

IDO-EVB3020 Android SDK编译手册

一、源码获取

二、开发板环境配置

三、SDK编译

1. uboot编译步骤

2. kernel编译步骤

3. Android 编译及固件生成步骤

4. 固件打包

一、源码获取

源码下载路径

链接: <https://pan.baidu.com/s/1jBknasoctjD5JOulfpCP4Q>

提取码: 88lc

解压源码

```
1 cat ido_px30_base.tar.gza* | tar -xz -C your/target/path
```

二、开发板环境配置

开发环境搭建, 请参考 [RKDocs/rk3326/RK3326_ANDROID8.1_SDK发布说明_V1.00.pdf](#), 安装 OpenJDK 8和一些编译依赖软件。

三、SDK编译

1. uboot编译步骤

进入 sdk 根目录执行命令

▼ Shell

```
1 $ cd u-boot
2 $ ./make.sh evb-px30
```

编译完会生成 trust.img, px30_loader_v1.10.115.bin, uboot.img 三个文件。

2.kernel编译步骤

内核配置文件路径: kernel/arch/arm64/configs/ido_px30_defconfig

设备树文件路径: kernel/arch/arm64/boot/dts/rockchip/

3020开发板dts

序号	文件名	功能说明
1	ido-px30-sbc3020-lvds-1024-600.dts	7寸 1024x600 LVDS屏
2	ido-px30-sbc3020-mipi-1200-1920.dts	10.1寸 1200x1920 MIPI屏

以编译7寸屏内核为例, 编译方法如下:

▼ Shell

```
1 $ cd kernel
2 $ make ARCH=arm64 ido_px30_defconfig
3 $ make ARCH=arm64 ido-px30-sbc3020-lvds-1024-600.img -j10
```

编译完成后, kernel 根目录, 生成 kernel.img, resource.img 两个文件

3.Android 编译及固件生成步骤

切换至SDK根目录

```
▼ Shell |
1  $ source build/envsetup.sh
2  $ lunch
3
4  You're building on Linux
5
6  Lunch menu... pick a combo:
7      1. aosp_arm-eng
8      2. aosp_arm64-eng
9      3. aosp_mips-eng
10     4. aosp_mips64-eng
11     5. aosp_x86-eng
12     6. aosp_x86_64-eng
13     7. px30_evb-userdebug
14     8. px30_evb-user
15
16  Which would you like? [aosp_arm-eng] 7
17
```

或者

```
▼ Shell |
1  userdebug 版本:
2  $ lunch px30_evb-userdebug
3  user 版本:
4  $ lunch px30_evb-user
```

编译Android

```
▼ Bash |
1  $ make -j20
```

4. 固件打包

编译完成后，执行 SDK 根目录下的 mkimage.sh 脚本生成固件，所有烧写所需的各分区镜像将都 rockdev/Image-px30_evb/ 目录下

```
▼ Shell |
1
2 $ ./mkimage.sh
```

将所有分区镜像合并成单个的镜像

```
▼ Bash |
1 $ cd RKTools/linux/Linux_Pack_Firmware/rockdev/
2 $ ln -sf ../../../../../../rockdev/Image-px30_evb/ Image
3 $ ./mkupdate.sh
```

执行 mkupdate.sh 命令将各分区镜像合并成一个 update.img 的镜像文件。

```
/ido_px30_base/RKTools/linux/Linux_Pack_Firmware/rockdev$ ls -l
afptool
Image -> ../../../../../../rockdev/Image-px30_evb/
mkupdate.sh
package-file
readme.txt
rkImageMaker
unpack.sh
update.img
```