软件开发环境

SDK的软件需要用到Linux环境，比如Ubuntu桌面系统：**Ubuntu 22.04 64bit Desktop / 20.04.4 64bit Desktop**。

安装系统依赖

此部分可自行参考Google官网或相关网络文档。

```
# 可参考Google原生Android源码编译依赖库
$ sudo apt-get install git-core git-lfs gnupg flex bison build-essential zip curl
zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 libncurses5 lib32ncurses5-dev
x11proto-core-dev libx11-dev lib32z1-dev libgl1-mesa-dev libxml2-utils xsltproc
unzip fontconfigc clang
```

完成环境安装后，在没拿到RTD861B SDK前，可自行下载AOSP官方源码（比如：`android14-r38`）进行环境验证。

SDK获取

通过FAE窗口获取。

压缩包大小约80G，建议再预留300G空间用于解压和编译。

SDK编译

可直接运行`build_all.sh`进行完整编译。

注意事项

docker配置

默认配置了docker编译，在根目录下的`build.cfg`中：

```
BUILD_CONFIG_USE_DOCKER_BUILD true
```

需要系统支持docker环境，并将当前用户添加到docker用户组。

如果不支持docker环境编译方式，可以在`build.cfg`中将`BUILD_CONFIG_USE_DOCKER_BUILD`设置为`false`：

```
BUILD_CONFIG_USE_DOCKER_BUILD false
```

CCACHE_DIR配置

AOSP编译会使用ccache加速编译。需要配置一下ccache的目录。例如：

```
# 路径改为自己的实际路径
export CCACHE_DIR=~/disk2/.ccache
```

可能遇到的编译问题

1. `error:Please update ABI references with: SANDROID BUILD TOPheader-checker/utils/create reference dumps.py -l libcrypto`

   ```
   # 按顺序执行以下三步后，再执行build_all.sh
   $ source env.sh
   $ lunch RTD1861AutoBox-userdebug
   $ development/vndk/tools/header-checker/utils/create_reference_dumps.py -p RTD1861AutoBox -l libcrypto
   ```

2. `Failed to generate BTF for vmlinux`

   错误信息：

   ```
   BTF: .tmp_vmlinux.btf: pahole (pahole) is not available
   Failed to generate BTF for vmlinux
   Try to disable CONFIG_DEBUG_INFO_BTF
   make: *** [Makefile:1291: vmlinux] Error 1

   #### failed to build some targets (06:24 (mm:ss)) ####
   ```

   处理方式：

   ```
   # 安装dwarves后正常
   $ sudo apt install dwarves
   ```

3. GLIBC版本过低

   出现类似这样的错误信息，需要确认GLIBC的版本，通常升级版本后会OK。

   ```
   scripts/kconfig/conf: /lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.33' not found (required by scripts/kconfig/conf)
   scripts/kconfig/conf: /lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.34' not found (required by scripts/kconfig/conf)
   ```

编译产出文件

- 正常编译结束的日志参考

  ```
  INFO: ##Creating image(s)... done

  wic create ...done

  real    0m9.106s
  user    1m6.282s
  sys     0m4.138s
  rtk_installer
  customer_drm.tar
  install.bmap
  ```

```
boot_emmc.bind.bin
./build.sh create kent car 8gb [OK]
~/disk2/kent/kent_aosp_auto_bu_250124
img_packing [OK]
docker_run ./build_all.sh [OK]
Deleted Containers:
5300ca885ee103f3f1e8dc919b53e67a90c1cf370afa90100fc37845dc574ab3
7196b1abdf3d630bfc1faa6936422534d2ee52f35dba9808b465fad29fb176cc

Total reclaimed space: 658.7kB
```

以上采用docker方式编译，所以多了些docker相关信息。

- 生存的USB升级文件

```
# USB升级文件
./image_file/out/install.img
./image_file/out/uda_install.bin.gz
./image_file/tmp/rescue.root.emmc.cpio.gz_pad.img
./image_file/tmp/rescue.emmc.dtb
./image_file/tmp/emmc.uImage
```

空片烧录

新的板子还没烧录任何程序，需要参考此章节进行一次空片升级。

现阶段空片烧录只能先通过`UART0`烧录`LK`固件。

- 通过`UART0`连接平台和PC。
  - 需要使用带`Ymode`发送协议的串口助手。比如：`Tera Term`
- 同时按住`Ctrl+Q`后再给平台上电，系统进入recovery mode后可看到以下提示符：

```
d/g/r>
```

- 通过`Ymode`协议烧录LK固件。固件由Realtek FAE提供。

#	command	description
1	'c'	先傳 certificate，等一下驗證程式時會用到 (ex: `Recovery_Certificate.0x01500000.bin`)
2	'h'	傳 hwsetting 以 init ddr (ex: `RL6991_hwsetting_lpddr4x_32gb_1600_final.bin`)
3	'd'	傳 flash_writer 編好的程式 (ex: `demo-0000-rtd1861_car_drm_smpv3-dev.bin`)
4	'v'	用前面的 certificate 驗證程式，verify ok 後就會跳到 ddr (0x01500000) 執行程式燒 flash

最后一步`v`验证OK后，会启动进入LK，可看到如下提示符：

```
Realtek>
```

完成以上步骤后，可进行下一节通过USB升级android系统固件。

USB升级系统固件

通过串口`UART0`连接电脑，按住键盘`Esc`键，再给设备上电，待系统进行升级状态可松开按键，此时可看到以下提示符：

```
Realtek>
```

将前面编译产出文件拷贝到`FAT32`格式的U盘根目录，并插入板子的USB2端口。使用`boot ru`指令启动升级：

```
Realtek>boot ru
```

升级完成后会自动重启并进行Android系统。