

RK3308 快速入门指南

发布版本：1.0

作者邮箱：jkand.huang@rock-chips.com

日期：2019.04

文件密级：公开资料

前言

概述

本文档旨在指导工程师拿到Rockchip RK3308 SDK之后，如何快速bring up 板子。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

产品版本

芯片名称	内核版本
RK3308	4.4

修订记录

日期	版本	作者	修改说明
2019-07-04	V1.0	黄开辉、余永镇	初始版本

RK3308 快速入门指南

- 编译
 - 32bit 编译说明
 - 64bit 编译说明
- 音频
- 显示
- WIFI/BT
- USB
- 文件系统
- 常见问题
 - BuildRoot 如何增加一个包？
 - BuildRoot 如何单独编译某个一包？
 - BuildRoot 什么时候需要全部重新编译，什么时候只需要单独编译某一个包？
 - 如何保存u-boot，kernel，buildroot 的配置文件，以及保存后如何编译
 - 如何修改userdata、oem、rootfs 固件的打包格式
 - 为什么Flash 的大小是128M，但是实际可只有107M左右？
 - 如何单独打包uboot.img、boot.img、recovery.img、rootfs.img、userdata.img、oem.img
 - 烧写完固件，为何一直停在Recovery 模式

1、编译

本章简述如何快速编译SDK，更详细说明请参见<Rockchip Linux Software_Developer_Guide_CN.pdf> 第六章SDK 编译。

1.1、32bit 编译说明

1. 一键编译步骤

1. `source envsetup.sh`
2. 选择`rockchip_rk3308_32_debug`
3. 修改编译配置文件，如下节所介绍
4. `./build.sh`
5. `IMAGE` 目录会生成相应的固件

2. 编译配置文件

如一键编译，`source envsetup.sh` 之后，会自动的将`BoardConfig` 文件软连接到对应的配置文件，如下，选择`rockchip_rk3308_32_debug`之后：

```
$ ls -lh device/rockchip/.BoardConfig.mk
lrwxrwxrwx 1 hkh hkh 27 Jul  5 09:04 device/rockchip/.BoardConfig.mk ->
rk3308/BoardConfig_32bit.mk
```

该文件每一行配置都有详细注释，请工程师打开查阅，以下是SDK常规修改项：

```
# Uboot defconfig
# SDK 提供两个配置，rk3308-aarch32(使用kernel dtb)，evb-aarch32-rk3308(使用
uboot dtb)
export RK_UBOOT_DEFCONFIG=rk3308-aarch32
# Kernel defconfig
# SDK 提供两个配置，rk3308_linux_aarch32_debug_defconfig
rk3308_linux_aarch32_defconfig
export RK_KERNEL_DEFCONFIG=rk3308_linux_aarch32_debug_defconfig
# Kernel dts
export RK_KERNEL_DTS=rk3308-voice-module-board-v10-aarch32
# parameter for GPT table
# 分区表配置，修改分区表参考文档 Rockchip-Parameter-File-Format-Version1.4.pdf
export RK_PARAMETER=parameter-32bit.txt
```

3. 参考固件

1.2、64bit 编译说明

1. 键编译步骤

1. `source envsetup.sh`
2. 选择`rockchip_rk3308_release`
3. 修改编译配置文件，如下节所介绍
4. `./build.sh`
5. `IMAGE` 目录会生成相应的固件

2. 编译配置文件

如一键编译，source envsetup.sh 之后，会自动的将BoardConfig 文件软连接到对应的配置文件，如下，选择rockchip_rk3308_release之后：

```
$ ls -lh device/rockchip/.BoardConfig.mk
lrwxrwxrwx 1 hkh hkh 21 Jul  5 10:16 device/rockchip/.BoardConfig.mk ->
rk3308/BoardConfig.mk
```

该文件每一行配置都有详细注释，请工程师打开查阅，以下是SDK常规修改项：

```
# Uboot defconfig
# SDK 提供两个配置，rk3308(使用kernel dtb)，evb-rk3308(使用uboot dtb)
export RK_UBOOT_DEFCONFIG=rk3308
# Kernel defconfig
# SDK 提供两个配置，rk3308_linux_debug_defconfig rk3308_linux_defconfig
export RK_KERNEL_DEFCONFIG=rk3308_linux_defconfig
# Kernel dts
export RK_KERNEL_DTS=rk3308-evb-dmic-pdm-v13
# parameter for GPT table
# 分区表配置，修改分区表参考文档 Rockchip-Parameter-File-Format-Version1.4.pdf
export RK_PARAMETER=parameter-64bit.txt
```

3. 参考固件

2、音频

参考文档<RK3308_Audio_Codec_Introduction_v0.3.0_CN.pdf>

3、显示

参考文档<Rockchip_Developer_Guide_RK3308_DISPLAY_CN.pdf>

4、WIFI/BT

参考文档：<Rockchip Linux WIFI BT 开发指南 V6.0.pdf>

5、USB

参考文档<Rockchip-Developer-Guide-Linux4.4-USB-Gadget-UAC-CN.pdf>

6、文件系统

如果需要使用MTD，参考文档<Rockchip_Developer_Guide_Linux_SPL_MTD_CN.pdf>

7、常见问题

7.1、BuildRoot 如何增加一个包？

参考 < The Buildroot User Manual.pdf > 第17章

7.2、BuildRoot 如何单独编译某个一包？

1. 如果修改了源码，在编译前运行 `make < package >-dirclean`
2. 如果只是修改output 目录下的东西，编译前运行 `make < package >-rebuild`

7.3、BuildRoot 什么时候需要全部重新编译，什么时候只需要单独编译某一个包？

1. 目标体系结构修改时，需要全部重新编译
2. 编译工具链修改时，需要全部重新编译
3. 新增一个包无需全部重新编译，但是如果新增的是一个库，且别其他文件所引用，则需一起重新编译，或者全部重编。
4. 删除一个包的时候，需要全部重新编译，因为BuildRoot不会去删除编译产生的文件，这样会照成文件系统臃肿等问题，但是你没有必要马上重新编译，可以等到最后一起编译。
5. 文件系统框架修改时，除了overlay 之外，都需要全部重新编译。

7.4、如何保存u-boot， kernel， buildroot 的配置文件，以及保存后如何编译

1. u-boot

```
# 进入uboot 目录
cd u-boot
# 选择配置文件
make rk3308_defconfig
# 执行menuconfig命令，打开菜单配置选项，根据需求进行选项配置
make menuconfig
# 保存配置文件
make savedefconfig
cp defconfig configs/rk3308_defconfig
# 编译
# ./make.sh rk3308
```

2. kernel

```
# 进入kernel 目录
cd kernel
# 执行命令，配置rk3308 defconfig:
make ARCH=arm rk3308_linux_aarch32_debug_defconfig
# 执行 menuconfig命令，打开菜单配置选项，根据需求进行选项配置
make ARCH=arm menuconfig
# 配置完成后保存配置文件
make ARCH=arm savedefconfig
# 备份保存修改的配置项
cp defconfig arch/arm/configs/rk3308_linux_aarch32_debug_defconfig
# 编译执行的dts
make ARCH=arm rk3308-voice-module-board-v10-aarch32.img -j20
#单独编译后， 目录下的 zboot.img 替换固件中的 boot.img 即可。
```

3. buildroot

```
# SDK根目录
make menuconfig
# 保存配置文件
make savedefconfig
# 编译
make
```

7.5、如何修改userdata、oem、rootfs 固件的打包格式

BoardConfig.mk 文件，修改下面字段

```
export RK_OEM_FS_TYPE=ext2
export RK_USERDATA_FS_TYPE=ext2
export RK_ROOTFS_TYPE=squashfs
```

修改完成之后，运行./mkfirmware.sh 重新打包即可。

7.6、为什么Flash 的大小是128M，但是实际可只有107M左右？

Rockchip 使用FTL算法进行全盘坏块管理，少掉的20M用于坏块管理。

7.7、如何单独打包uboot.img、boot.img、recovery.img、rootfs.img、userdata.img、oem.img

参考<Rockchip Linux Software_Developer_Guide_CN.pdf> 第六章，第五节模块部分编译。

7.8、烧写完固件，为何一直停在Recovery 模式

V13 的DTB默认存储器配置为NAND，如果是EMMC或者SPI NAND存储器，需要重新配置，修改方法如下：

1. EMMC

```
diff --git a/arch/arm64/boot/dts/rockchip/rk3308-evb-v13.dtsi
b/arch/arm64/boot/dts/rockchip/rk3308-evb-v13.dtsi
index 92675be..5891ff0 100644
--- a/arch/arm64/boot/dts/rockchip/rk3308-evb-v13.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3308-evb-v13.dtsi
@@ -234,6 +234,15 @@
         regulator-max-microvolt = <3300000>;
     };

+    vccio_flash: vccio-flash {
+        compatible = "regulator-fixed";
+        regulator-name = "vccio_flash";
+        regulator-always-on;
+        regulator-boot-on;
+        regulator-min-microvolt = <1800000>;
+        regulator-max-microvolt = <1800000>;
+    };
+
+    vcc_phy: vcc-phy-regulator {
+        compatible = "regulator-fixed";
+        regulator-name = "vcc_phy";
@@ -307,7 +316,7 @@
```

```

        disable-wp;
        non-removable;
        num-slots = <1>;
-       status = "disabled";
+       status = "okay";
    };

    &fiq_debugger {
@@ -333,7 +342,7 @@
        vccio0-supply = <&vcc_io>;
        vccio1-supply = <&vcc_io>;
        vccio2-supply = <&vcc_1v8>;
-       vccio3-supply = <&vcc_io>;
+       vccio3-supply = <&vccio_flash>;
        vccio4-supply = <&vccio_sdio>;
        vccio5-supply = <&vccio_sd>;
    };

```

2. SPI NAND

```

diff --git a/arch/arm64/boot/dts/rockchip/rk3308-evb-amic-v13.dts
b/arch/arm64/boot/dts/rockchip/rk3308-evb-amic-v13.dts
index 2ae880f..1a467b7 100644
--- a/arch/arm64/boot/dts/rockchip/rk3308-evb-amic-v13.dts
+++ b/arch/arm64/boot/dts/rockchip/rk3308-evb-amic-v13.dts
@@ -55,3 +55,7 @@
        rockchip,mode = <1>;
        #sound-dai-cells = <0>;

    };
+
+&sfc {
+    status = "okay";
+};

```