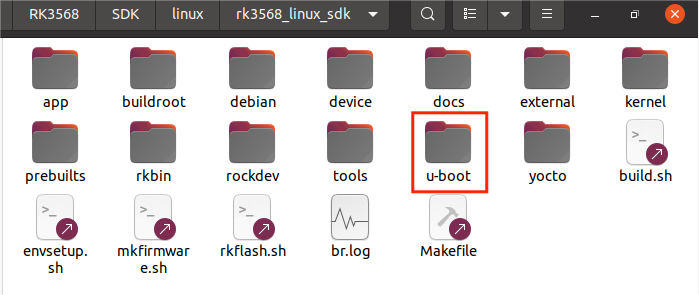
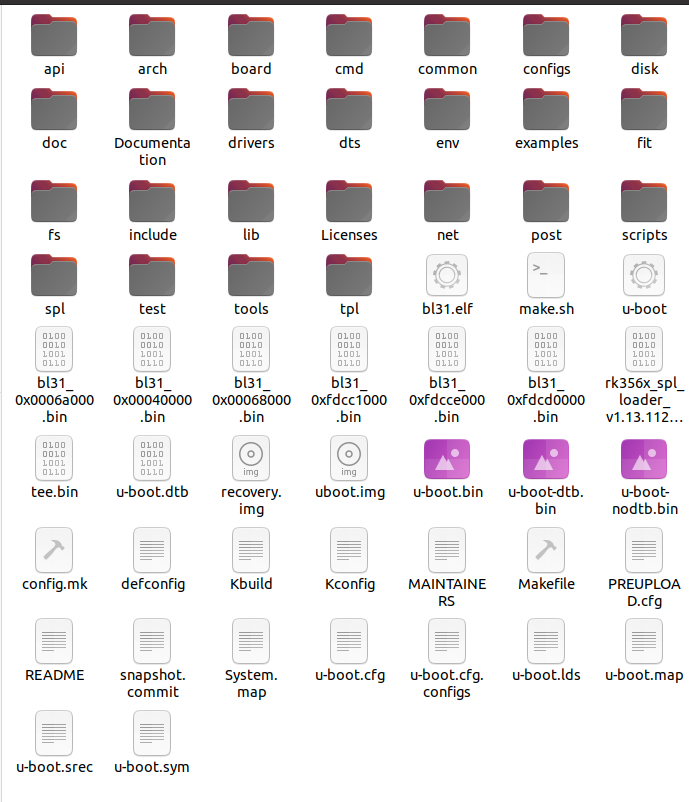
uboot路径：

/home/alientek/RK3568/SDK/linux/rk3568\_linux\_sdk/u-boot



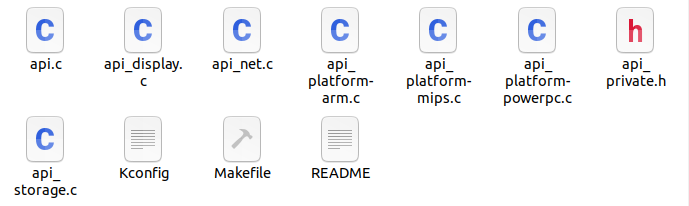
Uboot的内容：



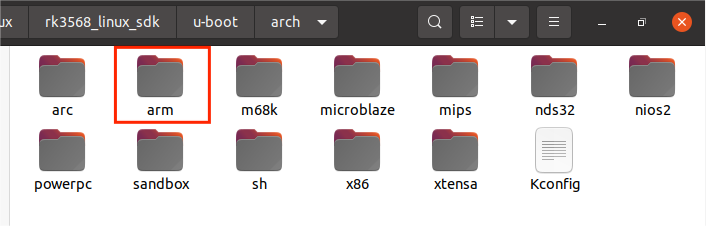
————————————————————————————————

Uboot文件分析：

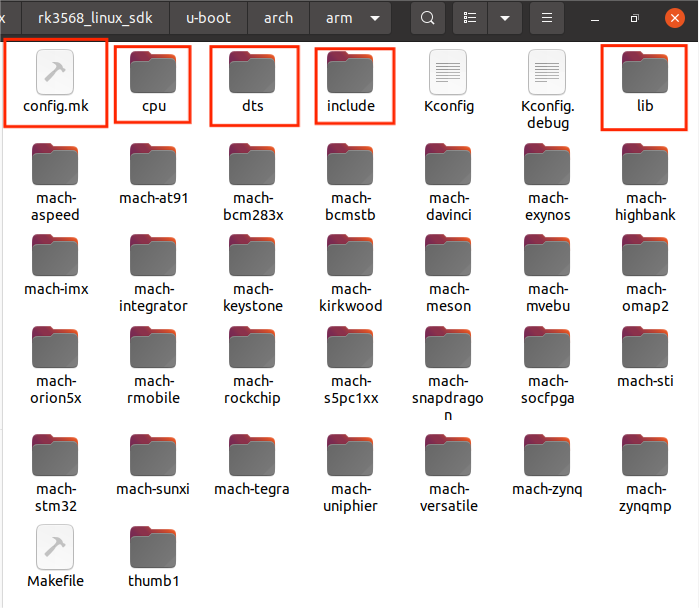
1.Api:一些不常用的接口：



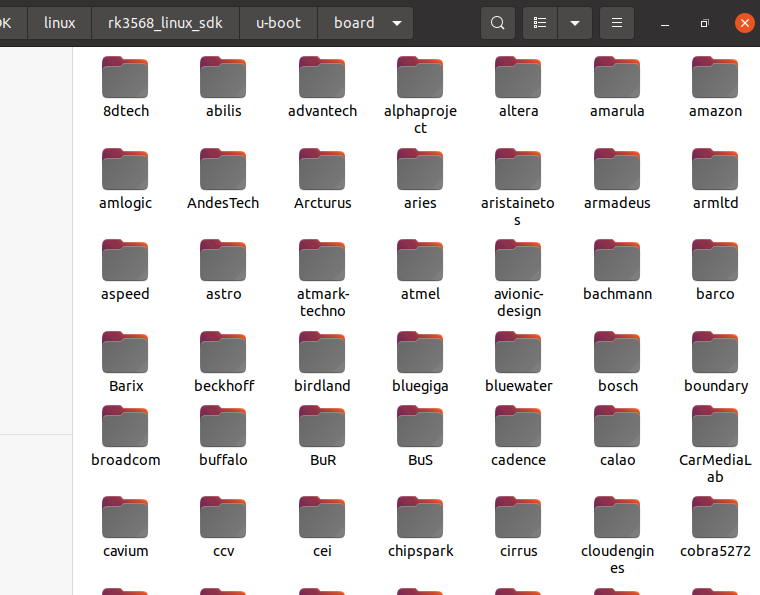
2.Arch:该目录非常重要，但千峰课程里面只留下了这一个文件，官方的SDK里面有很多文件，每一个文件代表一种架构因此工程会非常庞大，为了以后工程阅读的时候不麻烦，他把一些架构删掉了。



并且其arm架构里面也只保留了这几个文件。



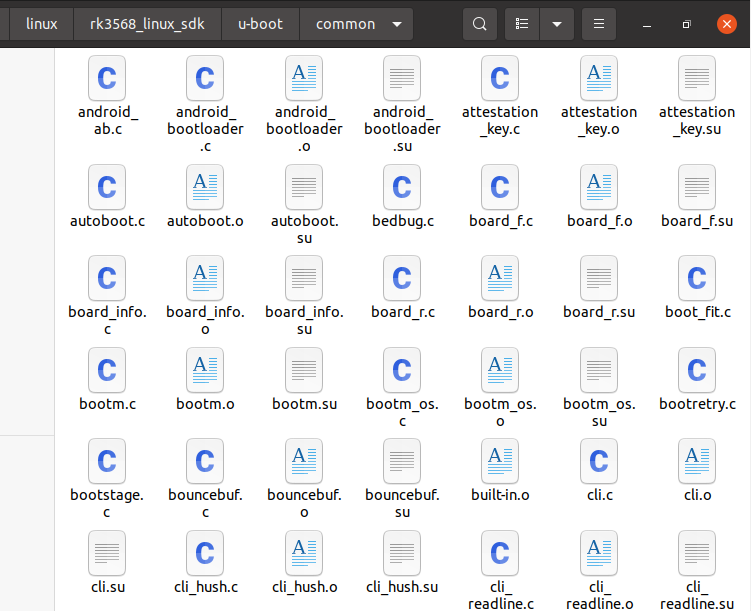
3.board:



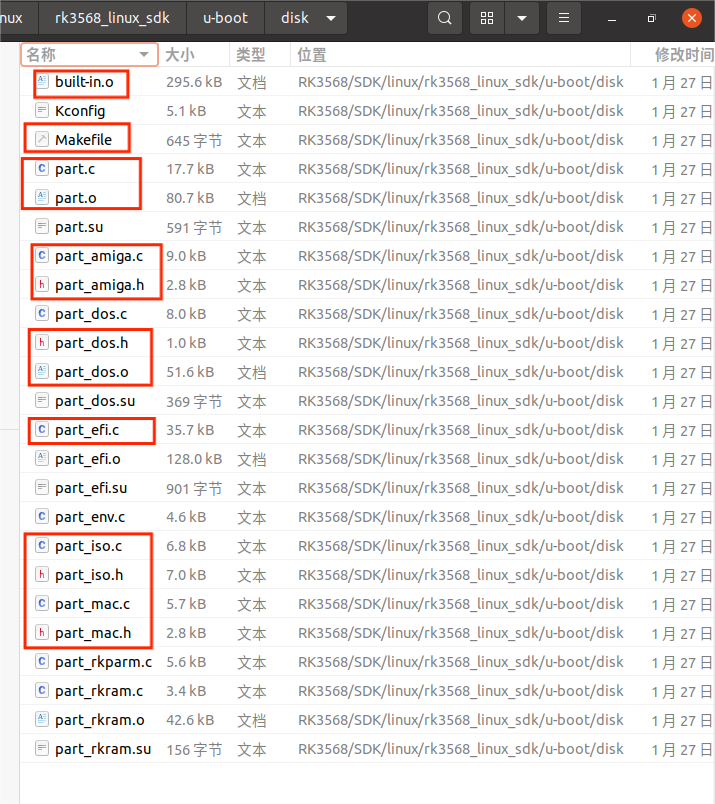
board与arch非常像，arch代表的是架构；board代表的是芯片，代表的是平台uboot支持很多的平台。

千峰把除了使用的s5p6816平台的文件夹保留了，其余全部也删除了。

4.cmd、common、configs:这里面的文件绝大部分都是命令。



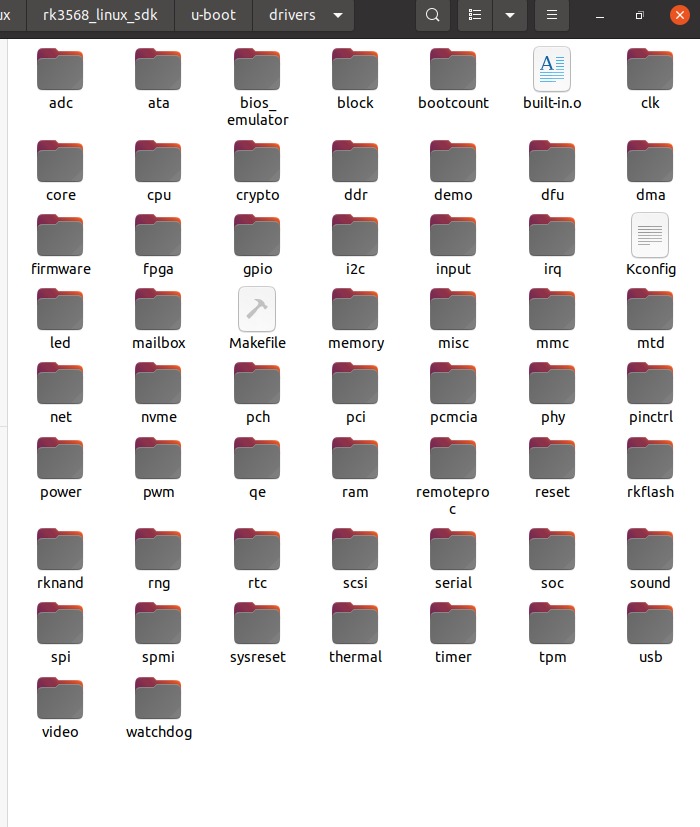
5.disk:磁盘（磁盘的驱动都在这里）他只保留了如下文件：



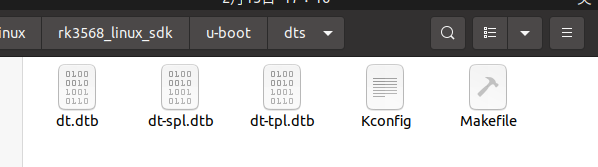
1. doc：针对于uboot开源软件的各个模块的介绍



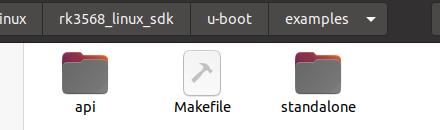
1. Driver：驱动，这些驱动主要就是位于uboot引导程序的第二阶段的代码，第一阶段只是DDR初始化和拷贝，用不到这么多的程序，如果能直接把内核启动起来的话，实际上这些驱动大部分都是可以删除的，



1. dts：设备树，现在已经可以 直接把设备树和内核镜像打成一个包，下载的时候，就把它放到指定的低地址就可以了。早期需要使用uboot去解析一些设备树的内容，现在不需要了。



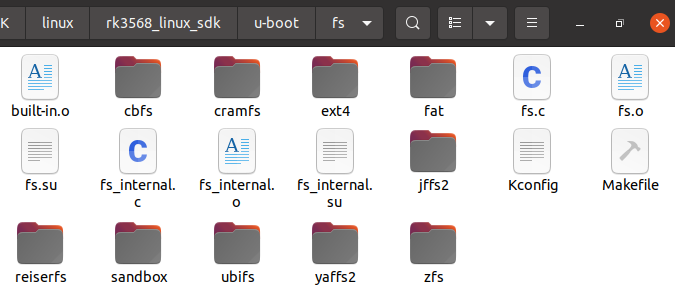
1. Examples：例子。



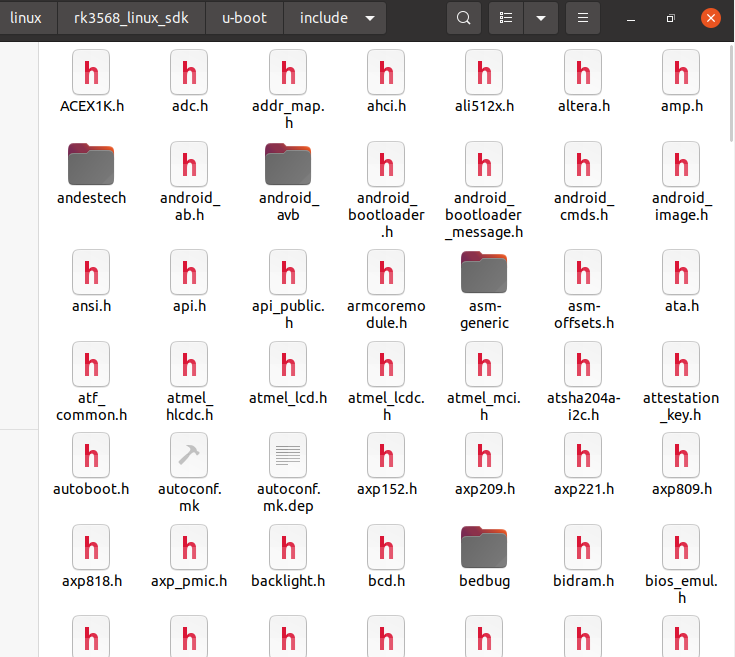
1. fs：文件系统，在uboot环境下能支持很多的文件系统。像ext4直接就可以通过命令查看到它里面的内容。

如：在开发板上电启动uboot后：输入命令：ext4ls mmc 2:1,就是指第二个SDK的第一个分区。

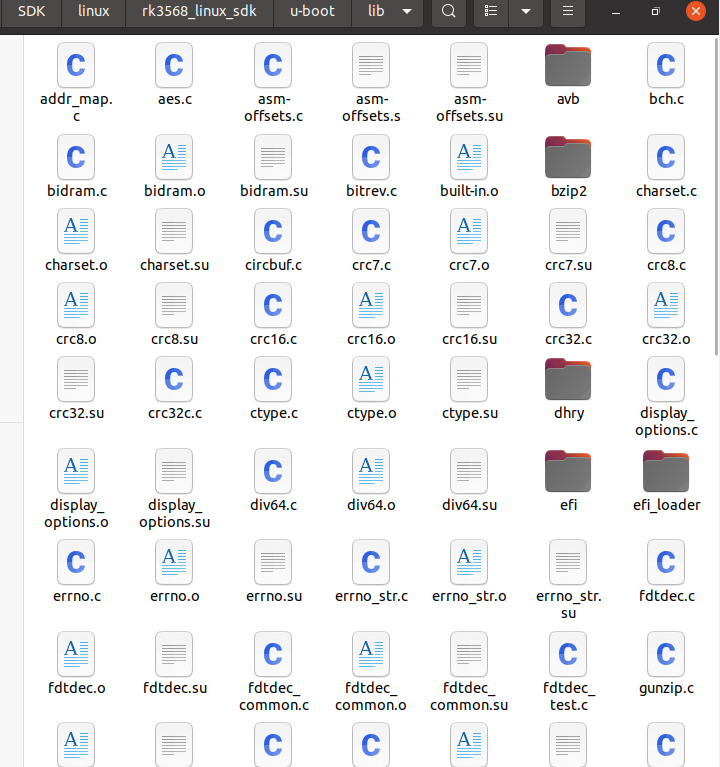
甚至可以看第8个分区：ext4ls mmc 2:8



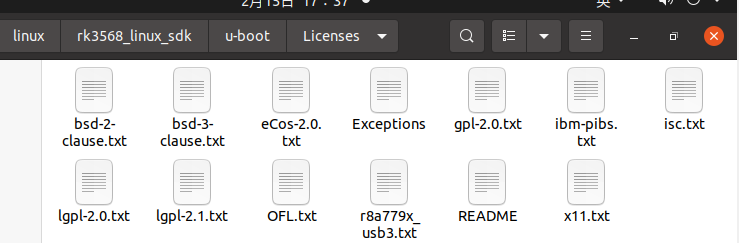
1. include：包含最常用的头文件。头文件一般就在这里查找。



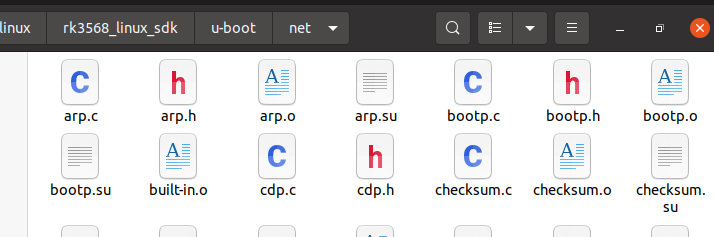
1. lib：这里面是库，在裸机开发的时候就讲过，因为裸机开发的时候是不能使用带有系统调用的库，但我们平常写程序可能用库用习惯了，所以他给了一些和c类似的库便于我们做引导程序开发的调用。



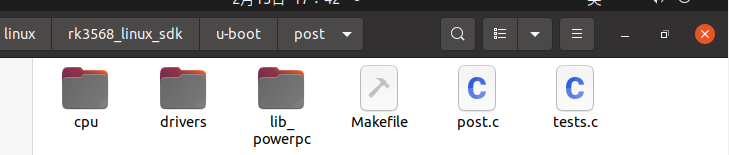
1. Licenses:开源许可协议，暂时不看了：



1. net：这里面并不是网卡驱动，这里面更多的是协议相关的，也就是说我们其实在将来刷机升级的时候除了串口、USB之外，也可以支持网络下载，比如这里支持tftp。这是和网络的协议栈相关的东西。



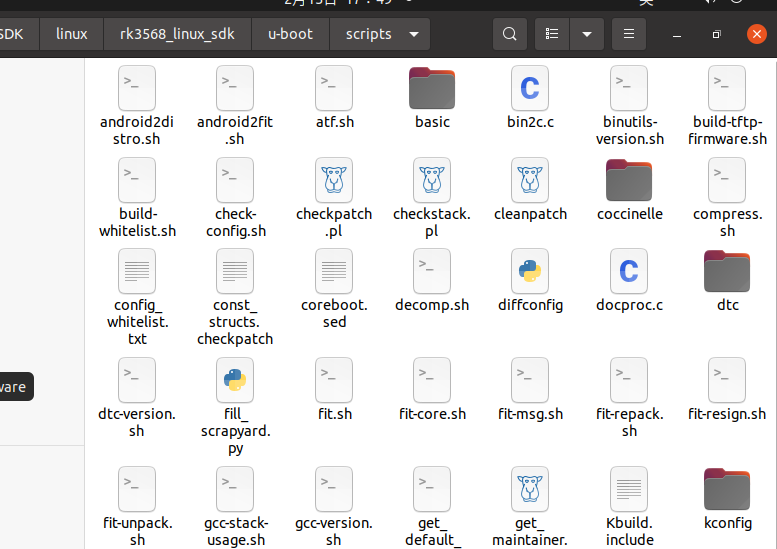
1. Post：一般是用于自检的，也就是说检测系统正常相关的代码，不看。



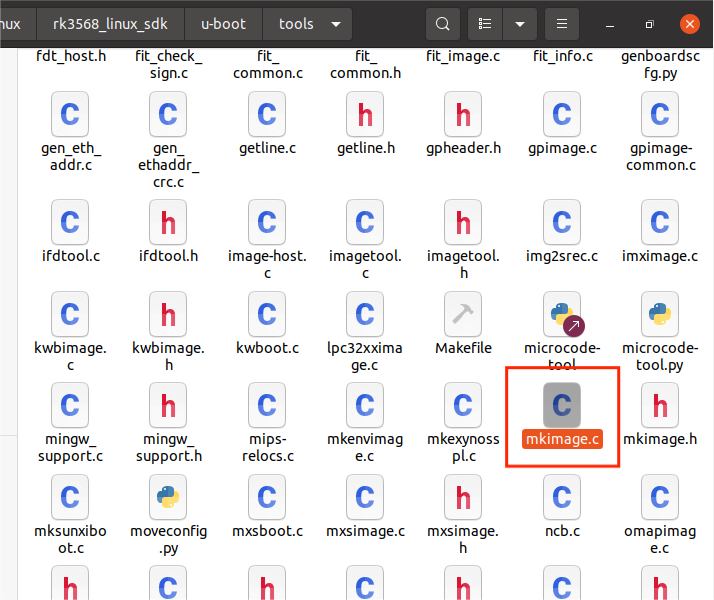
1. Prototype：我们没有该文件，是一些和兼容性相关的东西，比如6818、4418同一个引导程序是不是可以在这上面跑起来。有哪些差异。



1. Scripts:这里面都是一些脚本，这些脚本其实主要是完成系统的配置。后面会看到一些：我们在编译引导程序之前，我们需要对引导程序进行配置相关的工作。这些配置是让他更好地去编出目标平台，所以很多的工作是靠这些脚本文件。



1. Test：测试，暂时不看。
2. Tools：除了制作uboot镜像用到的工具,他还给了我们内核制作成镜像所用到的工具，比如mkimage:



Mkimage尽量拷贝到别的目录:拷贝内核镜像生成工具（编译uimage时用到）：

sudo cp tools/mkimage /bin

——————————————————————————————————

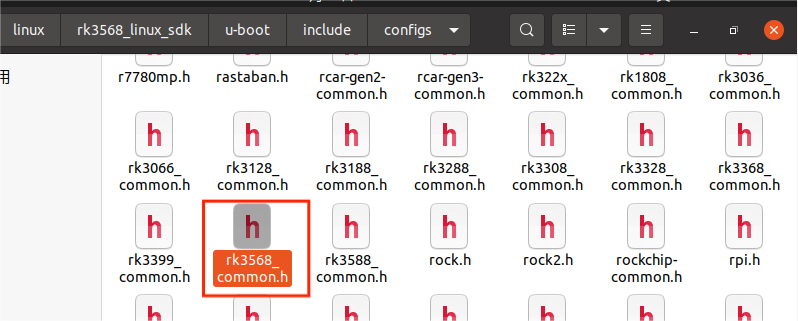
编译uboot:

1. cd uboot
2. 编译之前先清除之前编译的文件（清除编译痕迹，不必每次都做）

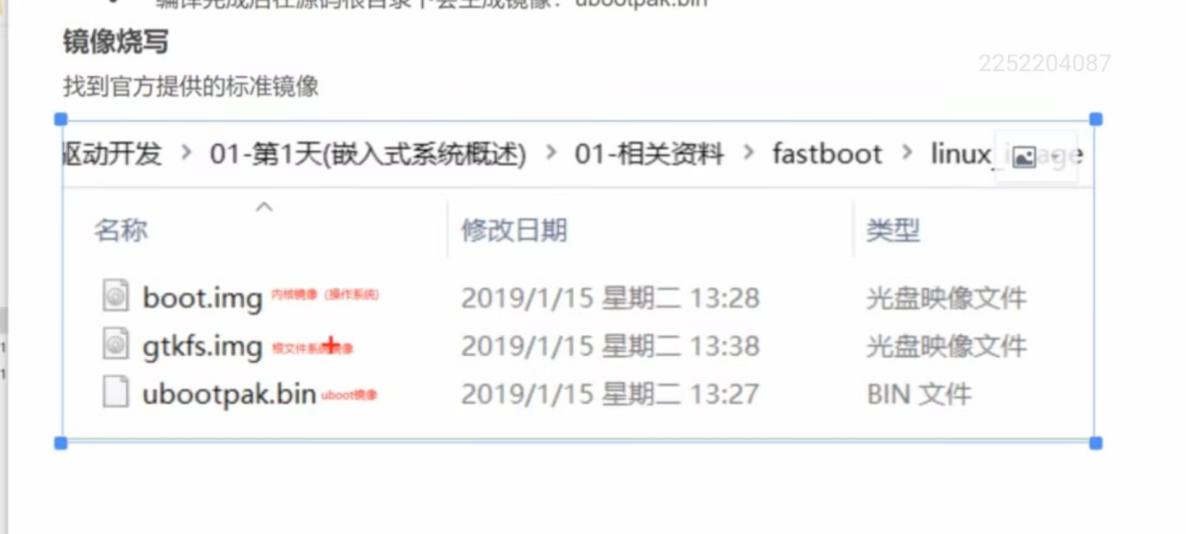
Make distclean

1. 编译前配置成目标平台：例如：make x6818\_config
2. 编译源码：make
3. 编译完成之后会在源码根目录下会生成镜像：ubootpak.bin

为了区分裸机和加入了uboot之后的区别：在下面的路径下，有一个rk3568common.h文件，这个文件是用户自定义的文件，我们可以在这里面稍微修改一些内容。



（这里除了这个文件，其它大多数其实都可以删除）



Boot.img就是内核镜像（操作系统）

gtkfs.img就是根文件系统 镜像

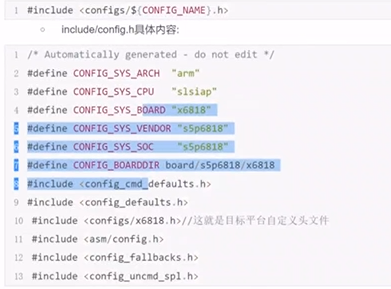
ubootpak.bin就是uboot镜像

mkconfig 创建 include/config.h 并写入一个定制的头文件及目标平台相关的信息：

include/config.h默认是被所有源码所包含的不变的头文件。

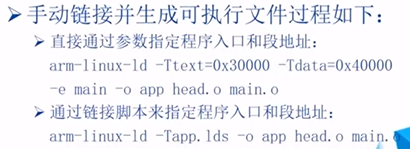
所以只要包含了include/config.h这个头文件的，就会间接x6818.h。

包含方法：#include <config/$(CONFIG\_NAME).h>



IMG_256

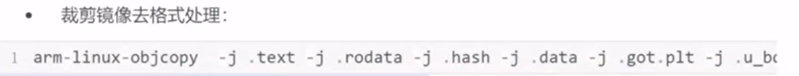




同时有 通过参数指定程序入口和段地址 和 通过链接脚本来指定程序入口和段地址 那么哪一种有效？



使用objcopy工具裁剪镜像去格式处理：



IMG_256

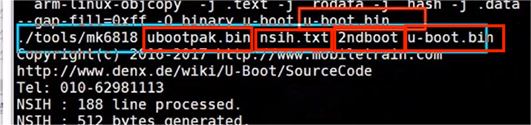
通过这个可以看出来，裁剪镜像去格式之后，最终得到的是uboot.bin。

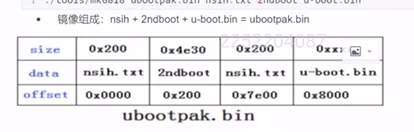
也就是说：uboot源码生成的可执行程序是(真正的源码)：uboot.bin，而不是下载的ubootpak.bin。

uboot.bin如何生成ubootpak.bin：

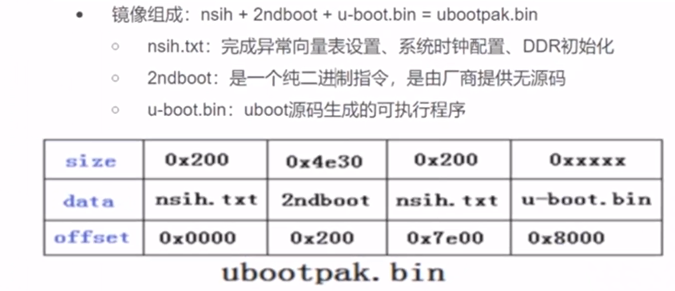
uboot.bin又经过下面的工具，进行镜像打包，最终生成ubootpak.bin

Uboot.bin添加2ndboot和nsih.txt这两个文件 然后通过./tools/mk6818工具最终生成了ubootpak.bin。（三个文件打包成了一个文件）





三个文件的作用：



——————————————————————————————————

Uboot控制命令：

