**RV1126文件系统分区**

目录：

1. 查看分区
2. SDK中分区文件
3. 分区文件介绍
4. 尝试修改分区

前言：

由adb shell进入后台。

将本地文件内拷贝至设备:adb push local remote

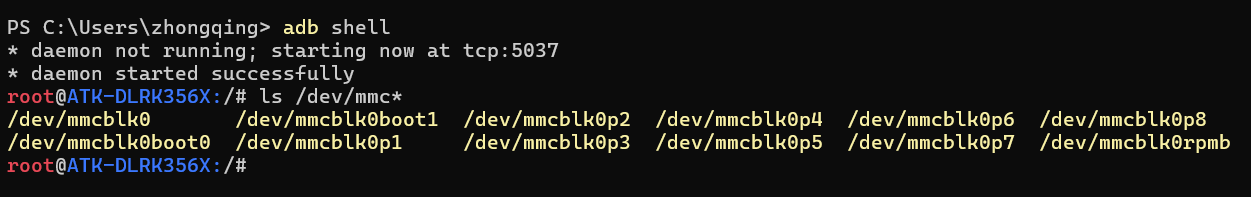
将设备中的文件拷贝至 Windows 下:adb pull remote local

1.查看开发板的分区：



1、//查看设备节点（查看mmc设备：SDK卡或eMMC统称为mmc设备）

ls /dev/mmc\*



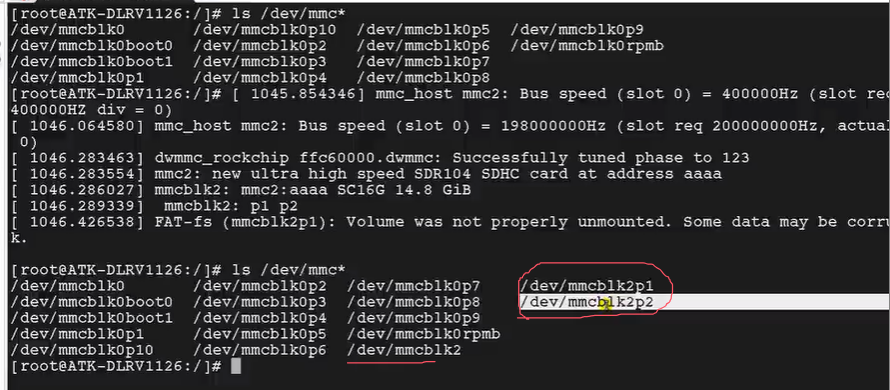
这里，mmc设备的设备名为：/dev/mmcblk0

这里mmc表示emmc是一个mmc设备，

后面的blk表示为快设备，

0表示设备编号。

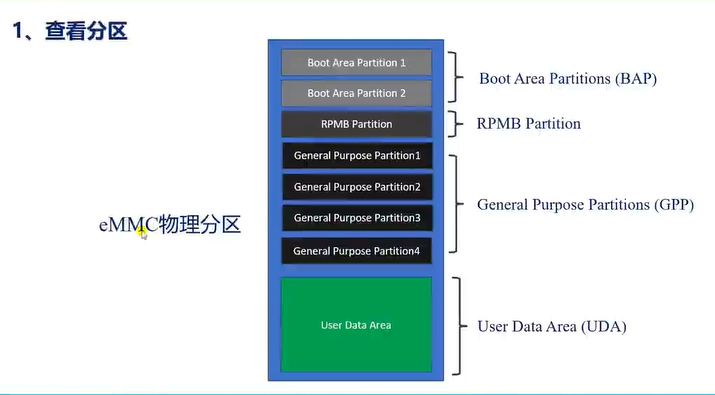
如果插入SD卡，再执行命令，可以看到：多出来如下3个。



/dev/mmcblk2表示新插入的SD卡， /dev/mmcblk2p1和/dev/mmcblk2p2表示SD卡的分区（她之前已经将SD卡格式化了两个分区）。

/\*

为了理解图1所示目录下的文件的作用，就需要先了解emmc的硬件分区：



标准emmc可以有8个硬件分区：

BAP（boot area partitions）分区主要用于存储bootloader，目的就是为了支持从emmc启动系统。BAP一般有2个分区。一般出厂时就已经配置好了。

RPMB分区：具有安全性，当emmc往RPMB分区写数据的时候，需要校验数据的合法性。当读取RPMB区域数据时，需要提供签名机制。可防止数据被窃取。一般存放密钥数据、指纹数据等等，如果要使用该分区一般是需要emmc厂家来进行配置才可。

GPP分区：GPP可以配置1~4个分区。主要用于存储系统或应用数据

UDA分区：用户数据分区。UDA主要也是用于存储系统或者存储应用数据的。

（GPP和UDA的功能是差不多的，所以一般直接使用UDA分区即可）

对于emmc来说：一般的分区就是：

包含两个分区的BAF分区。

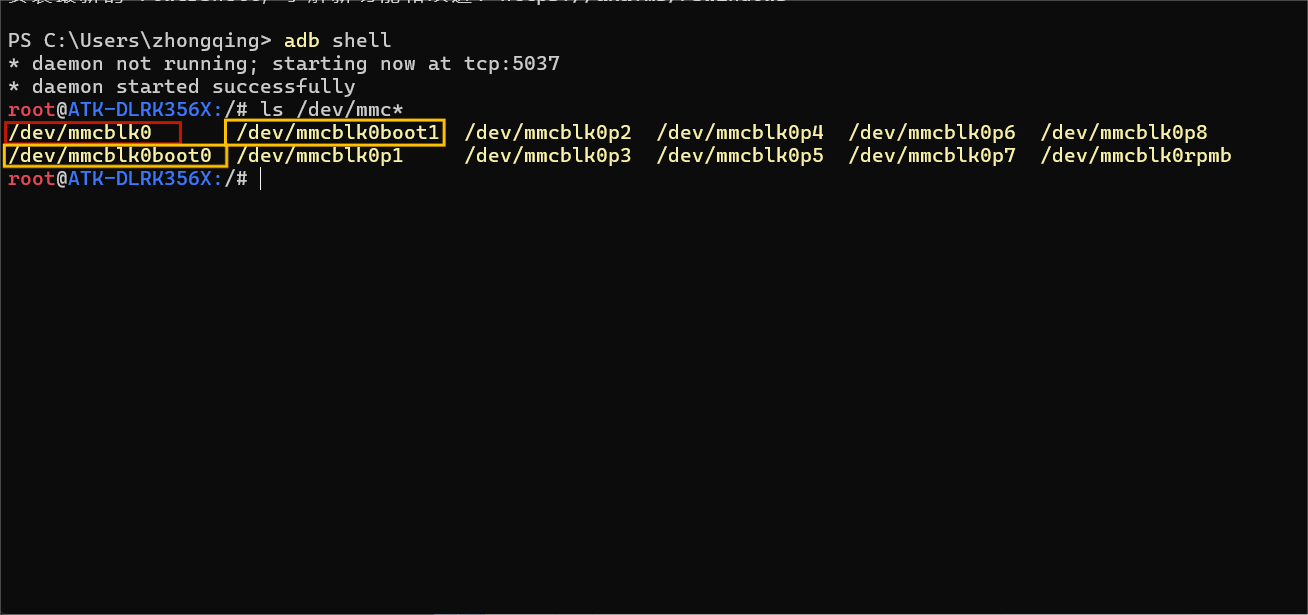
具有安全性的RPMB分区

用于存储系统和数据的UDA分区。

（GPP分区可以没有）

\*/

有以上知识基础之后我们就知道：



**红色框**表示：mmc设备的设备名，并且也是UDA分区,供用户使用。

橙色框表示：mmc的两个BAP分区

Boot0~boot8表示：在UDA分区上创建的子分区。

P1: P为partition就是分区的意思

最后的rpmb：就是上面提到的RPMB分区。这个分区具有安全性。

2、//目前开发板文件系统下没这个命令

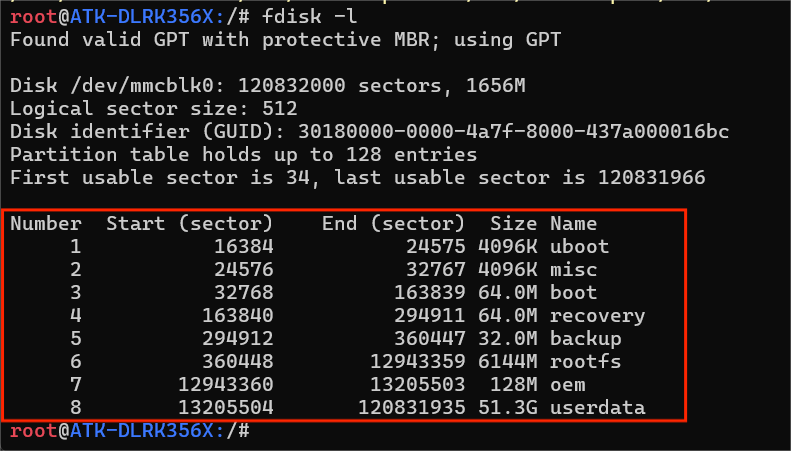
lsblk

3、//使用fdisk命令可以获取（UDA）分区列表

Fdisk -l

执行该命令后，可以看到这个分区情况：

查看的是UDA分区的情况。



可以看到：

第一个分区存放uboot，大小是：4096kbits

第二个分区存放misc， 大小是：4096kb

...

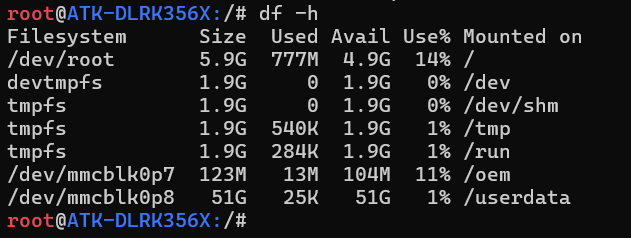
第六个文件系统分区rootfs,大小是6144M 字节

4、//查看（UDA）分区情况

cat /proc/partitions 或者 fdisk /dev/mmcblk0

5、//查看已挂载文件系统的磁盘空间占用情况

df -h



查看已经挂载的文件系统分区：

/dev/root 5.9G 777M 4.9G 14% /

这个就是挂载的文件系统，而文件系统位于第6个分区：

6 360448 12943359 6144M rootfs

Root是挂载到根目录：‘/’下的。

/dev/mmcblk0p7 123M 13M 104M 11% /oem

这个就是挂载的第7个oem分区，

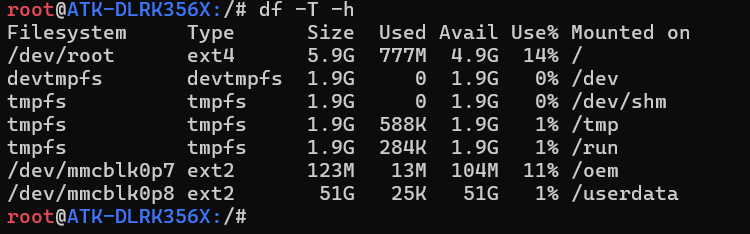
mmcblk0p7挂载到oem下的。

（对照3和5的图看），可以看到图3中的12345分区是没有挂载的（因为在图5的右侧没看到1~5）

**特别是第1和第3个分区，用于存放uboot、内核、设备树，一般这两个分区不要动。**

6、//查看已挂载的文件系统格式

df -T -h



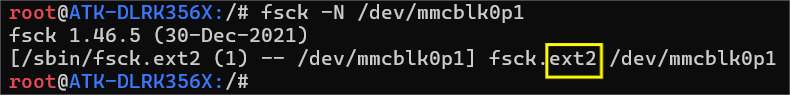
可见：/dev/root ext4 5.9G 777M 4.9G 14% /

根文件系统：/dev/root的格式是：ext4。

oem下存放的就是一些可执行文件，如RKMedia的可执行文件。Oem格式是ext

7、//查看对应分区的文件系统格式

fsck -N /dev/mmcblk0pl



可看到格式为：ext2。

8、查看文件系统分区表：

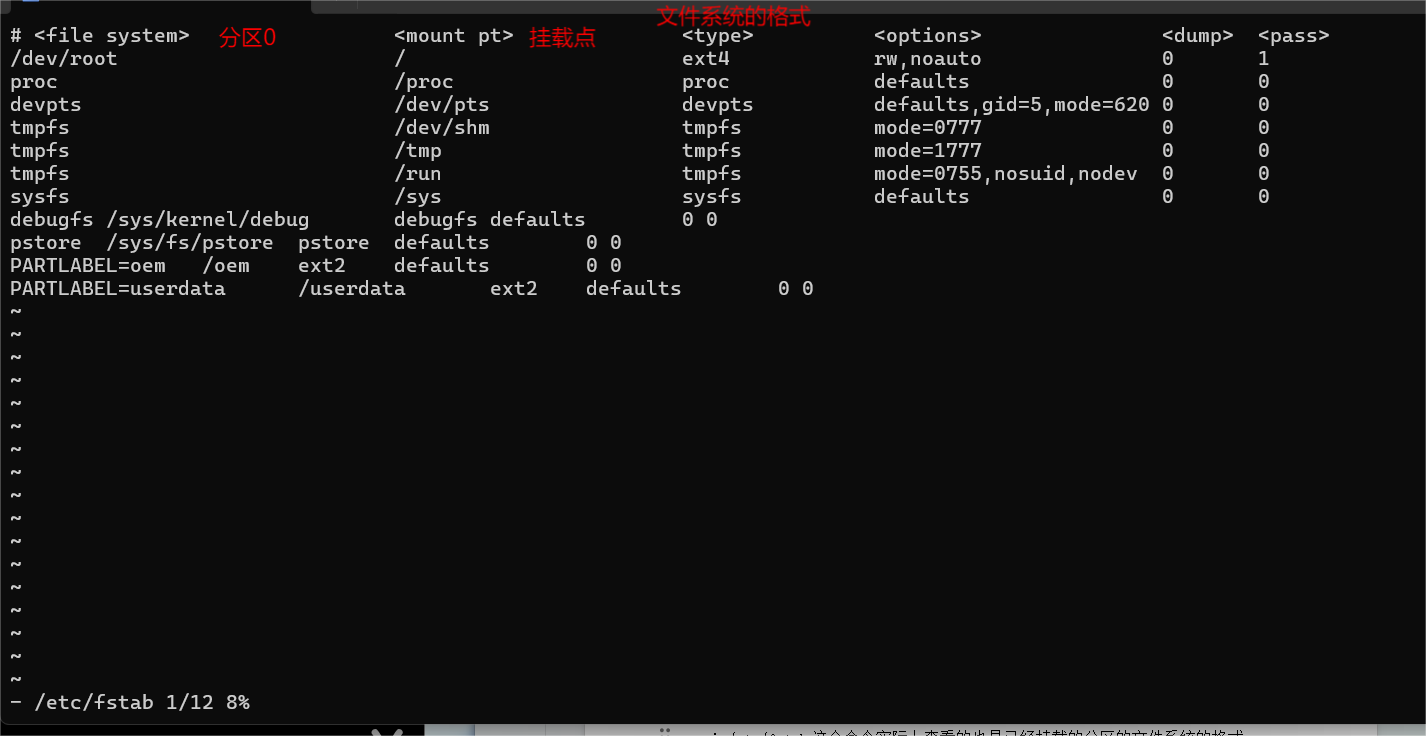
//查看文件系统分区表

vi /etc/fstab

//或者

More /etc/fstab

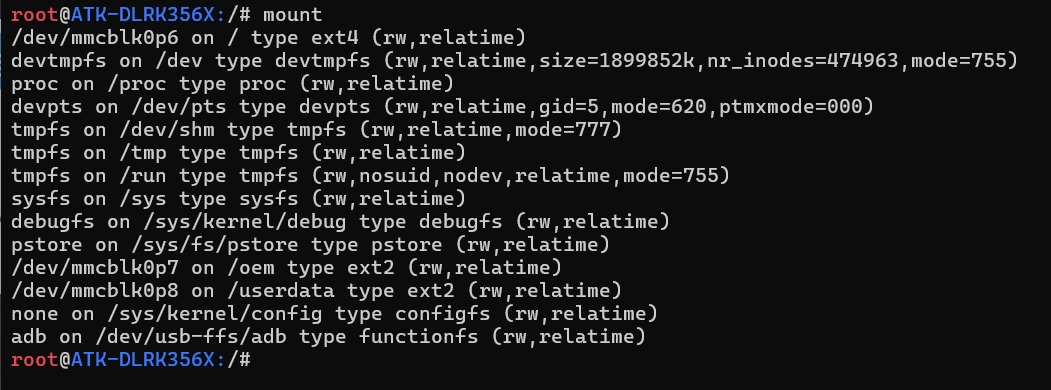
vi /etc/fstab这个命令实际上查看的也是已经挂载的分区的文件系统的格式。



从左到右分别为：分区0，挂载点，文件系统的格式

9、//查看分区挂载情况

mount



执行mount命令后，可以查看到：那个分区挂载到那个路径下，以及它的文件系统格式。

如:

/dev/mmcblk0p6 on / type ext4 (rw,relatime)

表示分区6mmcblk0p6 挂载到 根目录/下，格式为ext4。

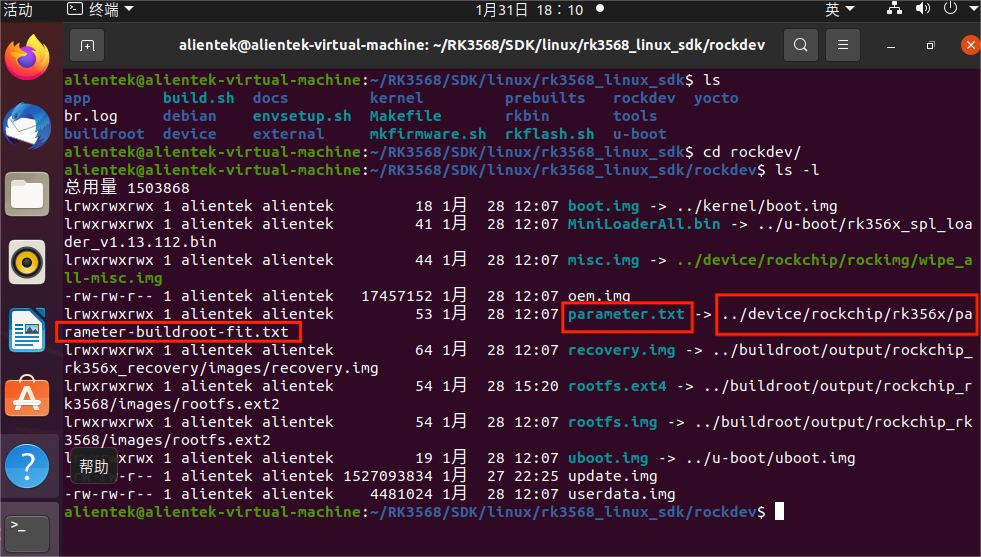
2.SDK中分区文件：

Parameter.txt-> ../device/rockchip/rv1126\_rv1109/parameter-buildroot-fit.txt

在:

/home/alientek/RK3568/SDK/linux/rk3568\_linux\_sdk/rockdev路径下的parameter.txt 文件就是分区文件。它软链接到了

./device/rockchip/rk356x/parameter-buildroot-fit.txt 路径下。



也就是说这两个文件是一样的。（我们可以使用vim命令分别打开查看）

RV1126开发板\开发板光盘A-基础资料\09、系统镜像\01、出厂系统SDK镜像\parameter.txt。



出厂SDK镜像在环境搭建和编译课程已经简单介绍过了。此处再次简单总结：

boot.img：就是设备树、内核和logo的系统镜像

demo.img：正点原子官方出厂demo的目录，里面存放的是测试文件。

也就是如下目录被打包成了demo.img

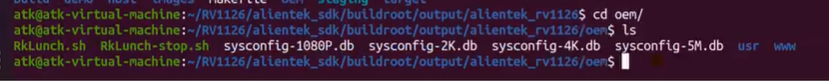


MiniLoaderAll.bin：此文件负责初始化DDR,引导Uboot。相当于SPL或者TF-A.

misc.img：存放升级标志，用于升级系统或恢复出厂设置。

oem.img：存放 给厂商用的分区文件 。或rkmedia的测试代码。

也就是如下目录被打包成了oem.img镜像。



parameter.txt： 就是本节课要重点讲解的分区文件。

recovery.img：前面也介绍过，是用于升级的一个文件系统

rootfs.img：buildroot的文件系统。

uboot.img：uboot和uboot下的设备树

update.img：整个SDK的img

userdata.img：用户分区

3.分区文件介绍：

Parmeter.txt文件内容如下：

FIRMWARE\_VER: 1.0

MACHINE\_MODEL: RK3568

MACHINE\_ID: 007

MANUFACTURER: RK3568

MAGIC: 0x5041524B

ATAG: 0x00200800

MACHINE: 0xffffffff

CHECK\_MASK: 0x80

PWR\_HLD: 0,0,A,0,1

TYPE: GPT

CMDLINE: mtdparts=rk29xxnand:0x00002000@0x00004000(uboot),0x00002000@0x00006000(misc),0x00020000@0x00008000(boot),0x00020000@0x00028000(recovery),0x00010000@0x00048000(backup),0x00c00000@0x00058000(rootfs),0x00040000@0x00c58000(oem),-@0x00c98000(userdata:grow)

uuid:rootfs=614e0000-0000-4b53-8000-1d28000054a9

包含**固件版本**、**芯片平台** 以及 **CMDLINE这些参数**等等。

我们主要看的是CMDLINE信息：其包含分区信息，还有文件系统挂载的信息。

mtdparts就包含文件分区信息。

uuid就包含了文件系统挂载信息。

如果要修改分区，就要先修改这个parameter文件，重点就是修改CMDLINE,CMDLINE参数是uboot引导内核启动时，传递给内核的，其作用就是引导内核启动。那么内核在启动阶段就会去解析CDMLINE参数。

此处，我们可以将CMDLINE分为两个部分：

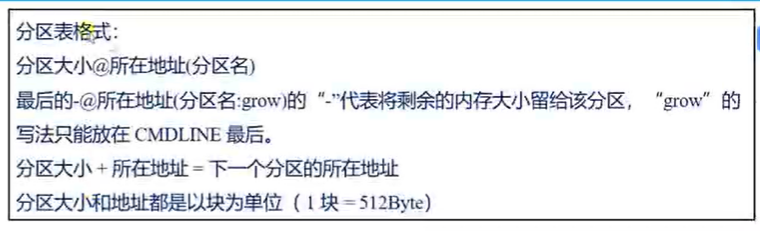
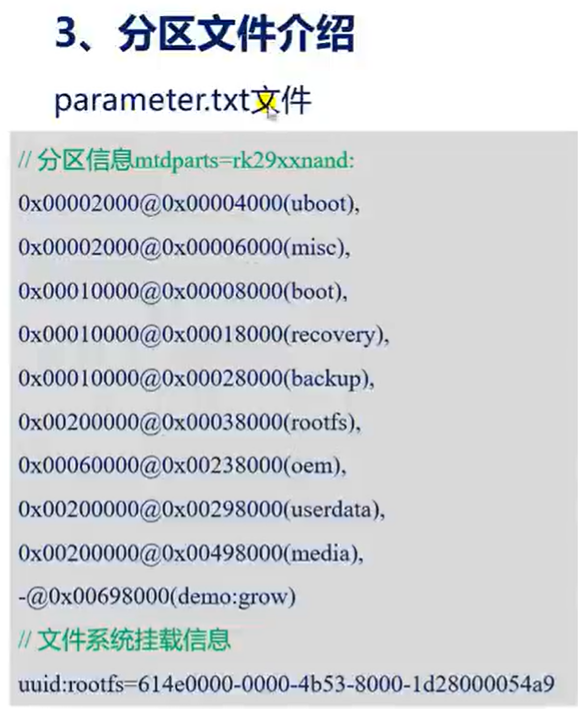


如图：前半部分就是分区信息。后半部分就是文件系统挂载信息。

后半部分的uuid就是通用唯一标识码，作用就是帮助Linux系统识别一个磁盘分区。例如我们可以执行一个 blkid 指令：

执行该命令以后可以查看Linux系统中，各个磁盘的分区的uuid信息。

3、分区文件介绍：



分区表格式：

前面部分，如0x00002000就是分区的大小。

@符号后面，如0x00004000就是分区的地址。（分区的起始地址）

括号（）里面的就是分区名。

前面几个写法都是统一类似的。我们主要看与众不同的最后一个：

-@0x00698000(demo:grow):

其0x00698000的作用与前面的类似，都是指起始分区地址。

-：表示将剩余的内存大小都给这个分区使用。

demo：就是分区的名字。

grow：grow只能写在CMDLINE文件里，且在这个mtdparts后面

这一行的意思是指：demo分区的起始地址是：0x00698000，大小是剩余的空间中可以利用的空间都给了demo使用。

分区的地址和大小都是以块为单位，一块有512个字节。



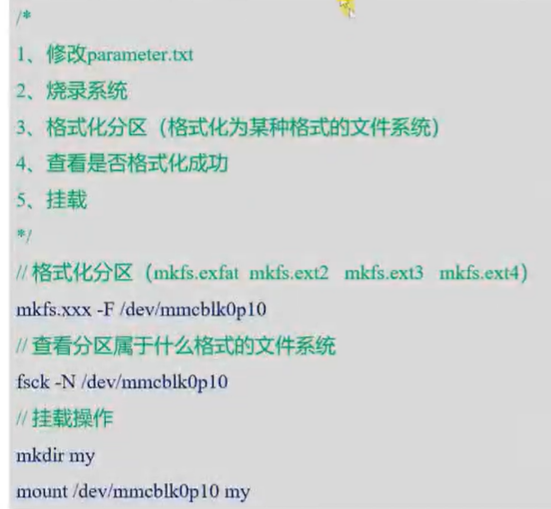
所以各个分区分别有如上图所示的MB大小。

4、尝试修改分区：

添加一个分区：

操作流程如图：有5步。

1. 修改parameter.txt
2. 烧录系统
3. 格式化分区（格式化为某种格式的文件系统）
4. 查看是否格式化成功
5. 挂载



修改为：

FIRMWARE\_VER: 1.0

MACHINE\_MODEL: RK3568

MACHINE\_ID: 007

MANUFACTURER: RK3568

MAGIC: 0x5041524B

ATAG: 0x00200800

MACHINE: 0xffffffff

CHECK\_MASK: 0x80

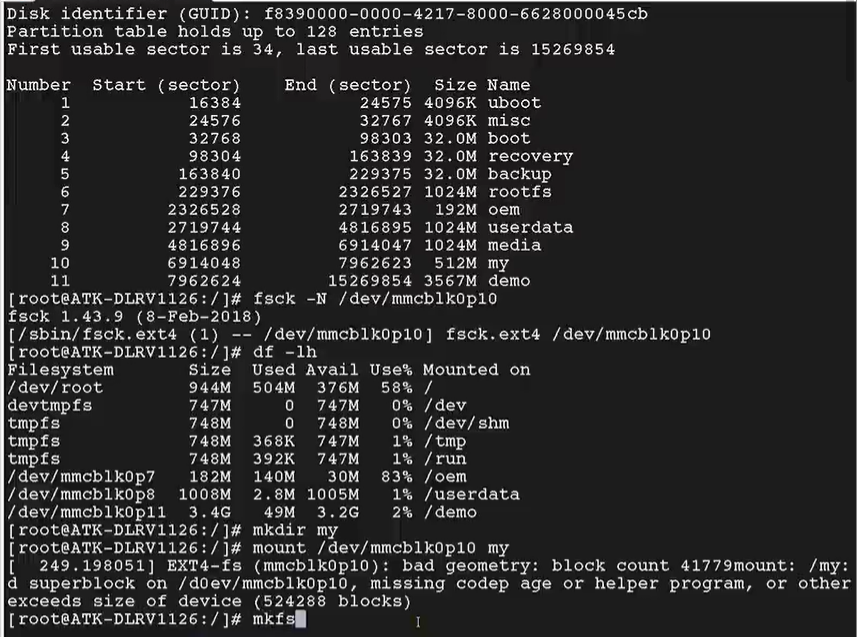
PWR\_HLD: 0,0,A,0,1

TYPE: GPT

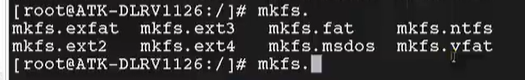
CMDLINE: mtdparts=rk29xxnand:0x00002000@0x00004000(uboot),0x00002000@0x00006000(misc),0x00020000@0x00008000(boot),0x00020000@0x00028000(recovery),0x00010000@0x00048000(backup),0x00c00000@0x00058000(rootfs),0x00040000@0x00c58000(oem),0x00100000@0x00c98000(my),-@0x00d98000(userdata:grow)

uuid:rootfs=614e0000-0000-4b53-8000-1d28000054a9

保存后，重新烧录该系统：导入配置-》进入MASKROM模式烧录。-》执行前面的命令查看分区。（ls /dev/mmc\* 和 fdisk -l）。（可以查看一下my分区以及格式：fsck -N /dev/mmcblk0p10,这里写10是因为她是第10个分区，我们自己的需要修改，再使用df -lh查看该分区挂载情况,假设直接去挂载：1.先创建一个目录mkdir my，2.再通过mount命令去挂载到该目录下： mount /dev/mmcblk0p10 my,此时会提示文件系统格式错误，所以需要先格式化文件系统）使用命令：mkfs.xxx -F /dev/mmcblk0p10格式化分区

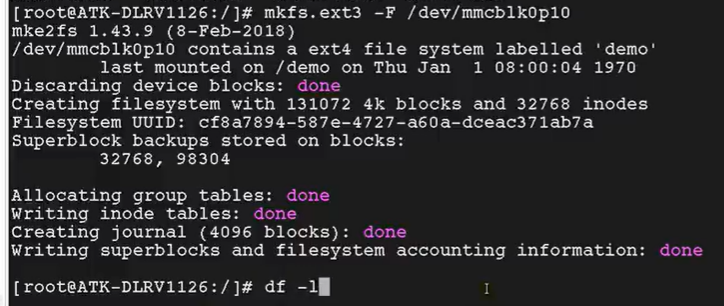


(在输入mkfs. + Tab键之后，可以看到)能够格式化的格式有如下多种：

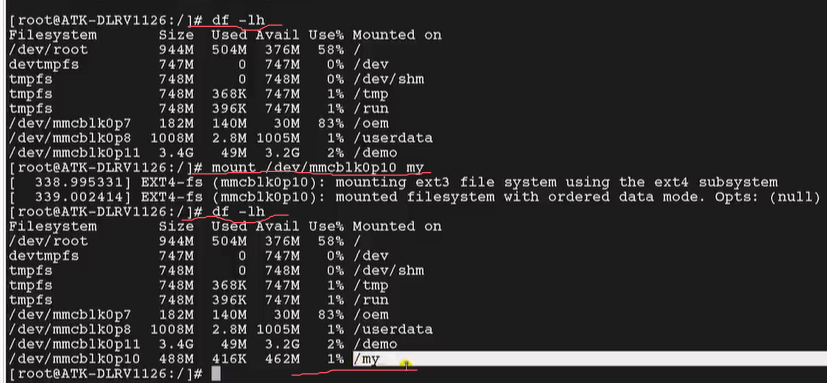


我们可以选择其中任意一种格式，例如：ext3 ，然后加上mkfs后面的参数,/dev/mmcblk0p10是指要格式化的分区。例如执行如下指令且结果如下：

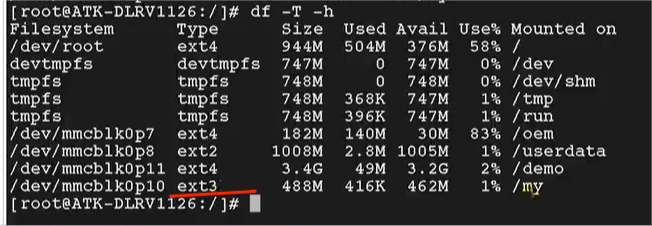
mkfs.ext3 -F /dev/mmcblk0p10



格式化完成后，进行挂载格式化好了的文件系统：mount /dev/mmcblk0p10 my



执行命令：df -T -h就可以查看到已经挂载到文件系统它的文件系统格式：



cd进入my目录下，即进入挂载点。我们就可以在该文件系统中操作文件。如新建目录等。

