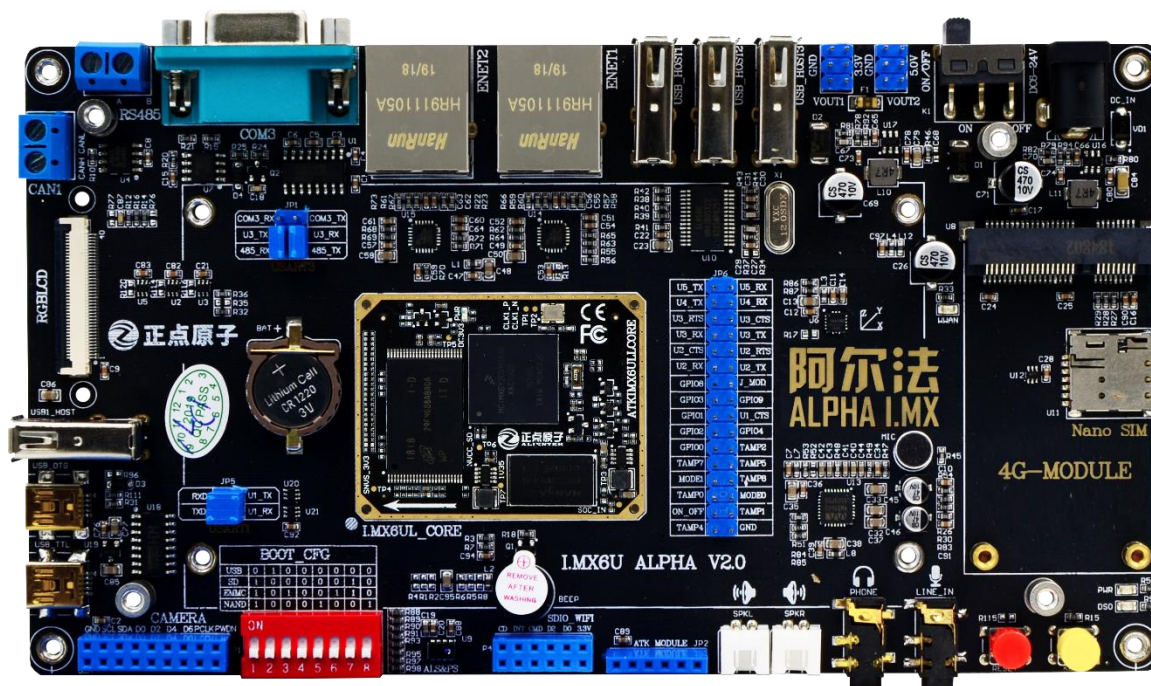


I.MX6U 开发板与 pc 传输文件、单步更新固件参考手册 V1.1



正点原子广州市星翼电子科技有限公司

淘宝店铺 1: <http://eboard.taobao.com>

淘宝店铺 2: <http://openedv.taobao.com>

技术支持论坛 (开源电子网) : www.openedv.com

原子哥在线教学: www.yuanzige.com

官方网站: www.alientek.com

最新资料下载链接: <http://www.openedv.com/posts/list/13912.htm>

E-mail: 389063473@qq.com QQ: [389063473](https://www.qq.com/389063473)

咨询电话: [020-38271790](tel:020-38271790)

传真号码: [020-36773971](tel:020-36773971)

团队: [正点原子团队](#)

正点原子, 做最全面、最优秀的嵌入式开发平台软硬件供应商。

友 情 提 示

如果您想及时免费获取“正点原子”最新资讯, 敬请关注正点原子微信公众平台, 我们将及时给您发布最新消息和重要资料。



关注方法:

- (1) 微信“扫一扫”, 扫描右侧二维码, 添加关注
- (2) 微信→添加朋友→公众号→输入“正点原子”→关注
- (3) 微信→添加朋友→输入“alientek_stm32”→关注



文档更新说明

版本	版本更新说明	负责人	校审	发布日期
V1.0	初稿:	正点原子 linux 团队	正点原子 linux 团队	2020.1.7
V1.1	更正: 1.更正 2.2.1 小节中, uboot 向 emmc 的 boot 分区烧写的错误, 由 mmcblk1boot0 改为/dev/mmcblk1boot0。	正点原子 linux 团队	正点原子 linux 团队	2020.7.2

目录

前言	V
第一章 Linux 开发板文件拷贝篇.....	6
1.1 Linux 通过 U 盘或者 SD 卡拷贝文件.....	6
1.2 开发板通过 scp 指令拷贝文件	11
1.3 开发板使用 MobaXterm 与 Windows 互传文件	17
第二章 I.MX6U 更新固件	19
2.1 NandFlash 更新固件	20
2.2 eMMC 更新固件.....	26
2.3 SD 卡更新固件	29

前言

正点原子 Linux 用户，在开发过程中，新手不知道怎么拷贝文件，所以就出了这份文档，可以说拷贝是新手必备的技能吧，后面还加了如何单步去更新开发板的固件。

这里必须注意：这里使用的是出厂的镜像，出厂的文件系统，编译的时候是使用正点原子修改过的源码。如果是教程的源码，请自行参考！

第一章 Linux 开发板文件拷贝篇

大家知道,在 Windows 这种图形界面的系统拷贝文件,直接用鼠标点击或者拖动文件就可以了。对的,在 Ubuntu 这种含图形界面的 Linux 系统也是可以这样做的。但是一般的 Linux 系统,除了别人去写图形界面,否它只有一个黑色的终端呈现在用户面前。“几乎”让人摸不着头脑!下面就介绍我们在学习或者开发时经常用到的文件拷贝方法。传文件的方法有很多种,下面介绍以下常用的几种。

1.1 Linux 通过 U 盘或者 SD 卡拷贝文件

1.1.1 Linux 开发板通过 U 盘拷贝文件

准备一个 FAT32 格式的 U 盘,注意这里不能用 NTFS 格式的 U 盘!开发板上电,打开串口终端,如果不知道怎么打开串口终端的用户,请看【正点原子】I.MX6U 用户快速体验 V1.x.pdf 里面第二章 ALIENTEK I.MX6U 使用前准备的内容,安装串口软件。

如下准备一个 U 盘(**FAT32 格式,不能用 NTFS 格式的**)如下图,拍个照片给大伙看看。(可用 SD 卡小卡(TF 卡),用读卡器插上也是可以的)。



1.1.1 1 FAT32 格式的 U 盘

开发板上电启动系统后,在串口终端里输入指令 df 查看当前挂载的内容

USER# df

```
root@ALIENTEK-IMX6U:~# df
Filesystem      1K-blocks    used  available  Use%  Mounted on
/dev/root        7350760 497336   6456984    8%  /
devtmpfs         187680      4    187676    1%  /dev
tmpfs            40          0      40        0%  /mnt/.psplash
tmpfs           253436    152   253284    1%  /run
tmpfs           253436    156   253280    1%  /var/volatile
root@ALIENTEK-IMX6U:~#
```

1.1.1 2 插 U 盘前的挂载内容

插上 U 盘如下图



1.1.1.3 开发板的 USB 接口插上 U 盘

开发板串口终端打印信息如下，可以看到 U 盘的实际大小、格式与 U 盘的节点（sda）等一些信息。

```
root@ALIENTEK-IMX6U:~# [ 73.852809] random: nonblocking pool is initialized
[ 3138.552855] usb 2-1.3: new high-speed USB device number 3 using ci_hdrc
[ 3138.723589] usb-storage 2-1.3:1.0: USB Mass Storage device detected
[ 3138.734262] scsi host0: usb-storage 2-1.3:1.0
[ 3140.168040] scsi 0:0:0:0: Direct-Access Kingston DataTraveler 3.0 PMAP PQ: 0 ANSI: 6
[ 3140.186359] sd 0:0:0:0: [sda] 30277632 512-byte logical blocks: (15.5 GB/14.4 GiB)
[ 3140.196831] sd 0:0:0:0: [sda] write Protect is off
[ 3140.204627] sd 0:0:0:0: [sda] No Caching mode page found
[ 3140.210005] sd 0:0:0:0: [sda] Assuming drive cache: write through
[ 3140.229119] sda: sda1
[ 3140.604881] sd 0:0:0:0: [sda] Attached SCSI removable disk
[ 3141.043047] FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
```

1.1.1.4 识别到 U 盘的大小信息

再使用 df 指令查看挂载的节点，可以看到/dev/sda1 挂载的目录为/run/media/sda1。因为我的 U 盘只有一个分区，所以只挂载了一个 sda1 分区，如果你的 SD 卡有两个分区，挂载的目录有两个分别是 sda1 或者 sda2。如果你还有别一个 U 盘，再插到开发板，它的挂载节点会是 sdb*了，以此类推。

USER# df

```
root@ALIENTEK-IMX6U:~# df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/root        7350760    497336   6456984   8% /
devtmpfs         187680         4    187676   1% /dev
tmpfs            40             0         40      0% /mnt/.psplash
tmpfs           253436      164    253272   1% /run
tmpfs           253436      156    253280   1% /var/volatile
/dev/sda1       14643200  8647312  5995888  60% /run/media/sda1
root@ALIENTEK-IMX6U:~#
```

1.1.1.5 U 盘挂载的节点与挂载的目录等信息

所以我们只要进入/run/media/sda1 下去拷贝文件就可以了，直接用 cd 指令进入/run/media/sda1 目录下拷贝文件，使用 ls 指令可查看 U 盘里的内容。

在下图中，已经进入了/run/media/sda1 目录下。可以看到我的 U 盘下有很多文件，其中有一些有“???”的文件，是因为文件系统字符终端不显示中文。

USER# cd /run/media/sda1

USER# ls

```
root@ALIENTEK-IMX6U:~# cd /run/media/sda1
root@ALIENTEK-IMX6U:/run/media/sda1# ls
'9????' '??' '??????.IMX6U??????.v1.0.docx' 'CH340??(USB?????)_XP_WIN???' FOUND.000 'System volume Information' imx6mboot misc
root@ALIENTEK-IMX6U:/run/media/sda1#
```

1.1.1.6 查看 U 盘下的内容

本次新建一个文件，然后将这个文件拷贝到家目录（/home/root）目录下。

使用 touch 指令创建一个 test 文件，然后把这个 test 文件拷贝到/home/root 目录下，如下图，已经通过 touch 创建了一个 test 文件。

USER# touch test

```
root@ALIENTEK-IMX6U:/run/media/sda1# touch test
root@ALIENTEK-IMX6U:/run/media/sda1# ls
'9?????' '??' '??????.MX6U??????.0.docx' 'CH340??(USB????)_XP_WIN???' FOUND.000 'system volume information' imx6mkboot misc test
root@ALIENTEK-IMX6U:/run/media/sda1#
```

1.1.1 7 新建 test 文件

使用 cp 指令拷贝这个 test 复制一份到/home/root 目录下，或者你也可以使用 mv（移动/重命名）指令把它移动到/home/root 目录下。下面只演示 cp 指令(拷贝文件夹用 cp)。拷贝过去了，用 ls 查看/home/root 目录下有 test 这个文件，说明拷贝成功！

USER# cp test /home/root/

USER# ls /home/root

```
root@ALIENTEK-IMX6U:/run/media/sda1# cp test /home/root/
root@ALIENTEK-IMX6U:/run/media/sda1# ls /home/root
driver shell test
root@ALIENTEK-IMX6U:/run/media/sda1#
```

1.1.1 8 拷贝 test 文件到/home/root 目录下

问题又来了！怎么退出 U 盘？暴力做法直接拔出？为了数据写入写出完整，你需要在终端命令下执行指令 sync 来同步数据。

USER# sync

```
root@ALIENTEK-IMX6U:/run/media/sda1# sync
root@ALIENTEK-IMX6U:/run/media/sda1#
```

1.1.1 9 同步数据

退出 U 盘时你需要退出这个挂载的目录，然后用 umount 指令去卸载这个挂载的目录就可以退出了！

使用 cd - 指令返回到上一次目录，再使用 umount /run/media/sda1/ 卸载这个 sda1 目录。然后用 df 指令查看 sda1 这个目录已经不存在了，表明已经卸载了，U 盘可以正常拔出了！

USER# cd -

USER# umount /run/media/sda1

USER# df

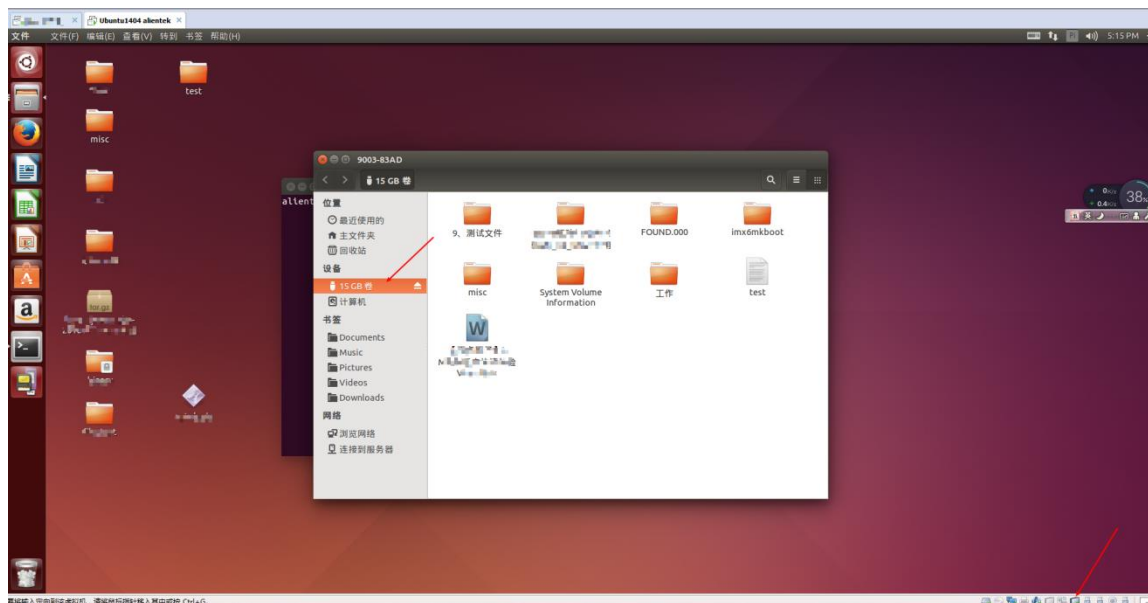
```
root@ALIENTEK-IMX6U:/run/media/sda1# cd -
/home/root
root@ALIENTEK-IMX6U:~# umount /run/media/sda1/
root@ALIENTEK-IMX6U:~# df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/root        7350760 497336   6456984   8% /
devtmpfs        187680      4    187676   1% /dev
tmpfs           40          0         40    0% /mnt/.psplash
tmpfs          253436     164    253272   1% /run
tmpfs          253436     156    253280   1% /var/volatile
root@ALIENTEK-IMX6U:~#
```

1.1.1 10 卸载 U 盘

1.1.2 Ubuntu 下通过 U 盘拷贝文件

本文使用的 Ubuntu 版本是 14.04 和 VMware® Workstation 12 Pro，其他版本的可能图标有点差异！使用 1.1.1 小节的 U 盘 16GB 的，先把鼠标点击到虚拟机，再把 U 盘直接插到 PC 电脑上的 USB 接口。U 盘插入的过程中可能会提示是否连接到虚拟机，选是就可以了。作者默认已经选是了，并且不让它再提示。如下图，已经连接到了 Ubuntu 上。并且 Ubuntu 会弹出一个

文件管理窗口，显示的正是 U 盘的内容。用鼠标点击拖动文件的拷贝方式在这里我就不说了，和 windows 的差不多。我们要用指令 cp/mv 来拷贝/移动。



1.1.2 1 U 盘插到 Ubuntu 上所显示的文件管理窗口

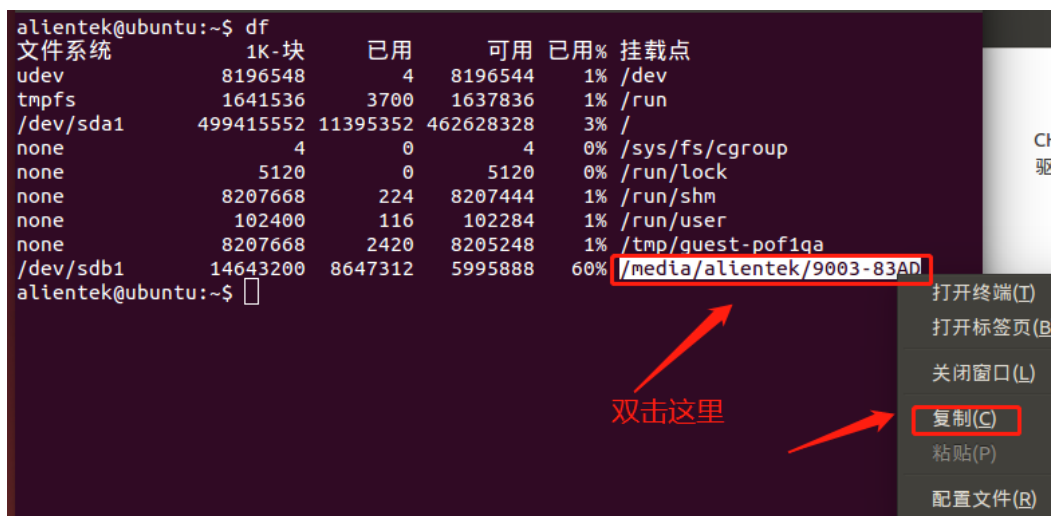
同理，和 1.1.1 小节一样，先用 df 指令来查看 U 盘挂载的目录。

Ubuntu# df

```
alientek@ubuntu: ~
alientek@ubuntu:~$ df
文件系统      1K-块      已用      可用  已用%  挂载点
udev          8196548         4    8196544     1%  /dev
tmpfs         1641536      3700    1637836     1%  /run
/dev/sda1     499415552 11395352 462628328     3%  /
none           4             0         4     0%  /sys/fs/cgroup
none          5120             0        5120     0%  /run/lock
none         8207668      224    8207444     1%  /run/shm
none         102400       116    102284     1%  /run/user
none         8207668      2420   8205248     1%  /tmp/quest-pof1qa
/dev/sdb1     14643200  8647312  5995888    60%  /media/alientek/9003-83AD
alientek@ubuntu:~$
```

1.1.2 2 使用 df 指令查看 U 盘挂载的节点与目录等信息

我们直接双击就可以选中这个目录的路径了，然后右键选择复制，就可以复制它的路径名了。



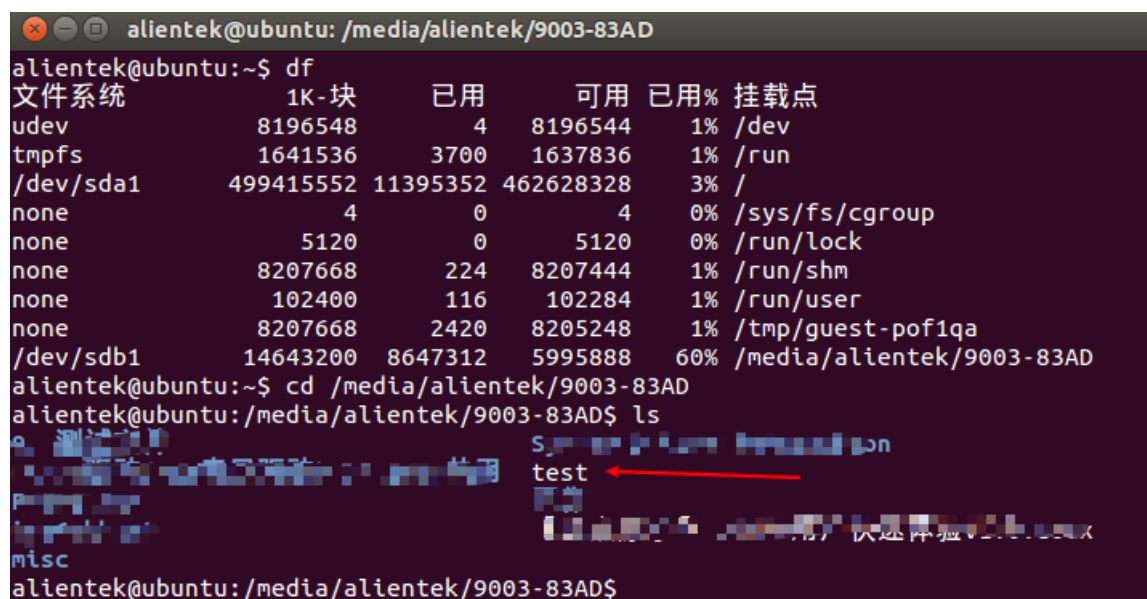
```
alientek@ubuntu:~$ df
文件系统      1K-块      已用      可用 已用% 挂载点
udev          8196548         4 8196544    1% /dev
tmpfs         1641536      3700 1637836    1% /run
/dev/sda1     499415552 11395352 462628328    3% /
none           4           0         4    0% /sys/fs/cgroup
none          5120         0        5120    0% /run/lock
none          8207668      224 8207444    1% /run/shm
none          102400       116 102284    1% /run/user
none          8207668      2420 8205248    1% /tmp/guest-pof1qa
/dev/sdb1     14643200 8647312 5995888   60% /media/alientek/9003-83AD
alientek@ubuntu:~$
```

1.1.2.3 双击复制路径名

输入 `cd` 指令, 然后按 `ctrl+shift+v` 快捷键去粘贴这个路径。再用 `ls` 指令查看当前目录下的文件。其他文件打了码, 可以看到 `test` 这个文件, 是 1.1.1 小节从开发板文件系统里测试拷贝出来的。现在要把它拷贝到 Ubuntu 里。

Ubuntu# `cd + SD 卡的挂载路径`

Ubuntu# `ls`



```
alientek@ubuntu:~$ df
文件系统      1K-块      已用      可用 已用% 挂载点
udev          8196548         4 8196544    1% /dev
tmpfs         1641536      3700 1637836    1% /run
/dev/sda1     499415552 11395352 462628328    3% /
none           4           0         4    0% /sys/fs/cgroup
none          5120         0        5120    0% /run/lock
none          8207668      224 8207444    1% /run/shm
none          102400       116 102284    1% /run/user
none          8207668      2420 8205248    1% /tmp/guest-pof1qa
/dev/sdb1     14643200 8647312 5995888   60% /media/alientek/9003-83AD
alientek@ubuntu:~$ cd /media/alientek/9003-83AD
alientek@ubuntu:/media/alientek/9003-83AD$ ls
test
```

1.1.2.4 查看 test 文件

同样地, 也是使用 `cp/mv` 指令进行拷贝或者移动文件即可(拷贝文件夹是 `cp -r` 文件夹名)。普通用户拷贝到 `/home/` 这样的目录需要权限, 请在前面加上 `sudo`, 再输入密码, 这样就可以拷贝文件了!

Ubuntu# `sudo cp test /home/`

```
alientek@ubuntu:/media/alientek/9003-83AD$ cp test /home/
cp: 无法创建普通文件"/home/test": 权限不够 输入密码
alientek@ubuntu:/media/alientek/9003-83AD$ sudo cp test /home/
[sudo] password for alientek:
alientek@ubuntu:/media/alientek/9003-83AD$ ls /home/
alientek test
alientek@ubuntu:/media/alientek/9003-83AD$
```

1.1.2.5 拷贝 test 文件到 /home 目录下

1.2 开发板通过 scp 指令拷贝文件

初学者很以为只能用 u 盘去给开发板拷贝文件，既然有网口，那么在想是否可以通过网络来传输文件呢，答案是肯定的。下面介绍常用的网络传输指令 scp。

1.2.1 开发板与 ubuntu 在同一路由器下拷贝文件

在路由器能联网的情况下，开发板与主机（ubuntu 或者 windows）都连在同一路由器，或者同一网段内的网络环境下。Windows 要与开发板传文件，请安装 windows Git 软件。这里就不多说了，与 linux 的传输指令是一样的。

开发板上电启动，插上网线，在串口终端下输入指令 ifconfig 来查看开发板自动获取的 ip 地址。开发板上电，插网线，使用 ifconfig 指令查看网络的 ip。如下图 192.168.1.12 就是开发板的 ip 地址，记下来。

USER# ifconfig

```
root@ALIENTEK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 0e:69:97:f2:a2:da
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr 9a:64:6e:c9:35:65
          inet addr:192.168.1.12  Bcast:192.168.1.255  Mask:255.255.255
          inet6 addr: fe80::9864:6eff:fec9:3565/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST DYNAMIC MTU:1500 Metric:1
          RX packets:176 errors:0 dropped:0 overruns:0 frame:0
          TX packets:46 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13826 (13.5 KiB)  TX bytes:8323 (8.1 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:140 (140.0 B)  TX bytes:140 (140.0 B)

root@ALIENTEK-IMX6U:~#
```

1.2.1.1 在串口终端查看开发板的 ip

在 ubuntu 虚拟机上也同样的使用 ifconfig 来查看 ubuntu 的 ip 地址。

Ubuntu# ifconfig

```

alientek@ubuntu:~/test$ ifconfig
eth0      Link encap:以太网  硬件地址 00:0c:29:c5:42:e8
          inet 地址:192.168.1.103  广播:192.168.1.255  掩码:255.255.255.0
          inet6 地址: fe80::20c:29ff:fec5:42e8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
          接收数据包:13100126  错误:0  丢弃:1  过载:0  帧数:0
          发送数据包:1551262  错误:0  丢弃:0  过载:0  载波:0
          碰撞:0  发送队列长度:1000
          接收字节:1637418937 (1.6 GB)  发送字节:2013764403 (2.0 GB)

lo        Link encap:本地环回
          inet 地址:127.0.0.1  掩码:255.0.0.0
          inet6 地址: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  跃点数:1
          接收数据包:1648  错误:0  丢弃:0  过载:0  帧数:0
          发送数据包:1648  错误:0  丢弃:0  过载:0  载波:0
          碰撞:0  发送队列长度:1
          接收字节:193258 (193.2 KB)  发送字节:193258 (193.2 KB)

alientek@ubuntu:~/test$

```

1.2.1 2 查看 ubuntu 上的 ip, 判断是否同一网段

例如, 在 test 文件夹里有个 test.c 文件我们要把这个 test 文件传到开发板的/home/root 目录(我们一般传文件都是传文件到家目录(/home/root)的)。指令格式如下

拷贝文件

scp 文件 用户名@ip 地址:路径

拷贝文件夹

scp -r 文件夹 用户名@ip 地址:路径

例: scp test.c root@192.168.1.12:/home/root

指令格式分析:

test.c 要传输的文件
root 为用户名, 开发板默认的就是 root 用户, 拥有最高权限
@ 一个符号
192.168.1.12 开发板 ip
: 这里要加一个英文字符的“:”, 不要忘记了!
/home/root 要传输到开发板的路径

Ubuntu# scp test.c root@192.168.1.12:/home/root

```

alientek@ubuntu:~/test$ ls
a.out  main.c  repo  test.c
alientek@ubuntu:~/test$ scp test.c root@192.168.1.12:/home/root
The authenticity of host '192.168.1.12 (192.168.1.12)' can't be established.
RSA key fingerprint is ce:53:55:e7:a1:23:e0:cf:82:ed:7d:bb:37:dc:b3:d9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.12' (RSA) to the list of known hosts.
test.c
100% 64 0.1KB/s 00:00
alientek@ubuntu:~/test$

```

1.2.1 3 将 test 文件使用 scp 指令传到开发板

在开发板/home/root 可以看到 test.c 文件已经通过网络传输到开发板了。

USER# ls

```
root@ALIENTEK-IMX6U:~# ls
driver  shell  t.sh  test.c
root@ALIENTEK-IMX6U:~#
```

1.2.1 3 在开发板/home/root 目录下指令查看传过来的 test 文件

1.2.2 开发板与 ubuntu 直连拷贝文件

在上一小节中，用户需要在有网的情况下才能拷贝文件。假若用户没有网络，只有一根网线怎么办？下面作者为只有一根网线或者路由器不能上网但是又想拷贝文件的方法！

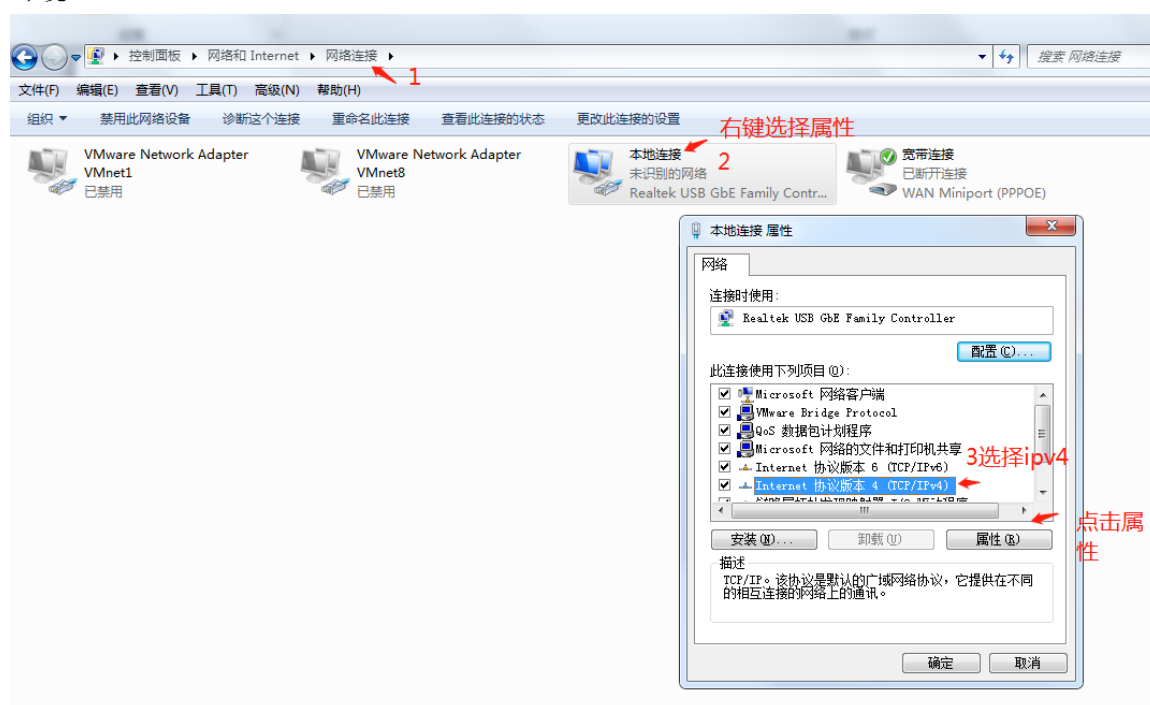
开发板上电，电脑与开发板用一根网线直接连接，或者电脑与开发板同连一个路由器的 lan 口处。

此时电脑上出现一个感叹号，表示无法联网，如下图。如果是一个红 x，表示硬件断开。请检查硬件是否连接正常！



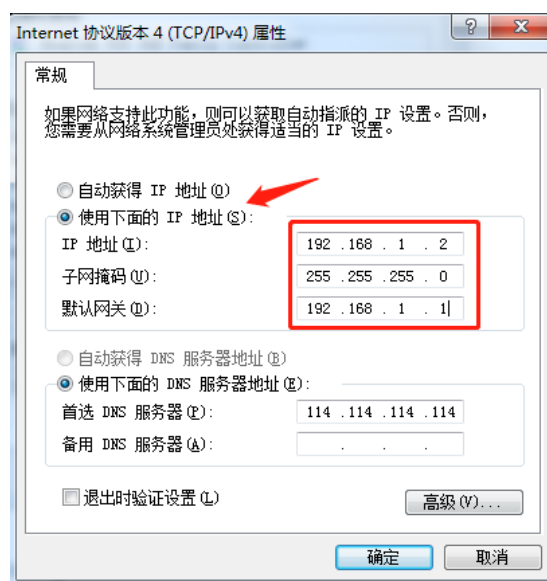
1.2.2 1 开发板直连 PC（电脑）

我们需要为电脑（pc 机）设置静态 ip，下面介绍静态 ip 的方法，如下图。作者用的是 windows7 环境。



1.2.2 2 在本地连接属性里设置属性

设置静态 ip 地址，如下图，点击使用下面的 IP 地址，按如下图设置 ip 地址，子网掩码，默认网关。



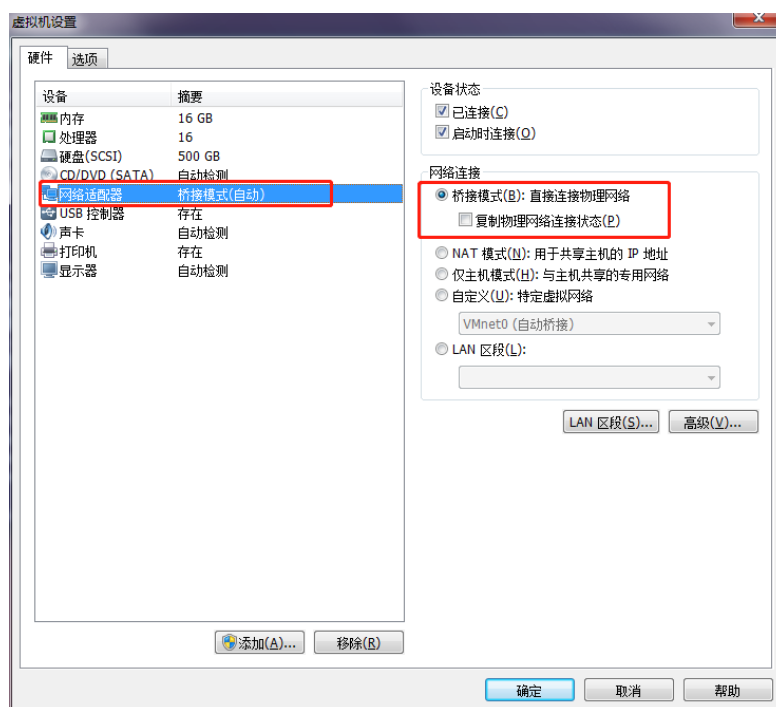
1.2.2.3 设置静态 ip

再到 ubuntu 虚拟机查看，此时，网络已经无法连接，所以显示无网络连接的符号或者是正在尝试连接的符号。



1.2.2.4 虚拟机正在尝试连接网络

请把虚拟机设置成桥接模式，按下图这样设置。



1.2.2 5 ubuntu 虚拟机设置成桥接模式

在 ubuntu 虚拟机右上角设置启用联网，如下图。



1.2.2 6 再点击启用联网

设置 ubuntu 的静态 ip，如下图，`sudo ifconfig eth0 192.168.1.3`，掩码 ubuntu 它自己会补上的。

Ubuntu# `sudo ifconfig eth0 192.168.1.3`

Ubuntu# `ifconfig`

```

alientek@ubuntu:~/test$ sudo ifconfig eth0 192.168.1.3
[sudo] password for alientek:
alientek@ubuntu:~/test$ ifconfig
eth0      Link encap:以太网 硬件地址 00:0c:29:c5:42:e8
          inet 地址:192.168.1.3 广播:192.168.1.255 掩码:255.255.255.0
          inet6 地址: fe80::20c:29ff:fec5:42e8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
          接收数据包:13162381 错误:0 丢弃:1 过载:0 帧数:0
          发送数据包:1551693 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:1000
          接收字节:1641535541 (1.6 GB)  发送字节:2013851479 (2.0 GB)

lo        Link encap:本地环回
          inet 地址:127.0.0.1 掩码:255.0.0.0
          inet6 地址: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  跃点数:1
          接收数据包:1648 错误:0 丢弃:0 过载:0 帧数:0
          发送数据包:1648 错误:0 丢弃:0 过载:0 载波:0
          碰撞:0 发送队列长度:1
          接收字节:193258 (193.2 KB)  发送字节:193258 (193.2 KB)

alientek@ubuntu:~/test$

```

1.2.2 7 设置虚拟机的静态 ip

再在开发板设置开发板的静态 ip。输入指令 `ifconfig eth1 192.168.1.4`。作者插网络到 eth1 处了。所以设置了 eth1 的静态 ip。如下图。

USER# `ifconfig eth1 192.168.1.4`

```

root@ALIENTEK-IMX6U:~# ifconfig eth1 192.168.1.4
root@ALIENTEK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 0e:69:97:f2:a2:da
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr 9a:64:6e:c9:35:65
          inet addr:192.168.1.4 Bcast:192.168.1.255 Mask:255.255.255.0
          inet6 addr: fe80::9864:6eff:fec9:3565/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST DYNAMIC MTU:1500 Metric:1
          RX packets:77453 errors:0 dropped:186 overruns:0 frame:0
          TX packets:291 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5626941 (5.3 MiB)  TX bytes:95055 (92.8 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:140 (140.0 B)  TX bytes:140 (140.0 B)

root@ALIENTEK-IMX6U:~#

```

1.2.2 8 设置开发板的静态 ip

到时设置静态 ip 结束。这样就可以与开发板进行文件传送了。如下图，和 1.2.1 小节一样，使用相同的指令传送相同的文件。（备注：电脑设置了静态 ip 后会不能上网，请设置回自动获取

ip 即可。)

Ubuntu# scp test.c root@192.168.1.4:/home/root

```
alientek@ubuntu:~/test$ sudo ifconfig eth0 192.168.1.3
alientek@ubuntu:~/test$ ifconfig
eth0      Link encap:以太网  硬件地址 00:0c:29:c5:42:e8
          inet 地址:192.168.1.3  广播:192.168.1.255  掩码:255.255.255.0
          inet6 地址: fe80::20c:29ff:fec5:42e8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  跃点数:1
          接收数据包:13163134  错误:0  丢弃:1  过载:0  帧数:0
          发送数据包:1552076  错误:0  丢弃:0  过载:0  载波:0
          碰撞:0  发送队列长度:1000
          接收字节:1641617862 (1.6 GB)  发送字节:2013918249 (2.0 GB)

lo        Link encap:本地环回
          inet 地址:127.0.0.1  掩码:255.0.0.0
          inet6 地址: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  跃点数:1
          接收数据包:1718  错误:0  丢弃:0  过载:0  帧数:0
          发送数据包:1718  错误:0  丢弃:0  过载:0  载波:0
          碰撞:0  发送队列长度:1
          接收字节:198012 (198.0 KB)  发送字节:198012 (198.0 KB)

alientek@ubuntu:~/test$ scp test.c root@192.168.1.4:/home/root
The authenticity of host '192.168.1.4 (192.168.1.4)' can't be established.
RSA key fingerprint is ce:53:55:e7:a1:23:e0:cf:82:ed:7d:bb:37:dc:b3:d9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.4' (RSA) to the list of known hosts.
test.c                                          100% 64    0.1KB/s   00:00
alientek@ubuntu:~/test$
```

1.2.2.9 使用 scp 指令向开发板发送文件

1.3 开发板使用 MobaXterm 与 Windows 互传文件

使用前提: 开发板与 PC 机用网线连接在同一路由器上, 路由器能上网。

请注意, 这里使用的是出厂的文件系统, 支持 ssh 协议。默认开发板文件系统不支持 FTP 传输, 其他文件系统请确认是否支持 ssh 协议。

【正点原子】I.MX6U 嵌入式 Linux 驱动开发指南 V1.x 第 4.9 小节已经安装过 MobaXterm, 使用 ifconfig 在 MobaXterm 查看开发板的 ip, 如下图, 记下开发板的 ip 为 192.168.1.222。

USER# ifconfig

```
root@ATK-IMX6U:~# ifconfig
eth0      Link encap:Ethernet  HWaddr f6:a7:85:f3:de:fb
          inet addr:192.168.1.222  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::f4a7:85ff:fef3:defb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3317  errors:0  dropped:12  overruns:0  frame:0
          TX packets:84  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:228833 (223.4 KiB)  TX bytes:12374 (12.0 KiB)

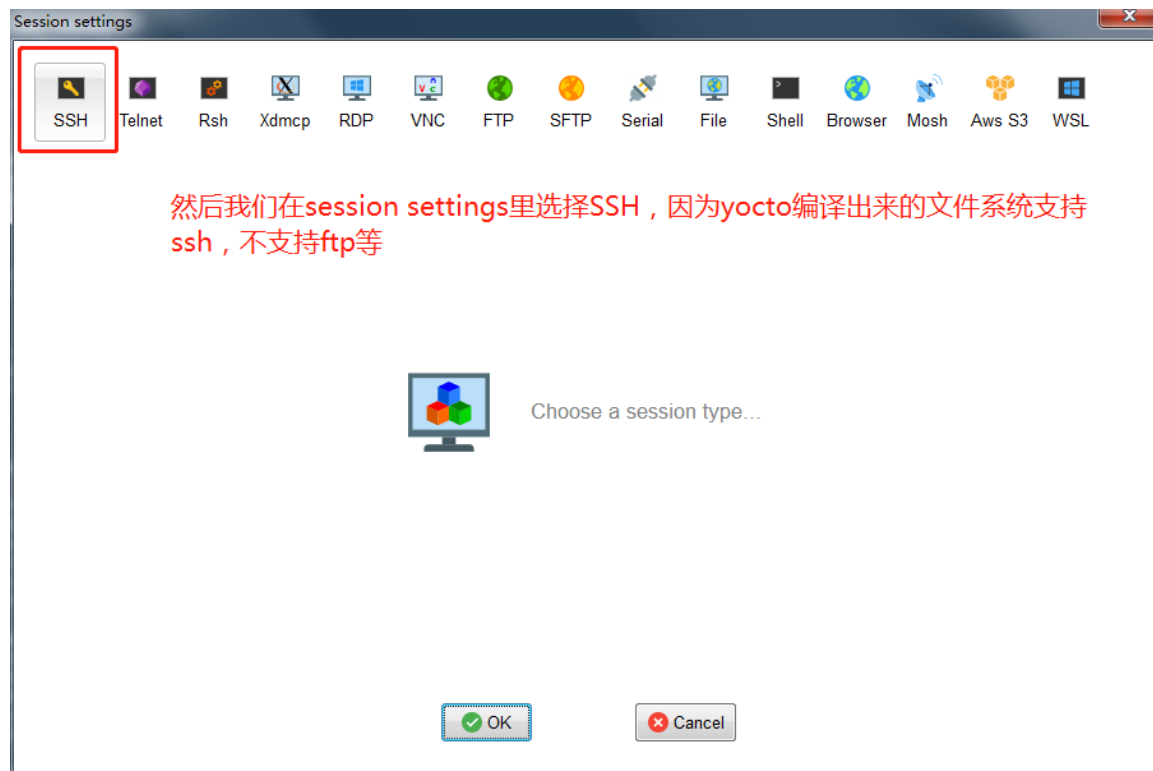
eth1      Link encap:Ethernet  HWaddr 0a:aa:10:8e:90:2d
          UP BROADCAST MULTICAST DYNAMIC  MTU:1500  Metric:1
          RX packets:0  errors:0  dropped:0  overruns:0  frame:0
          TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

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:14  errors:0  dropped:0  overruns:0  frame:0
          TX packets:14  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1932 (1.8 KiB)  TX bytes:1932 (1.8 KiB)

root@ATK-IMX6U:~#
```

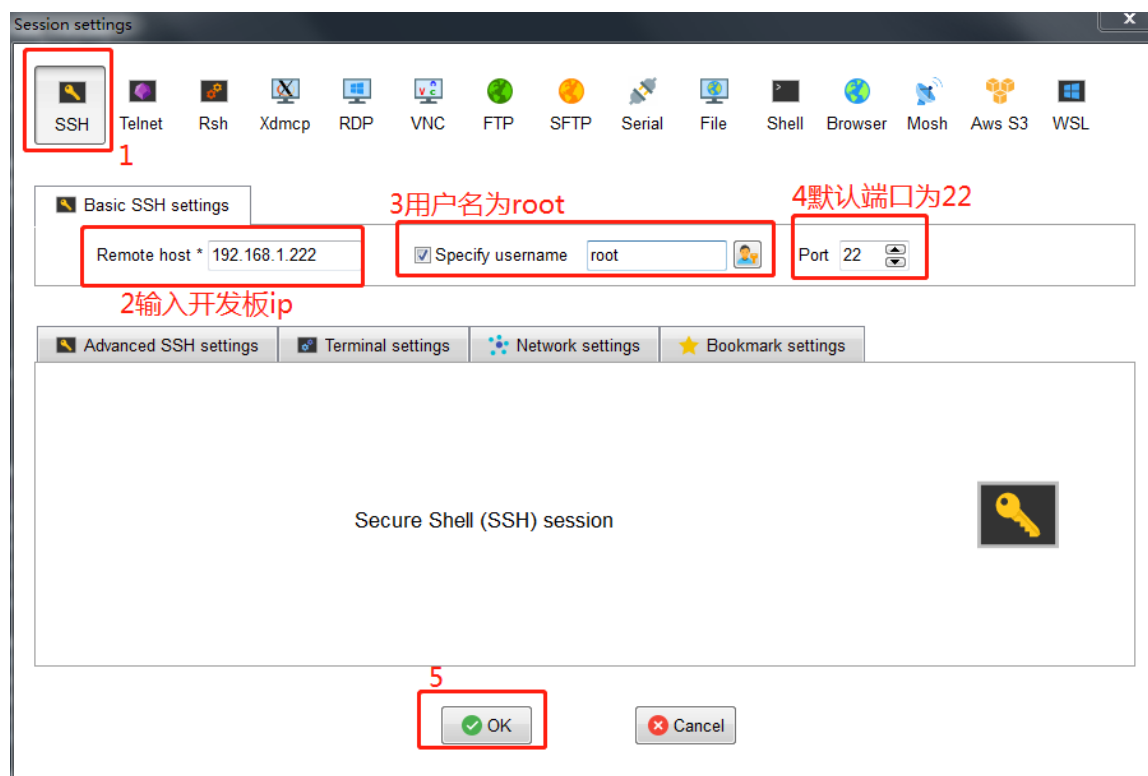
1.3 1 查看开发的 ip

然后我们在 MobaXterm 选择 SSH



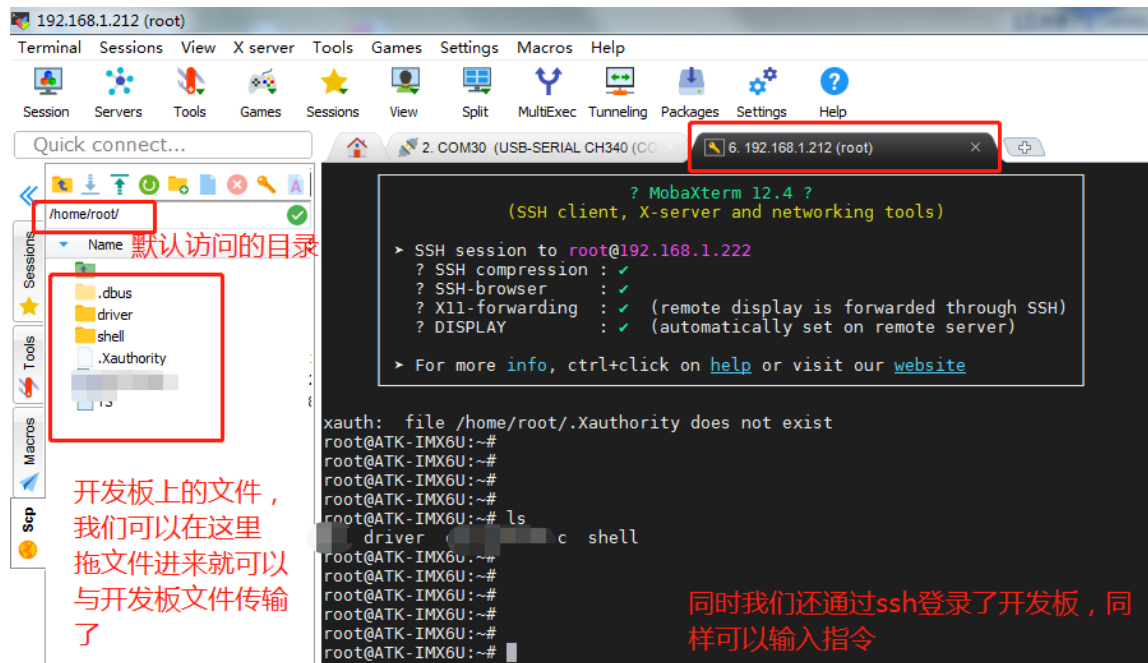
1.3 2 选择 SSH

按如下步骤使用 ssh 去连接开发板。



1.3.3 输入用户名、ip 连接开发板

按如下说明进行 windows 与开发板进行文件传输就可以了，左边的是开发板默认访问的目录，我们可以看到开发板上/home/root 目录下的文件。要想进行文件传输，直接拖拽文件即可。



1.3.4 在 mobaxterm 里可以看到开发板的目录

第二章 I.MX6U 更新固件

在【正点原子】I.MX6U 用户快速体验 V1.x.pdf 的第二章 2.2 小节我们已经成功固化了 Linux 文

件系统，系统也已经成功启动了。但有时候我们开发过程中，尝试自己编译 uboot、内核与文件系统，编译后我们只需要更新 uboot、设备树和文件系统中的某一个，特别是在开发过程中可能要频繁更新，为了节省开发时间，我们可以采用单步更新的方法来更新我们需要更新的部分，而不必花更长的时间去重新制作一张 SD 系统启动卡或者使用脚本固化系统到 eMMC 或者 NAND FLASH 中。

这里说明一下，有些用户会使用 tftp 去更新 uboot、内核等。对于新手来说，这种方式难度还是有点大，不直观。所以这里我们考虑到新手的程度，就不说网络更新了！另外使用 nfs 从网络加载 zImage 和设备树，挂载文件系统的方法，请自行替换相应的文件进行更新就好了。

2.1 NandFlash 更新固件

如果用户使用的是 NandFlash 的核心板，正点原子出厂都有把固件固化到 NandFlash 中。可以直接从 **Nand Flash 启动** 或者 **SD 卡启动** 进入系统去更新这固化在 Nand Flash 的系统固件。

2.1.1 单步更新 uboot 到 NAND FLASH 中

请大家基于正点原子提供的镜像，光盘路径如下：[开发板光盘 A-基础资料->8、开发板系统镜像->uboot-imx-2016.03-2.1.0-xxxxxxx-v1.x](#)，然后把 **u-boot-imx6ull-14x14-ddr256-nand.imx**，文件拷贝到文件系统的/home/root 中（本文作者是拷贝到了/home/root，如下图，下图的 u-boot-imx6ull-14x14-ddr256-nand.imx 实际上是一个 u-boot.imx 文件，也可以根据自己的习惯拷贝到其它目录），为后面更新 uboot 做准备。（备注：可以参考【正点原子】I.MX6U 用户快速体验 V 1.x.pdf 第 2.2.1.1 小节查看 **u-boot-imx6ull-14x14-ddr256-nand.imx** 命名的含义）。

```
root@ALIENTEK-IMX6U:~# pwd
/home/root
root@ALIENTEK-IMX6U:~# ls
u-boot-imx6ull-14x14-ddr256-nand.imx
root@ALIENTEK-IMX6U:~#
```

2.1.1 1 将 NandFlash 用的 uboot 文件拷贝到/home/root

查看 NAND FLASH 分区情况：

USER# cat /proc/mtd

```
root@ALIENTEK-IMX6U:~# cat /proc/mtd
dev:   size   erasesize  name
mtd0: 00400000 00020000  "u-boot"
mtd1: 00020000 00020000  "env"
mtd2: 00100000 00020000  "logo"
mtd3: 00100000 00020000  "dtb"
mtd4: 00800000 00020000  "kernel"
mtd5: 1f1e0000 00020000  "rootfs"
root@ALIENTEK-IMX6U:~#
```

2.1.1 2 查看 NandFlash 的分区情况

使用指令 `cat /proc/mtd` 可以查看 mtd 中保存的系统磁盘分区信息，由以上的输出信息可知，设备 mtd0 是 u-boot 分区，mtd1 是 env（环境变量）分区，mtd2 是 logo 分区，mtd3 是 dtb（设备树文件）分区，mtd4 是 kernel（内核文件）分区，mtd5 是 rootfs（文件系统）分区。

size 是对应 mtd 分区的最大字节数空间，erasesize 是对应分区的最小擦除字节数空间（以块为单位），例如，u-boot 分区，size=00400000(表示 16 进制 0x400000，单位为 Byte)=4MB，erasesize=00020000=128KB

烧写前使用 `flash_erase` 指令先擦除 u-boot 对应的分区 mtd0，可以输入 `flash_erase --help` 指令来查看其用法，如果不在 MTD_DEVICE 后面加上 <start offset> 和 <block count>，直接执行 `flash_erase /dev/mtd0` 的话将会报错。

USER# `flash_erase /dev/mtd0 0 0`

```
root@ALIENTEK-IMX6U:~# flash_erase --help
Usage: flash_erase [options] MTD_DEVICE <start offset> <block count>
Erase blocks of the specified MTD device.
Specify a count of 0 to erase to end of device.

-j, --jffs2      format the device for jffs2
-N, --noskipbad  don't skip bad blocks
-u, --unlock     unlock sectors before erasing
-q, --quiet      do not display progress messages
--silent        same as --quiet
--help          display this help and exit
--version       output version information and exit

root@ALIENTEK-IMX6U:~# flash_erase /dev/mtd0 0 0
Erasing 128 Kibyte @ 3e0000 -- 100 % complete
root@ALIENTEK-IMX6U:~#
```

2.1.1 3 擦除 mtd0 分区

USER# `kobs-ng init -x -v --chip_0_device_path=/dev/mtd0 u-boot-imx6ull-14x14-ddr256-nand.imx`

USER# `sync`

```
root@ALIENTEK-IMX6U:~# ls
u-boot-imx6ull-14x14-ddr256-nand.imx
root@ALIENTEK-IMX6U:~#
root@ALIENTEK-IMX6U:~# kobs-ng init -x -v --chip_0_device_path=/dev/mtd0 u-boot-imx6ull-14x14-ddr256-nand.imx
MTD CONFIG:
  chip_0_device_path = "/dev/mtd0"
  chip_1_device_path = "(null)"
  search_exponent = 2
  data_setup_time = 80
  data_hold_time = 60
  address_setup_time = 25
  data_sample_time = 6
  row_address_size = 3
  column_address_size = 2
  read_command_code1 = 0
  read_command_code2 = 48
  boot_stream_major_version = 1
  boot_stream_minor_version = 0
  boot_stream_sub_version = 0
  ncb_version = 3
  boot_stream_1_address = 0
  boot_stream_2_address = 0
  -- We add the 1k-padding to the uboot.
  .tmp_kobs_ng: verifying using key '00000000000000000000000000000000'
  .tmp_kobs_ng: is a valid bootstream for key '00000000000000000000000000000000'
  mtd: use new bch layout raw access mode
  mtd: opening: "/dev/mtd0"
  NFC geometry :
```

2.1.1 4 烧写 uboot 到 mtd0 分区

以上使用 `kobs-ng` 烧录命令来更新 uboot，在此次操作中已经将 uboot 文件放在了文件系统的 /home/root 下（即当前执行指令的目录），所以指令的后面省去了当前的路径/home/root，写成了 u-boot-imx6ull-14x14-ddr256-nand.imx。在执行以上的烧写 uboot 的指令的时候，根据自己 uboot 文件放置的路径情况来修改，如以上的指令将路径补全以后为：

`kobs-ng init -x -v --chip_0_device_path=/dev/mtd0 /home/root/u-boot-imx6ull-14x14-ddr256-nand.imx`

更新完 uboot 文件以后，最好执行一遍 **sync** 命令来将缓存内容同步到 NAND FLASH 中再重启系统，千万不要直接关机或者按 **RESET** 键来重启，否则有可能缓存还未同步导致文件的丢失，烧写失败。

2.1.2 单步更新设备树到 NAND FLASH 中

由前面的查看分区指令可知 mtd1 是 env（环境变量）分区，mtd2 是 logo 分区，mtd3 是 dtb（设备树文件）分区，按照前面更新 uboot 的步骤，如果我们要更新设备树的话，先将对应的设备树文件拷贝到文件系统的 /home/root 分区，然后擦除对应的分区再进行烧写。

```
root@ALIENTEK-IMX6U:~# pwd
/home/root
root@ALIENTEK-IMX6U:~# ls
imx6ull-14x14-emmc-10.1-1280x800-c.dtb  imx6ull-14x14-emmc-7-800x480-c.dtb  imx6ull-14x14-nand-7-1024x600-c.dtb
imx6ull-14x14-emmc-4.3-480x272-c.dtb  imx6ull-14x14-nand-10.1-1280x800-c.dtb  imx6ull-14x14-nand-7-800x480-c.dtb
imx6ull-14x14-emmc-4.3-800x480-c.dtb  imx6ull-14x14-nand-4.3-480x272-c.dtb  u-boot-imx6ull-14x14-ddr256-nand.imx
imx6ull-14x14-emmc-7-1024x600-c.dtb  imx6ull-14x14-nand-4.3-800x480-c.dtb
```

2.1.2.1 拷贝相关设备树到/home/root 目录下

如上图，为了文档的完整，本文中是将对应各个分辨率的设备树文件拷贝到了文件系统的 /home/root 中。

擦除设备树对应的分区：

USER# `flash_erase /dev/mtd3 0 0`

根据自己显示屏的尺寸和分辨率的情况，使用 **nandwrite** 指令将对应的设备树文件烧写到 mtd3 对应的地址中。如 imx6ull-14x14-nand-4.3-480x272-c.dtb 对应的是显示屏为 4.3 寸、分辨率为 480*272 的设备树文件，在 mtd3 的偏移地址 0x0 处将此设备树的数据写入到了 mtd3 中。同理，imx6ull-14x14-nand-7-800x480-c.dtb 对应的是显示屏为 7 寸、分辨率为 800*480 的设备树文件，在 mtd3 的偏移地址 0x40000 处将该设备树的数据写入到了 mtd3 中。同理，更新完设备树文件以后，最好执行一遍 **sync** 命令以后再重启系统。

后面红色的两个是新增的正点原子 **hdmi** 或者 **vga** 设备树

USER# `nandwrite -p /dev/mtd3 /home/root/imx6ull-14x14-nand-4.3-480x272-c.dtb`

USER# `nandwrite -s 0x20000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-4.3-800x480-c.dtb`

USER# `nandwrite -s 0x40000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-7-800x480-c.dtb`

USER# `nandwrite -s 0x60000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-7-1024x600-c.dtb`

USER# `nandwrite -s 0x80000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-10.1-1280x800-c.dtb`

USER# `nandwrite -s 0xa0000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-hdmi.dtb`

USER# `nandwrite -s 0xc0000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-vga.dtb`

USER# `sync`

```

root@ALIENTEK-IMX6U:~# pwd
/home/root
root@ALIENTEK-IMX6U:~# ls
imx6ull-14x14-emmc-10.1-1280x800-c.dtb  imx6ull-14x14-emmc-7-800x480-c.dtb  imx6ull-14x14-nand-7-1024x600-c.dtb
imx6ull-14x14-emmc-4.3-480x272-c.dtb  imx6ull-14x14-nand-10.1-1280x800-c.dtb  imx6ull-14x14-nand-7-800x480-c.dtb
imx6ull-14x14-emmc-4.3-800x480-c.dtb  imx6ull-14x14-nand-4.3-480x272-c.dtb  u-boot-imx6ull-14x14-ddr256-nand.imx
imx6ull-14x14-emmc-7-1024x600-c.dtb  imx6ull-14x14-nand-4.3-800x480-c.dtb
root@ALIENTEK-IMX6U:~#
root@ALIENTEK-IMX6U:~# flash_erase /dev/mtd3 0 0
Erasing 128 Kibyte @ e0000 -- 100 % complete
root@ALIENTEK-IMX6U:~# nandwrite -p /dev/mtd3 /home/root/imx6ull-14x14-nand-4.3-480x272-c.dtb
Writing data to block 0 at offset 0x0
root@ALIENTEK-IMX6U:~# nandwrite -s 0x20000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-4.3-800x480-c.dtb
Writing data to block 1 at offset 0x20000
root@ALIENTEK-IMX6U:~# nandwrite -s 0x40000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-7-800x480-c.dtb
Writing data to block 2 at offset 0x40000
root@ALIENTEK-IMX6U:~# nandwrite -s 0x60000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-7-1024x600-c.dtb
Writing data to block 3 at offset 0x60000
root@ALIENTEK-IMX6U:~# nandwrite -s 0x80000 -p /dev/mtd3 /home/root/imx6ull-14x14-nand-10.1-1280x800-c.dtb
Writing data to block 4 at offset 0x80000
root@ALIENTEK-IMX6U:~#

```

2.1.2.2 烧写各个设备树，注意设备树要对应相应的偏移地址

2.1.3 单步更新内核到 NAND FLASH 中

将内核文件 zImage 拷贝到文件系统的/home/root 中，执行如下指令，擦除对应的内核文件分区再进行内核文件的烧写，再重启系统。

```

USER# flash_erase /dev/mtd4 0 0
USER# nandwrite -p /dev/mtd4 /home/root/zImage
USER# sync

```

```

root@ALIENTEK-IMX6U:~# pwd
/home/root
root@ALIENTEK-IMX6U:~# ls zImage
zImage
root@ALIENTEK-IMX6U:~# flash_erase /dev/mtd4 0 0
Erasing 128 Kibyte @ 7e0000 -- 100 % complete
root@ALIENTEK-IMX6U:~# nandwrite -p /dev/mtd4 /home/root/zImage
Writing data to block 0 at offset 0x0
Writing data to block 1 at offset 0x20000
Writing data to block 2 at offset 0x40000
Writing data to block 3 at offset 0x60000
Writing data to block 4 at offset 0x80000
Writing data to block 5 at offset 0xa0000
Writing data to block 6 at offset 0xc0000
Writing data to block 7 at offset 0xe0000
Writing data to block 8 at offset 0x100000
Writing data to block 9 at offset 0x120000
Writing data to block 10 at offset 0x140000
Writing data to block 11 at offset 0x160000
Writing data to block 12 at offset 0x180000

```

2.1.3.1 先擦除内核分区再烧写内核

2.1.3 单步更新文件系统到 NAND FLASH 中

更新文件系统到 NADN FLASH 中需要从 SD 卡系统卡启动，请参照【正点原子】I.MX6U 用户快速体验 V1.x.pdf 的第二章 2.2.1.1 小节固化系统到 SD 卡，从 SD 卡启动！注意 Nand Flash 只有 512MB，不能从 Nandflash 去启动文件系统，正在运行的系统不能把自己文件系统分区给格式化！。

将文件系统压缩包和模块压缩包（文件系统压缩包可以在开发板光盘 A-基础资料->8、开发板系统镜像下找到 fsl-image-qt5-v1.x.tar.bz2，然后重命名为 rootfs.tar.bz2，内核模块压缩包可以在开发板光盘 A-基础资料->8、开发板系统镜像\linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x 下找到 modules.tar.bz2）拷贝到文件系统的/home/root 目录下，如下图，rootfs.tar.bz2 是文件系统压缩包，modules.tar.bz2 是模块压缩包。

```

root@ALIENTEK-IMX6U:~# pwd
/home/root
root@ALIENTEK-IMX6U:~# ls
imx6ull-14x14-emmc-10.1-1280x800-c.dtb  imx6ull-14x14-nand-10.1-1280x800-c.dtb  modules.tar.bz2
imx6ull-14x14-emmc-4.3-480x272-c.dtb   imx6ull-14x14-nand-4.3-480x272-c.dtb   rootfs.tar.bz2
imx6ull-14x14-emmc-4.3-800x480-c.dtb   imx6ull-14x14-nand-4.3-800x480-c.dtb   u-boot-imx6ull-14x14-ddr256-nand.imx
imx6ull-14x14-emmc-7-1024x600-c.dtb    imx6ull-14x14-nand-7-1024x600-c.dtb    zImage
imx6ull-14x14-emmc-7-800x480-c.dtb     imx6ull-14x14-nand-7-800x480-c.dtb
root@ALIENTEK-IMX6U:~#

```

2.1.3.2 拷贝文件系统和内核模块到 SD 卡中

```

USER# flash_erase /dev/mtd5 0 0
USER# ubiformat /dev/mtd5
USER# ubiattach /dev/ubi_ctrl -m 5
USER# ubimkvol /dev/ubi0 -Nrootfs -m
USER# mkdir -p /mnt/mtd5
USER# mount -t ubifs ubi0:rootfs /mnt/mtd5

```

以上使用了 UBI 相关的一些指令来烧写 UBIFS 格式的文件系统，FLASH 常用的文件系统有 JFFS2、YAFFS2、LOGFS 和 UBIFS，而 UBIFS 专门为了解决 MTD 设备所遇到的瓶颈而设计的，其在设计与性能上均较 YAFFS2、JFFS2 更适合 NAND Flash。关于更详细的 UBIFS 格式文件系统和 UBI 工具的介绍可自行查找相关资料，这里只是做简单的描述。

ubiformat /dev/mtd5 作用是格式 mtd5 分区；ubiattach /dev/ubi_ctrl -m 5 指将/dev/mtd5 分区关联到 UBI 上；ubimkvol /dev/ubi0 -Nrootfs -m 的作用是为新创建的分区，设置大小以及标识名；mkdir -p /mnt/mtd5 用于创建一个旧时目录/mnt/mtd5，即创建用户挂载文件系统的地方；mount -t ubifs ubi0:rootfs /mnt/mtd5 是指按照 UBI 的分区名进行挂载 UBI 分区到新建的目录/mnt/mtd5 中。

```

root@ALIENTEK-IMX6U:~# flash_erase /dev/mtd5 0 0
Erasing 128 KiByte @ 1f140000 -- 99 % complete flash_erase: Skipping bad block at 1f160000
flash_erase: Skipping bad block at 1f180000
flash_erase: Skipping bad block at 1f1a0000
flash_erase: Skipping bad block at 1f1c0000
Erasing 128 KiByte @ 1f1c0000 -- 100 % complete
root@ALIENTEK-IMX6U:~# ubiformat /dev/mtd5
ubiformat: mtd5 (nand), size 522059776 bytes (497.9 MiB), 3983 eraseblocks of 131072 bytes (128.0 KiB), min. I/O size 2048 bytes
libscan: scanning eraseblock 3982 -- 100 % complete
ubiformat: 3979 eraseblocks are supposedly empty
ubiformat: 4 bad eraseblocks found, numbers: 3979, 3980, 3981, 3982
ubiformat: formatting eraseblock 3982 -- 100 % complete
root@ALIENTEK-IMX6U:~# ubiattach /dev/ubi_ctrl -m 5
[ 191.441311] ubi0: attaching mtd5
[ 192.457979] ubi0: scanning is finished
[ 192.477662] ubi0: attached mtd5 (name "rootfs", size 497 MiB)
[ 192.483583] ubi0: PEB size: 131072 bytes (128 KiB), LEB size: 126976 bytes
[ 192.490472] ubi0: min./max. I/O unit sizes: 2048/2048, sub-page size 2048
[ 192.498408] ubi0: VID header offset: 2048 (aligned 2048), data offset: 4096
[ 192.508511] ubi0: good PEBs: 3979, bad PEBs: 4, corrupted PEBs: 0
[ 192.515679] ubi0: user volume: 0, internal volumes: 1, max. volumes count: 128
[ 192.522929] ubi0: max/mean erase counter: 0/0, WL threshold: 4096, image sequence number: 1746448873
[ 192.532933] ubi0: available PEBs: 3899, total reserved PEBs: 80, PEBs reserved for bad PEB handling: 76
[ 192.542446] ubi0: background thread "ubi_bgt0d" started, PID 710
UBI device number 0, total 3979 LEBs (505237504 bytes, 481.8 MiB), available 3899 LEBs (495079424 bytes, 472.1 MiB), LEB size 126976 bytes (124.0 KiB)
root@ALIENTEK-IMX6U:~# ubimkvol /dev/ubi0 -Nrootfs -m
Set volume size to 495079424
Volume ID 0, size 3899 LEBs (495079424 bytes, 472.1 MiB), LEB size 126976 bytes (124.0 KiB), dynamic, name "rootfs", alignment 1
root@ALIENTEK-IMX6U:~# mkdir -p /mnt/mtd5
root@ALIENTEK-IMX6U:~# mount -t ubifs ubi0:rootfs /mnt/mtd5
[ 236.828731] UBIFS (ubi0:0): default file-system created
[ 236.834874] UBIFS (ubi0:0): background thread "ubifs_bgt0_0" started, PID 716
[ 236.871525] UBIFS (ubi0:0): UBIFS: mounted UBI device 0, volume 0, name "rootfs"
[ 236.879125] UBIFS (ubi0:0): LEB size: 126976 bytes (124 KiB), min./max. I/O unit sizes: 2048 bytes/2048 bytes
[ 236.890101] UBIFS (ubi0:0): FS size: 493047808 bytes (470 MiB, 3883 LEBs), journal size 24633344 bytes (23 MiB, 194 LEBs)
[ 236.901218] UBIFS (ubi0:0): reserved for root: 4952683 bytes (4836 KiB)
[ 236.908969] UBIFS (ubi0:0): media format: w4/r0 (latest is w4/r0), UUID ABAF4FA4-BB0E-4169-A337-FF943193571C, small LPT model
root@ALIENTEK-IMX6U:~#

```

2.1.3.3 格式化文件系统分区，然后挂载文件系统分区

将文件系统解压到/mnt/mtd5/中，如下图是解压过程的部分截图：

```

USER# tar jxvf rootfs.tar.bz2 -C /mnt/mtd5/

```



```

var/lib/alsa/asound.state
var/lib/arpd/
var/lib/dbus/
var/lib/sudo/
var/lib/sudo/lectured/
var/lib/logrotate.status
var/spool/
var/spool/cron/
var/spool/cron/root
var/spool/at/
var/spool/at/jobs/
var/spool/at/jobs/.SEQ
var/spool/at/spool/
var/spool/mail/
var/volatile/
var/tmp
var/backups/
var/cache/
var/cache/fontconfig/
var/cache/fontconfig/3830d5c3ddfd5cd38a049b759396e72e-1e32d8.cache-6
var/cache/fontconfig/CACHEDIR.TAG
var/cache/fontconfig/6ba42ae0000f58711b5caaf10d690066-1e32d8.cache-6
var/cache/fontconfig/94322f4d3cdf1bd54794b691285ca062-1e32d8.cache-6
var/cache/ldconfig/
var/cache/ldconfig/aux-cache
root@ALIENTEK-IMX6U:~#

```

2.1.3 4 解压文件系统到 NandFlash 文件系统分区完成

以上指令是将 rootfs.tar.bz2 解压到/mnt/mtd5/中，jxvfm 中 j 指有 bz2 属性的解压，x 指解压缩，v 指将解压缩的过程展示出来，如果不想展示解压缩的过程可以将 v 去掉，f 指指定要操作的文件名，m 指保留文件不被覆盖，-C 指变更解压的目标目录，默认是当前目录，后面紧跟的是要解压缩到/mnt/mtd5/这个目录中。

解压完文件系统以后，如果有改动过内核模块选项，编译好内核以后还需要编译内核模块，所以更新了内核以后，我们还需要更新一下内核模块（如果没有更改过内核模块选项的，此步可以跳过）。还需要解压模块到/mnt/mtd5/lib/modules 中。在设备驱动开发中会将某些驱动程序以模块的形式来编译，所以我们还需要将这些编译好的模块解压到对应的分区中以支持相应的功能。

USER# tar jxvfm modules.tar.bz2 -C /mnt/mtd5/lib/modules

USER# sync

```

root@ALIENTEK-IMX6U:~# tar jxvfm modules.tar.bz2 -C /mnt/mtd5/lib/modules
./
./4.1.15-g2689732/
./4.1.15-g2689732/modules.devname
./4.1.15-g2689732/modules.alias
./4.1.15-g2689732/modules.dep.bin
./4.1.15-g2689732/modules.symbols.bin
./4.1.15-g2689732/modules.order
./4.1.15-g2689732/modules.builtin.bin
./4.1.15-g2689732/kernel/
./4.1.15-g2689732/kernel/drivers/
./4.1.15-g2689732/kernel/drivers/input/
./4.1.15-g2689732/kernel/drivers/input/evbug.ko
./4.1.15-g2689732/kernel/drivers/input/mouse/
./4.1.15-g2689732/kernel/drivers/input/mouse/psmouse.ko
./4.1.15-g2689732/kernel/drivers/input/serio/
./4.1.15-g2689732/kernel/drivers/input/serio/serport.ko
./4.1.15-g2689732/kernel/drivers/video/
./4.1.15-g2689732/kernel/drivers/video/fbdev/
./4.1.15-g2689732/kernel/drivers/video/fbdev/mxc/
./4.1.15-g2689732/kernel/drivers/video/fbdev/mxc/mxc_dci.ko
./4.1.15-g2689732/kernel/drivers/media/
./4.1.15-g2689732/kernel/drivers/media/platform/
./4.1.15-g2689732/kernel/drivers/media/platform/mxc/
./4.1.15-g2689732/kernel/drivers/media/platform/mxc/capture/
./4.1.15-g2689732/kernel/drivers/media/platform/mxc/capture/ov5640_camera_int.ko
./4.1.15-g2689732/kernel/drivers/media/platform/mxc/capture/ipu_bg_overlay_sdc.ko
./4.1.15-g2689732/kernel/drivers/media/platform/mxc/capture/adv7180_tvin.ko

```

2.1.3 5 解压内核模块到 NandFlash 文件系统 lib/modules 目录

更新完内核模块以后，我们需要卸载这个目录，然后再删除这个目录，注意，顺序不能变。

```
USER# umount /mnt/mtd5
```

```
USER# rm -rf /mnt/mtd5
```

```
USER# ubidetach -p /dev/mtd5
```

```
USER# sync
```

ubidetach 是 ubiattach 相反的操作，即将/dev/mtd5 设备从 UBI 设备上去掉关联，在使用此命令之前先将/mnt/mtd5 卸载掉。执行以上指令以后再进行一次 sync 同步。

至此，我们已经将文件系统更新到了 NAND FLASH 中了，开发板选择从 NAND FLASH 启动，查看系统是可以成功启动了的。

```
root@ALIENTEK-IMX6U:~# umount /mnt/mtd5
[ 4415.150960] UBIFS (ubi0:0): un-mount UBI device 0
[ 4415.157164] UBIFS (ubi0:0): background thread "ubifs_bgt0_0" stops
root@ALIENTEK-IMX6U:~# rm -rf /mnt/mtd5
root@ALIENTEK-IMX6U:~# ubidetach -p /dev/mtd5
[ 4436.961925] ubi0: detaching mtd5
[ 4436.971023] ubi0: mtd5 is detached
root@ALIENTEK-IMX6U:~# sync
root@ALIENTEK-IMX6U:~#
```

2.1.3.6 卸载目录再删除文件系统目录的临时目录

2.2 eMMC 更新固件

如果用户使用的是 eMMC 的核心板，正点原子出厂都有把固件固化到 eMMC 中。我们需要从 eMMC 启动或者从 SD 卡启动替换自己固件。

2.2.1 单步更新 uboot 到 eMMC 中

从 eMMC 启动系统，或者从 SD 卡启动系统来更新 uboot 到 eMMC 中。光盘路径如下：[开发板光盘 A-基础资料->8、开发板系统镜像->uboot-imx-2016.03-2.1.0-xxxxxxx-v1.x](#)，然后把 [u-boot-imx6ull-14x14-ddr512-emmc.imx](#) 文件拷贝到文件系统的/home/root 中（本文作者是拷贝到了/home/root，如下图，下图的 u-boot-imx6ull-14x14-ddr512-emmc.imx 实际上是一个 u-boot.imx 文件，也可以根据自己的习惯拷贝到其它目录），为后面更新 uboot 做准备。（备注：可以参考【正点原子】I.MX6U 用户快速体验 V1.x.pdf 第 2.2.1.1 小节查看 [u-boot-imx6ull-14x14-ddr512-emmc.imx](#) 命名的含义）。

```
root@ATK-IMX6U:~# ls
driver  ...  shell  u-boot-imx6ull-14x14-ddr512-emmc.imx
root@ATK-IMX6U:~#
root@ATK-IMX6U:~#
```

2.2.1.1 拷贝 eMMC 所用的 uboot 到文件系统/home/root 目录下

执行下面的指令，先使能 emmc 启动分区，才能进行烧写。

```
USER# echo 0 > /sys/block/mmcblk1boot0/force_ro
```

开始把当前目录下的烧写至 eMMC 的启动分区

```
USER# dd if=u-boot-imx6ull-14x14-ddr512-emmc.imx of=/dev/mmcblk1boot0 bs=1024 seek=1 conv=fsync
```

烧写完成后，关闭要烧写的启动分区

USER# echo 1 >/sys/block/mmcblk1boot0/force_ro

```
root@ATK-IMX6U:~# echo 0 > /sys/block/mmcblk1boot0/force_ro
root@ATK-IMX6U:~# dd if=u-boot-imx6ull-14x14-ddr512-emmc.imx of=/dev/mmcblk1boot0 bs=1024 seek=1 conv=fsync
415+0 records in
415+0 records out
424960 bytes (425 kB, 415 KiB) copied, 0.065932 s, 6.4 MB/s
root@ATK-IMX6U:~# echo 1 >/sys/block/mmcblk1boot0/force_ro
root@ATK-IMX6U:~#
```

2.2.1 2 烧写 uboot 到 eMMC 的启动分区

2.2.2 单步更新设备树到 eMMC 中

使用 ls 指令查看 eMMC 的 boot 分区设备树所在的目录，如下图，有各种屏对应的设备树，根据个人的屏的使用对应的设备树即可，**如果没有屏**，默认使用的是 imx6ull-14x14-emmc-4.3-480x272-c.dtb 这个设备树，更新这个设备树即可。

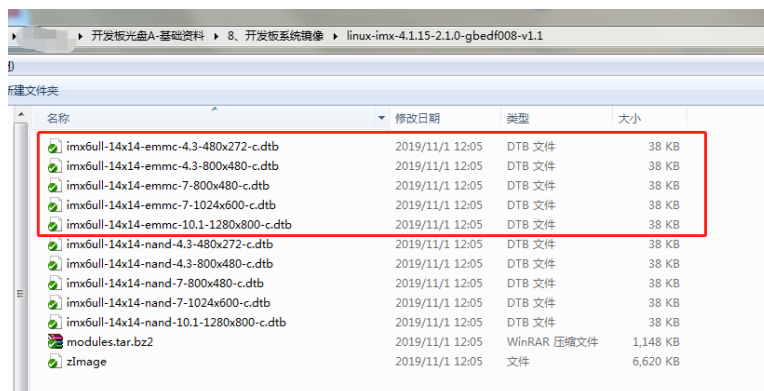
USER# ls /run/media/mmcblk1p1

```
root@ATK-IMX6U:~# ls /run/media/mmcblk1p1
imx6ull-14x14-emmc-10.1-1280x800-c.dtb  imx6ull-14x14-emmc-4.3-480x272-c.dtb  imx6ull-14x14-emmc-4.3-800x480-c.dtb  imx6ull-14x14-emmc-7-1024x600-c.dtb  imx6ull-14x14-emmc-7-800x480-c.dtb  zImage
root@ATK-IMX6U:~#
```

2.2.2 1 查看 eMMC 出厂的设备树所在的目录

我们可以直接拷贝光盘路径如下 [开发板光盘 A-基础资料->8、开发板系统镜像->linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x](#) 目录下的对应设备树文件到/run/media/mmcblk1p1 这个目录下就可以了。

这次我们把全部的 eMMC 所有的设备树全部拷贝到/run/media/mmcblk1p1 目录下，把下图所有的 eMMC 所使用的设备树拷贝到 eMMC 的 boot 分区（挂载目录为/run/media/mmcblk1p1）就可以了。



2.2.2 2 eMMC 所用的各种分辨率的设备树

2.2.3 单步更新内核到 eMMC 中

和 2.2.2 小节一样，我们可以直接拷贝光盘路径如下 [开发板光盘 A-基础资料->8、开发板系统镜像->linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x](#) 目录下的 **zImage** 直接替换到 eMMC 的挂载目录 /run/media/mmcblk1p1 下就可以了。

➤ 更新内核模块

将光盘路径如下 [开发板光盘 A-基础资料->8、开发板系统镜像->linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x](#) 下的内核模块拷贝到 eMMC 文件系统目录下，用 tar 指令解压内核模块到/lib/modules 目录下就可以了。

如下图, 已经将 modules 压缩包传至开发板

```
root@ATK-IMX6U:~# ls
driver  modules.tar.bz2  shell
```

2.2.3 1 将内核模块压缩包传到开发板目录下

在**当前目录**下使用 tar 指令, 将 modules.tar.bz2 解压至 /lib/modules 目录下就可以了, 开发板启动时会去这个目录找对应的内核模块来加载的。

USER# tar xf modules.tar.bz2 -C /lib/modules/

```
root@ATK-IMX6U:~# tar xf modules.tar.bz2 -C /lib/modules/
root@ATK-IMX6U:~#
```

2.2.3 2 解压内核模块到/lib/modules 目录

2.2.4 单步更新文件系统到 eMMC 中

注意: 不能用 eMMC 启动来更新 eMMC 分区里的文件系统。应要使用 SD 卡启动来更新 eMMC 的文件系统! 正在运行的一个系统不能把自己给格式化。还需要注意的是在 2.2.3 小节里, 因为内核模块是在 eMMC 的第二个分区 (rootfs 分区) 里的, 更新文件系统时会把这个分区里的东西全部删除, 请更新文件系统后, 自行按照 2.2.3 小节按照步骤来更新内核模块即可。

从 SD 卡启动, 将**开发板光盘 A-基础资料->8、开发板系统镜像**下的 fsl-image-qt5-v1.x.tar.bz2 拷贝到开发板的文件系统中。如下图, 本文已经拷贝到 /home/root 目录了。

```
root@ATK-IMX6U:~# ls
driver  fsl-image-qt5-v1.3.tar.bz2  modules  shell
```

2.2.4 1 拷贝文件系统压缩包到开发板的 /home/root 目录下

我们直接删除 eMMC 的第二个分区 (rootfs 分区) 文件系统分区, 执行下面的指令。

USER# rm -rf /run/media/mmcblk1p2/*

```
root@ATK-IMX6U:~# rm -rf /run/media/mmcblk1p2/*
root@ATK-IMX6U:~#
```

2.2.4 2 删除 eMMC 的文件系统分区所挂载的目录下所有的内容

然后就可以直接将 fsl-image-qt5-v1.x.tar.bz2 文件系统解压至 eMMC 的文件系统分区目录就可以了。执行下面的指令。(注:解压文件系统时间比较长, 请耐心等待)

USER# tar vxvf fsl-image-qt5-v1.3.tar.bz2 -C /run/media/mmcblk1p2/

```
var/lib/dbus/
var/lib/sudo/
var/lib/sudo/lectured/
var/lib/logrotate.status
var/spool/
var/spool/cron/
var/spool/cron/root
var/spool/at/
var/spool/at/jobs/
var/spool/at/jobs/.SEQ
var/spool/at/spool/
var/spool/mail/
var/volatile/
var/tmp
var/backups/
var/cache/
var/cache/fontconfig/
var/cache/fontconfig/3838d5c3ddfd5cd38a049b759396e72e-le32d8.cache-6
var/cache/fontconfig/CACHEDIR.TAG
var/cache/fontconfig/6ba42ae000f58711b5caaf10d690066-le32d8.cache-6
var/cache/fontconfig/94322f4d3cdf1bd54794b691285ca062-le32d8.cache-6
var/cache/ldconfig/
var/cache/ldconfig/aux-cache
root@ATK-IMX6U:~#
```

2.2.4 3 解压文件系统到 eMMC

解压完成后, 执行一次 sync 指令, 同步一下数据, 可防止数据未完全写入。

2.3 SD 卡更新固件

SD 卡启动卡，更新固件比较简单，我们可以直接在 ubuntu 上操作。**首先确保你已经制作了一张 SD 卡系统启动卡。**

2.3.1 单步更新 uboot 到 SD 卡中

➤ 如果你的核心板所带的存储类型是 eMMC

我们将 SD 启动卡连接到 ubuntu 虚拟机，然后把[开发板光盘 A-基础资料->8、开发板系统镜像->uboot-imx-2016.03-2.1.0-xxxxxxx-v1.x/u-boot-imx6ull-14x14-ddr512-emmc.imx](#) 文件拷贝到 ubuntu 里，为后面更新 uboot 做准备。（备注：可以参考【正点原子】IMX6U 用户快速体验 V1.x.pdf 第 2.2.1.1 小节查看 [u-boot-imx6ull-14x14-ddr512-emmc.imx](#) 命名的含义）。

➤ 如果你的核心板所带的存储类型是 NandFlash

我们将 SD 启动卡连接到 ubuntu 虚拟机，然后把[开发板光盘 A-基础资料->8、开发板系统镜像->uboot-imx-2016.03-2.1.0-xxxxxxx-v1.x/u-boot-imx6ull-14x14-ddr512-nand-sd.imx](#) 文件拷贝到 ubuntu 里，为后面更新 uboot 做准备。（备注：可以参考【正点原子】IMX6U 用户快速体验 V1.x.pdf 第 2.2.1.1 小节查看 [u-boot-imx6ull-14x14-ddr512-nand-sd.imx](#) 命名的含义）。

本次示范的是核心板所带的存储类型为 eMMC 的 uboot 更新到 SD 卡中。如下图，本文已经复制 [u-boot-imx6ull-14x14-ddr512-emmc.imx](#) 到虚拟机目录了。



```
alientek@ubuntu:~$ ls
files          nfs            test
filetest      misc          u-boot-imx6ull-14x14-ddr512-emmc.imx
alientek@ubuntu:~$
```

2.3.1.1 将 uboot 镜像拷贝到 ubuntu

在 ubuntu 上插上连接 SD 卡，使用 fdisk 指令查看 SD 卡挂载的节点。如下图，可以看到 SD 卡所挂载的节点为/dev/sdb。

USER# `sudo fdisk -l`

```

alientek@ubuntu:~$ sudo fdisk -l
[sudo] password for alientek:

Disk /dev/sda: 536.9 GB, 536870912000 bytes
255 heads, 63 sectors/track, 65270 cylinders, total 1048576000 sectors
Units = 扇区 of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000028cb

   设备 启动   起点       终点   块数   Id  系统
/dev/sda1 *      2048   1015021567   507509760   83  Linux
/dev/sda2      1015023614   1048573951   16775169    5  扩展
/dev/sda5      1015023616   1048573951   16775168   82  Linux 交换 / Solaris

Disk /dev/sdb: 15.9 GB, 15931539456 bytes
64 heads, 32 sectors/track, 15193 cylinders, total 31116288 sectors
Units = 扇区 of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x55eeffb3

   设备 启动   起点       终点   块数   Id  系统
/dev/sdb1 *      2048     133119     65536    c  W95 FAT32 (LBA)
/dev/sdb2      133120   31116287   15491584   83  Linux
alientek@ubuntu:~$

```

2.3.1 2 查看 SD 卡挂载节点

在当前目录下使用 `dd` 指令将 `u-boot-imx6ull-14x14-ddr512-emmc.imx` 烧写到 SD 卡中, 执行下面的指令。

USER# `sudo dd if=u-boot-imx6ull-14x14-ddr512-emmc.imx of=/dev/sdb bs=1024 seek=1 conv=fsync`

```

alientek@ubuntu:~$ sudo dd if=u-boot-imx6ull-14x14-ddr512-emmc.imx of=/dev/sdb bs=1024 seek=1 conv=fsync
[sudo] password for alientek:
记录了415+0 的读入
记录了415+0 的写出
424960字节(425 kB)已复制, 0.347772 秒, 1.2 MB/秒
alientek@ubuntu:~$

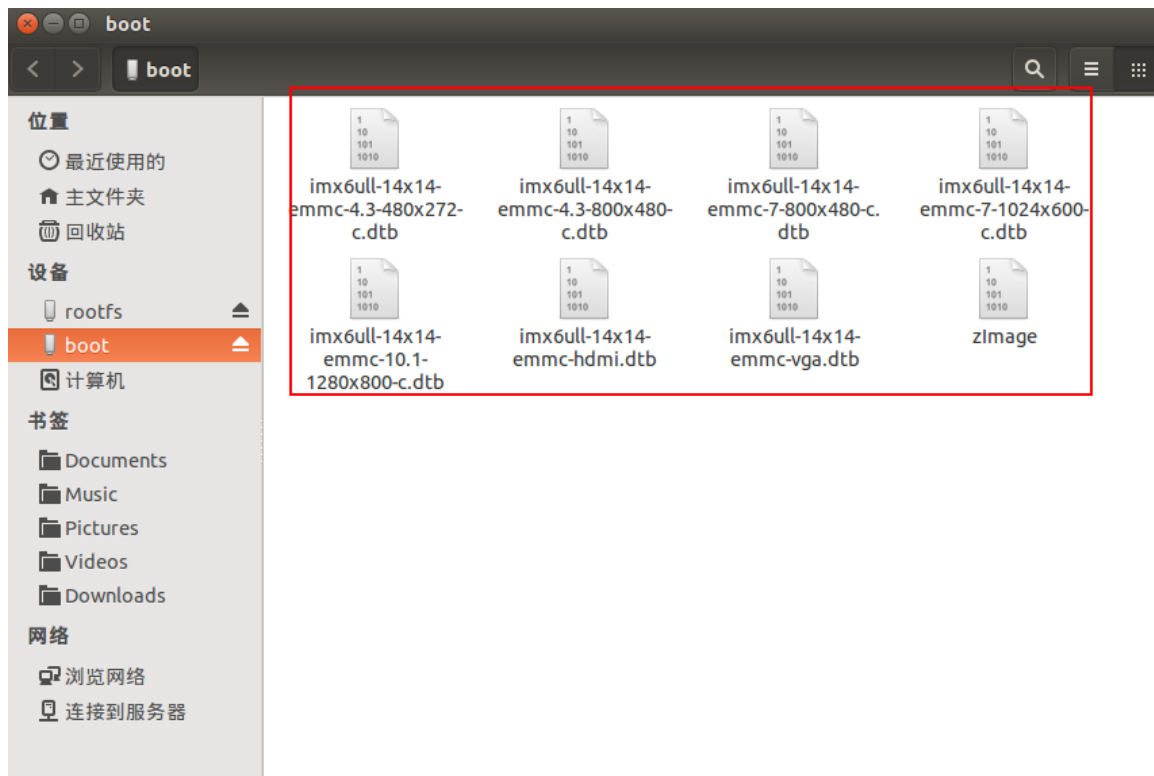
```

2.3.1 3 使用 dd 指令烧写 uboot 到 SD 卡 1K 地址处

至此更新 uboot 成功, 需要注意的是: 如果你的 SD 卡曾经保存过其他 uboot 环境变量, 你需要在 SD 卡时恢复一次环境变量再保存, 才能使用新的环境变量启动!

2.3.2 单步更新设备树到 SD 中

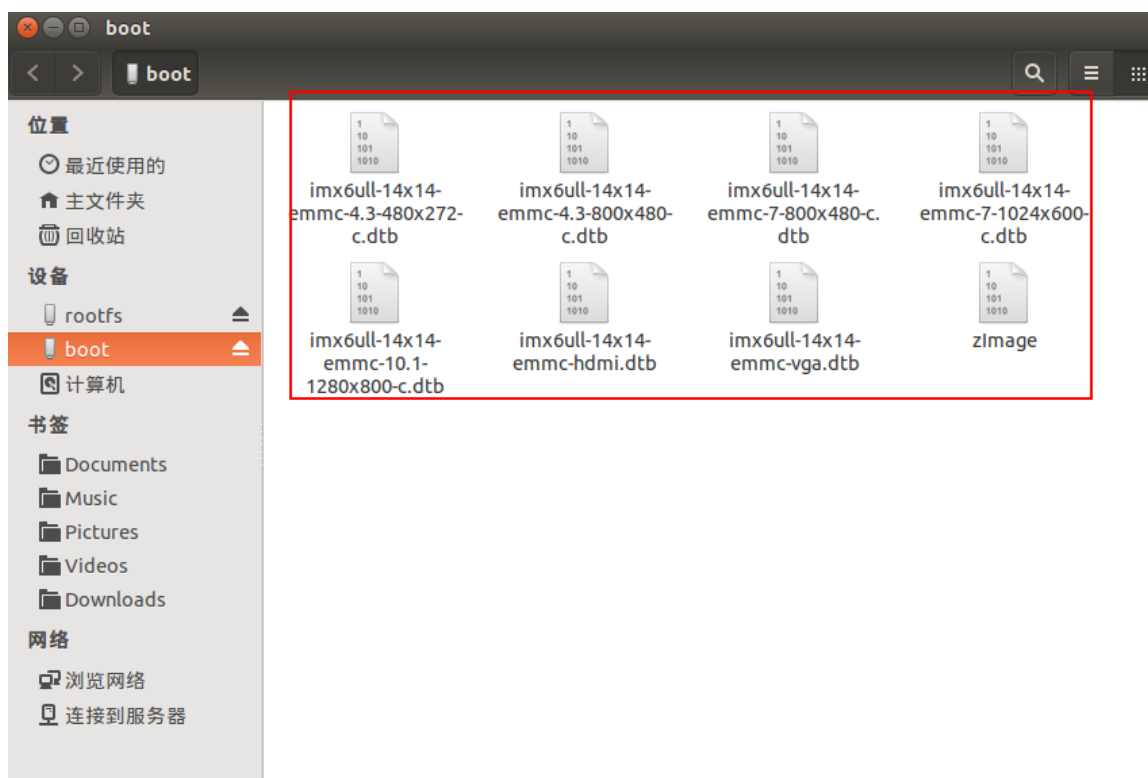
直接拷贝光盘路径如下开发板光盘 A-基础资料->8、开发板系统镜像-> `linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x` 目录下的对应设备树文件到 SD 卡的 boot 分区即可。



2.3.2 1 SD 卡的 boot 分区下的设备树*dtb

2.3.3 单步更新内核到 SD 中

直接拷贝光盘路径如下开发板光盘 A-基础资料->8、开发板系统镜像-> linux-imx-4.1.15-2.1.0-gxxxxxxx-v1.x 目录下的 zImage 到 SD 卡的 boot 分区即可。



2.3.3 1 SD 卡的 boot 分区下的内核 zImage

➤ 更新内核模块

同理，只需要将 [modules.tar.bz2](#) 解压更新到 SD 卡的 rootfs 分区下的/lib/modules 目录即可。

2.3.4 单步更新文件系统到 SD 中

将开发板光盘 A-基础资料->8、开发板系统镜像下的 [fsl-image-qt5-v1.x.tar.bz2](#) 拷贝到 ubuntu 中。

```
alientek@ubuntu:~$ ls
alientek@ubuntu:~$
```

2.3.4 1 拷贝文件系统压缩包到 ubuntu

使用 df 指令查看 SD 卡的挂载路径，如下图，本文的挂载路径为/media/alientek/rootfs，请根据个人 SD 卡挂载的目录具体填写

```
alientek@ubuntu:~$ df
文件系统      1K-块    已用    可用  已用%  挂载点
udev          8196548      4  8196544    1% /dev
tmpfs         1641536    3708  1637828    1% /run
/dev/sda1     499415552 11579696 462443984    3% /
none           4          0        4    0% /sys/fs/cgroup
none          5120        0     5120    0% /run/lock
none          8207668    224  8207444    1% /run/shm
none          102400     128   102272    1% /run/user
none          8207668    2436  8205232    1% /tmp/quest-pof1qa
/dev/sdb2     15117196  716700 13609536    6% /media/alientek/rootfs
/dev/sdb1      64511     9904   54607   16% /media/alientek/boot
alientek@ubuntu:~$
```


首先删除 rootfs 分区下的根文件系统，执行如下指令

USER# `sudo rm -rf /media/alientek/rootfs/*`

```
alientek@ubuntu:~$ sudo rm -rf /media/alientek/rootfs/*  
[sudo] password for alientek:  
alientek@ubuntu:~$
```

2.3.4 2 删除 SD 卡的 rootfs 分区下的文件系统

然后将 `fsl-image-qt5-v1.x.tar.bz2` 解压至 SD 卡根文件系统分区即可，执行下面的指令

USER# `sudo tar xf fsl-image-qt5-v1.3.tar.bz2 -C /media/alientek/rootfs/`

```
alientek@ubuntu:~$ sudo tar xf fsl-image-qt5-v1.3.tar.bz2 -C /media/alientek/rootfs/  
[sudo] password for alientek:  
alientek@ubuntu:~$
```

2.3.4 3 解压文件系统到 SD 卡 rootfs 分区

解压完成后执行 `sync` 同步数据，防止数据未完全写入。

USER# `sync`

```
alientek@ubuntu:~$ sync  
alientek@ubuntu:~$
```

2.3.4 4 同步数据