

# Linux USB 3G\_4G 模块配置使用教程

作者：向仔州

3G 模块移植.....	2
现在我们来切换 3G 模块模式.....	8
APN 添加方法.....	14
用 AT 指令集查看 SIM 卡信息和信号强度.....	16
LinuxUSB 4G 模块使用.....	17
4G 模块调试过程在 ARM9 平台下编译 pppd 软件.....	20
3G 和 4G ping 百度域名问题.....	24
Scsi 驱动 4G 模块代码.....	25

## 3G 模块移植

CPU: IMX6 平台

操作系统: linux-3.14.28

1. 先要准备以下文件

```
libusb-1.0.9.tar.bz2  
ppp-2.4.5.tar.gz  
usb-modeswitch-2.0.1.tar.bz2  
usb-modeswitch-data-20131113.tar.bz2
```

将文件拷贝进 ubuntu 系统

```
root@ubuntu:/home/xiang/IMX6/3G# ls  
libusb-1.0.9.tar.bz2 ppp-2.4.5.tar.gz usb-modeswitch-2.0.1.tar.bz2 usb-modeswitch-data-20131113.tar.bz2
```

2. 先安装 libusb 软件

```
[6/3G# tar -vxf libusb-1.0.9.tar.bz2 ] 解压 libusb  
root@ubuntu:./home/xiang/IMX6/3G# ls  
libusb-1.0.9 libusb-1.0.9.tar.bz2  
root@ubuntu:/home/xiang/IMX6/3G# 进入 libusb-1.0.9 目录
```

编译 libusb

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# ls  
aclocal.m4 compile config.sub COPYING examples libusb Makefile.am msvc README  
AUTHORS config.guess configure depcomp INSTALL libusb-1.0.pc.in Makefile.in NEWS THANKS  
ChangeLog config.h.in configure.ac doc install-sh ltmain.sh missing PORTING TODO  
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# ./configure --host=arm-poky-linux --prefix=$PWD/tmp
```

配置./configure

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# make  
make all-recursive  
make[1]: Entering directory `/home/xiang/IMX6/3G/libusb-1.0.9'  
Making all in libusb  
make[2]: Entering directory `/home/xiang/IMX6/3G/libusb-1.0.9/libusb'  
CC libusb_1_0_la-core.lo  
CC libusb_1_0_la-descriptor.lo  
CC libusb_1_0_la-io.lo  
CC libusb_1_0_la-sync.lo  
CC libusb_1_0_la-linux_usbfs.lo  
CC libusb_1_0_la-threads_posix.lo  
CCLD libusb-1.0.la  
make[2]: Leaving directory `/home/xiang/IMX6/3G/libusb-1.0.9/libusb'  
Making all in doc  
make[2]: Entering directory `/home/xiang/IMX6/3G/libusb-1.0.9/doc'  
make[2]: Nothing to be done for `all'.  
make[2]: Leaving directory `/home/xiang/IMX6/3G/libusb-1.0.9/doc'  
make[2]: Entering directory `/home/xiang/IMX6/3G/libusb-1.0.9'  
make[2]: Leaving directory `/home/xiang/IMX6/3G/libusb-1.0.9'  
make[1]: Leaving directory `/home/xiang/IMX6/3G/libusb-1.0.9'  
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9#
```

make

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# make install
```

这就是编译之后得到的 tmp 目录

然后到 libusb /tmp 目录下看一下

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# ls
aclocal.m4 config.guess config.status COPYING INSTALL libusb-1.0.pc Makefile.am NEWS THANKS
AUTHORS config.h config.sub depcomp install-sh libusb-1.0.pc.in Makefile.in PORTING tmp
ChangeLog config.h.in configure doc libtool ltmain.sh missing README TODO
compile config.log configure.ac examples libusb Makefile msvc stamp-h1
```

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9# cd tmp/
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp# ls
include lib
```

把头文件和 lib 库文件复制进交叉编译工具链

因为我的 CC 编译器用的是下面这个路径

```
CC="arm-poky-linux-gnueabi-gcc -march=armv7-a -mthumb-interwork -mfloating-abi=hard -
mfpu=neon -mtune=cortex-a9 --sysroot=/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-
gnueabi"
```

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/include/libusb-1.0# ls
libusb.h
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/include/libusb-1.0#
```

我要把 libusb.h 复制到交叉工具链这个路径

```
root@ubuntu:/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi/usr/include#
```

我还要把 lib 库复制进交叉工具链里面

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/lib# ls
libusb-1.0.a libusb-1.0.la libusb-1.0.so libusb-1.0.so.0 libusb-1.0.so.0.1.0 pkgconfig
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/lib# cp * -rfd /opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi/usr/lib
cp * -rfd 这个 d 就是说如果你是链接文件，就按照链接文件属性拷贝
libgplaycore-1.0.so.0.0.0 libopencv_highgui.so.2.4 xml2Conf.sh
libgstallocators-1.0.la libopencv_highgui.so.2.4.11 xtables
root@ubuntu:/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi/usr/lib#
```

然后在将 usblib 库再拷贝一份到开发板文件系统的 lib 里面

```
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/lib# ls
libusb-1.0.a libusb-1.0.la libusb-1.0.so libusb-1.0.so.0 libusb-1.0.so.0.1.0 pkgconfig
root@ubuntu:/home/xiang/IMX6/3G/libusb-1.0.9/tmp/lib# cp * -rfd /opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi/usr/lib
```

其实拷贝一份 usblib 到开发板文件系统不就行了，为什么还要向 X86 的交叉编译工具链里面拷贝呢？ 那是因为我们下面编译 usb\_modeswitch 软件的时候要用到 libusb 里面的库。

如果不拷贝一份 include 和 lib 的 libusb 库到交叉编译工具链里面，在编译 usb\_modeswitch 会报错

```
book@book-desktop:/work/projects/3G/usb-modeswitch-2.0.1$ make DESTDIR=$PWD/tmp
No installed jimsh or tclsh, building local bootstrap jimsh
cc -o usb_modeswitch usb_modeswitch.c -Wall -lusb-1.0
In file included from usb_modeswitch.c:59:
usb_modeswitch.h:26:20: error: libusb.h: No such file or directory
```

### 3. 编译 usb\_modeswitch

```
root@ubuntu:/home/xiang/IMX6/3G# tar -vxf usb-modeswitch-2.0.1.tar.bz2
```

**usb-modeswitch-2.0.1**

进入 usb-modeswitch-2.0.1 目录

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1# ls
ChangeLog      jlm        README      usb_modeswitch.conf      usb_modeswitch@.service  usb-modeswitch-upstart.conf
COPYING        Makefile    usb_modeswitch.1  usb_modeswitch.dispatcher.1  usb_modeswitch.sh
dispatcher.c   make_string.tcl  usb_modeswitch.c  usb_modeswitch.h  usb_modeswitch.tcl
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1# vim Makefile
```

修改 Makefile

```
CC           ?= gcc
CFLAGS       += -Wall
LIBS         = `pkg-config --libs --cflags libusb-1.0`
```

修改 CC 指定的交叉编译工具

修改 LIBS，指定我们前面编译的 libusb 库

```
CC           = arm-poky-linux-gnueabi-gcc -march=armv7-a -mthumb-interwork -mfpu=neon -mtune=cortex-a9 --sysroot=/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi
CFLAGS       += -Wall
LIBS         = -lusb-1.0
```

保存 Makefile

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1# make DESTDIR=$PWD/tmp
arm-poky-linux-gnueabi-gcc -march=armv7-a -mthumb-interwork -mfpu=neon -mtune=a7hf-vfp-neon-poky-linux-gnueabi -o usb_modeswitch usb_modeswitch.c -Wall -lusb-1.0
sed 's_!/usr/bin/tclsh_!"/usr/bin/tclsh"'_ < usb_modeswitch.tcl > usb_modeswitch_dispatcher
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1#
```

编译成功

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1# make install DESTDIR=$PWD/tmp
```

Make install 会在目录下出现 tmp

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1# ls
ChangeLog      Makefile      usb_modeswitch      usb
COPYING        make_string.tcl  usb_modeswitch.1  usb
dispatcher.c   README       usb_modeswitch.c  usb
jlm            tmp          usb_modeswitch.conf  usb
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1#
```

这就是编译之后得到的 tmp 目录

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1/tmp# ls
etc  lib  usr  var
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1/tmp#
```

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1/tmp# ls
etc  lib  usr  var
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-2.0.1/tmp# cp * -rfd .....
```

将这四个目录拷贝到开发板相应的文件系统下面去

#### 4. 编译 usb\_modeswitch-data

```
root@ubuntu:/home/xiang/IMX6/3G# tar -vxf usb-modeswitch-data-20131113.tar.bz2
```

编译

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-data-20131113# ls  
40-usb_modeswitch.rules ChangeLog COPYING gen-rules.tcl Makefile README usb_modeswitch.d
```

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-data-20131113# make install DESTDIR=$PWD/tmp  
install -d /home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp/usr/share/usb_modeswitch  
install -d /home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp/etc/usb_modeswitch.d  
install -D --mode=644 40-usb_modeswitch.rules /home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp/lib/udev/rules.d/40-usb_modeswitch.rules  
install --mode=644 -t /home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp/usr/share/usb_modeswitch ./usb_modeswitch.d/*  
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-data-20131113#
```

```
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp# ls  
etc lib usr  
root@ubuntu:/home/xiang/IMX6/3G/usb-modeswitch-data-20131113/tmp#
```

将 `usb_modeswitch_data` 里面的文件全部拷贝到开发板的文件系统里面。

#### 5. 编译 pppd

```
root@ubuntu:/home/xiang/IMX6/3G# tar -vxf ppp-2.4.5.tar.gz
```

**ppp-2.4.5** 进入目录

错误的 `pppd` 编译方法，正确的编译方法在下面

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# ls  
Changes-2.3 configure FAQ modules pppdump README.cbcP README.MPPE README.pppoe README.sol2 solaris  
chat contrib include PLUGINS pppstats README.eap-srp README.MSCHAP80 README.pppol2tp scripts  
common etc.ppp linux pppd README README.linux README.MSCHAP81 README.pwfd SETUP  
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# ./configure --host=arm-poky-linux --prefix=$PWD/tmp
```

配置 `configure`

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# ./configure --host=arm-poky-linux --prefix=$PWD/tmp  
Configuring for Linux  
Creating Makefiles.  
Makefile <= linux/Makefile.top  
pppd/Makefile <= pppd/Makefile.linux  
pppstats/Makefile <= pppstats/Makefile.linux  
chat/Makefile <= chat/Makefile.linux  
pppdump/Makefile <= pppdump/Makefile.linux  
pppd/plugins/Makefile <= pppd/plugins/Makefile.linux  
pppd/plugins/rp-pppoe/Makefile <= pppd/plugins/rp-pppoe/Makefile.linux  
pppd/plugins/radius/Makefile <= pppd/plugins/radius/Makefile.linux  
pppd/plugins/pppoatm/Makefile <= pppd/plugins/pppoatm/Makefile.linux  
pppd/plugins/pppol2tp/Makefile <= pppd/plugins/pppol2tp/Makefile.linux
```

配置之后是这样

然后我们编译

因为 `makefile` 我们没有看到 CC 所以我们在 `Make` 外面指定 `GCC`

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# make CC="arm-poky-linux-gnueabi-gcc -march=armv7-a -mthumb-interwork -mfloat-abi=hard -mfpu=neon -mtune=cortex-a9 --sysroot=/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi"
```

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# make install  
install -d -m 755 /home/xiang/IMX6/3G/ppp-2.4.5/tmp/sbin  
install -d -m 755 /home/xiang/IMX6/3G/ppp-2.4.5/tmp/share/man/man8  
cd chat; make install  
make[1]: Entering directory `/home/xiang/IMX6/3G/ppp-2.4.5/chat'  
mkdir -p /home/xiang/IMX6/3G/ppp-2.4.5/tmp/sbin /home/xiang/IMX6/3G/ppp-2.4.5/tmp/share/man/man8  
install -s -c chat /home/xiang/IMX6/3G/ppp-2.4.5/tmp/sbin  
strip: Unable to recognise the format of the input file `/home/xiang/IMX6/3G/ppp-2.4.5/tmp/sbin/chat'  
install: strip process terminated abnormally  
make[1]: *** [install] Error 1  
make[1]: Leaving directory `/home/xiang/IMX6/3G/ppp-2.4.5/chat'  
make: *** [install-progs] Error 2  
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5#
```

编译完成后在 `make install` 的时候发生错误。

```
strip: Unable to recognise the format of the input file `/home/xiang/IMX6/3G/ppp-2.4.5/tmp/sbin/chat'
```

根据这条错误问题，我们知道要用 `arm-linux-strip` 这个命令来消除符号。但是我觉得这样太麻烦了。还要一种方法就是去文件里面删除 `-s`。

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# grep "INSTALL) \-s" * -nR
chat/Makefile.linux:28: $(INSTALL) -s -c chat $(BINDIR)
chat/Makefile:28:     $(INSTALL) -s -c chat $(BINDIR)
pppd/Makefile.linux:102:EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry
pppd/Makefile.linux:203:           $(INSTALL) -s -c -m 555 pppd $(BINDIR)/pppd
pppd/plugins/rp-pppoe/Makefile.linux:46:           $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)
pppd/plugins/rp-pppoe/Makefile.linux:48:           $(INSTALL) -s -c -m 555 ppoe-discovery $(BINDIR)
pppd/plugins/rp-pppoe/Makefile:46:           $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)
pppd/plugins/rp-pppoe/Makefile:48:           $(INSTALL) -s -c -m 555 ppoe-discovery $(BINDIR)
pppd/plugins/radius/Makefile.linux:39:           $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)
pppd/plugins/radius/Makefile.linux:40:           $(INSTALL) -s -c -m 755 radattr.so $(LIBDIR)
pppd/plugins/radius/Makefile.linux:41:           $(INSTALL) -s -c -m 755 radrealms.so $(LIBDIR)
pppd/plugins/radius/Makefile:39:           $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)
pppd/plugins/radius/Makefile:40:           $(INSTALL) -s -c -m 755 radattr.so $(LIBDIR)
pppd/plugins/radius/Makefile:41:           $(INSTALL) -s -c -m 755 radrealms.so $(LIBDIR)
pppd/Makefile:102:EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry
pppd/Makefile:203:           $(INSTALL) -s -c -m 555 pppd $(BINDIR)/pppd
pppdump/Makefile.linux:20:           $(INSTALL) -s -c pppdump $(BINDIR)
pppdump/Makefile:20:           $(INSTALL) -s -c pppdump $(BINDIR)
pppstats/Makefile.linux:25:           $(INSTALL) -s -c pppstats $(BINDIR)
pppstats/Makefile:25:           $(INSTALL) -s -c pppstats $(BINDIR)
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5#
```

有这麼多-s 我们一个一个去删除吧。

正确的 `pppd` 编译方法

```
root@ubuntu:/home/xiang/IMX6/3G# tar -vxf ppp-2.4.5.tar.gz
```

**ppp-2.4.5** 进入目录

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# ls
Changes-2.3  configure  FAQ  modules  pppdump  README.cbcP  README.MPPE  README.pppoe  README.sol2  solaris
chat  contrib  include  PLUGINS  pppstats  README.eap-srp  README.MSCHAP80  README.pppol2tp  scripts
common  etc.ppp  linux  pppd  README  README.linuX  README.MSCHAP81  README.pwfd  SETUP
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5#
```

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# grep "INSTALL) \-s" * -nR
chat/Makefile.linux:28: $(INSTALL) -s -c chat $(BINDIR)
chat/Makefile:28:     $(INSTALL) -s -c chat $(BINDIR)
pppd/Makefile.linux:102:EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry
pppd/Makefile.linux:203:           $(INSTALL) -s -c -m 555 pppd $(BINDIR)/pppd
pppd/plugins/rp-pppoe/Makefile.linux:46:           $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)
pppd/plugins/rp-pppoe/Makefile.linux:48:           $(INSTALL) -s -c -m 555 ppoe-discovery $(BINDIR)
pppd/plugins/rp-pppoe/Makefile:46:           $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)
pppd/plugins/rp-pppoe/Makefile:48:           $(INSTALL) -s -c -m 555 ppoe-discovery $(BINDIR)
pppd/plugins/radius/Makefile.linux:39:           $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)
pppd/plugins/radius/Makefile.linux:40:           $(INSTALL) -s -c -m 755 radattr.so $(LIBDIR)
pppd/plugins/radius/Makefile.linux:41:           $(INSTALL) -s -c -m 755 radrealms.so $(LIBDIR)
pppd/plugins/radius/Makefile:39:           $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)
pppd/plugins/radius/Makefile:40:           $(INSTALL) -s -c -m 755 radattr.so $(LIBDIR)
pppd/plugins/radius/Makefile:41:           $(INSTALL) -s -c -m 755 radrealms.so $(LIBDIR)
pppd/Makefile:102:EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry
pppd/Makefile:203:           $(INSTALL) -s -c -m 555 pppd $(BINDIR)/pppd
pppdump/Makefile.linux:20:           $(INSTALL) -s -c pppdump $(BINDIR)
pppdump/Makefile:20:           $(INSTALL) -s -c pppdump $(BINDIR)
pppstats/Makefile.linux:25:           $(INSTALL) -s -c pppstats $(BINDIR)
pppstats/Makefile:25:           $(INSTALL) -s -c pppstats $(BINDIR)
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5#
```

就是要把这些 `Makefile.linux` 里面的-s 全部去掉，不去掉会出现 make 的时候 `strip` 报错

`strip: Unable to recognise the format of the input file XXXX` 文件或者`.so`

有些交叉编译器只需要取消下面 6 个-s 就可以了，IMX6 的交叉编译器就是，但是 ARM9 的不是

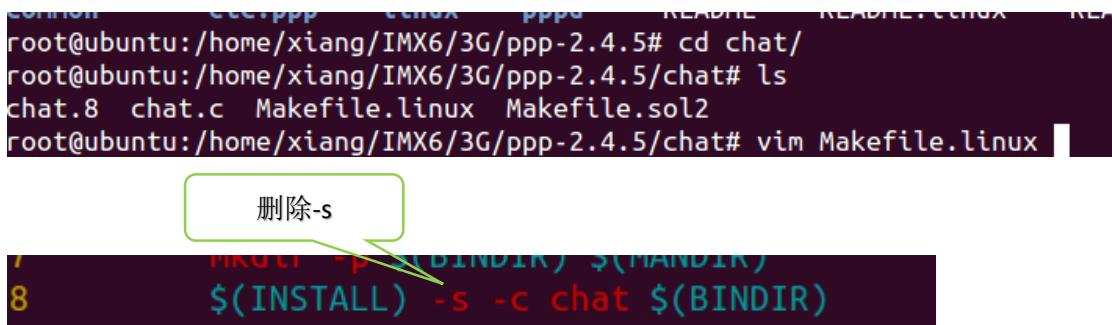
修改以下 6 个文件路径的 Makefile.linux

```
chat/Makefile.linux:28: $(INSTALL) -s -c chat $(BINDIR)
pppd/plugins/radius/Makefile.linux:39: $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)
pppd/plugins/rp-pppoe/Makefile.linux:46: $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)
pppd/Makefile.linux:102:EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry
pppdump/Makefile.linux:20: $(INSTALL) -s -c pppdump $(BINDIR)
pppstats/Makefile.linux:25: $(INSTALL) -s -c pppstats $(BINDIR)
```

我们先去修改这个 chat/Makefile.linux: 文件

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# cd chat/
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5/chat# ls
chat.8  chat.c  Makefile.linux  Makefile.sol2
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5/chat# vim Makefile.linux
```

删除-s



```
8 $(INSTALL) -s -c chat $(BINDIR)
```

修改后就是下面这样

```
28 $(INSTALL) -c chat $(BINDIR)
```

然后保存退出

其余 5 个文件也是这样修改，删除所有-s 的位置。

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# ./configure --host=arm-poky-linux --prefix=$PWD/tmp
Configuring for Linux
Creating Makefiles.
Makefile <= linux/Makefile.top
pppd/Makefile <= pppd/Makefile.linux
pppstats/Makefile <= pppstats/Makefile.linux
chat/Makefile <= chat/Makefile.linux
pppdump/Makefile <= pppdump/Makefile.linux
pppd/plugins/Makefile <= pppd/plugins/Makefile.linux
pppd/plugins/rp-pppoe/Makefile <= pppd/plugins/rp-pppoe/Makefile.linux
pppd/plugins/radius/Makefile <= pppd/plugins/radius/Makefile.linux
pppd/plugins/ppoatm/Makefile <= pppd/plugins/ppoatm/Makefile.linux
pppd/plugins/pppol2tp/Makefile <= pppd/plugins/pppol2tp/Makefile.linux
```

然后配置文件

```
root@ubuntu:/home/xiang/IMX6/3G/ppp-2.4.5# make CC="arm-poky-linux-gnueabi-gcc" -march=armv7-a -mthumb-interwork -mfloating-abi=hard -mfpu=neon -mtune=cortex-a9 -sysroot=/opt/poky/1.8/sysroots/cortexa7hf-vfp-neon-poky-linux-gnueabi"
```

编译 make

然后 make install 安装文件就在 tmp 目录下了

```
include  lib  sbin  share
```

将这些文件拷贝进开发板文件系统。

3G 用到的库文件移植就算完成了

## 现在我们来切换 3G 模块模式

将 3G 模块切换成 ttyUSB 功能有两种方法。

第一种：有些模块厂商已经在 3G 模块内部自动安装的模块切换协议。

我们就以 M7281 为例

```
root@imx6qdlolo:/# lsusb
Bus 001 Device 002: ID 1782:0002 Spreadtrum Communications Inc.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
root@imx6qdlolo:/#
```

模块插入 USB 后会连接进 USB 设备，我们 lsusb 命令来查看模块的 PID,VID。

我们 M7281 的 VID 是 1782，PID 是 0002，这个不要搞反咯，我就是搞反了搞了几天。

```
/fsl-linux/drivers/usb/serial#
```

 进入内核 usb 转串口目录

```
vim option.c
```

 修改 option 文件

下面是修改具体步骤

```
1 static int option_send_setup(struct usb_serial_port *port);
2 static void option_instat_callback(struct urb *urb);
3 /*M7281 3G model*/
4 #define MPID 0x0002
5 #define MVID 0x1782
6
```

在 option.c 文件里面加上 PID,VID 的宏，自己根据模块型号来定义。

```
5 static const struct usb_device_id option_ids[] = {
6     { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_COLT) },
7     { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA) },
8     { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_LIGHT) },
9     { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_QUAD) },
10    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_QUAD_LIGHT) },
11    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_NDIS) },
12    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_NDIS_LIGHT) },
13    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_NDIS_QUAD) },
14    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_RICOLA_NDIS_QUAD_LIGHT) },
15    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_COBRA) },
16    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_COBRA_BUS) },
17    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_VIPER) },
18    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_VIPER_BUS) },
19    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_GT_MAX_READY) },
20    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_FUJI_MODEM_LIGHT) },
21    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_FUJI_MODEM_GT) },
22    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_FUJI_MODEM_EX) },
23    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_KOI_MODEM) },
24    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_SCORPION_MODEM) },
25    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_ETNA_MODEM) },
26    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_ETNA_MODEM_LITE) },
27    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_ETNA_MODEM_GT) },
28    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_ETNA_MODEM_EX) },
29    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_ETNA_KOI_MODEM) },
30    { USB_DEVICE(OPTION_VENDOR_ID, OPTION_PRODUCT_GTM380_MODEM) },
31    { USB_DEVICE(QUANTA_VENDOR_ID, QUANTA_PRODUCT_Q101) },
32    { USB_DEVICE(QUANTA_VENDOR_ID, QUANTA_PRODUCT_Q111) },
33    { USB_DEVICE(QUANTA_VENDOR_ID, QUANTA_PRODUCT_GLX) },
34    { USB_DEVICE(QUANTA_VENDOR_ID, QUANTA_PRODUCT_GKE) },
35    { USB_DEVICE(QUANTA_VENDOR_ID, QUANTA_PRODUCT_GLE) },
36    { USB_DEVICE(MVID, MPID) },//M7281 3Gmodel
37    { USB_DEVICE(QUANTA_VENDOR_ID, 0xe042),
```

在 option\_ids[]数组里面加一项自己模块的 VID,PID 宏。

```

[*] Networking support --->
  Device Drivers --->
    File systems --->
      HID support --->
      [*] USB support --->
        MMC/SD/USB port drivers ...
          <*> USB Serial Converter support --->
            ...
              USB Serial Converter support
                ...
                  ...
                    ...
                      ...
                        ...
                          ...
                            ...
                              ...
                                ...
                                  ...
                                    ...
                                      ...
                                        ...
                                          ...
                                            ...
                                              ...
                                                ...
                                                  ...
                                                    ...
                                                      ...
                                                        ...
                                                          ...
                                                            ...
                                                              ...
                                                                ...
                                                                    ...
................................................................

```

全部编译进内核，然后开发板上电。上电完成后启动 3G 模块就可以看到下面打印了。

```

root@imx6qdlolo:/mnt# usb 1-1: new high-speed USB device number 2 using ci_hdrc
option 1-1:1.0: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB0
option 1-1:1.1: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB1
option 1-1:1.2: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB2

```

这就是模块成功转换成了 `ttyUSB` 模式

<code>tty25</code>	<code>tty8</code>
<code>tty26</code>	<code>tty9</code>
<code>tty27</code>	<code>ttyUSB0</code>
<code>tty28</code>	<code>ttyUSB1</code>
<code>tty29</code>	<code>ttyUSB2</code>
<code>tty3</code>	<code>ttymxc0</code>
<code>tty30</code>	<code>ttymxc2</code>
<code>tty31</code>	<code>ttymxc3</code>
<code>tty32</code>	<code>ubi_ctrl</code>
<code>tty33</code>	<code>udev_network</code>

- 你在 `/dev` 设备节点下面也能看到。

第二种：有些 3G 厂商模块内部不带自动切换模式功能，那么我们要用 `usb_modeswitch` 来切换

所以我们前面编译的 `usb_modeswitch` 就是用在这里的。

假如我们用的是中兴的 3G 模块，型号为 MF637，那么我们建立一个 **MF637.cfg** 文件先插入 MF637 USB 3G 模块

```
/ # lsusb  
Bus 001 Device 002: ID 19d2:2000  
Bus 001 Device 001: ID 1d6b:0001  
/ #
```

搜索到了 VID,PID VID=19d2 , PID=2000

然后我们拿到厂家给出的模式切换文件，寻找对应的 PID,VID

全部复制粘贴进  
MF637.cfg 文件

找到对应的 PID:VID 后将上面所有的字符，复制粘贴进我刚才建立的 MF637.cfg 文件。

然后将 `MF637.cfg` 文件拷贝进开发板的文件系统/`etc` 目录下。

```
# usb_modeswitch -c /etc/mf637.cfg ━ 执行 usb_modeswitch
```

现在还没有出现 `ttyUSB`, 那是因为你还没有装驱动程序。

```
/ # ls /dev/usb 1-1: new full-speed USB device number 4 using s3c2410-ohci
```

```
/ # lsusb  
Bus 001 Device 004: ID 19d2:0031  
Bus 001 Device 001: ID 1d6b:0001  
/ # █
```

你可以用 lsusb 来查看，你会发现 VID,PID 已经是 19d2:0031 了，而不是之前的 19D2: 2000

```

/ # insmod option.ko
option: Unknown symbol usb_wwan_write (err 0)
option: Unknown symbol usb_wwan_suspend (err 0)
option: Unknown symbol usb_wwan_close (err 0)
option: Unknown symbol usb_wwan_release (err 0)
option: Unknown symbol usb_serial_disconnect (err 0)
option: Unknown symbol usb_wwan_tiocmget (err 0)
option: Unknown symbol usb_wwan_chars_in_buffer (err 0)
option: Unknown symbol usb_serial_suspend (err 0)
option: Unknown symbol usb_wwan_write_room (err 0)
option: Unknown symbol usb_wwan_disconnect (err 0)
option: Unknown symbol usb_wwan_startup (err 0)

```

安装 option.ko 发现找不到调用函数，这个问题是安装驱动程序顺序不对。

先安装 usbserial.ko

```

/ # insmod usbserial.ko
usbcore: registered new interface driver usbserial
usbserial: USB Serial Driver core
/ # insmod usb_wwan.ko
/ # insmod option.ko
usbcore: registered new interface driver option
USB Serial support registered for GSM modem (1-port)
option 1-1:1.0: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB0
option 1-1:1.1: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB1
option 1-1:1.3: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB2
/ #

```

再安装 wwan

最后是 option

你看 ttyUSB 已经出现了，证明模块以及正常转换了，下面我们来 pppd 拨号。

给开发板文件系统添加 ppp 文件

```

root@ubuntu:/home/xiang/IMX6/ppp# ls
chat peers
root@ubuntu:/home/xiana/IMX6/#

```

将该 ppp 文件放在文件系统/etc 目录下

```

root@imx6qdlolo:/test# echo 0 >/sys/ONOFF_3G/ONOFF_3G
kobj_test_store
write: 0

root@imx6qdlolo:/test# echo 1 >/sys/3G_Reset/SG_Reset
kobj_test_store
write: 1

root@imx6qdlolo:/test# echo 0 >/sys/3G_Reset/SG_Reset
kobj_test_store
write: 0

root@imx6qdlolo:/test# usb 1-1: new high-speed USB device number 2 using ci_hdrc
usb 1-1: device descriptor read/all, error -71
usb 1-1: new high-speed USB device number 3 using ci_hdrc
option 1-1:1.0: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB0
option 1-1:1.1: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB1
option 1-1:1.2: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB2

```

重复上面的 3G 模块启动动作，控制模块启动开关和复位开关的 GPIO

执行 pppd call wcdma-dailer & //这是联通 sim 卡的启动脚本 移动的又不一样哦!!

```
root@imx6qdlolo:/test# pppd call wcdma-dailer &
[1] 945
root@imx6qdlolo:/test# timeout set to 15 seconds
abort on (NO CARRIER)
abort on (ERROR)
abort on (NO DIALTONE)
abort on (BUSY)
abort on (NO ANSWER)
send (AT^M)
expect (OK)
alarm
Failed
Script /usr/sbin/chat -s -v -f /etc/ppp/chat/wcdma-connect-chat finished (pid 946), status
= 0x3
Connect script failed
```

发现出现连接脚本错误

```
root@imx6qdlolo:/test# vi /etc/ppp/peers/wcdma-dailer
去开发板文件系统/etc/ppp/peer 目录下查看 wcdma-dailer 文件
```

```
debug
lock
nodetach
/dev/ttyUSB0
115200
user "card"
password "card"
crtscs
show-password
usepeerdns
noauth
noipdefault
```

我们映射了三个 ttyUSB。但是有个 USB 是执行 AT 指令集的，还有个 USB 是执行 3G 模块 ppp 协议的，所以我们要试试到底是哪个 USB 口

```
1 COM46
debug
lock
nodetach
/dev/ttyUSB1
115200
user "card"
password "card"
crtscs
show-password
usepeerdns
noauth
noipdefault
```

我改成 ttyUSB1 试试，然后保存 wcdma-dailer 文件

```

root@imx6qdlolo:/test# pppd call wcdma-dailer &
[1] 948
root@imx6qdlolo:/test# timeout set to 15 seconds
abort on (NO CARRIER)
abort on (ERROR)
abort on (NO DIALTONE)
abort on (BUSY)
abort on (NO ANSWER)
send (AT^M)
expect (OK)
^M
OK
-- got it

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

Script /usr/sbin/chat -s -v -f /etc/ppp/chat/wcdma-connect-chat finished (pid 949), status
= 0x0
Serial connection established.
using channel 1
Using interface ppp0
Connect: ppp0 <--> /dev/ttyUSB1
sent [LCP ConfReq id=0x1 <asyncmap 0x0> <magic 0x93dc27d9> <pcomp> <accomp>]
rcvd [LCP ConfRej id=0x1 <magic 0x93dc27d9>]
sent [LCP ConfReq id=0x2 <asyncmap 0x0> <pcomp> <accomp>]
rcvd [LCP ConfReq id=0x2 <auth chap MD5> <accomp>]
sent [LCP ConfAck id=0x2 <auth chap MD5> <accomp>]
rcvd [LCP ConfAck id=0x2 <asyncmap 0x0> <pcomp> <accomp>]
rcvd [CHAP Challenge id=0x1 <f23ff40a7c2e922936e4293c14b445e2>, name = "UMTS_CHAP_SRVR"]
sent [CHAP Response id=0x1 <7a2548c2f5bd267783ec90ec49baf35d>, name = "card"]
rcvd [CHAP Success id=0x1 ""]
CHAP authentication succeeded
CHAP authentication succeeded
sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP ConfRej id=0x1]
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [LCP ProtRej id=0x1 00 01 01 02 00 16 03 06 00 00 00 00 81 06 00 00 00 00 83 06 00 00
00 00]
Protocol-Reject for unsupported protocol 0x1
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]

```

再次执行 `pppd call wcdma-dailer &` 可以进行网络连接了，证明联网功能已经实现，但是还有一个问题

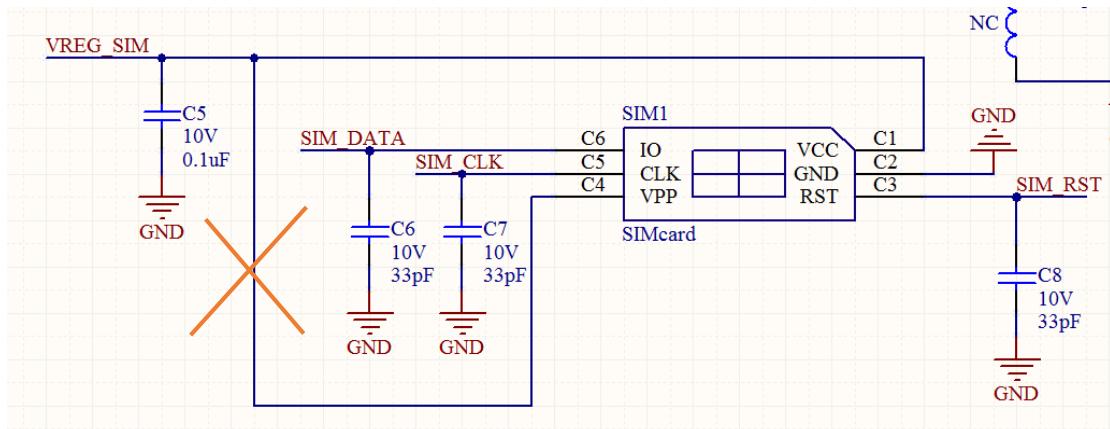
```

.... ...
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP TermAck id=0x2 03 06 00 00 00 00 81 06 00 00 00 00 83 06 00 00 00 00 00 00]
IPCP: timeout sending Config-Requests
sent [LCP TermReq id=0x3 "No network protocols running"]
rcvd [LCP TermAck id=0x3 "No network protocols running"]
Connection terminated.
abort on (BUSY)
abort on (ERROR)
abort on (NO DIALTONE)
send (/K^M)reak to the modem/n
send (+++ATH^M)
Script /usr/sbin/chat -s -v -f /etc/ppp/chat/disconnect-chat finished (pid 962), status =
0x0
Serial link disconnected.

```

连着连着出现这个，这是因为 SIM 卡供电可能存在~~问题~~。

经过检查发现 SIM 卡有个引脚必须悬空，不能接电源。



VPP 引脚要悬空，否则 SIM 卡会进入保护模式，这样就无法获取联通的 DNS 地址。

还有记住要给联通卡加 APN，这个在 AT 指令集文件里面加

## APN 添加方法

```
root@imx6qdlolo:/mnt/ppp/chat# pppd call wcdma-dailer
```

这个 pppd 联网命令是去执行/etc/ppp/目录下面的 wcdma-dailer 文件  
那我们去查看 wcdma-dailer 文件

```
root@imx6qdlolo:/etc/ppp# ls
evdo-dailer  td-dailer  wcdma-dailer
root@imx6qdlolo:/etc/ppp#
```

在/etc/ppp/perrs 目录下

```
debug
lock
nodetach
/dev/ttyUSB0
115200
user "card"
password "card"
crtsccts
show-password
usepeerdns
noauth
noipdefault
novj
novjccomp
noccp
defaultroute
ipcp-accept-local
ipcp-accept-remote
connect '/usr/sbin/chat -s -v -f /etc/ppp/chat/wcdma-connect-chat'
disconnect '/usr/sbin/chat -s -v -f /etc/ppp/chat/disconnect-chat'
```

我们发现 connect 连接的是/etc/ppp/chat/这个路径的 wcdma-connect-char 文件  
我们去查看这个文件

```

TIMEOUT 15
ABORT 'NO CARRIER'
ABORT 'ERROR'
ABORT 'NO DIALTONE'
ABORT 'BUSY'
ABORT 'NO ANSWER'
"" 'AT'
OK 'AT+CGDCONT=1,"IP","UNIM2M.NJM2MAPN"'
OK 'ATDT*99#'
CONNECT
~

```

这是联通卡的 APN

在 wcdma-connect-char 文件里面加入 APN。我这里加入的是联通的 APN，因为我用的是联通的上网卡。注意格式不要写错。

经过以上测试之后还是出现

```

----- 
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0> <ms-dns1 0.0.0.0> <ms-dns2 0.0.0.0>]
rcvd [IPCP TermAck id=0x2 03 06 00 00 00 81 06 00 00 00 00 83 06 00 00 00 00 00]
IPCP: timeout sending Config-Requests
sent [LCP TermReq id=0x3 "No network protocols running"]
rcvd [LCP TermAck id=0x3 "No network protocols running"]
Connection terminated.
abort on (BUSY)
abort on (ERROR)
abort on (NO DIALTONE)
send (/K^M)reak to the modem/n
send (+++ATH^M)
Script /usr/sbin/chat -s -v -f /etc/ppp/chat/disconnect-chat finished (pid 962), status =
0x0
Serial link disconnected.

```

或者出现

```

primary DNS address 58.240.57.33
secondary DNS address 221.6.4.66

root@imx6qdlolo:/test# ping 119.75.217.109
PING 119.75.217.109 (119.75.217.109): 56 data bytes

```

卡在这里

这是因为天线没有匹配好造成的，你看 DNS 都已经获取了。  
然后要在硬件上匹配天线。

```

root@imx6qdlolo:/test# ping 119.75.217.109
PING 119.75.217.109 (119.75.217.109): 56 data bytes
64 bytes from 119.75.217.109: seq=0 ttl=52 time=1398.276 ms
64 bytes from 119.75.217.109: seq=1 ttl=52 time=420.058 ms
64 bytes from 119.75.217.109: seq=2 ttl=52 time=137.325 ms
64 bytes from 119.75.217.109: seq=3 ttl=52 time=137.438 ms
64 bytes from 119.75.217.109: seq=4 ttl=52 time=137.310 ms
64 bytes from 119.75.217.109: seq=5 ttl=52 time=138.721 ms
64 bytes from 119.75.217.109: seq=6 ttl=52 time=138.714 ms
64 bytes from 119.75.217.109: seq=7 ttl=52 time=140.202 ms
64 bytes from 119.75.217.109: seq=8 ttl=52 time=134.075 ms
64 bytes from 119.75.217.109: seq=9 ttl=52 time=136.591 ms
64 bytes from 119.75.217.109: seq=10 ttl=52 time=139.107 ms

```

你看这样我 ping 百度的 IP 地址就没有问题的。

还有个问题就是 ping 域名

```
root@imx6qdlolo:/test# ping www.baidu.com
ping: bad address 'www.baidu.com'
root@imx6qdlolo:/test#
```

上面能 ping IP 地址，但是无法 ping 英文域名

## 用 AT 指令集查看 SIM 卡信息和信号强度

microcom -s 115200 /dev/ttyUSB1

-s 指定波特率 指定 3G/4G 模块 AT 串口设备节点

如果这样写，串口就会卡在界面上使用

```
# microcom -s 115200 /dev/ttyUSB1
AT+CPIN?
+CPIN: READY

OK
AT+CSQ
+CSQ: 31,99

OK
```

就只能这样使用，无法退出 micromodem 程序

microcom -t 10 -s 115200 /dev/ttyUSB1

加上-t 就是 microcom 程序退出时间

```
----- -----
# microcom -t 10 -s 115200 /dev/ttyUSB1
# echo -e "AT+CSQ\r\n" > /dev/ttyUSB1
# microcom -t 10 -s 115200 /dev/ttyUSB1
AT+CSQ
+CSQ: 31,99

OK
# echo -e "AT+CPIN?\r\n" > /dev/ttyUSB1
# microcom -t 10 -s 115200 /dev/ttyUSB1
AT+CPIN?
+CPIN: READY
```

READY 表示 SIM 卡插入 4G 模块

这样你可以先用 echo 把 AT 指令集发给 4G 模块，然后用 microcom 加上-t 查看

### AT 指令集返回信息解析

#### AT+CSQ

<rssi> GSM 制式 : 0-31 , 99 ; TD 制式 : 100-199 ; LTE 制式 : 100-199 GSM 制式的映射关系 :

取值	含义
0	小于或等于 -113 dBm
1	-111 dBm
2...30	-109...-53 dBm
31	大于或等于 -51 dBm
99	未知或不可测

+CSQ:31,99 正常信号强度

+CSQ:99,99 无信号

## LinuxUSB 4G 模块使用

我用的 marvell 88MP1802 的 4G 模块

我在 driver/...option.c 里面增加了该模块的 VID/PID

插入 USB4G 模块

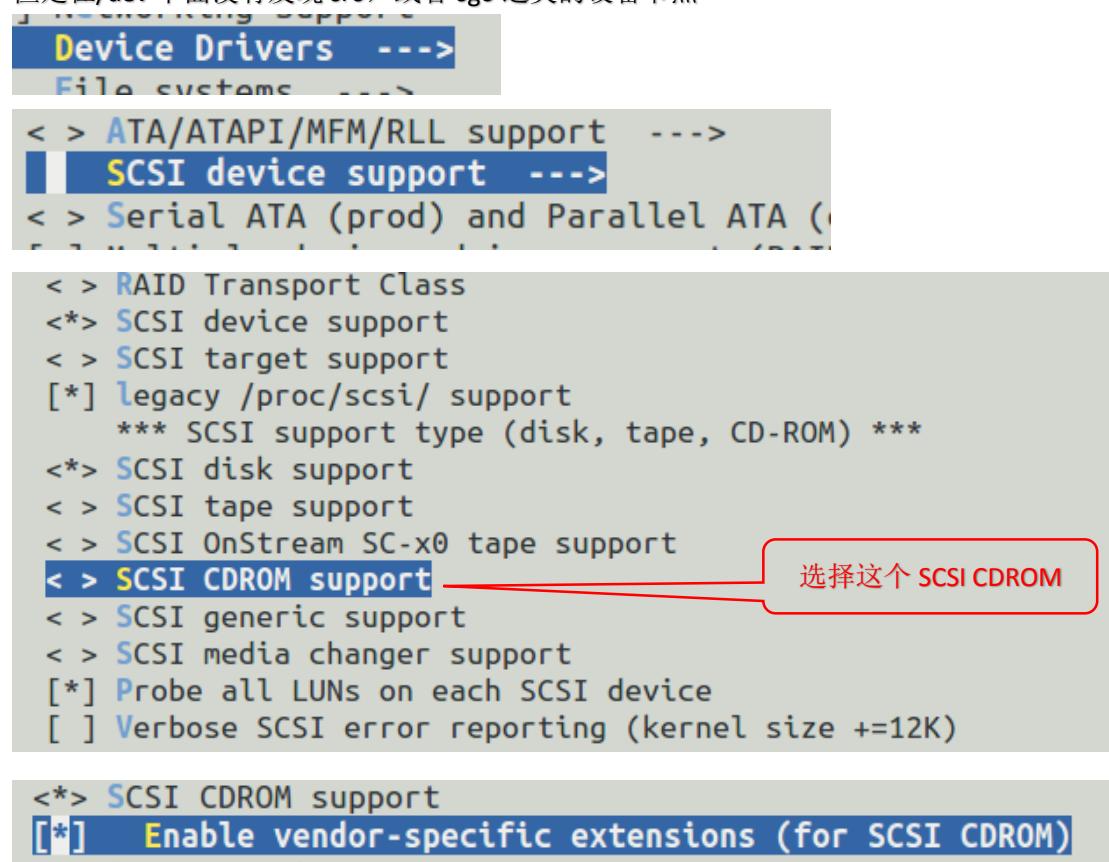
```
# arch_interrupt 85: VBUS error workaround (delay&reset coming)
usb 1-1: new high speed USB device using musb_hdrc and address 2
www device reset speed:3,oldspeed:3
usb 1-1: New USB device found, idVendor=05c6, idProduct=f000
usb 1-1: New USB device strings: Mfr=3, Product=2, SerialNumber=4
usb 1-1: Product: Qualcomm CDMA Technologies MSM
usb 1-1: Manufacturer: Qualcomm, Incorporated
usb 1-1: SerialNumber: 1234567890ABCDEF
usb 1-1: configuration #1 chosen from 1 choice
scsi0 : SCSI emulation for USB Mass Storage devices
scsi 0:0:0:0: CD-ROM          4G      MMC Storage    2.31 PQ: 0 ANSI: 2
#
#
```

显示的是 CD-ROM，证明不是普通的 4G 模块那样在 option.c 加 VID/PID

或者用 usb-modeswitch 就能切换到 ttyUSB 节点

这种设备要用 SCSI 命令发送 CDB 码去让 CD-ROM 转换成 ttyUSB。

但是在 /dev 下面没有发现 sr0，或者 sg0 之类的设备节点



设置完毕，编译内核，将内核烧录进开发板

再次插入 USB4G 模块

```
# arch_interrupt 85: VBUS error workaround (delay&reset coming)
usb 1-1: new high speed USB device using musb_hdrc and address 2
www device reset speed:3,oldspeed:3
usb 1-1: New USB device found, idVendor=05c6, idProduct=f000
usb 1-1: New USB device strings: Mfr=3, Product=2, SerialNumber=4
usb 1-1: Product: Qualcomm CDMA Technologies MSM
usb 1-1: Manufacturer: Qualcomm, Incorporated
usb 1-1: SerialNumber: 1234567890ABCDEF
usb 1-1: configuration #1 chosen from 1 choice
scsi0 : SCSI emulation for USB Mass Storage devices
scsi 0:0:0:0: CD-ROM           4G      MMC Storage      2.31 PQ: 0 ANSI: 2
sr0: scsi-1 drive
Uniform CD-ROM driver Revision: 3.20
```

在 /dev 目录下产生了 sr0

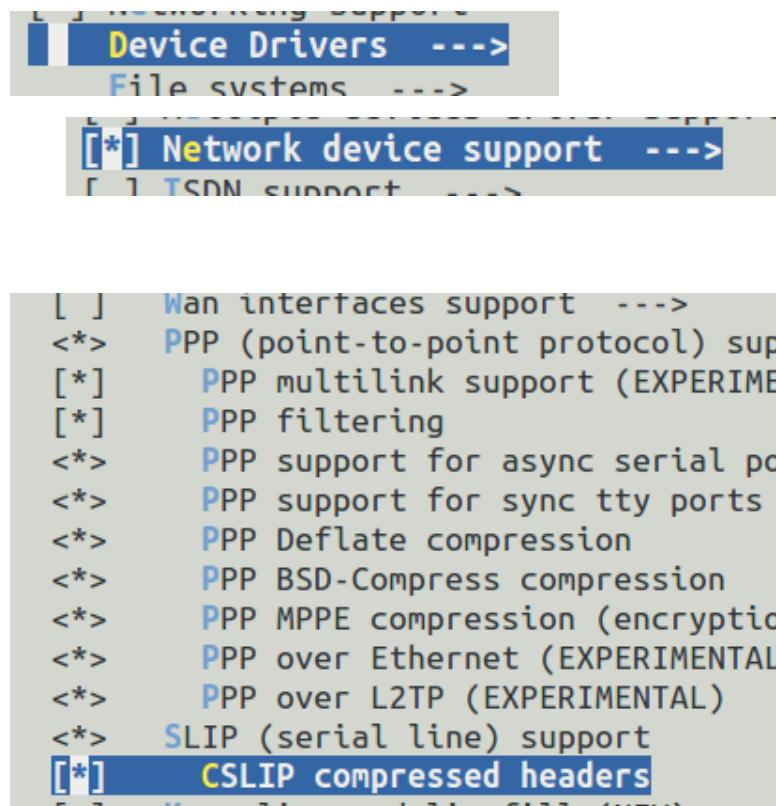
这里要写一个 scsi 的发命令程序，程序我写在了 scsi 驱动 4G 模块章节，在 25 页

```
# ./scsitest
[storage_scsi_inquiry][65]fd=3 cdb=0xebb4fc38 data=0xebb4fb3a sense=0xebb4fa3c
musb_stage2_irq 834: unhandled DISCONNECT transition (a_wait_vrise)
usb 1-1: reset high speed USB device using musb_hdrc and address 2
www device reset speed:3,oldspeed:3
usb 1-1: device descriptor read/64, error -110
www device reset speed:3,oldspeed:3
usb 1-1: device descriptor read/64, error -110
musb_h_ep0_irq 1038: no URB for end 0
usb 1-1: USB disconnect, address 2
[storage_scsi_inquiry][98]V: R:
[storage_scsi_inquiry][114]len=83
[storage_scsi_inquiry][116]S:
[storage_scsi_inquiry][134]H:scsi0 C:00 I:00 L:00
Segmentation fault
# usb 1-1: new high speed USB device using musb_hdrc and address 3
www device reset speed:3,oldspeed:3
usb 1-1: New USB device found, idVendor=05c6, idProduct=6000
usb 1-1: New USB device strings: Mfr=3, Product=2, SerialNumber=4
usb 1-1: Product: Qualcomm CDMA Technologies MSM
usb 1-1: Manufacturer: Qualcomm, Incorporated
usb 1-1: SerialNumber: 1234567890ABCDEF
usb 1-1: configuration #1 chosen from 1 choice
option 1-1:1.0: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB0
option 1-1:1.1: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB1
option 1-1:1.2: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB2
option 1-1:1.3: GSM modem (1-port) converter detected
usb 1-1: GSM modem (1-port) converter now attached to ttyUSB3
scsi1 : SCSI emulation for USB Mass Storage devices
scsi 1:0:0:0: Direct-Access      4G          MMC Storage      2.31 PQ: 0 ANSI: 2
sd 1:0:0:0: [sda] Attached SCSI removable disk
```

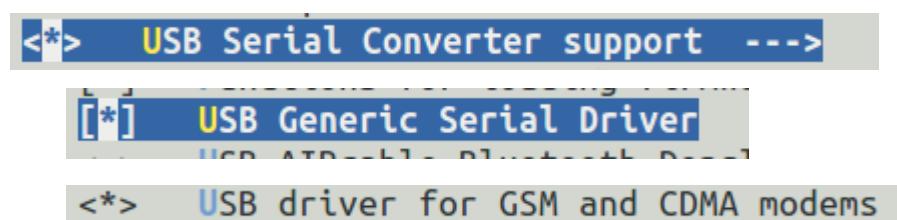
```
ttySDBG  
ttyUSB0  
ttyUSB1  
ttyUSB2  
ttyUSB3  
random
```

在设备节点下产生了 ttyUSB，证明 4G 模块驱动成。

还要记得配置内核的 USB 转串口，和 ppp 协议



全部 PPP, SLIP, CSLIP 都选择上



USB 转串口全部选择上。

然后 make 编译成内核，烧录进开发板

## 4G 模块调试过程在 ARM9 平台下编译 pppd 软件

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd# ls  
ppp-2.4.5  ppp-2.4.5.tar.gz
```

进入 ppp-2.4.5 目录

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd# ls  
Changes-2.3  configure  FAQ  modules  pppdump  README.cbc  README.MPPE  README.pppoe  README.sol2  solaris  
chat  contrib  include  PLUGINS  pppstats  README.eap-srp  README.MSCHAP80  README.pppol2tp  scripts  
common  etc.ppp  linux  pppd  README  README.linux  README.MSCHAP81  README.pwfd  SETUP
```

现在我们要将 Makefile.linux 文件的-s 取消掉

ppp-2.4.5/chat/Makefile.linux

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/chat# ls  
chat.8  chat.c  Makefile.linux  Makefile.sol2
```

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/chat#  
26 install: chat  
27 |     mkdir -p $(BINDIR) $(MANDIR)  
28 |     $(INSTALL) -s -c chat $(BINDIR)  
29 |     $(INSTALL) -c -m 644 chat.8 $(MANDIR)  
30
```

将-s 取消掉

```
27 |     mkdir -p $(BINDIR) $(MANDIR)  
28 |     $(INSTALL) -c chat $(BINDIR)  
29 |     $(INSTALL) -c -m 644 chat.8 $(MANDIR)  
30
```

然后保存退出

-s 取消掉就成这样了

ppp-2.4.5/pppd/plugins/radius/Makefile.linux 将-s 取消掉

```
37 install: all  
38 |     $(INSTALL) -d -m 755 $(LIBDIR)  
39 |     $(INSTALL) -s -c -m 755 radius.so $(LIBDIR)  
40 |     $(INSTALL) -s -c -m 755 radattr.so $(LIBDIR)  
41 |     $(INSTALL) -s -c -m 755 radrealms.so $(LIBDIR)  
42 |     $(INSTALL) -c -m 444 pppd-radius.8 $(MANDIR)
```

然后保存退出

ppp-2.4.5/pppd/plugins/rp-pppoe/Makefile.linux 将-s 取消掉

```
44 install: all  
45 |     $(INSTALL) -d -m 755 $(LIBDIR)  
46 |     $(INSTALL) -s -c -m 4550 rp-pppoe.so $(LIBDIR)  
47 |     $(INSTALL) -d -m 755 $(BINDIR)  
48 |     $(INSTALL) -s -c -m 555 pppoe-discovery $(BINDIR)
```

然后保存退出

ppp-2.4.5/pppd/Makefile.linux 将-s 取消掉

```
102 TARGETS += srp-entry  
102 EXTRAINSTALL = $(INSTALL) -s -c -m 555 srp-entry $(BINDIR)/srp-entry  
103 MANDICES += SRP_ENTRY.8  
104  
104 |     $(EXTRAINSTALL)  
103 |     $(INSTALL) -s -c -m 555 pppd $(BINDIR)/pppd  
104 |     $(INSTALL) -c -m 444 pppd $(BINDIR)/pppd  
104 |     $(INSTALL) -c -m 444 pppd $(MANDIR)
```

取消掉-s

然后保存退出

取消掉-s

ppp-2.4.5/pppdump/Makefile.linux 将-s 取消掉

```
18 install:  
19 |     mkdir -p $(BINDIR) $(MANDIR)  
20 |     $(INSTALL) -s -c pppdump $(BINDIR)  
21 |     $(INSTALL) -c -m 444 pppdump.8 $(MANDIR)
```

取消掉-s

然后保存退出

ppp-2.4.5/pppstats/Makefile.linux 将-s 取消掉

```
24 |     mkdir -p $(MANDIR)  
25 |     $(INSTALL) -s -c pppstats $(BINDIR)  
26 |     $(INSTALL) -c -m 444 pppstats.8 $(MANDIR)
```

取消掉-s

保存并退出

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5# ls  
changes-2.3  configure  FAQ  modules  pppdump  README.cbcpc  README.MPE  README.pppoe  README.sol2  solaris  
chat  contrib  include  PLUGINS  pppstats  README.eap-srp  README.MSCHAPB0  README.pppol2tp  scripts  
common  etc.ppp  linux  pppd  README  README.linux  README.MSCHAPB1  README.pwfd  SETUP  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5# ./configure --host=arm-linux --prefix=$PWD/tmp
```

回到顶层目录配置 configure，指定安装路径当前目录下 tmp，回车

```
Creating Makefiles.  
Makefile <= linux/Makefile.top  
pppd/Makefile <= pppd/Makefile.linux  
pppstats/Makefile <= pppstats/Makefile.linux  
chat/Makefile <= chat/Makefile.linux  
pppdump/Makefile <= pppdump/Makefile.linux  
pppd/plugins/Makefile <= pppd/plugins/Makefile.linux  
pppd/plugins/rp-pppoe/Makefile <= pppd/plugins/rp-pppoe/Makefile.linux  
pppd/plugins/radius/Makefile <= pppd/plugins/radius/Makefile.linux  
pppd/plugins/pppoatm/Makefile <= pppd/plugins/pppoatm/Makefile.linux  
pppd/plugins/pppol2tp/Makefile <= pppd/plugins/pppol2tp/Makefile.linux  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5#
```

表示配置成功

```
pppd/plugins/pppol2tp/Makefile <= pppd/plugins/pppol2tp/Makefile.linux  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5# make CC=asm9260t-gcc-4.3.2
```

指定交叉编译器编译

```
cd pppdump; make all  
make[1]: Entering directory '/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/pppdump'  
asm9260t-gcc-4.3.2 -O -I../include/net -c -o pppdump.o pppdump.c  
asm9260t-gcc-4.3.2 -O -I../include/net -c -o bsd-comp.o bsd-comp.c  
asm9260t-gcc-4.3.2 -O -I../include/net -c -o deflate.o deflate.c  
asm9260t-gcc-4.3.2 -O -I../include/net -c -o zlib.o zlib.c  
asm9260t-gcc-4.3.2 -O -I../include/net -c -o pppdump.o pppdump.c  
make[1]: Leaving directory '/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/pppdump'  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5#
```

编译成功就是这样。

然后在安装 make install

```
~/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/pppdump' 9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5# make install  
install -c -M 644 ccp.h session.h chap-new.h ecn.h fsm.h lpcp.h lpxcp.h lcp.h Magtc.h mos.h patchlevel.h patch.h pppcrypt.h tdb.h spinlock.h /home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/tmp/include  
make[1]: Leaving directory `/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/pppd'  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5#
```

安装完成

```
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/tmp# ls  
include lib sbin share  
root@ubuntu:/home/xiang/ASM9260T/ASM9260T_driver/USB4Gmodule/ppp/pppd/ppp-2.4.5/tmp#
```

到 tmp 目录下

将 lib 库拷贝到开发板文件系统/usr/lib 下，将 sbin 里面的软件拷贝到开发板文件系统 /usr/sbin 目录下。

确保开发板文件系统/usr/sbin 下面有 chat 软件

```
apt-get update  
arecord  flashc  
chat  flash_
```

然后在开发板文件系统/var 下创建 lock 和 run 文件

```
@ubuntu:~/.local/share  
lock  run  
Ubuntu:~/.local/share
```

最后添加配置文件

配置文件分为 chat 和 peers 目录

```
root@ubuntu:~/.local/share  
chat  peers
```

chat 目录下是这两个文件

wcdma-connect-chat 脚本文件内容

```
1 #!/etc/ppp/peers/quentel-chat-connect  
2 ABORT "DELAYED"  
3 ABORT "BUSY"  
4 ABORT "NO CARRIER"  
5 ABORT "NO DIALTONE"  
6 ABORT "ERROR"  
7 ABORT "NO ANSWER"  
8 TIMEOUT 30  
9 "" AT  
10 OK ATE0  
11 #OK ATI;+CSUB;+CSQ;+CPIN?;+COPS?;+CGREG?;&D2  
12 OK ATI;+CSQ;+CPIN?;+COPS?;+CGREG?;&D2  
13 # Insert the APN provided by your network operator, default apn is 3gnet  
14 OK 'AT+CGDCONT=1,"IP","UNIM2M.NJM2MAPN"'  
15 OK ATD*99#  
16 CONNECT
```

要取消掉+CSUB

这里有个问题，这个 APN 是 3G 模块联通物联网卡

4G 模块一定要改成 CMNET。不然 ping 不通 IP 和域名

### disconnect-chat 脚本文件内容

```
1 #!/etc/ppp/peers/quectel-chat-disconnect
2 ABORT "ERROR"
3 ABORT "NO DIALTONE"
4 SAY "\nSending break to the modem\n"
5 """
6 """
7 """
8 SAY "\nGoodbay\n"
```

然后我们再 peers 目录下创建拨号的执行文件

### wcdma-dailer 拨号脚本

```
1 debug
2 确定 chat 软件路径
3 lock
4 nodetach
5 /dev/ttyUSB0
6 115200
7 user "card"
8 crtscs
9 modem
10 show-password
11 usepeerdns
12 noauth
13 noipdefault
14 novj
15 novjccomp
16 noccp
17 defaultroute
18 ipcp-accept-local
19 ipcp-accept-remote
20 connect '/usr/sbin/chat -s -v -f /etc/ppp/chat/wcdma-connect-chat'
21 disconnect '/usr/sbin/chat -s -v -f /etc/ppp/chat/disconnect-chat'
```

根据 4G 模块厂家规格确定是 ttyUSB 几？为数据接口

确定 chat 软件路径

确定你的脚本放在文件系统什么路径下

然后将 chat 和 peers 目录拷贝到文件系统/etc/ppp 目录下，烧写进开发板

```
# cd peers/
# ls
wcdma-dailer
# pppd call wcdma-dailer & 启动开发板执行拨号脚本
```

```
Could not determine remote IP address: defaulting to 10.64.64.64
local IP address 10.96.5.135
remote IP address 10.64.64.64
primary DNS address 221.7.92.100
secondary DNS address 221.5.203.100
```

最后显示这个证明拨号成功

```
# ifconfig
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.96.5.135 P-t-P:10.64.64.64 Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST MTU:1500 Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:60 (60.0 B) TX bytes:102 (102.0 B)

#
```

然后在 ifconfig 下可以看到 ppp0 被分配好了 IP 地址。

```
primary   DNS address 221.7.92.100
secondary  DNS address 221.5.203.100

# ping 119.75.217.109
PING 119.75.217.109 (119.75.217.109): 56 data bytes
64 bytes from 119.75.217.109: seq=0 ttl=52 time=1770.000 ms
64 bytes from 119.75.217.109: seq=1 ttl=52 time=760.000 ms
64 bytes from 119.75.217.109: seq=2 ttl=52 time=100.000 ms
64 bytes from 119.75.217.109: seq=3 ttl=52 time=80.000 ms
64 bytes from 119.75.217.109: seq=4 ttl=52 time=80.000 ms
^C
--- 119.75.217.109 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 80.000/558.000/1770.000 ms
#
```

ping 百度的 ip 地址没有问题

```
# ping www.baidu.com
ping: bad address 'www.baidu.com'
#
```

但是 ping 百度的域名就不行，这是因为你的 DNS 没有放在/etc 目录下

## 3G 和 4G ping 百度域名问题

```
# cd /etc/ppp/
# ls
chat      peers      resolv.conf
#
#                                         我们看到域名文件 resolv.conf 只在 ppp 目录下
chat      peers      resolv.conf
# cp resolv.conf /etc/
#
# cp resolv.conf /etc/
# ls /etc/
fstab      init.d      mdev.conf      ppp      resolv.conf
hotplug    inittab     mtab         profile
#
#
```

我们将 resolv.conf 文件拷贝到 etc 目录下

```
# ping www.baidu.com
PING www.baidu.com (61.135.169.121): 56 data bytes
64 bytes from 61.135.169.121: seq=0 ttl=52 time=90.000 ms
64 bytes from 61.135.169.121: seq=1 ttl=52 time=80.000 ms
64 bytes from 61.135.169.121: seq=2 ttl=52 time=90.000 ms
^C
--- www.baidu.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 80.000/86.666/90.000 ms
```

ping 域名就可以了

## SCSI 驱动 4G 模块代码

```
1 #include<stdio.h>
2 #include<string.h>
3 #include<stdarg.h>
4 #include<fcntl.h>
5 #include<scsi/scsi.h>
6 #include<scsi/scsi.h>
7 #include<error.h>
8 #include<errno.h>
9
10 #include <scsi/scsi_ioctl.h> //scsi发送命令需要使用的头文件
11
12 typedef struct{
13     unsigned char vendor[32];
14     unsigned char product[32];
15     unsigned char revision[32];
16     unsigned char serial[32];
17     int host;
18     int channel;
19     int id;
20     int lun;
21     long size;
22     long ifree;
23 }device_info;
24
25 typedef struct scsi_id
26 {
27     int a[2];
28 } SCSI_ID;
```

```
int main()
{
    int fd=-1;
    unsigned char cdb[12],*ptr;
    unsigned char data_buffer[254],sense_buffer[254];
    //获取供应商和产品的信息都放在这两个字符数组里面，所以这两个大小要一致，否则会出问题
    unsigned char EVPD=0,page_code=0;//返回标准Inquiry数据
    int ret=0,len=0,i=0;
    SCSI_ID scsi_id;
    struct sg_io_hdr p_hdr;//定义要传入的scsi数据结构

    device_info dev={0};
    fd = open('/dev/sr0', O_RDONLY); //插入4G USB网卡会生成设备节点，我们将其打开
    if(fd<0)
    {
        printf("open sr0 failed\n");
    }
    /*下面6个cdb命令根据4G供应商手册来*/
    cdb[0] = 0x71;          //对于支持了CDROM功能的4G网卡(数据卡)，检测VID=0x05C6,PID=0xF000。
    cdb[1] = 0x01;          //名称为4G MMC Storage的光盘，发送SCSI命令0x71，+
    cdb[2] = 0x00;          //（CDB格式为0x71 0x01 0x00 0x00 0x00 0x00）完成光盘到串口的切换，即设备转换为
    cdb[3] = 0x00;          //如上VID=0x05C6,PID=0x6000的四个端口。
    cdb[4] = 0x00;
    cdb[5] = 0x00;
```

```
memset(data_buffer,0,254);
memset(sense_buffer,0,254);
p_hdr.interface_id = 'S'; //S必须写入
p_hdr.dxfer_direction = SG_DXFER_FROM_DEV; //读取4G网卡信息的命令
p_hdr.flags = SG_FLAG_LUN_INHIBIT; //将LUN放到cdb指令区域的第二字节
p_hdr.cmdp = cdb;
p_hdr.cmd_len = 6;
p_hdr.dxferp = data_buffer;
p_hdr.dxfer_len = 254 ;
p_hdr.sbp = sense_buffer;
p_hdr.mx_sb_len = 254;

ret = ioctl(fd,SG_IO,&p_hdr);
if(ret == -1)
{
    printf("ioctl SG_IO error\n");
    return -1;
}
// close(fd); //如果你要持续使用4G模块那么这里就不能close
// 如果这里close了就会关闭4G模块通信
// 所以建议不要close

return 0;
```

就是这样编译执行