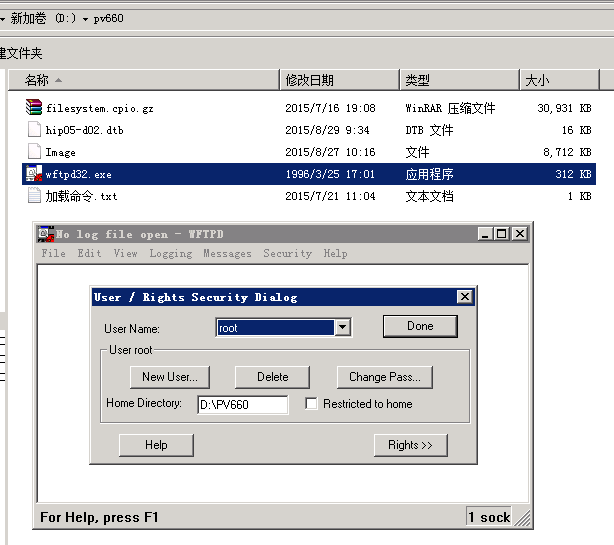
ODP使用手册

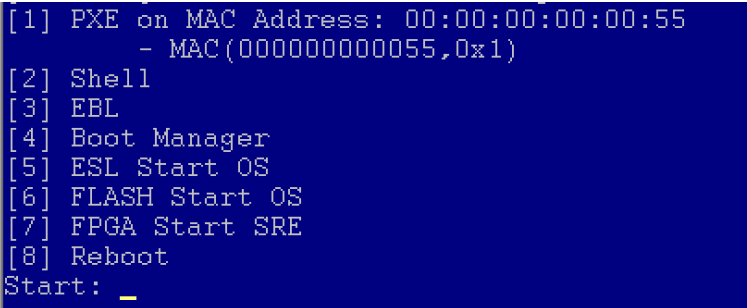
# 环境搭建，这里以D02单板为例

第一步：连接串口，串口波特率：115200 数据位：8 停止位：1 校验位：0

第二步：保证D02单板的网口和加载电脑在同一网段，且网络是能互通。

第三步：把内核文件、DTB文件、文件系统放入PC机中，并打开wftpd32.exe，设置好用户名和密码，指定文件加载路径。



第四步：当串口出现下面界面时

依据提示输出下面的脚本：

3

ifconfig -s eth0 192.168.1.20 255.255.255.0 192.168.1.1

provision 192.168.1.151 -u root -p huawei -f Image -a 0x80000

provision 192.168.1.151 -u root -p huawei -f hip05-d02.dtb -a 0x6000000

provision 192.168.1.151 -u root -p huawei -f filesystem.cpio.gz -a 0x7000000

exit

5

ifconfig eth1 192.168.1.20

其中黑色IP为前面所提示FTP服务器的IP，标红部分别为内核文件、DTB文件、文件系统。

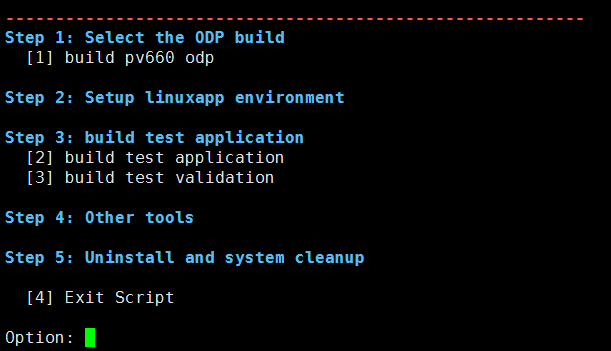
单板加载成功后，配置好IP后，会有下面提示：



这时可以确认单板已加载成功，并且网络已互通，这时可以使用SSH连接单板了。

# 工程编译

编译目录为：/platform/linux-hisilicon/build，执行setup.sh文件：



根据提示，就可编译相关内容。

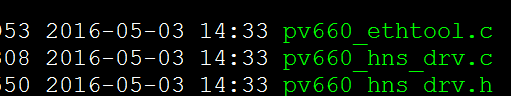
编译后，生成的文件在“/platform/linux-hisilicon/build/objs”目录下。

# 内核驱动编译

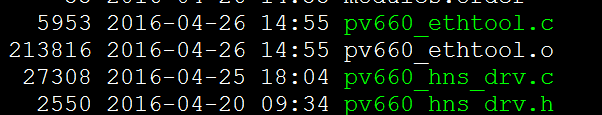
注：platform\linux-hisilicon\build\objs\kernal目录下存放当前我们单板可以运行的内核Image、dtb和文件系统；

**pv660\_hns.ko编译步骤：**

1. 内核驱动文件在：platform\linux-hisilicon\drivers\net\pv660\kernel目录。



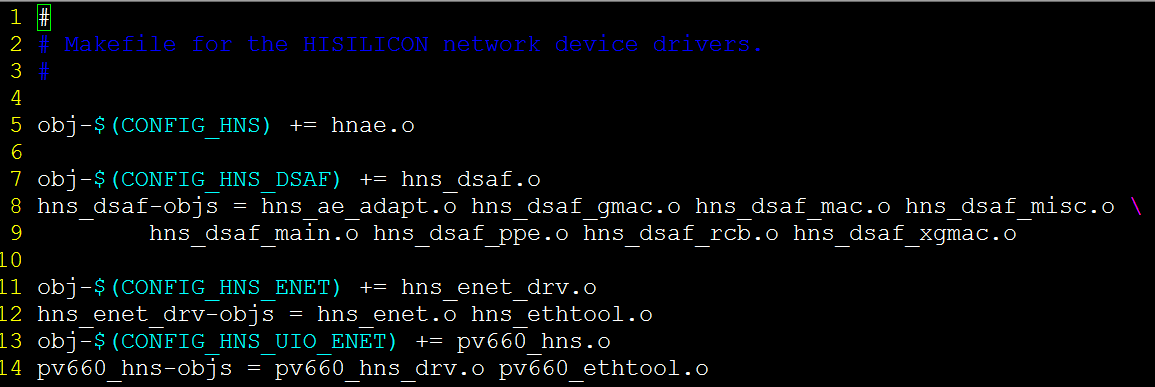
1. 编译服务器上，将驱动文件放入内核目录：drivers/net/ethernet/hisilicon/hns



3、修改hisilicon/hns目录下Makefile文件，添加:

obj-$(CONFIG\_HNS\_UIO\_ENET) += pv660\_hns.o

pv660\_hns-objs = pv660\_hns\_drv.o



4、修改hisilicon/目录下Kconfig文件，添加：

config HNS\_UIO\_ENET

tristate "Hisilicon HNS UIO Ethernet Device Support"

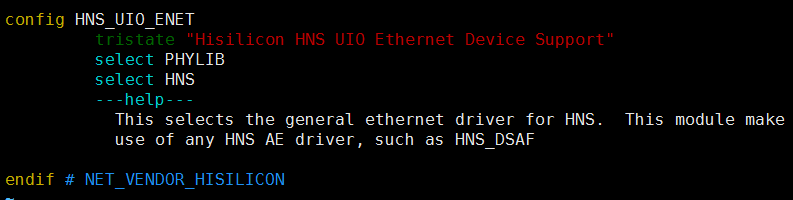
select PHYLIB

select HNS

---help---

This selects the general ethernet driver for HNS. This module make

use of any HNS AE driver, such as HNS\_DSAF



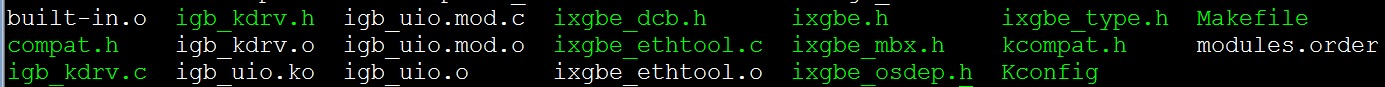
5、重新编译内核，会在hisilicon/hns/目录下生产uio内核驱动。

**82599网卡的内核态驱动igb\_uio.ko编译步骤：**

1、内核驱动文件在：platform\linux-hisilicon\drivers\net\ixgbe\kernel目录。

C:\Users\c00306396\AppData\Roaming\eSpace_Desktop\UserData\c00306396\imagefiles\882732BB-01D2-4CFF-95D6-0FFC631150E5.png

1. 编译服务器上，将驱动文件放入内核目录：drivers\char\igb\_uio



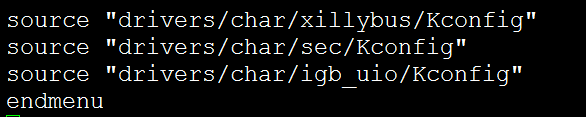
1. 修改drivers\char目录下Makefile文件，添加:

obj-$(CONFIG\_IGBUIO) += igb\_uio/

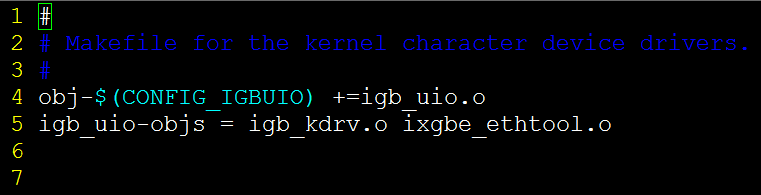


1. 修改drivers\char目录下Kconfig文件，添加：

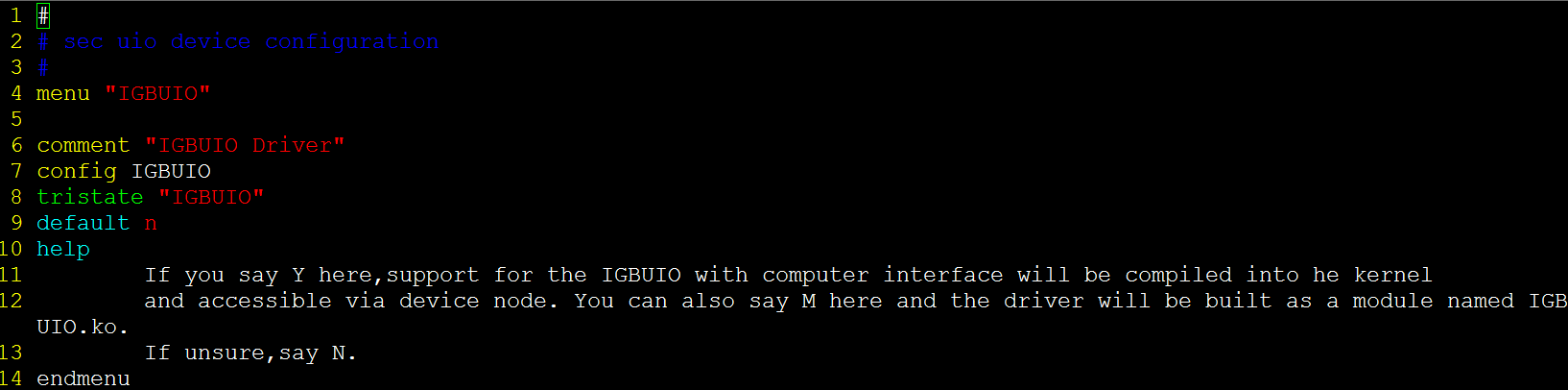
source "drivers/char/igb\_uio/Kconfig"



5、修改drivers\char\igb\_uio目录下Makefile文件，添加:



6、修改drivers\char\igb\_uio目录下Kconfig文件，添加：



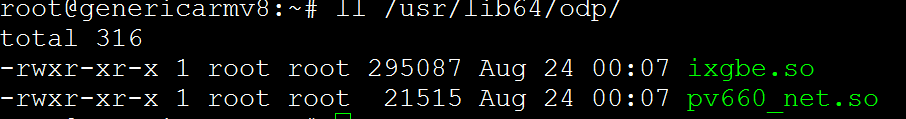
7、重新编译内核，会在drivers\char\igb\_uio目录下生成igb\_uio.ko内核驱动。

# 加载ODP网卡驱动

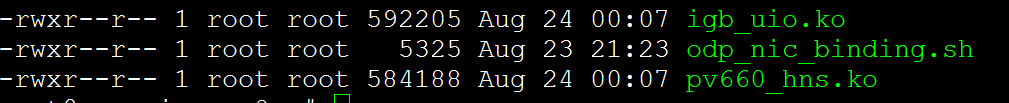
**第一步：**在单板的usr目录下，创建lib64/odp目录



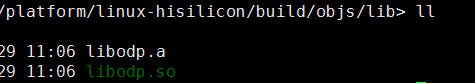
**第二步：** 把编译机的/build/objs/drv目录下的文件上传到单板的上面目录下



**第三步：**把/build/objs/ko目录下的ko文件和platform\linux-hisilicon\build\scripts下的odp\_nic\_binding.sh文件上传到单板，目录随便



**第四步：**把编译机的build/objs/lib目录下的libodp.so文件上传到单板的/usr/lib目录下

****

**第五步：**执行 insmod pv660\_hns.ko和insmod igb\_uio.ko插入SOC用户态网口驱动

和8259用户态网口驱动

小技巧：执行/build/copy.sh，该脚本会把相应的文件拷贝到对应的目录中，具体目录可以查看脚本。

# 编译ODP应用程序

包含ODP应用程序编译，当前是在编译的时候需要显示的指定libodp.so，这样才能把相应的符号表编译到应用程序中，在应用程序中才能正确的调用到ODP当中的API接口。下面就具例应用程序的makefile编写。

LIBS := -L$(ROOT)/build/bin/ -lODP -lpthread -ldl -lrt -lm

显示指定libODP.so的路径和SO名称。

$(APP\_OUT) : $(OBJ\_FILE)

$(ECHO) "LD " $@;\

$(CC) $(LIBS) $(CCFLAGS) -o $(ROOT)/build/app/$@ $^

生成应用程序规则，编译后存放路径为：build/objs/examples 目录下，生成的名字规则为应用程序的 目录名\_app 。

具体应用程序举例可以参照 odp\_hisilicon\_1.7/example 下面的应用程序。

小技巧：直接在ODP的工程 odp\_hisilicon\_1.7/example 下新建自已的应用程序目录，并把odp\_hisilicon\_1.7/example/classifier 下的 Makefile 、 build.sh 两个文件拷入自已新的应用程序目录中，然后在 odp\_hisilicon\_1.7\build\app\_build.sh 中，加入相应的 $ROOT\_DIR/example/新建目录名/build.sh 即可。然后在build目录中调用 app\_build.sh 即会编译相应的应用程序了。

# 运行ODP应用程序

在运行ODP应用程序之前，需要执行以下几个操作：

**1、需要在单板上执行如下几条脚本，创建大页内存。**

mkdir /mnt/huge

mount none /mnt/huge -t hugetlbfs

echo 500 > /proc/sys/vm/nr\_hugepages

echo 0 > /proc/sys/kernel/randomize\_va\_space

**2、设置环境变量，选择HNS\_NET\_PKTIO网卡收发包模式**

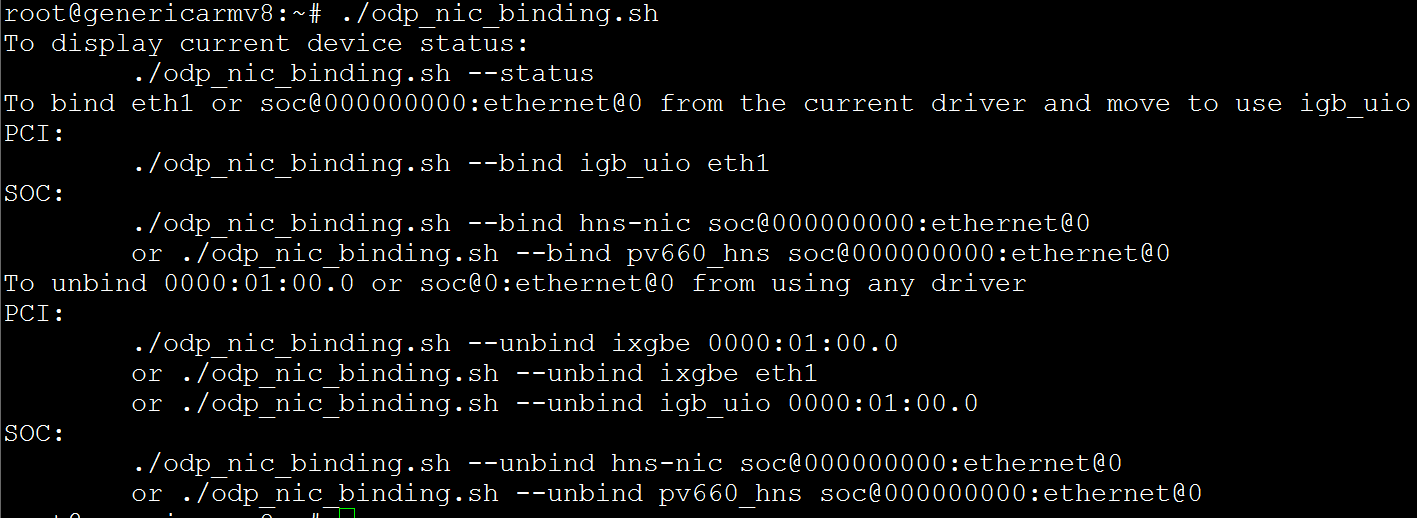
export ODP\_PKTIO\_DISABLE\_SOCKET\_MMAP=DISABLE

export ODP\_PKTIO\_DISABLE\_SOCKET\_MMSG=DISABLE

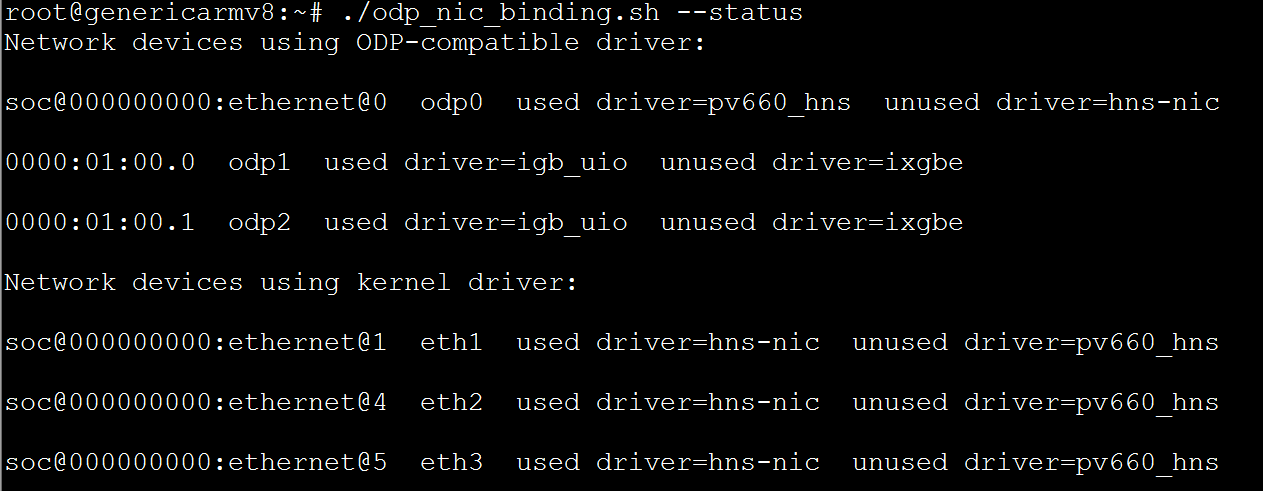
export ODP\_PKTIO\_DISABLE\_NETMAP=DISABLE

**3、解绑linux内核态驱动，绑定uio驱动**

执行./odp\_nic\_binding.sh，可以看到网卡绑定解绑的命令格式：



执行./odp\_nic\_binding.sh --status，可以看到当前网卡设备关联的驱动，以及可以关联的但未关联的驱动：（其中pv660\_hns是SOC网卡用户态驱动，hns-nic是SOC网卡内核态驱动，igb\_uio是82599网卡用户态驱动，ixgbe是82599网卡内核态驱动）



从上图可以看出：

soc网卡的设备soc@000000000:ethernet@0与用户态驱动pv660\_hns关联，

82599网卡的设备0000:01:00.0与用户态驱动igb\_uio关联

82599网卡的设备0000:01:00.1与用户态驱动igb\_uio关联

soc网卡的设备soc@000000000:ethernet@1与内核态驱动hns-nic关联

soc网卡的设备soc@000000000:ethernet@4与内核态驱动hns-nic关联

soc网卡的设备soc@000000000:ethernet@5与内核态驱动hns-nic关联

**命令使用举例：**

* SOC网卡的网口0（soc@000000000:ethernet@0）与pv660\_hns用户态驱动关联的命令：

./odp\_nic\_binding.sh --bind pv660\_hns soc@000000000:ethernet@0

* SOC网卡的网口0（soc@000000000:ethernet@0）与hns-nic内核态驱动关联的命令：

./odp\_nic\_binding.sh --bind hns-nic soc@000000000:ethernet@0

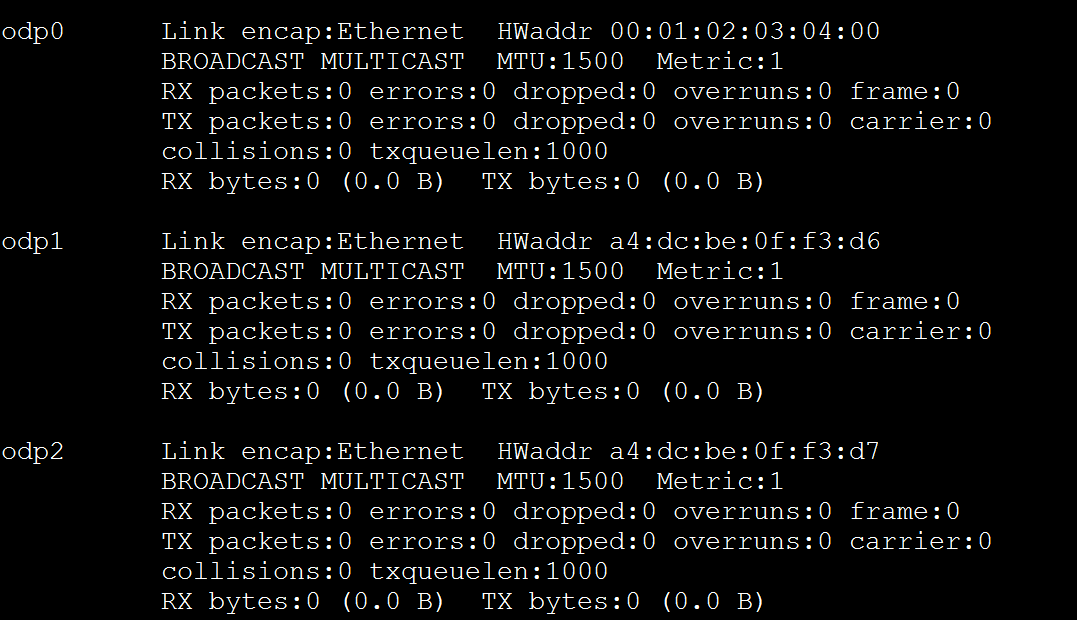
* 82599网卡的PCIE号（0000:01:00.0）与igb\_uio用户态驱动关联的命令：

./odp\_nic\_binding.sh --bind igb\_uio 0000:01:00.0

* 82599网卡的PCIE号（0000:01:00.0）与ixgbe内核态驱动关联的命令：

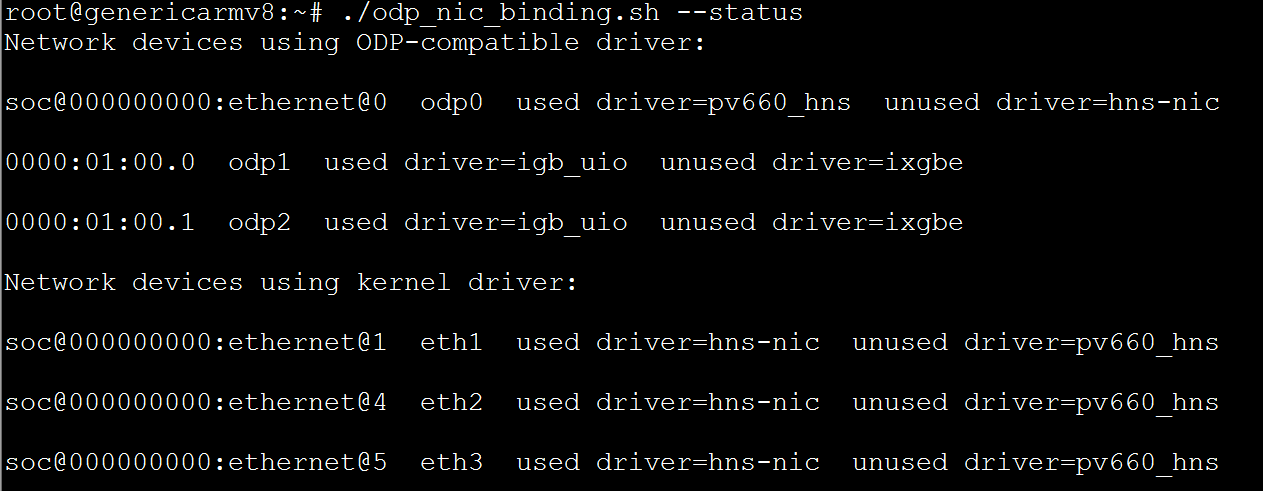
./odp\_nic\_binding.sh --bind ixgbe 0000:01:00.0

用ifconfig -a 命令可以看到：



表示网卡绑定成功！

执行./odp\_nic\_binding.sh –status，可以看到 网卡设备对应的odp\*



**4、运行ODP相关的应用程序**

Demo程序执行命令：

64位ODP层2转发：

./l2fwd\_app -i pktio\_0 -c 1 -m 0

64位ODP层82599的RSS功能：

./l2fwd\_app\_rss -i pktio\_0 -c 1 -m 0

64位ODP加解密：

./sec\_app -i pktio\_0 -c 1 -m 0 -s 0

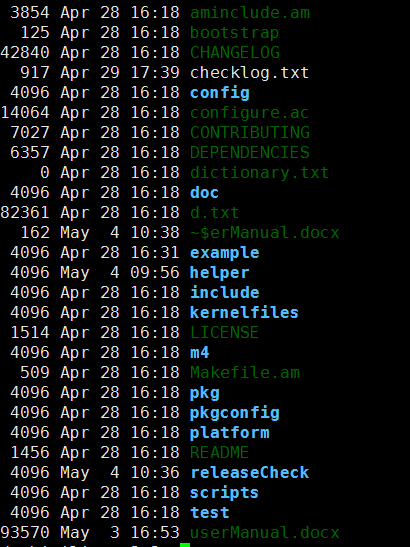
32位ODP层2转发：

./l2fwd\_app32 -i pktio\_0 -c 1 -m 0

32位ODP层2转发：

./sec\_app32 -i pktio\_0 -c 1 -m 0 -s 0

# odp工程文件目录介绍



config：doc：helper：m4：pkgconfig：test：暂时没用到，可不用关注；

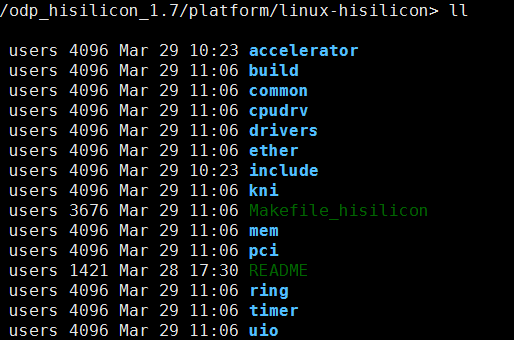
example：demo程序存放路径，在创建ODP的应用程序时参考；

include：ODP对外API接口申明文件；

platform：ODP平台源文件；

kernelfiles：存放当前我们单板可以运行的内核Image、dtb和文件系统；

releaseCheck：版本发布时测试用例、自检日志。



Build:编译目录，编译相关文件都在里面，同时编译后的目标文件也放这里。

Drivers：驱动文件，除uio内核驱动文件外，其他暂时为空，uio用户态驱动提供so文件。