

iTOP-4418&6818-QtE4.7WIFI_MT6620 热点

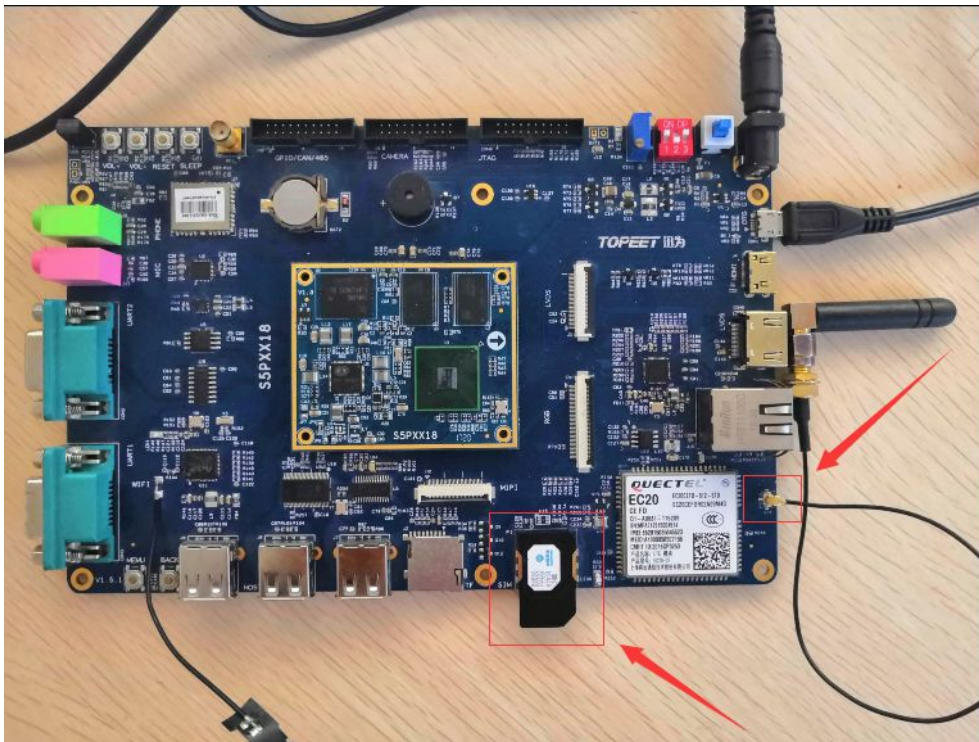
本文档介绍如何在 4418 和 6818qt4.7 上配置 WIFI_MT6620 热点，本文档以 4418_4g 联网和网线联网示范，6818 与 4418 步骤基本一样，不一样的地方已经用红色字体标注出来了。在移植前需要做充分的准备工作，请详细看文档的第一节。如想快速使用可跳过第二节，直接参考第一，三节。

一．移植前准备

4418 使用 4G 移植前准备：

烧写可以使用 4G 上网的 qt 镜像。镜像在网盘：iTOP4418 开发板资料汇总（不含光盘内容）\04_iTOP-4418 开发板 QtE 和 Qtopia 系统源码以及镜像\04_QtE 最新 root 文件目录下。

插入手机卡（该实验使用的是移动 4G 卡），连接好天线。如下图所示：



输入 “./etc/ppp/peers/netec20” 命令。输入 ping www.baidu.com 。测试成功如下图所示：

```
~ #  
~ # ping www.baidu.com  
PING www.baidu.com (111.13.100.92): 56 data bytes  
64 bytes from 111.13.100.92: seq=0 ttl=52 time=115.893 ms  
64 bytes from 111.13.100.92: seq=1 ttl=52 time=51.693 ms  
64 bytes from 111.13.100.92: seq=2 ttl=52 time=57.016 ms  
64 bytes from 111.13.100.92: seq=3 ttl=52 time=65.585 ms
```

4418 使用有线网移植前准备：

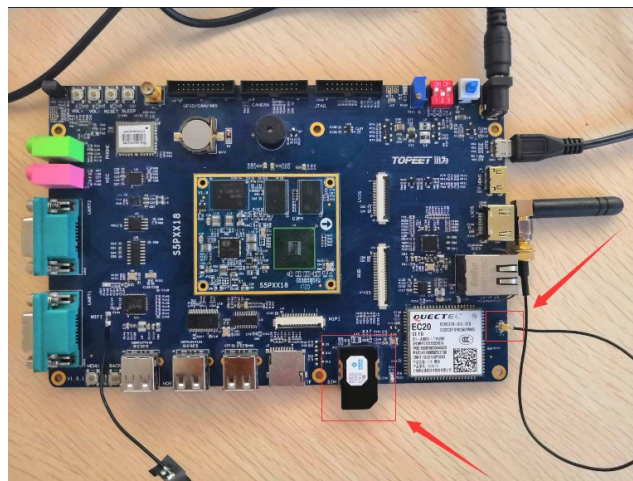
连接好网线，可以使用 ping 命令连通外网即可。如下图所示：

```
~ # ping www.baidu.com  
PING www.baidu.com (180.149.132.151): 56 data bytes  
64 bytes from 180.149.132.151: seq=0 ttl=54 time=14.947 ms  
64 bytes from 180.149.132.151: seq=1 ttl=54 time=15.545 ms  
64 bytes from 180.149.132.151: seq=2 ttl=54 time=15.375 ms  
64 bytes from 180.149.132.151: seq=3 ttl=54 time=15.256 ms  
64 bytes from 180.149.132.151: seq=4 ttl=54 time=15.218 ms  
64 bytes from 180.149.132.151: seq=5 ttl=54 time=14.543 ms  
64 bytes from 180.149.132.151: seq=6 ttl=54 time=15.488 ms  
64 bytes from 180.149.132.151: seq=7 ttl=54 time=15.411 ms  
64 bytes from 180.149.132.151: seq=8 ttl=54 time=15.308 ms
```

6818 使用 4G 移植前准备：

需要更新到最新的 qt 镜像。镜像在网盘：“iTOP6818 开发板资料汇总（不含光盘内容）\04_iTOP-6818 开发板 QtE 和 Qtopia 系统源码以及镜像\04_QtE 最新 root 文件”目录下。

插入手机卡（该实验使用的是移动 4G 卡），连接好天线。如下图所示：



输入命令 `pppd call wcdma &` , 用这条命令最后打印出来的 IP 和 dns,如下图 , 替换这两条命令中的红色部分 , `echo "nameserver 111.11.1.3" >> /etc/resolv.conf` , `route add default gw 10.14.165.1` 。然后输入命令。

```
Could not determine remote IP address: defaulting to 10.64.64.64
not replacing existing default route via 192.168.1.1
local IP address 10.25.145.22
remote IP address 10.64.64.64
primary DNS address 111.11.1.3
secondary DNS address 111.11.1.3
```

输入 `ping www.baidu.com` 。测试成功如下图所示：

```
~ # ping www.baidu.com
PING www.baidu.com (111.13.100.91): 56 data bytes
64 bytes from 111.13.100.91: seq=0 ttl=53 time=169.017 ms
64 bytes from 111.13.100.91: seq=1 ttl=53 time=48.312 ms
64 bytes from 111.13.100.91: seq=2 ttl=53 time=76.503 ms
```

6818 使用有线网移植前准备：

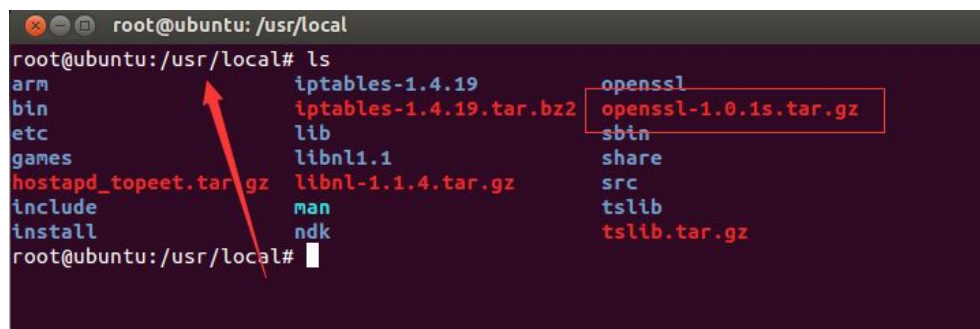
连接好网线，可以 ping 通外网即可。如下图所示：

```
~ # ping www.baidu.com
PING www.baidu.com (180.149.132.151): 56 data bytes
64 bytes from 180.149.132.151: seq=0 ttl=54 time=14.947 ms
64 bytes from 180.149.132.151: seq=1 ttl=54 time=15.545 ms
```

二 . 移植

1 移植 openssl

把 `hostapd_topeet.tar.gz` 压缩包拷贝到 `/usr/local` 并解压输入命令 `tar -vxf openssl-1.0.1s.tar.gz` 到当前目录下。如下图所示。



```
root@ubuntu: /usr/local
root@ubuntu:/usr/local# ls
arm          iptables-1.4.19  openssl
bin          iptables-1.4.19.tar.bz2  openssl-1.0.1s.tar.gz
etc          lib              sbin
games        libnl1.1        share
hostapd_topeet.tar.gz  libnl-1.1.4.tar.gz  src
include      man             tslib
install      ndk             tslib.tar.gz
root@ubuntu:/usr/local#
```


输入命令 `cd openssl-1.0.1s` 进入到 `openssl-1.0.1s` 目录下。在此目录下输入命令 `./config no-asm shared` 。成功后如下图所示：

```
make[1]: Entering directory `/usr/local/openssl-1.0.1s/test'
md2test.c => dummytest.c
rc5test.c => dummytest.c
jpaketest.c => dummytest.c
make[1]: Leaving directory `/usr/local/openssl-1.0.1s/test'

Configured for linux-x86_64.

*** Because of configuration changes, you MUST do the following before
*** building:

        make depend
root@ubuntu:/usr/local/openssl-1.0.1s#
```

输入命令 `vim Makefile` 打开 `Makefile` 文件，修改 `Makefile` 文件配置为下面内容。

```
INSTALLTOP=/usr/local/openssl
OPENSSLDIR=/usr/local/openssl
```

修改前：

```
INSTALL_PREFIX=
INSTALLTOP=/usr/local/ssl

# Do not edit this manually. Use Configure --openssldir=DIR do change this!
OPENSSLDIR=/usr/local/ssl

# NO_IDEA - Define to build without the IDEA algorithm
```

修改后：

```
INSTALL_PREFIX=
INSTALLTOP=/usr/local/openssl

# Do not edit this manually. Use Configure --openssldir=DIR do change this!
OPENSSLDIR=/usr/local/openssl

# NO_IDEA - Define to build without the IDEA algorithm
```

删除 `CFLAG` 中的 `"-m64"` 参数

修改前：

```
CFLAG= -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -m64 -DL_ENDIAN -O3 -Wall
OPENSSL_CFLAGS= -DOPENSSL_NO_EC_NISTP_64_GCC_128 -DOPENSSL_NO_GMP -DOPENSSL_NO_JPAKE -
```

修改后：

```
CC= arm-none-linux-gnueabi-gcc
CFLAG= -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -DL_ENDIAN -O3 -Wall
```

```
CC= arm-none-linux-gnueabi-gcc
EX_LIBS= -ldl
AR= arm-none-linux-gnueabi-ar $(ARFLAGS) r
RANLIB= arm-none-linux-gnueabi-ranlib
NM= arm-none-linux-gnueabi-nm
```

修改前：

```
CC= gcc
PEX_LIBS=
EX_LIBS= -ldl
EXE_EXT=
ARFLAGS=
AR= arm-none-linux-gnueabi-ar $(ARFLAGS) r
RANLIB= arm-none-linux-gnueabi-ranlib
NM= arm-none-linux-gnueabi-nm
PERL= /usr/bin/perl
TAR= tar
TARFLAGS= --no-recursion --record-size=10240
MAKEDEPPROG= gcc
```

修改后：

```
CC= arm-none-linux-gnueabi-gcc
PEX_LIBS=
EX_LIBS= -ldl
EXE_EXT=
ARFLAGS=
AR= arm-none-linux-gnueabi-ar $(ARFLAGS) r
RANLIB= arm-none-linux-gnueabi-ranlib
NM= arm-none-linux-gnueabi-nm
PERL= /usr/bin/perl
TAR= tar
TARFLAGS= --no-recursion --record-size=10240
MAKEDEPPROG= gcc
```

修改完成后保存退出，在当前目录输入 make，编译成功后如下图所示：

```
arm-none-linux-gnueabi-gcc -I.. -I../include -fPIC -DOPENSSL_PIC -DOPENSSL_THRE
ADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -DL_ENDIAN -O3 -Wall -c -o dummyte
st.o dummytest.c
make[2]: Entering directory `/usr/local/openssl-1.0.1s/test'
make[2]: Leaving directory `/usr/local/openssl-1.0.1s/test'
make[1]: Leaving directory `/usr/local/openssl-1.0.1s/test'
making all in tools...
make[1]: Entering directory `/usr/local/openssl-1.0.1s/tools'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/usr/local/openssl-1.0.1s/tools'
root@ubuntu:/usr/local/openssl-1.0.1s#
```

输入命令 make install 安装成功后如下图所示：

```
chmod 644 /usr/local/openssl/lib/pkgconfig/libcrypto.pc
cp libssl.pc /usr/local/openssl/lib/pkgconfig
chmod 644 /usr/local/openssl/lib/pkgconfig/libssl.pc
cp openssl.pc /usr/local/openssl/lib/pkgconfig
chmod 644 /usr/local/openssl/lib/pkgconfig/openssl.pc
root@ubuntu:/usr/local/openssl-1.0.1s#
```

2 移植 libnl

拷贝压缩包 libnl-1.1.4.tar.gz 到/usr/local 目录下。输入命令 tar -vxf libnl-1.1.4.tar.gz 解压到当前目录，如下图所示：

```

root@ubuntu:/usr/local# ls
arm          iptables-1.4.19.tar.bz2  openssl-1.0.1s
bin          lib                      openssl-1.0.1s.tar.gz
etc          libnl1.1                sbin
games        libnl-1.1.4             share
hostapd_topeet.tar.gz  libnl-1.1.4.tar.gz      src
include      man                     tslib
install      ndk                     tslib.tar.gz
iptables-1.4.19  openssl
root@ubuntu:/usr/local#

```

输入命令 `cd libnl-1.1.4` 进入到 `libnl-1.1.4` 文件夹，输入 `./configure -prefix=/usr/local/libnl1.1`。成功后如下图所示：

```

config.status: creating libnl-1.pc
config.status: creating doc/Doxyfile
config.status: creating lib/defs.h
config.status: lib/defs.h is unchanged

-----
SUMMARY:

Included in Compilation:
  libnl:  Yes  -lm -lpthread

Dependencies:
  libm           Yes      (required)
root@ubuntu:/usr/local/libnl-1.1.4#

```

输入命令 `make CC=arm-none-linux-gnueabi-gcc` 编译成功后如下图所示：

```

LD test-cache-mngr
LD test-genl
LD test-nf-cache-mngr
LD test-socket-creation
root@ubuntu:/usr/local/libnl-1.1.4#

```

输入命令 `make install` 进行安装，安装成功后如下图所示：

```

LD test-cache-mngr
LD test-genl
LD test-nf-cache-mngr
LD test-socket-creation
root@ubuntu:/usr/local/libnl-1.1.4#
install -m 0644 netlink/*.h /usr/local/libnl1.1/include/netlink/
install -m 0644 netlink/route/*.h /usr/local/libnl1.1/include/netlink/route/
install -m 0644 netlink/route/sch/*.h /usr/local/libnl1.1/include/netlink/route/
sch/
install -m 0644 netlink/route/cls/*.h /usr/local/libnl1.1/include/netlink/route/
cls/
install -m 0644 netlink/genl/*.h /usr/local/libnl1.1/include/netlink/genl/
install -m 0644 netlink/fib_lookup/*.h /usr/local/libnl1.1/include/netlink/fib_l
lookup/
Entering doc
Entering src
Entering tests
mkdir -p /usr/local/libnl1.1/lib/pkgconfig/
install -m 0644 libnl-1.pc /usr/local/libnl1.1/lib/pkgconfig/
root@ubuntu:/usr/local/libnl-1.1.4#

```

3 移植 hostapd

把 hostapd_topeet.tar.gz 压缩包拷贝到/usr/local 并输入 tar -vxf

hostapd_topeet.tar.gz 解压到当前目录，如下图所示：

```
root@ubuntu:/usr/local# ls
arm          iptables-1.4.19      openssl
bin          iptables-1.4.19.tar.bz2 openssl-1.0.1s
etc          lib                  openssl-1.0.1s.tar.gz
games        libnl1.1             sbin
hostapd_topeet libnl-1.1.4          share
hostapd_topeet.tar.gz libnl-1.1.4.tar.gz  src
include      man                  tslib
install      ndk                  tslib.tar.gz
root@ubuntu:/usr/local#
```

输入命令 cd hostapd_topeet/hostapd 进入到 hostapd 目录，输入 cp defconfig .config 进行缺省配置，如下图所示：

```
root@ubuntu:/usr/local# cd hostapd_topeet/hostapd
root@ubuntu:/usr/local/hostapd_topeet/hostapd# cp defconfig .config
root@ubuntu:/usr/local/hostapd_topeet/hostapd#
```

打开.config 文件，注释 CONFIG_DRIVER_HOSTAP=y，并取消注释 CONFIG_DRIVER_NL80211=y，如下图，保存退出

```
# To override previous values of the variables.

# Driver interface for Host AP driver
#CONFIG_DRIVER_HOSTAP=y

# Driver interface for wired authenticator
#CONFIG_DRIVER_WIRED=y

# Driver interface for madwifi driver
#CONFIG_DRIVER_MADWIFI=y
#CFLAGS += -I.././madwifi # change to the madwifi source directory

# Driver interface for drivers using the nl80211 kernel interface
CONFIG_DRIVER_NL80211=y

# Driver interface for FreeBSD net80211 layer (e.g., Atheros driver)
```

输入命令 vim Makefile 打开当前路径下的 Makefile 文件，根据自己编译器头文件和库的路径，注意前一步编译的 OpenSSL 路径。修改完成后如下图所示：


```

root@ubuntu: /usr/local/hostapd_topeet/hostapd
C=arm-none-linux-gnueabi-gcc

ifndef CFLAGS
CFLAGS = -MMD -O2 -Wall -g
endif

CFLAGS += -I../src
CFLAGS += -I../src/utils

CFLAGS += -I/usr/local/openssl/include
CFLAGS += -I/usr/local/libn1.1/include

LIBS += -L/usr/local/openssl/lib/
LIBS += -L/usr/local/libn1.1/lib/
Uncomment following line and set the path to your kernel tree include
directory if your C library does not include all header files.
CFLAGS += -DUSE_KERNEL_HEADERS -I/usr/src/linux/include

include .config

ifndef CONFIG_OS
ifdef CONFIG_NATIVE_WINDOWS
CONFIG_OS=win32
endif
endif

Makefile" 934L, 19661C 10,3 Top

```

在当前目录输入 make 进行编译，编译成功后如下图所示：

```

CC ../src/drivers/driver_common.c
/usr/local/arm/4.4.1/bin/../lib/gcc/arm-none-linux-gnueabi/4.4.1/../../../../arm-
none-linux-gnueabi/bin/ld: warning: library search path "/usr/local/libn1.1/li
b/" is unsafe for cross-compilation
LD hostapd
CC hostapd_cli.c
CC ../src/common/wpa_ctrl.c
CC ../src/utils/edit_simple.c
LD hostapd_cli
root@ubuntu: /usr/local/hostapd_topeet/hostapd#

```

在当前目录输入 make install 进行安装，安装成功后如下图所示：

```

LD hostapd_cli
root@ubuntu: /usr/local/hostapd_topeet/hostapd# make install
mkdir -p /usr/local/bin
for i in hostapd hostapd_cli; do cp -f $i /usr/local/bin/$i; done
root@ubuntu: /usr/local/hostapd_topeet/hostapd#

```

4.移植 iptables

拷贝 iptables-1.4.19.tar.bz2 压缩包到/usr/local 目录下并输入命令 tar -vxvf iptables-1.4.19.tar.bz2 解压到当前目录下。如下图所示：

```

tar: Exiting with failure status due to previous errors
root@ubuntu: /usr/local# ls
arm      iptables-1.4.19      openssl
bin      iptables-1.4.19.tar.bz2  openssl-1.0.1s
etc      lib                  openssl-1.0.1s.tar.gz

```

输入命令 cd iptables-1.4.19 进入到 iptables-1.4.19 目录下，在此目录下输入 mkdir install 命令创建安装目录。如下图所示：


```

root@ubuntu:/usr/local/iptables-1.4.19# mkdir install
root@ubuntu:/usr/local/iptables-1.4.19# ls
aclocal.m4      configure      include        libipq         Makefile.in
autogen.sh      configure.ac   INCOMPATIBILITIES  libiptc        release.sh
build-aux       COPYING       install        libxtables     tests
COMMIT_NOTES    etc           INSTALL        m4             utils
config.h.in     extensions    iptables       Makefile.am
root@ubuntu:/usr/local/iptables-1.4.19#

```

在 iptables-1.4.19 目录下输入命令 `./configure --host=arm-none-linux-gnueabi --prefix=/usr/local/iptables-1.4.19/install/ --enable-static --disable-shared`

注意：填写自己的路径

如下图所示：

```

root@ubuntu:/usr/local/iptables-1.4.19# ./configure --host=arm-none-linux-gnueabi --prefix=/usr/local/iptables-1.4.19/install/ --enable-static --disable-shared

```

成功后如下图所示：

```

checking pkg-config is at least version 0.9.0... yes
checking for libnftnl... no
configure: creating ./config.status
config.status: creating Makefile
config.status: creating extensions/GNUMakefile
config.status: creating include/Makefile
config.status: creating iptables/Makefile
config.status: creating iptables/xtables.pc
config.status: creating libipq/Makefile
config.status: creating libipq/libipq.pc
config.status: creating libiptc/Makefile
config.status: creating libiptc/libiptc.pc
config.status: creating libiptc/libip4tc.pc
config.status: creating libiptc/libip6tc.pc
config.status: creating libxtables/Makefile
config.status: creating utils/Makefile
config.status: creating include/xtables-version.h
config.status: creating include/iptables/internal.h
config.status: creating config.h
config.status: executing depfiles commands
config.status: executing libtool commands

```

在 iptables-1.4.19 目录输入 `make` 进行编译，编译成功后如下图所示：

```

8.in >iptables-extensions.8;
make[2]: Leaving directory `/usr/local/iptables-1.4.19/iptables'
make[2]: Entering directory `/usr/local/iptables-1.4.19'
Makefile:810: warning: overriding commands for target `install-data-hook'
Makefile:806: warning: ignoring old commands for target `install-data-hook'
make[2]: Leaving directory `/usr/local/iptables-1.4.19'
make[1]: Leaving directory `/usr/local/iptables-1.4.19'

```

在 iptables-1.4.19 目录输入 `make install` 进行安装，安装成功后如下图所示：

```

make[3]: Entering directory `/usr/local/iptables-1.4.19'
Makefile:810: warning: overriding commands for target `install-data-hook'
Makefile:806: warning: ignoring old commands for target `install-data-hook'
make[3]: Leaving directory `/usr/local/iptables-1.4.19'
make[2]: Leaving directory `/usr/local/iptables-1.4.19'
make[1]: Leaving directory `/usr/local/iptables-1.4.19'

```

进入到 `install/lib` 目录下，使用命令 `tar zcvf lib_iptables.tar.gz *` 压缩 lib 下的文件。如下图所示：

```
root@ubuntu:/usr/local/iptables-1.4.19/install/lib# ls
libip4tc.a  libip6tc.a  lib_iptables.tar.gz  libxtables.a  pkgconfig
libip4tc.la  libip6tc.la  libiptc.a  libxtables.so  xtables
```

三 . 拷贝文件

将第二节自己生成的文件拷贝到 u 盘，也可以直接用我们提供的，按照自己的路径将/usr/local/openssl/lib 下的 libcrypto.so.1.0.0，libssl.so.1.0.0，/usr/local/iptables-1.4.19/install/lib 路径下的 lib_iptables.tar.gz，/usr/local/hostapd_topeet/hostapd 下的 hostapd。
/usr/local/iptables-1.4.19/install/sbin 下的 xtables-multi
将本教程提供的 hostapd.conf，udhcpd.conf，mt6620_AP_4G，mt6620_AP_eth0 拷贝进 u 盘。如下图所示。

名称	修改日期	类型	大小
libssl.so.1.0.0	2018/11/14 17:44	0 文件	377 KB
libcrypto.so.1.0.0	2018/11/14 17:44	0 文件	1,952 KB
lib_iptables.tar.gz	2018/11/14 18:41	好压 GZ 压缩文件	130 KB
xtables-multi	2018/11/14 18:28	文件	1,457 KB
hostapd	2017/11/22 17:48	文件	3,385 KB
hostapd.conf	2017/11/22 13:14	CONF 文件	1 KB
mt6620_AP_4G	2017/11/22 16:13	文件	2 KB
mt6620_AP_eth0	2018/11/9 16:46	文件	2 KB
udhcpd.conf	2017/11/22 13:13	CONF 文件	2 KB

挂载 u 盘（可以参考使用手册的 11.3.3.1linux 下 qt 挂载 U 盘），在开发板 etc 目录下，使用命令 mkdir hostapd 创建 hostapd 文件夹。
将 libcrypto.so.1.0.0 和 libssl.so.1.0.0 拷贝到开发板的 lib 下
将 hostapd.conf” 拷贝到开发板 “/etc/hostapd/” 目录，
将 “udhcpd.conf” 拷贝到开发板 “/etc/” 目录，
将 “mt6620_AP_4G” 拷贝到开发板 “/etc/init.d/” 目录，
将 “mt6620_AP_eth0” 拷贝到开发板 “/etc/init.d/” 目录
将 “hostapd” 拷贝到开发板 “/etc/hostapd/” 目录，
将压缩包 lib_iptables.tar.gz 拷贝到开发板 bin 目录下并解压

将 xtables-multi 拷贝到开发板的 bin 目录下，输入 chmod 777 xtables-multi 修改权限，并改名为 iptables。如下图所示：

```

/mnt/udisk # ls
hostapd          libcrypto.so.1.0.0  mt6620_AP_eth0
hostapd.conf     libssl.so.1.0.0    udhcpd.conf
lib_iptables.tar.gz mt6620_AP_4G      xtables-multi
/mnt/udisk # cp libcrypto.so.1.0.0 /lib/
/mnt/udisk # cp libssl.so.1.0.0 /lib/
/mnt/udisk # cp udhcpd.conf /etc/
/mnt/udisk # cp mt6620_AP_4G /etc/init.d/
/mnt/udisk # cp mt6620_AP_eth0 /etc/init.d/
/mnt/udisk # cp lib_iptables.tar.gz /bin/
/mnt/udisk # cp xtables-multi /bin/
/mnt/udisk # cd
~ # cd etc/
/etc # mkdir hostapd
/etc # cd
~ # cd /bin/
/bin # chmod 777 xtables-multi
/bin # mv xtables-multi iptables
/bin # cd
~ # cd bin/
/bin # tar -xvf lib_iptables.tar.gz
libip4tc.a
libip4tc.la
libip6tc.a
libip6tc.la
lib_iptables.tar.gz
libiptc.a
libiptc.la
libxtables.a
libxtables.la
pkgconfig/
pkgconfig/xtables.pc
pkgconfig/libip4tc.pc
pkgconfig/libip6tc.pc
pkgconfig/libiptc.pc
xtables/
/bin # cd
~ # cd mnt/udisk/
/mnt/udisk # cp hostapd /etc/hostapd/
/mnt/udisk # cp hostapd.conf /etc/hostapd/
/mnt/udisk #

```

进入开发板的 /etc/init.d/ 目录，这里作者以 4G 上网为例，输入 vi mt6620_AP_4G 打开脚本文件 mt6620_AP_4G。使用网线连接，需要修改脚本文件 mt6620_AP_eth0 并修改相同地方，找到以下命令：

```

chmod 0660 /dev/ttymxcl
/usr/bin/6620_launcher -m 1 -b 921600 -n /etc/firmware/mt6620_patch_hdr.bin -d /dev/ttymxcl &

```

4418 修改成以下内容：

```

chmod 0660 /dev/ttyAMA2
/usr/bin/6620_launcher -m 1 -b 921600 -n /etc/firmware/mt6620_patch_hdr.bin -d /dev/ttyAMA2 &

```

6818 修改成以下内容：

```

chmod 0660 /dev/ttySAC2
/usr/bin/6620_launcher -m 1 -b 921600 -n /etc/firmware/mt6620_patch_hdr.bin -d /dev/ttySAC2 &

```

这里作者就以 4418 (4g 上网) 为例：

修改前：

```

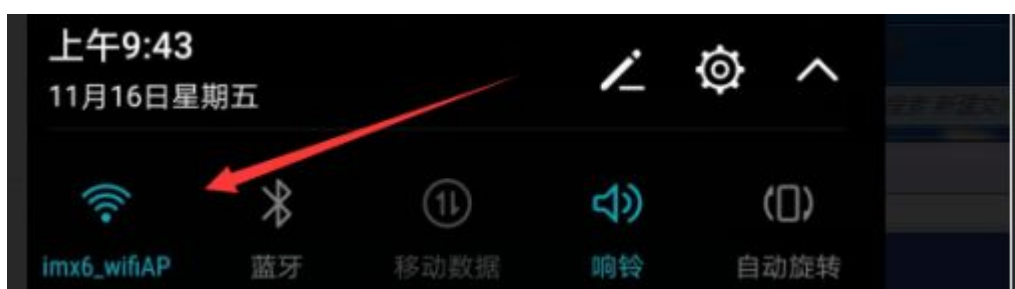
-----
chmod 0666 /dev/wmtWifi
chmod 0666 /dev/gps
chmod 0660 /dev/ttymxcl

/usr/bin/6620_launcher -m 1 -b 921600 -n /etc/firmware/mt6620_patch_hdr.bin -d /dev/ttymxcl &

sleep 4
echo 1 > /dev/wmtWifi

```

修改后：



打开 hostapd.conf，修改 ssid 和 wpa_passphrase 即可修改热点的名称和密码。如下图所示：

```
hostapd.conf
1 interface=wlan0
2 driver=nl80211
3 ssid=imx6_wifiAP
4 channel=9
5 hw_mode=g
6 macaddr_acl=0
7 ignore_broadcast_ssid=0
8 auth_algs=1
9 wpa=3
10 wpa_passphrase=12345678
11 wpa_key_mgmt=WPA-PSK
12 wpa_pairwise=TKIP
13 rsn_pairwise=CCMP
14
```

五．使用网线连接

连接网线，按照自己网络配置为同一网段后，输入 ping www.baidu.com，测试成功如下图，**务必保证可以 ping 通外网才可以进行下一步！**

```
~ # ping www.baidu.com
PING www.baidu.com (180.149.132.151): 56 data bytes
64 bytes from 180.149.132.151: seq=0 ttl=54 time=14.947 ms
64 bytes from 180.149.132.151: seq=1 ttl=54 time=15.545 ms
```

在第三节拷贝文件完成的基础上，进去/etc/init.d/" 目录，输入./mt6620_AP_eth0
输入密码，连接无线即可，连接成功后超级终端上会打印以下信息，如下图所示：

```
192.480000] [STP-PSM] [I]_stp_psm_stp_is_idle: **IDLE is over 60000 msec, go to sleep!!!**
ending OFFER of 192.168.2.10
ending OFFER of 192.168.2.10
ending ACK to 192.168.2.10
```


联系方式

北京迅为电子有限公司致力于嵌入式软硬件设计，是高端开发平台以及移动设备方案提供商；基于多年的技术积累，在工控、仪表、教育、医疗、车载等领域通过 OEM/ODM 方式为客户创造价值。

iTOP-4418/6818 开发板是迅为电子基于三星最新四核和八核处理器 4418/6818 研制的一款实验开发平台，可以通过该产品评估 4418 和 6818 处理器相关性能，并以此为基础开发出用户需要的特定产品。

本文档主要介绍 iTOP-4418/6818 开发板的使用方法，旨在帮助用户快速掌握该产品的应用特点，通过对开发板进行后续软硬件开发，衍生出符合特定需求的应用系统。

如需平板电脑案支持，请访问迅为平板方案网“<http://www.topeet.com>”，我司将有能力为您提供全方位的技术服务，保证您产品设计无忧！

本文档将持续更新，并通过多种方式发布给新老用户，希望迅为电子的努力能给您的学习和开发带来帮助。

迅为电子

2018 年 11 月