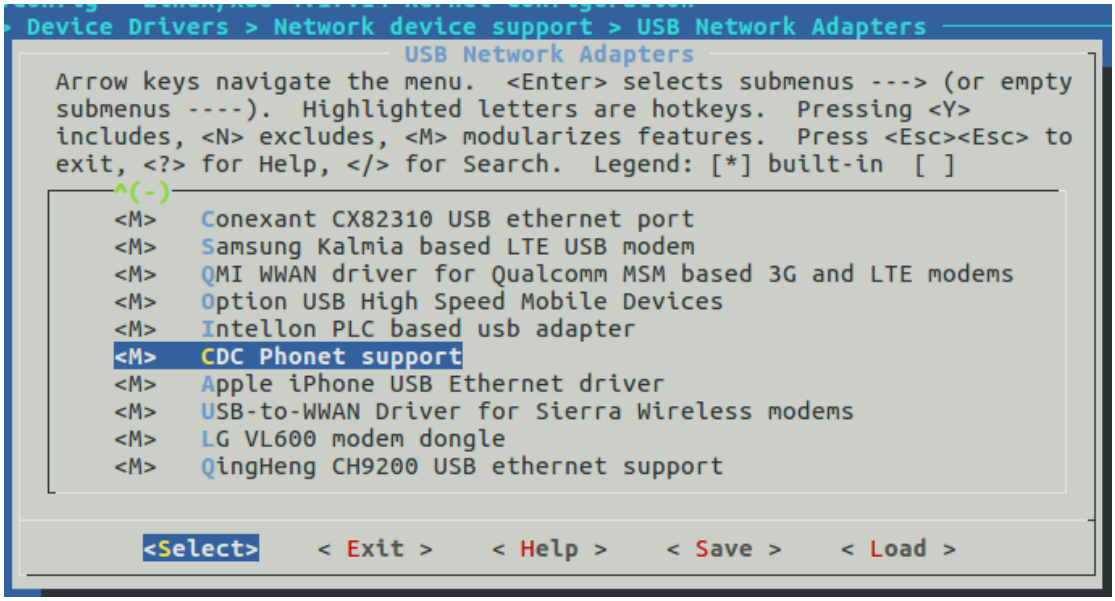
```
device drivers-->
  usb support-->
   usb serial converter support-->
  USB driver for GSM and CDMA modems
```

- 添加 USB 网卡驱动：

```
devices drivers-->
  Network device support-->
   usb Network Adapters-->
  Mulil-purpose USB Networking Framework
```



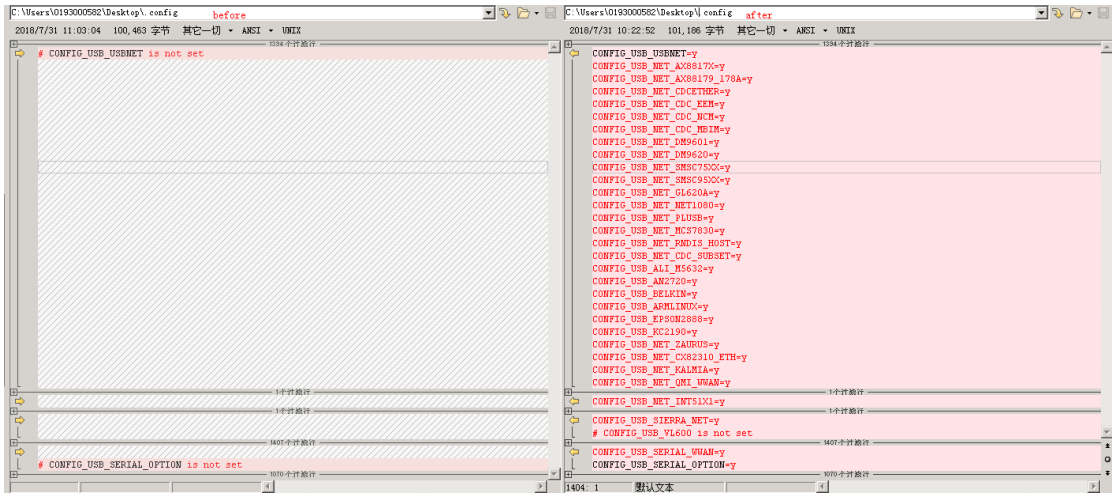




* 注：

①如果用户的内核结构与上面不一致，可能需要在其它的路径下面选择，总之，只要确保源文件中的 option.c 及其相关的部分（USB 串口驱动），cdc_ether.c 及其相关部分（USB 网卡驱动）参与编译即可

②如果您的内核不支持 make menuconfig 命令配置，可以直接修改 kernel 目录下的 .config 文件，手动添加配置项。



2.2.3. 内核源文件修改

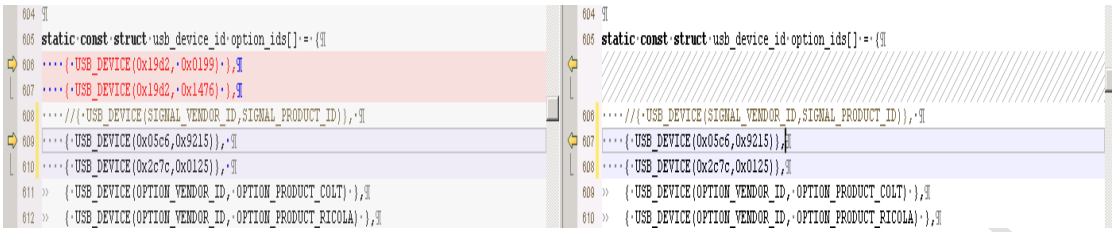
在内核驱动源文件中添加 GOSUNCN 模块相关的信息：USB 网卡驱动可以自动识别到模块，因此，其对应的 cdc_ether.c 文件中不需要添加任何内容。但是 USB 串口驱动不能自动识别，必须要添加 GOSUNCN 模块的设备信息到源文件 option.c 中。

文件路径：/kernel/drivers/usb/serial/option.c

添加 USB 端口的 VID 和 PID 信息，见下面蓝色部分。这里 0x0199 为 ME3860 模块和 ME3760_V2 模块的 PID，0x1476 为 ME3620 模块的 PID，如果您使用的是其它模块，将其中的 PID 值更换为相应的值即可。

```
static const struct usb_device_id option_ids[] = {
{ USB_DEVICE(0x19d2, 0x0199) },
```

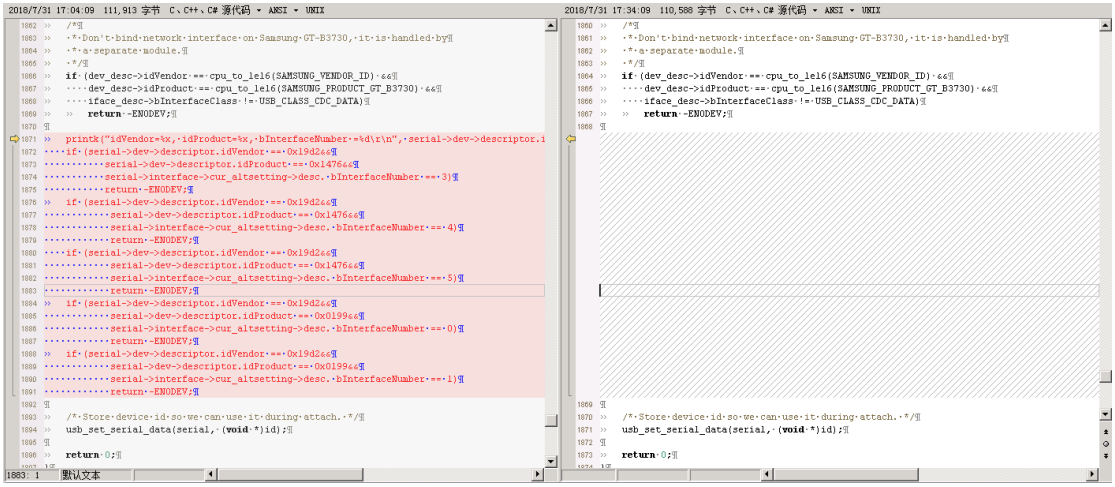
```
{ USB_DEVICE(0x19d2, 0x1476) },
……
}
```



添加黑名单信息，上面添加模块信息是只添加了设备的 VID 和 PID，没有附加任何额外的端口信息，这样会导致设备的网卡也被加载成为 USB 串口，下面提供的是一种类似于黑名单的方式，在 option_probe 函数中，将网卡对应的端口加入黑名单，防止 USB 网卡被加载成为 USB 串口。

对于 ME3860 和 ME3760_V2，其网卡对应的端口为 0 和 1，对于 ME3620，其网卡对应的端口为 3 和 4。请将以下代码添加到 option_probe 函数中

```
printk("idVendor=%x, idProduct=%x, bInterfaceNumber =%d\r\n",
serial->dev->descriptor.idVendor,
serial->dev->descriptor.idProduct,
serial->interface->cur_altsetting->desc.bInterfaceNumber);

if (serial->dev->descriptor.idVendor == 0x19d2
&&serial->dev->descriptor.idProduct == 0x1476
&&serial->interface->cur_altsetting->desc. bInterfaceNumber == 3)
      return -ENODEV;
if (serial->dev->descriptor.idVendor == 0x19d2
&&serial->dev->descriptor.idProduct == 0x1476
&&serial->interface->cur_altsetting->desc. bInterfaceNumber == 4)
      return -ENODEV;
if (serial->dev->descriptor.idVendor == 0x19d2
&&serial->dev->descriptor.idProduct == 0x1476
&&serial->interface->cur_altsetting->desc. bInterfaceNumber == 5)
      return -ENODEV;
if (serial->dev->descriptor.idVendor == 0x19d2
&&serial->dev->descriptor.idProduct == 0x0199
&&serial->interface->cur_altsetting->desc. bInterfaceNumber == 0)
      return -ENODEV;
if (serial->dev->descriptor.idVendor == 0x19d2
&&serial->dev->descriptor.idProduct == 0x0199
&&serial->interface->cur_altsetting->desc. bInterfaceNumber == 1)
      return -ENODEV;
```



* **注**：第一行的 printk 是为了方便调试而打印的，虽无实际效果，最好能带上。下面的几个 if 语句分别判断了需要加入黑名单的端口号，如果您使用的是除 ME3860，ME3760_V2 和 ME3620 之外的模块，上面 if 语句中的判断条件也要做相应修改。

2.3. 拨号命令

2.3.1. ECM 拨号 AT 命令

命令	功能	返回值	说明
AT+ZECMCALL=1	执行连网	+ZECMCALL: CONNECT OK	返回 CONNECT 表示连网成功
AT+ZECMCALL=0	执行断网	OK	返回 OK 表示断网成功

拨号 AT 命令（AT+ZECMCALL=1）会触发 ECM Module 向无线网络的数据连接，建立连接后，Linux PC 可以通过 ECM Iface 访问 internet。断网 AT 命令，发送给 ECM Module 后，Linux PC 无法访问 internet。

拨号 AT 命令的下发，依赖用户是否有插入 SIM 卡，SIM 卡如果有设置 PIN 码，需要先解锁 SIM 卡 PIN 之后，才能下发拨号 AT 命令，否则拨号 AT 命令执行不成功。

SIM 卡功能查询 AT 命令为 AT+CPIN，除此之外，拨号命令还依赖一些设备的状态，这些状态相关的 AT 命令，在第 2.3 节“ECM 拨号相关 AT 命令”中有详细的功能介绍,用户可以根据实际需求情况定制选择相应的 AT 命令。

2.3.2. 如何获取和使用 Minicom 工具

Minicom 工具是 Linux PC 系统选配的串口工具，只要用户打开 Minicom，发送命令“AT+ZECMCALL=1”，就可以完成拨号。

本节以 ubuntu16.04 系统为例，介绍 Minicom 工具如何从网络上获取和安装的方法。

注意：执行下面操作首先保证系统已联网

网络安装 Minicom 工具的步骤如下：

（1）执行命令 sudo apt-get update

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo apt-get update
Hit:1 http://us.archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://security.ubuntu.com/ubuntu xenial-security InRelease [107 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu xenial-security/main i386 DEP-11 Metada
a [67.7 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:6 http://security.ubuntu.com/ubuntu xenial-security/main DEP-11 64x64 Icons
[68.0 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe i386 Package
 [619 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe i386 DEP-11
etadata [249 kB]
Get:15 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe DEP-11 64x64
Icons [333 kB]
Get:16 http://us.archive.ubuntu.com/ubuntu xenial-updates/multiverse i386 DEP-1
 Metadata [7,144 B]
Get:17 http://us.archive.ubuntu.com/ubuntu xenial-backports/main i386 DEP-11 Me
adata [3,328 B]
Get:18 http://us.archive.ubuntu.com/ubuntu xenial-backports/universe i386 DEP-1
 Metadata [5,096 B]
Get:19 http://us.archive.ubuntu.com/ubuntu xenial-proposed/main i386 DEP-11 Met
data [14.7 kB]
Get:20 http://us.archive.ubuntu.com/ubuntu xenial-proposed/main DEP-11 64x64 Ic
ns [22.3 kB]
Get:21 http://us.archive.ubuntu.com/ubuntu xenial-proposed/universe i386 DEP-11
Metadata [212 B]
Fetched 3,514 kB in 14s (240 kB/s)
Reading package lists... Done
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-7 apt-get update 执行图示

（2）执行命令 sudo apt-get install minicom 安装 minicom 工具

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo apt-get install minicom
Reading package lists... Done
Building dependency tree
Reading state information... Done
minicom is already the newest version (2.7-1build1).
0 upgraded, 0 newly installed, 0 to remove and 40 not upgraded.
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-8 apt-get install minicom 执行图示

2.3.3. 如何使用 Minicom 工具

使用 Minicom 工具发送 AT 的使用方法请参照如下步骤：

（1）执行命令 sudo minicom -s 打开 Minicom 工具界面，打开后的界面如下图所示：

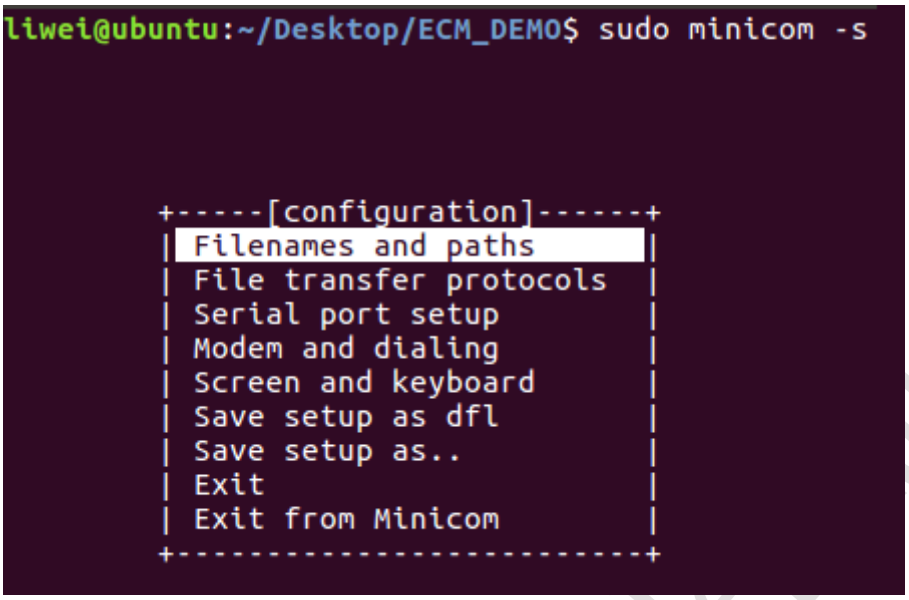


图 2-9 Minicom 界面图示

（2）移动光标到“Serial port setup”上。回车，出现串口 setting 界面。按 A 键，然后修改设备结点为 /dev/ttyUSB1，按回车键应用修改。之后再选择“Exit”退出 setting 界面，回到串口通讯界面。

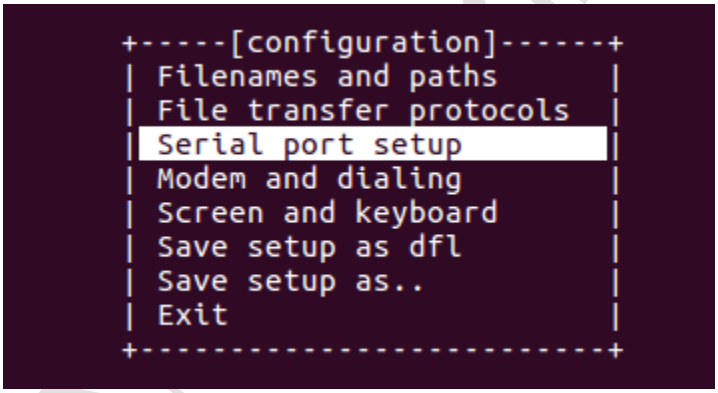


图 2-10 Minicom 界面图示

图 2-11 Minicom 设置界面图示

图 2-12 连接/dev/ttyUSB1 图示

（3）尝试输入简单 AT 命令，确认 AT 口是否接通。

出现通讯界面显示连接/dev/ttyUSB1 后，输入“ate”命令，此处打开 at 的反显功能，可以将输入的 AT 命令显示在 Minicom 上，成功则返回 OK。然后，再输入 at 命令，确认 at 口是否有数据返回，有返回则显示 OK。

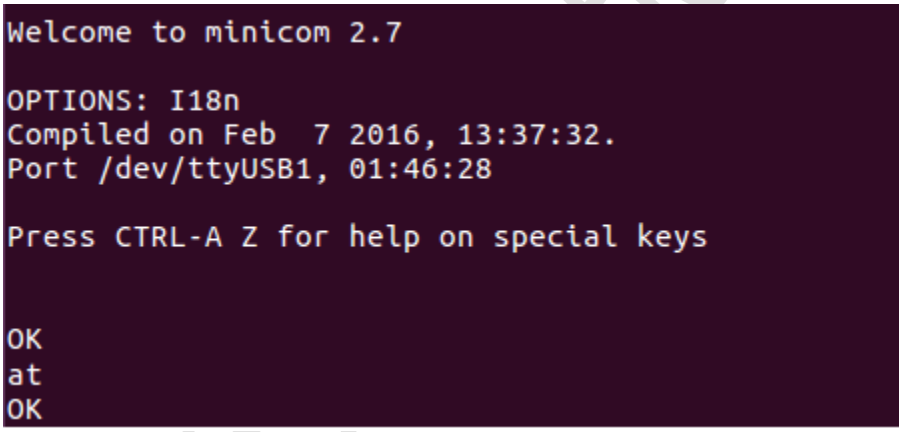


图 2-13 输入简单 AT 命令确认 AT 口已通

（4）查询 SIM 卡的状态，然后输入 ECM 拨号 AT 命令，连接网络。并且查询连网后 ECM 设备的 IP 地址，网关和 DNS。

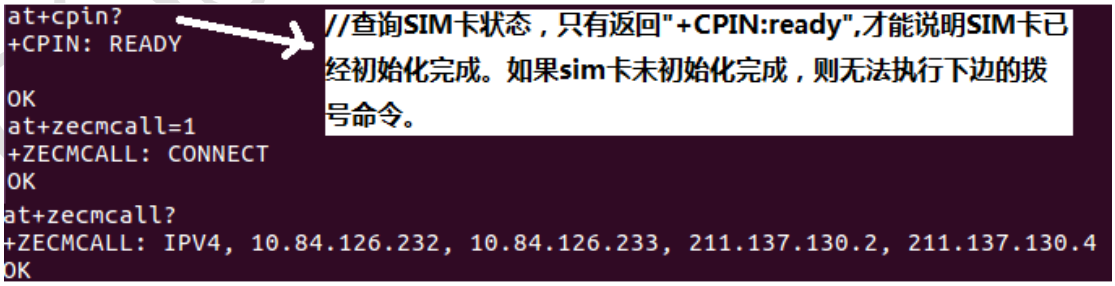


图 2-14 输入简单 AT 命令确认 AT 口已通

ECM 拨号 AT 命令及其相关命令，用户可以参考第 2.3 节 ECM 拨号相关 AT 命令。

（5）断开网络。用户使用完网络后，可以使用断网 AT 命令进行断网，如下图所示，断网后 IP 地址和 DNS 等信息将释放。

```
at+zecmcall=0
OK
at+zecmcall?
+ZECMCALL: IPV4, , , ,
OK
```

图 2-15 发送断网 AT 命令示意图

（6）关闭 Minicom 工具。

用户根据界面的提示，先按 CTRL+A，再按 Z 键，会出现功能选择界面。此时用户在功能选择界面再按下 Q 键，就会弹出退出提示界面；选择 YES 退出。

```
Welcome to minicom 2.7

OPTIONS: I18n
Compiled on Feb  7 2016, 13:37:32.
Port /dev/ttyUSB1, 18:13:12

Press CTRL-A Z for help on sp+-----------------------+
                             | Leave without reset? |
                             |   Yes       No       |
                             +-----------------------+
```

图 2-16 关闭 Minicom 图示

2.3.4. 如何使用 ECM DEMO 代码

ECM DEMO 代码，是高新兴物联为了让用户快速上手而开发的一套 ECM 拨号示例程序。

该示例程序包含源文件和 Makefile，用户可以使用 GCC 环境，按照以下步骤进行编译和测试。

（1）向高新兴物联获取 ECM_DEMO 代码，然后 COPY 到自己的 PC 系统，并查看 ECM_DEMO 代码。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ ls
ecm_demo_atctl.c   ecm_demo_mainauto.c  ecm_demo_msg.h     Makefile
ecm_demo_atctl.h   ecm_demo_main.c      ecm_demo_ttydev.c  README.txt
ecm_demo_config.h  ecm_demo_msg.c       ecm_demo_ttydev.h
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-17 查看 ECM_DEMO 目录

（2）确保 GCC 已经安装

用户可以使用 gcc –v 查看 gcc 版本，如果未安装，可以使用 sudo apt-get install gcc 进行 gcc 的安装。如果已经安装，则会显示 gcc 版本。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/i686-linux-gnu/5/lto-wrapper
Target: i686-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16
.04.10' --with-bugurl=file:///usr/share/doc/gcc-5/README.Bugs --enable-languages
=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-5 --en
able-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-ge
ttext --enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --e
nable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-de
fault-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --ena
ble-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin --enable-
java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-i
386/jre --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-i38
6 --with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-5-i386 --with-arch-dire
ctory=i386 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --ena
ble-targets=all --enable-multiarch --disable-werror --with-arch-32=i686 --with-m
ultilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-checkin
g=release --build=i686-linux-gnu --host=i686-linux-gnu --target=i686-linux-gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-18 查看 gcc 版本

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo apt-get install gcc
Reading package lists... Done
Building dependency tree
Reading state information... Done
gcc is already the newest version (4:5.3.1-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 40 not upgraded.
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-19 gcc 安装

（3）编译 ECM_DEMO

用户输入 sudo make Aslibrary，然后查看编译后文件，编译成功会多出来可执行文件 ECM_DEMO、ECM_DEMO_AUTO 以及静态链接库 libecmcall.a。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo make Aslibrary
rm -rf ECM_DEMO ECM_DEMO_AUTO *.so *.a *.o
gcc -c ecm_demo_atctl.c ecm_demo_msg.c ecm_demo_ttydev.c
ar -crvs libecmcall.a ecm_demo_atctl.o ecm_demo_msg.o ecm_demo_ttydev.o
a - ecm_demo_atctl.o
a - ecm_demo_msg.o
a - ecm_demo_ttydev.o
liwei@ubuntu:~/Desktop/ECM_DEMO$ ls
ECM_DEMO           ecm_demo_config.h    ecm_demo_msg.h     Makefile
ecm_demo_atctl.c   ecm_demo_mainauto.c  ecm_demo_ttydev.c  README.txt
ecm_demo_atctl.h   ecm_demo_main.c      ecm_demo_ttydev.h
ECM_DEMO_AUTO      ecm_demo_msg.c       libecmcall.a
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-20 ECM_DEMO 编译

（3）运行拨号程序(ECM_DEMO)或者自动拨号程序(ECM_DEMO_AUTO)进行拨号

ECM_DEMO 的运行，请参考下图所示，该图以拨号为例，用户需要输入端口参数/dev/ttyUSB1，APN 参数“3gnet”（使用了中国联通 SIM 卡），指令参数为拨号(up)。回车之后，执行该程序。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo ./ECM_DEMO -t up -p /dev/ttyUSB1 -a 3gnet
[2018_08_08_18:42:16] [info] ttyusb_recv_thxd up, tid(B7F3CB40)
[2018_08_08_18:42:17] [info] send at_cmd:ati

[2018_08_08_18:42:17] [info] recv:
Manufacturer: GOSUNCNWELINK
Model: ME3630-W
Revision: YDWLME3630C2CV1.0B01
IMEI: 866826030001797

OK
```

```
[2018_08_08_18:42:17] [info] send at_cmd:at+zswitch?

[2018_08_08_18:42:17] [info] recv:
+ZSWITCH: L
OK

[2018_08_08_18:42:17] [info] send at_cmd:at+zadset?

[2018_08_08_18:42:17] [info] recv:
+ZADSET: E
OK

[2018_08_08_18:42:18] [info] send at_cmd:at+cpin?

[2018_08_08_18:42:18] [info] recv:
+CPIN: READY

OK
```

```
[2018_08_08_18:42:18] [info] send at_cmd:at+cfun?

[2018_08_08_18:42:18] [info] recv:
+CFUN: 1

OK
```

```
[2018_08_08_18:42:18] [info] send at_cmd:at+zband?

[2018_08_08_18:42:18] [info] recv:
+ZBAND:
GW: 1800,900,1,5,8
CDMA: 0
TDS: 34,39
LTE: 1,3,5,8,38,39,40,41

OK
```

```
[2018_08_08_18:42:19] [info] send at_cmd:at+csq

[2018_08_08_18:42:19] [info] recv:
+csq: 14,99

OK
```

```
[2018_08_08_18:42:19] [info] send at_cmd:at+creg?

[2018_08_08_18:42:19] [info] recv:
+CREG: 0,1

OK
```

```
[2018_08_08_18:42:19] [info] send at_cmd:at+cgdcont=1,"IP","3gnet"

[2018_08_08_18:42:19] [info] recv:
OK
```

```
[2018_08_08_18:42:19] [info] send at_cmd:at+zsdt?

[2018_08_08_18:42:19] [info] recv:
+ZSDT: 1,0,3

OK
```

```
[2018_08_08_18:42:20] [info] send at_cmd:at+zecmcall=1

[2018_08_08_18:42:23] [info] recv:
+ZECMCALL: CONNECT
OK

[2018_08_08_18:42:23] [info] ttyusb_recv_thxd exit
[2018_08_08_18:42:23] ECM_DEMO success
liwei@ubuntu:~/Desktop/ECM_DEMO$
```

图 2-21 ECM_DEMO 拨号

从上图可以看出，ECM_DEMO 程序，传入 APN,PORT,拨号命令，最终通过发送 at+zswitch?，at+zadset?，at+cpin?，at+cfun?，at+zband?，at+csq？，at+creg?，at+cgdcont=，at+zsdt?，at+zecmcall=1 等 AT 命令实现与 ECM Module 的交互，达到连网的目的。

拨号程序 ECM_DEMO 运行完成之后会立刻退出，同时释放串口/dev/ttyUSB1。

除了联网，ECM_DEMO 程序还支持断网，配置 SIM 卡热插拔功能，配置 ECM Module 模式等功能。详细使用方法，用户可以参考 ECM_DEMO 代码文档《ECM_User_Guide_For_Embedded_Linux(GOSUNCN_ECM_CALL).pdf》。

ECM_DEMO_AUTO 的运行，请参考下图所示，用户只需要传入 APN 参数“3gnet”（此例使用了中国联通 SIM 卡，大部分模块对于中国国内运营商的 APN 可以自动适配，因此-a 3gnet 也可以省略）就可以。回车之后，执行该程序。

```
liwei@ubuntu:~/Desktop/ECM_DEMO$ sudo ./ECM_DEMO_AUTO -a 3gnet
[2018_08_08_19:03:41] ECM_DEMO_AUTO start...
[2018_08_08_19:03:42] [info] ttyusb_recv_thxd up, tid(B77C8B40)
[2018_08_08_19:03:43] [info] send at_cmd:ati

[2018_08_08_19:03:43] [info] recv:
Manufacturer: GOSUNCNWELINK
Model: ME3630-W
Revision: YDWLME3630C2CV1.0B01
IMEI: 866826030001797

OK
```

```
[2018_08_08_19:03:43] [info] send at_cmd:at+zswitch?

[2018_08_08_19:03:43] [info] recv:
+ZSWITCH: L
OK
```

```
[2018_08_08_19:03:43] [info] send at_cmd:at+zadset?

[2018_08_08_19:03:43] [info] recv:
+ZADSET: E
OK
```

```
[2018_08_08_19:03:44] [info] send at_cmd:at+cpin?

[2018_08_08_19:03:44] [info] recv:
+CPIN: READY

OK
```

```
[2018_08_08_19:03:44] [info] send at_cmd:at+cgsn

[2018_08_08_19:03:44] [info] recv:
866826030001797

OK
```

```
[2018_08_08_19:03:44] [info] send at_cmd:at+zgeticcid

[2018_08_08_19:03:44] [info] recv:
+ZGETICCID: 89860048261549204994

OK
```

```
[2018_08_08_19:03:45] [info] send at_cmd:at+cfun?

[2018_08_08_19:03:45] [info] recv:
+CFUN: 1

OK
```

```
[2018_08_08_19:03:45] [info] send at_cmd:at+zband?

[2018_08_08_19:03:45] [info] recv:
+ZBAND:
GW: 1800,900,1,5,8
CDMA: 0
TDS: 34,39
LTE: 1,3,5,8,38,39,40,41

OK
```

```
[2018_08_08_19:03:45] [info] send at_cmd:at+csq

[2018_08_08_19:03:45] [info] recv:
+csq: 14,99

OK
```

```
[2018_08_08_19:03:46] [info] send at_cmd:at+creg?

[2018_08_08_19:03:46] [info] recv:
+CREG: 0,1

OK
```

```
[2018_08_08_19:03:46] [info] send at_cmd:at+cgdcont=1,"IP","3gnet"

[2018_08_08_19:03:46] [info] recv:
OK
```

```
[2018_08_08_19:03:46] [info] send at_cmd:at+zsdt?

[2018_08_08_19:03:46] [info] recv:
+ZSDT: 1,0,3

OK
```

```
[2018_08_08_19:03:47] [info] send at_cmd:at+zecmcall=1

[2018_08_08_19:03:50] [info] recv:
+ZECMCALL: CONNECT
OK
```

```
[2018_08_08_19:03:50] [info] send at_cmd:at+zpdpcall?

[2018_08_08_19:03:50] [info] recv:
+ZPDPCALL: 1
OK
```

```
[2018_08_08_19:03:50] [info] send at_cmd:at+zpas?

[2018_08_08_19:03:50] [info] recv:
+ZPAS: "LTE","CS_PS","TDD"

OK
```

```
[2018_08_08_19:03:51] [info] send at_cmd:at+zcds?

[2018_08_08_19:03:51] [info] recv:
+ZCDS:38400,460,0,90F3,9040C03,-115,99,14,-11,56,460020848091849

OK
```

```
[2018_08_08_19:03:53] ECM_DEMO_AUTO Connected
[2018_08_08_19:03:53] ECM_DEMO_AUTO IMEI=[866826030001797]
[2018_08_08_19:03:53] ECM_DEMO_AUTO ICCID=[89860048261549204994]
[2018_08_08_19:03:53] ECM_DEMO_AUTO NetworkType=[LTE]
[2018_08_08_19:03:53] ECM_DEMO_AUTO SignalStrength=[115]
```

```
[2018_08_08_19:04:03] ECM_DEMO_AUTO Connected
[2018_08_08_19:04:03] ECM_DEMO_AUTO IMEI=[866826030001797]
[2018_08_08_19:04:03] ECM_DEMO_AUTO ICCID=[89860048261549204994]
[2018_08_08_19:04:03] ECM_DEMO_AUTO NetworkType=[LTE]
[2018_08_08_19:04:03] ECM_DEMO_AUTO SignalStrength=[115]
```

图 2-22 ECM_DEMO_AUTO 自动拨号

从上图可以看出，ECM_DEMO_AUTO 程序，传入 APN 参数，通过发送 at+zswitch?，at+zadset?，at+cpin?，at+cfun?，at+zband?，at+csq？，at+creg?，at+cgdcont=，at+zsdt?，at+zecmcall=1 等 AT 命令实现与 ECM Module 的交互，达到连网的目的。

自动拨号程序 ECM_DEMO_AUTO 与拨号程序 ECM_DEMO 相比。ECM_DEMO_AUTO 的目的是始终拨号，并一直占用串口/dev/ttyUSB1，并确保拨号状态，如果断网，则会再次发起拨号，直到收到退出指令。

ECM_DEMO_AUTO 自动拨号程序，在网络连接后，会以 15 秒的间隔去查询网络状态，如果发现断网，会再次重联，之后重新连接失败则会以 15 秒的间隔再次发起拨号，直到连接成功。

ECM_DEMO_AUTO 自动拨号的过程如下图所示：

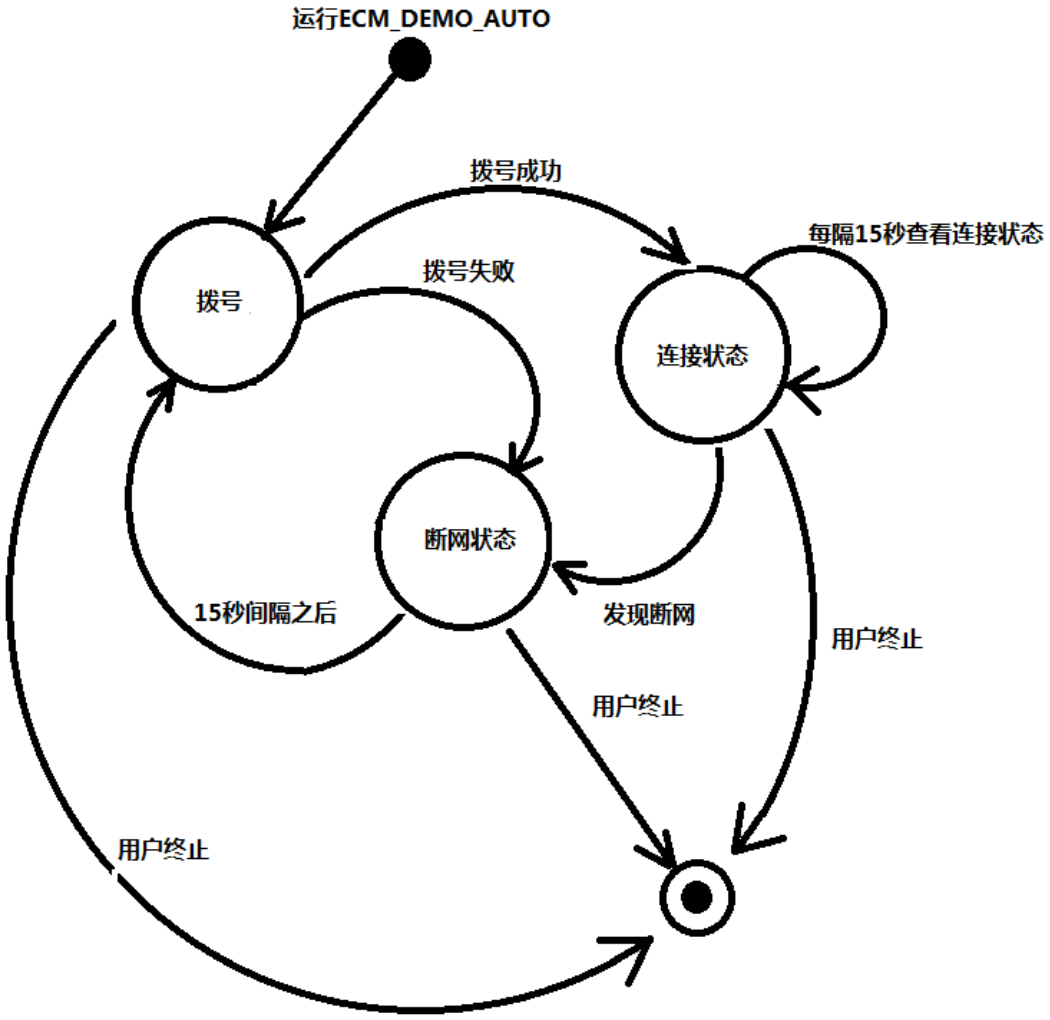


图 4.17 ECM_DEMO_AUTO 自动拨号过程的逻辑说明图示

关于自动拨号 ECM_DEMO_AUTO 程序，用户可以参考 ECM_DEMO 代码文档《ECM_User_Guide_For_Embedded_Linux(GOSUNCN_ECM_CALLVX.X.XBXX).pdf》做进一步研究。

用户也可以根据文档的描述对 ECM_DEMO 代码做定制化修改，然后将 ECM_DEMO 代码集成到自己的程序中。

2.3.5. ECM 拨号相关 AT 命令

ECM 拨号 AT 指令，相关的 AT 指令如下：

表 2-1 ECM 拨号相关 AT 命令

命令类型	命令格式	用途	与拨号命令的关系
设置参数	ATE	设置 AT 命令的回显	可选
查询版本	ATI	返回版本信息	**可选**
查询模式	AT+ZSWITCH?	查询是否在 ECM 网卡模式。 ECM 网卡模式返回值： +ZSWITCH:L	**可选，但需确认。** 只要设备工作在 L 模式，如果没有被修改，一直会工作在 L 模式，不必在拨

		OK 说明在 ECM 网卡模式	号时执行该命令。
查询模式	AT+ZADSET?	查询端口模式是否在多端口模式下。 ECM 网卡模式返回值： +ZADSET:E OK 说明在 ECM 多端口模式	**可选，但需确认。** 只要设备工作在 E 端口模式，如果没有被修改，一直会工作在 E 模式，不必在拨号时执行该命令。
查询状态	AT+CGSN	获取设备 IMEI 号	**可选**
查询状态	AT+CPIN?	获取 SIM 卡状态。 返回值： +CPIN:SIM Ready OK 说明 SIM 卡初始化完成。	**必选** 对于更换卡，或者插入有 PIN 码的 SIM 卡，该 AT 查询则返回： +CPIN:SIM PIN OK 需要解 PIN 后再使用。
查询状态	AT+ZGETICCID	获取 SIM 卡 ICCID	**可选**
查询状态	AT+CFUN?	获取设备上线状态	**可选**
设置参数	AT+CGDCONT=1,IP,PARA	设置拨号 APN	对于国内卡，可选
查询状态	AT+ZSDT?	查询是否开启支持 SIM 卡的热插拔功能。 ECM Module 默认支持热插拔功能，但是需要用户确认 SIM 卡卡座是否支持热插拔。有些卡座不支持探测 SIM 卡拔出和插入，所以无法支持该功能	**可选，但需确认。**
查询状态	AT+ZBAND?	查询频段信息	**可选**
查询状态	AT+CSQ	查询射频性能	**可选**
查询状态	AT+ZPAS?	查询当前网络制式	**可选**
查询状态	AT+CREQ?	查询是否注册成功	**可选**
查询状态	AT+ZCDS?	查询各网络制式信号强度	**可选**
拨号命令	AT+ZECMCALL=1	拨号	**必选**
断网命令	AT+ZECMCALL=0	断网	可选或必选

对于必选的 AT，如 AT+CPIN？ AT+ZECMCALL=1，对用户来讲是必选 AT。例如 AT+CPIN？ 返回“+CME ERROR”，可能用户根本没有插入 SIM 卡，未插入 SIM 卡也就没有必要进行拨号。

对于可选但需确认的 AT，只要用户确认过工作在该模式下，后续都无须再下发这些 AT 命令。

对于其它可选 AT，例如查询网络制式的 AT+ZPAS?；查询信号强度的 AT+ZCDS？；查询 IMEI 号 AT+CGSN 等，用户可以根据自己的需求自行选择和定制。

详细的 AT 指令说明请参考《高新兴物联 ME3630 模块 AT 指令手册》。

2.4. DHCP 服务说明

不同的 Linux 系统，开启 DHCP 服务的方式不同；不同的嵌入式 Linux 系统，开启 DHCP 服务的方式也不同。大概可以归纳成以下几种情况：

（1）Linux PC 系统开启 DHCP

对于 Linux PC，例如 Ubuntu，Debian 等系统，由于很多开源社区不同的在维护 OS 及安装包，DHCP 服务默认是自带的，或者可以通过安装包安装，网络更新的方式安装，所以开启 DHCP 功能，用户可以根据自己的系统不同，在 internet 网搜索相关的 DHCP 服务如何开启，就可以找到开启和安装方法。

例如 Debian 可以通过以下方法安装 DHCP：

root@debian:~# aptitude install dhcp3-server dhcp3-common

配置/etc/dhcp/dhcpd.conf，之后开启 DHCP，并且运行 DHCP 就可以为网卡获取 IP 地址。

（2）第三方发布的嵌入式 Linux 软件包。

这种软件包，可以是商用的，用户从第三方购买的软件；也可以是免费的，专用于某个领域的 Linux 软件包。

这种软件包的特点是定制化水平比较高，比如从第三方购买的软件，例如高通，MTK。这些软件本身自带一种编译方法和加载方法，所以 DHCP 软件的配置也是大不相同的，通常需要向第三方咨询开启方法。

（3）裁剪而来的嵌入式 Linux 系统。

对于裁剪而来的嵌入式 Linux 系统，例如目前市面上有用户通过下载 Linux Kernel 源码包和 BusyBox 工具源码包，以及 Linux 引导程序，拼装到一起的 Linux 系统。

此类嵌入式 Linux 系统，需要用户在编译配置 Kernel 和编译配置 busybox 时就要把 DHCP 选项给勾选上，因为裁剪后的系统甚至无法安装软件包或工具，因此需要编译 Kernel 和 busybox 时就要选配 DHCP 功能。

```
[*]Networking support -->
Networking support
Networking options -->
[*]Packet socket
[*]IP:kernel level autoconfiguration
[*] IP:DHCP support
[*]Network packet filtering framework(Netfilter)
```

```
<*> Packet socket
<*> Unix domain sockets
<*> Transformation user configuration interface
[ ] Transformation sub policy support (EXPERIMENTAL)
[ ] Transformation migrate database (EXPERIMENTAL)
[ ] Transformation statistics (EXPERIMENTAL)
< > PF_KEY sockets
[*] TCP/IP networking
[ ]   IP: multicasting
[ ]   IP: advanced router
[*]   IP: kernel level autoconfiguration
[*]     IP: DHCP support
[*]     IP: BOOTP support
[*]     IP: RARP support
```

图 2-23 内核编译选配 DHCP 图示

```
Networking Utilities
[*]udhcp server (udhcpd)
[*] dhcprelay
```

```
[*] Lease display utility (dumpleases)
[*]udhcp client (udhcpc)
[*] Verify that the offered address is free,using ARP ping
```

```
[*] dhcp server (udhcpd)
[*]   hcprelay
[*]   ease display utility (dumpleases)
[*]   ewrite the lease file at every new acknowledge
[ ]   elect IP address based on client MAC
(/var/lib/misc/udhcpd.leases) bsolute path to lease file
[*] dhcp client (udhcpc)
[*]   erify that the offered address is free, using ARP ping
[ ]   nable '-P port' option for udhcpd and udhcpc
(9)   M ximum verbosity level for udhcp applets (0..9)
[*]   upport for RFC3397 domain search (experimental)
[*]   upport for 802.1Q VLAN parameters
(/usr/share/udhcpc/default.script) Absolute path to config script
(80)  HCP options slack buffer size
```

图 2-24 BUSYBOX 编译选配 DHCP 图示

之后用户运行 udhcpcd 程序，就可以开启 dhcp 服务，去动态获取 IP 地址啦。

2.5. LINUX 系统下使用 ADB

（1）安装 android-tools-adb

```
sudo apt-get install android-tools-adbd
```

如果~/.android/adb_usb.ini 文件不存在,那么需要安装 android-tools-adbd，adb devices 才能显示。

（2）配置 adb_usb.ini

手动配置 adb_usb.ini 文件，lsusb 把 id 填进去，例如 ID 19D2:1476，那么把 19D2 写入 adb_usb.ini 即可。

```
root@zksoft-ThinkPad-X260:~/.android# cat adb_usb.ini
0x19D2
root@zksoft-ThinkPad-X260:~/.android#
```

（3）创建/etc/udev/rules.d/51-android.rules 文件，解决权限问题。

文件末尾加上：

```
SUBSYSTEM=="usb",ATTR{idVendor}=="19D2",MODE="0666",GROUP="plugdev"
```

ATTR 并非 ATTRS，不同 ubuntu 版本可能测试情况不一样，两种都试一试。另外一台 ubuntu ATTRS 也生效。

```
root@zksoft-ThinkPad-X260:/etc/udev/rules.d# ls
51-android.rules
root@zksoft-ThinkPad-X260:/etc/udev/rules.d# cat 51-android.rules
SUBSYSTEM=="usb",ATTR{idVendor}=="19D2",MODE="0666",GROUP="plugdev"

root@zksoft-ThinkPad-X260:/etc/udev/rules.d#
```

（4）拔插 usb 即可，adb device 查看设备是否正常显示，权限是否正常。

```
root@zksoft-ThinkPad-X260:/home/zksoft# adb devices
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
List of devices attached
daf368a6        device
```

```
root@zksoft-ThinkPad-X260:/home/zksoft# adb shell
/ # ls
WEBSERVER   cache       firmware    media       sbin        system      var
bin         data        home        mnt         sdcard      target      ztedata
boot        dev         lib         proc        share       tmp
build.prop  etc         linuxrc     run         sys         usr
/ # exit
root@zksoft-ThinkPad-X260:/home/zksoft#
```

（5）若 adb devices 无法列出有效 adb 设备，检查 adb 设备驱动是否安装正常。

- 列出 usb 设备列表

```
lsusb -t
```

```
root@zksoft-ThinkPad-X260:/home/zksoft# lsusb -t
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
    |__ Port 3: Dev 2, If 0, Class=Hub, Driver=hub/4p, 5000M
    |__ Port 6: Dev 3, If 0, Class=Mass Storage, Driver=usb-storage, 5000M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/12p, 480M
    |__ Port 1: Dev 2, If 0, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 1: Dev 2, If 1, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 1: Dev 2, If 2, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 2: Dev 3, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 2: Dev 5, If 0, Class=Vendor Specific Class, Driver=pl2303, 12M
        |__ Port 3: Dev 11, If 0, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 1, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 2, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 3, Class=Communications, Driver=cdc_ether, 480M
        |__ Port 3: Dev 11, If 4, Class=CDC Data, Driver=cdc_ether, 480M
        |__ Port 3: Dev 11, If 5, Class=Vendor Specific Class, Driver=option, 480M
    |__ Port 7: Dev 4, If 0, Class=Wireless, Driver=btusb, 12M
    |__ Port 7: Dev 4, If 1, Class=Wireless, Driver=btusb, 12M
    |__ Port 8: Dev 6, If 0, Class=Video, Driver=uvcvideo, 480M
    |__ Port 8: Dev 6, If 1, Class=Video, Driver=uvcvideo, 480M
    |__ Port 9: Dev 8, If 0, Class=Vendor Specific Class, Driver=, 12M
```

此设备是adb设备，发现被安装option驱动，不正确

- unbind 不正确的设备驱动

因为这里是 option 驱动，所以进入的驱动目录如下：(※注：与被安装的错误驱动有关)

```
cd /sys/bus/usb/drivers/option
```

```
root@zksoft-ThinkPad-X260:/sys/bus/usb/drivers/option# ls -all
total 0
drwxr-xr-x  2 root root    0 Aug 17 10:10 .
drwxr-xr-x 15 root root    0 Aug 17 10:10 ..
lrwxrwxrwx  1 root root    0 Aug 17 10:10 1-2.3:1.0 -> ../../../../devices/pci0000:00/0000:00:14.0/usb1/1-2
/1-2.3/1-2.3:1.0
lrwxrwxrwx  1 root root    0 Aug 17 10:10 1-2.3:1.1 -> ../../../../devices/pci0000:00/0000:00:14.0/usb1/1-2
/1-2.3/1-2.3:1.1
lrwxrwxrwx  1 root root    0 Aug 17 10:10 1-2.3:1.2 -> ../../../../devices/pci0000:00/0000:00:14.0/usb1/1-2
/1-2.3/1-2.3:1.2
lrwxrwxrwx  1 root root    0 Aug 17 10:10 1-2.3:1.5 -> ../../../../devices/pci0000:00/0000:00:14.0/usb1/1-2
/1-2.3/1-2.3:1.5
--w-------  1 root root 4096 Aug 17 10:10 bind
lrwxrwxrwx  1 root root    0 Aug 17 10:10 module -> ../../../../module/usbserial
--w-------  1 root root 4096 Aug 17 10:10 uevent
--w-------  1 root root 4096 Aug 17 10:10 unbind
root@zksoft-ThinkPad-X260:/sys/bus/usb/drivers/option# echo 1-2.3:1.5 > unbind
root@zksoft-ThinkPad-X260:/sys/bus/usb/drivers/option#
```

注意：这里为什么输入 1-2.3:1.5。和上图 if 5 对应。

- 执行成功后驱动结果如下。此时再执行 adb devices，既可以成功找到 adb 设备。

```
root@zksoft-ThinkPad-X260:/home/zksoft# lsusb -t
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/6p, 5000M
    |__ Port 3: Dev 2, If 0, Class=Hub, Driver=hub/4p, 5000M
    |__ Port 6: Dev 3, If 0, Class=Mass Storage, Driver=usb-storage, 5000M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/12p, 480M
    |__ Port 1: Dev 2, If 0, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 1: Dev 2, If 1, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 1: Dev 2, If 2, Class=Human Interface Device, Driver=usbhid, 12M
    |__ Port 2: Dev 3, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 2: Dev 5, If 0, Class=Vendor Specific Class, Driver=pl2303, 12M
        |__ Port 3: Dev 11, If 0, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 1, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 2, Class=Vendor Specific Class, Driver=option, 480M
        |__ Port 3: Dev 11, If 3, Class=Communications, Driver=cdc_ether, 480M
        |__ Port 3: Dev 11, If 4, Class=CDC Data, Driver=cdc_ether, 480M
        |__ Port 3: Dev 11, If 5, Class=Vendor Specific Class, Driver=, 480M
    |__ Port 7: Dev 4, If 0, Class=Wireless, Driver=btusb, 12M
    |__ Port 7: Dev 4, If 1, Class=Wireless, Driver=btusb, 12M
    |__ Port 8: Dev 6, If 0, Class=Video, Driver=uvcvideo, 480M
    |__ Port 8: Dev 6, If 1, Class=Video, Driver=uvcvideo, 480M
    |__ Port 9: Dev 8, If 0, Class=Vendor Specific Class, Driver=, 12M
root@zksoft-ThinkPad-X260:/home/zksoft#
```

2.6. LINUX 系统下电源管理

2.6.1. 内核配置项修改

linux 内核必须支持电源管理功能，请确认内核已经打开了以下配置选项：

1,CONFIG_SUSPEND 或者 CONFIG_HIBERNATION

2,CONFIG_PM

3,CONFIG_PM_RUNTIME

4,CONFIG_USB_SUSPEND (Linux 3.10 以上版本内核无此选项)

2.6.2. 电源管理设置

Linux 操作系统下为模块的 USB 端口正确绑定了驱动后，默认情况下 USB 的电源管理功能是被关闭的，需要设置才能使能电源管理功能。

2.6.2.1. 使能选择性挂起和远程唤醒功能

首先需要确认模块 USB 设备的 sysfs 路径。USB 设备的 sysfs 路径格式如下：

```
/sys/bus/usb/devices/总线（Bus）-端口（Port）/
```

其中总线和端口可以通过 lsusb 命令确定，以图表 1 为例，模块位于总线 1，端口 2，因此设备的 sysfs 路径为：

```
/sys/bus/usb/devices/1-2/
```

输入以下命令使能选择性挂起和远程唤醒功能：

```
echo auto >/sys/bus/usb/devices/1-2/power/control

echo auto >/sys/bus/usb/devices/1-2/power/level
```

要确认功能已正确打开请输入以下命令：

```
cat /sys/bus/usb/devices/1-2/power/control

cat /sys/bus/usb/devices/1-2/power/level
```

以上两个命令都返回“auto”时表明选择性挂起和远程唤醒功能打开。

2.6.2.2. 唤醒系统挂起

当系统挂起后，模块有能力将计算机从系统挂起状态中唤醒到正常工作状态，如果要打开此功能，请输入以下命令：

```
    echo enabled >/sys/bus/usb/devices/1-2/power/wakeup
```

要确认功能已正确打开请输入以下命令：

```
    cat /sys/bus/usb/devices/1-2/power/wakeup
```

返回“enabled”表明功能已正确打开。

2.6.2.3. 设置挂起延迟

当 USB 驱动检测到 USB 设备空闲一段时间后，才会使 USB 设备进入选择性挂起状态。设备空闲后挂起的延迟时间可以通过以下两个 sysfs 接口设置：

```
/sys/bus/usb/devices/1-2/power/autosuspend

/sys/bus/usb/devices/1-2/power/autosuspend_delay_ms
```

autosuspend 设置单位为秒，autosuspend_delay_ms 设置单位为毫秒。默认情况下延迟为 2s，推荐设置为 2~5s。设置命令如下：

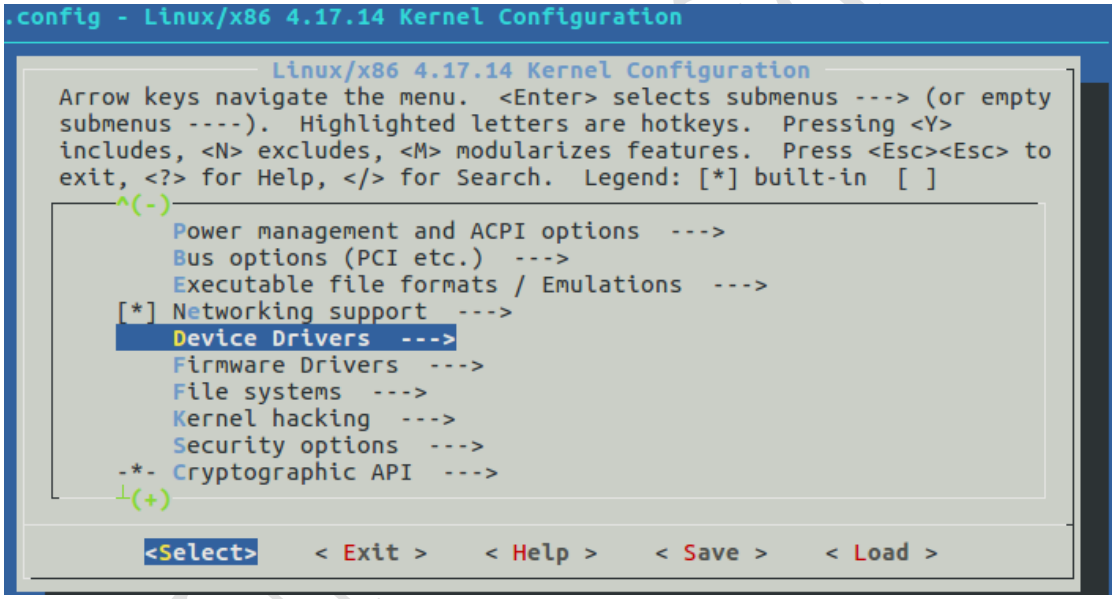
```
echo 2 > /sys/bus/usb/devices/1-2/power/autosuspend

或者：

echo 2000 > /sys/bus/usb/devices/1-2/power/autosuspend_delay_ms
```

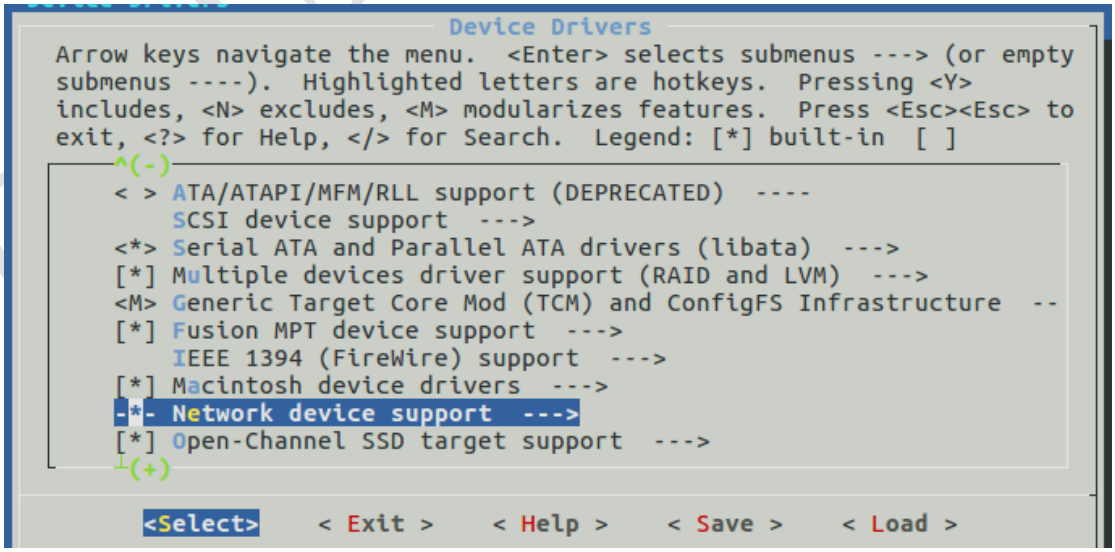
3. 嵌入式 LINUX PPP 配置

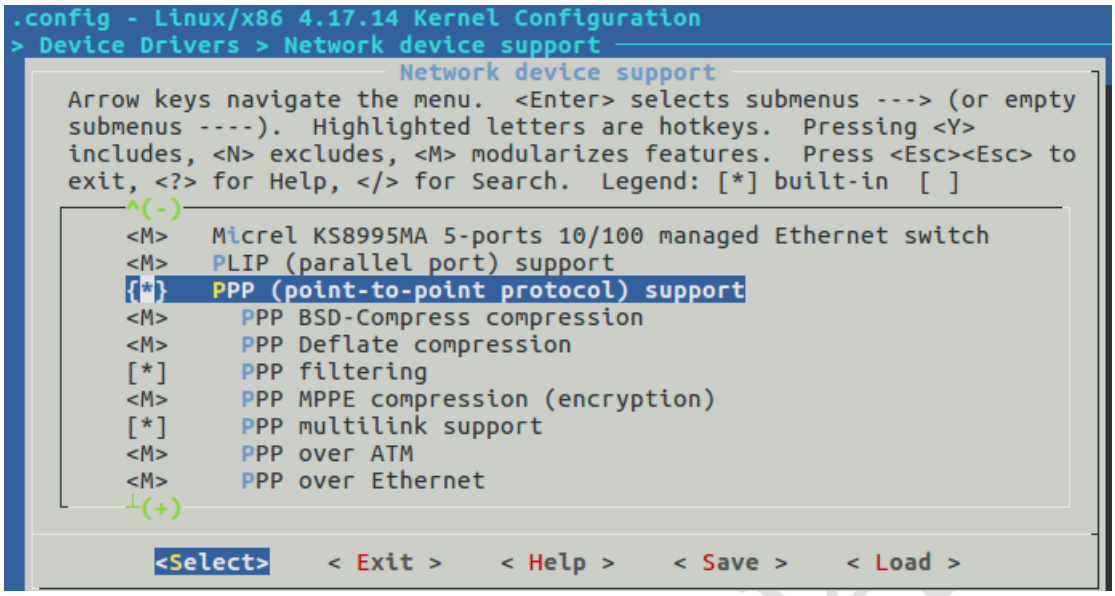
3.1. 内核中添加 PPP 组件

通常，配置内核是通过指令 make menuconfig，执行该指令后：

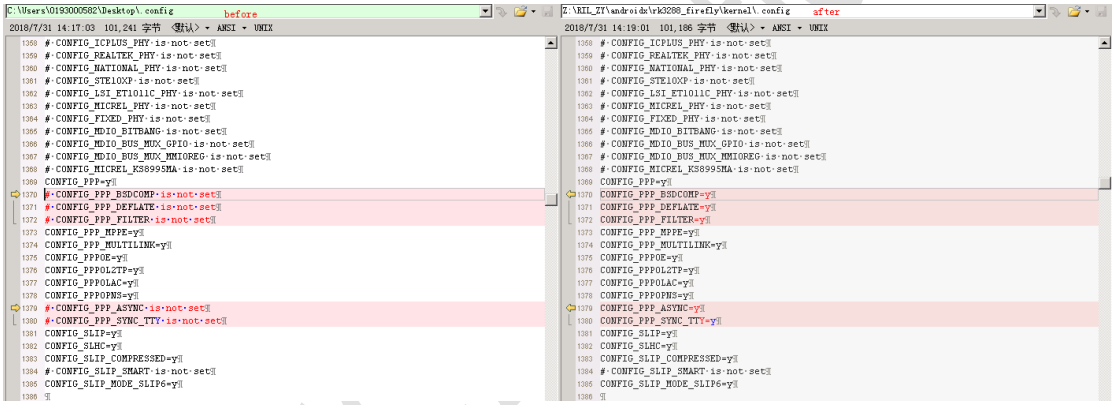
```
devices drivers-->
Network device support-->
  ppp support-->
  ppp filtering
  ppp support for async serial ports
  ppp support for sync tty ports
  ppp deflate compression
  ppp BSD-compress compression
```







* **注**：如果您的内核不支持 make menuconfig 命令配置，可以直接修改 kernel 目录下的 .config 文件，手动添加配置项。



3.2. 使用 PPP 和 CHAT 进行数据连接

3.2.1. PPP 数据连接脚本

解压缩 PPP_Script.7z 附件，将解压后的内容拷贝到 linux 设备中，并赋予可执行权限，以便可以执行。

3.2.2. 进行拨号连接

运行 ppp-on 脚本，进行数据拨号

./ppp-on

```
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script# ./ppp-on &
[1] 3769
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script#
pppd options in effect:
```

```
    debug        # (from gosuncn_options)
    nodetach     # (from gosuncn_options)
    persist      # (from gosuncn_options)
    dump     # (from gosuncn_options)
    noauth       # (from gosuncn_options)
    user Anyname      # (from gosuncn_options)
    password ??????       # (from gosuncn_options)
    /dev/ttyUSB2      # (from gosuncn_options)
    115200       # (from gosuncn_options)
    lock     # (from gosuncn_options)
    connect /usr/sbin/chat -v -f /home/zksoft/DATA/PPP_Script/gosuncn_ppp_dialer     #       (from
command line)
    crtscts      # (from gosuncn_options)
    modem        # (from gosuncn_options)
    asyncmap 0       # (from /etc/ppp/options)
    lcp-echo-failure 4       # (from /etc/ppp/options)
    lcp-echo-interval 30      # (from /etc/ppp/options)
    hide-password     # (from /etc/ppp/options)
    novj     # (from gosuncn_options)
    ipcp-accept-local     # (from gosuncn_options)
    ipcp-accept-remote       # (from gosuncn_options)
    noipdefault      # (from gosuncn_options)
    defaultroute     # (from gosuncn_options)
    usepeerdns       # (from gosuncn_options)
    noccp        # (from gosuncn_options)
    nobsdcomp    # (from gosuncn_options)
    noipx        # (from /etc/ppp/options)
    ATE
    OK
    ATH
    OK
    ATP
    OK
    ATD*99#
    CONNECT
    Script /usr/sbin/chat -v -f /home/zksoft/DATA/PPP_Script/gosuncn_ppp_dialer finished (pid 3772),
status = 0x0
    Serial connection established.
    using channel 2
    Using interface ppp0
    Connect: ppp0 <--> /dev/ttyUSB2
    sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x9f32d71c> <pcomp> <accomp>]
    sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x9f32d71c> <pcomp> <accomp>]
    sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x9f32d71c> <pcomp> <accomp>]
```

```
rcvd [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x269f4b16> <pcomp> <accomp>]
sent [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x269f4b16> <pcomp> <accomp>]
rcvd [LCP ConfAck id=0x1 <asyncmap 0x0> <magic 0x9f32d71c> <pcomp> <accomp>]
sent [LCP EchoReq id=0x0 magic=0x9f32d71c]
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP ConfReq id=0x1 <addr 10.9.186.37>]
sent [IPCP ConfAck id=0x1 <addr 10.9.186.37>]
rcvd [LCP EchoRep id=0x0 magic=0x269f4b16]
rcvd [IPV6CP ConfReq id=0x1 <addr fe80::8528:7420:f982:bc1a>]
Unsupported protocol 'IPv6 Control Protocol' (0x8057) received
sent [LCP ProtRej id=0x2 80 57 01 01 00 0e 01 0a 85 28 74 20 f9 82 bc 1a]
rcvd [IPCP ConfNak id=0x1 <addr 10.9.186.36> <ms-dns1 61.134.1.6> <ms-dns2 61.134.1.6>]
sent [IPCP ConfReq id=0x2 <addr 10.9.186.36> <ms-dns1 61.134.1.6> <ms-dns2 61.134.1.6>]
rcvd [IPCP ConfAck id=0x2 <addr 10.9.186.36> <ms-dns1 61.134.1.6> <ms-dns2 61.134.1.6>]
local  IP address 10.9.186.36
remote IP address 10.9.186.37
primary   DNS address 61.134.1.6
secondary DNS address 61.134.1.6
Script /etc/ppp/ip-up started (pid 3783)
Script /etc/ppp/ip-up finished (pid 3783), status = 0x0
```

```
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script# ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.9.186.36  P-t-P:10.9.186.37  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:84864 errors:0 dropped:0 overruns:0 frame:0
          TX packets:60318 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:113682221 (113.6 MB)  TX bytes:4311774 (4.3 MB)

root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script#
```

```
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script# ping www.baidu.com
PING www.a.shifen.com (180.149.132.151) 56(84) bytes of data.
64 bytes from 180.149.132.151: icmp_seq=1 ttl=53 time=55.7 ms
64 bytes from 180.149.132.151: icmp_seq=2 ttl=53 time=45.4 ms
64 bytes from 180.149.132.151: icmp_seq=3 ttl=53 time=45.8 ms
64 bytes from 180.149.132.151: icmp_seq=4 ttl=53 time=36.9 ms
```

3.2.3. 断开连接

1.在终端窗口输入“ctrl+c”

2.运行 disconnect 脚本:

./disconnect

```
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script# ./disconnect
Terminating on signal 15
Connect time 0.4 minutes.
Sent 641 bytes, received 236 bytes.
root@zksoft-ThinkPad-X260:/home/zksoft/DATA/PPP_Script#  Script  /etc/ppp/ip-down  started  (pid
```

```
4481)
    sent [LCP TermReq id=0x3 "User request"]
    rcvd [LCP TermAck id=0x3]
    Connection terminated.
    Script /etc/ppp/ip-down finished (pid 4481), status = 0x0
```