

Rockchip

RK3328 软件开发指南

发布版本:1.2

日期:2018.04

前言

概述

文档作为 Rockchip RK3328 软件开发指南，旨在帮助软件开发工程师、技术支持工程师更快上手 RK3328 的开发及调试。

产品版本

芯片名称	内核版本	Android 版本
RK3328	Linux4.4	Android8.1.0

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

修订记录

日期	版本	作者	审核	修改说明
2018-01-18	V1.00	yhc、tzb、zj	cw、zxz	创建初始发布版本
2018-02-10	V1.01	yhc	cw、zxz	更新文档索引目录结构 增加显示框架配置小节
2018-04-09	V1.2	yhc	cw、zxz	增加 5.2 关闭 system 和 vendor 分区 verify

目录

前言.....	I
目录.....	II
1 支持列表.....	1
1.1 DDR 支持列表.....	1
1.2 EMMC 支持列表.....	1
1.3 WiFi/BT 支持列表.....	2
1.4 SDK 软件包适用硬件列表.....	2
1.5 多媒体编解码支持列表.....	2
2 文档/工具索引.....	1
2.1 文档索引.....	1
2.2 工具索引.....	3
3 SDK 编译/烧写.....	1
3.1 SDK 获取.....	1
3.2 SDK 编译配置.....	1
3.3 量产烧写.....	4
4 U-Boot 开发.....	1
4.1 Rockchip U-Boot 简介.....	1
4.2 平台配置.....	1
4.3 固件生成.....	1
4.4 U-Boot 编译.....	2
4.5 U-Boot logo 相关的配置.....	2
5 内核开发常见配置.....	1
5.1 DTS 介绍.....	1
5.2 关闭 system 和 vendor 分区 verify.....	1
5.3 WiFi&BT 的配置.....	2
5.4 GPIO 对应关系注意.....	2
5.5 ARM、GPU、DDR 频率修改.....	2
5.6 温控配置.....	3
5.7 PWM IR 配置.....	3
5.8 DDR 频率修改说明.....	3
6 Android 开发常见配置.....	1
6.1 Android 编译配置.....	1
6.2 预置 APK.....	1
6.3 开/关机动画.....	2
6.4 Parameter 说明.....	2
6.5 新增分区配置.....	2
6.6 显示框架配置.....	2
6.7 OTA 升级.....	2
6.8 预制 Demo.....	3
6.9 DRM Widevine Level 1 配置.....	3
7 系统调试.....	1
7.1 ADB 工具.....	1

7.2 Logcat 工具.....	3
7.3 Procrank 工具.....	4
7.4 Dumpsys 工具.....	5
7.5 音视频问题调试工具及文档.....	6
8 常用工具说明.....	1
8.1 StressTest.....	1
8.2 PCBA 测试工具.....	1
8.3 DDR 测试工具.....	1
8.4 Android 开发工具.....	2
8.5 update.img 打包.....	4
8.6 固件签名工具.....	4
8.7 序列号/Mac/厂商信息烧写-WNpctool 工具.....	5
8.8 OemTool 打包工具.....	6
8.9 量产工具使用.....	7
8.10 Box 厂测工具.....	8

插图目录

图 1-1 EMMC Performance 示例.....	2
图 7-1 跟踪进程内存状态.....	5
图 8-1 Android 开发工具下载镜像.....	2
图 8-2 Android 开发工具升级固件.....	3
图 8-3 Android 开发工具高级功能.....	3
图 8-4 update.img 打包脚本.....	4
图 8-5 固件签名工具.....	4
图 8-6 WNPctool 工具.....	5
图 8-7 WNPctool 工具模式设置.....	6
图 8-8 Oem 工具.....	7
图 8-9 Oem 工具镜像制作文件夹路径要求.....	7
图 8-10 量产工具.....	8
图 8-11 功能测试界面.....	8
图 8-12 老化测试界面.....	8

表格目录

表 1-1 RK3328 DRAM Support Type.....1

表 1-2 RK3328 DDR Support Symbol.....1

表 1-3 RK3328 EMMC Support Symbol.....1

表 1-4 RK3328 硬件说明列表.....2

表 1-5 RK3328 多媒体编解码支持列表.....2

表 2-1 工具索引表格.....3

1 支持列表

1.1 DDR 支持列表

RK3328 DDR 支持 DDR3、DDR4、LPDDR3。

表 1-1 RK3328 DRAM Support Type

Chip	DRAM Support Type
RK3328	DDR3/DDR4/LPDDR3

RK3328 DDR 颗粒支持程度列表，详见 RKDocs\common\Platform support lists 目录下《RK DDR Support List Ver2.29》，下表中所标示的 DDR 支持程度表，只建议选用√、T/A 标示的颗粒。

表 1-2 RK3328 DDR Support Symbol

Symbol	Description
√	Fully Tested and Mass production
T/A	Fully Tested and Applicable
N/A	Not Applicable

1.2 EMMC 支持列表

RK3328 支持 eMMC 4.51, SDIO3.0, 支持 HS200 模式，详见 RKDocs\common\Platform support lists 目录下《RKeMMCSupportList Ver1.38_2018_01_22》，下表中所标示的 EMMC 支持程度表，只建议选用√、T/A 标示的颗粒。

表 1-3 RK3328 EMMC Support Symbol

Symbol	Description
√	Fully Tested , Applicable and Mass Production
T/A	Fully Tested , Applicable and Ready for Mass Production
D/A	Datasheet Applicable,Need Sample to Test
N/A	Not Applicable

1.2.1 高性能 EMMC 颗粒的选取

为了提高系统性能，需要选取高性能的 EMMC 颗粒。请在挑选 EMMC 颗粒前，参照 Rockchip 提供支持列表中的型号，重点关注下厂商 Datasheet 中 performance 一章节。

参照厂商大小以及 EMMC 颗粒读写的速率进行筛选。建议选取顺序读速率>200MB/s、顺序写速率>40MB/s。

如有选型上的疑问，也可直接联系 Rockchip Fae 窗口。

6.1.5 Performance

[Table 23] Performance

Density	Partition Type	Performance	
		Read(MB/s)	Write (MB/s)
16GB	General	285	40
32GB		310	70
64GB		310	140
128GB		310	140
16GB	Enhanced	295	80
32GB		320	150
64GB		320	245
128GB		320	245

图 1- 1 EMMC Performance 示例

1.3 WiFi/BT 支持列表

RK3328 Android8.1 的 Kernel 版本为 Linux4.4，WiFi/BT 支持列表，详见

RKDocs\common\Platform support lists 目录下《Rockchip_WiFi_Situation_20170214》，文档列表中为目前 RK3328 上大量测试过的 Wifi/Bt 芯片列表，建议按照列表上的型号进行选型。如果有其他 WiFi/BT 芯片调试，需要 WiFi/BT 芯片原厂提供 Linux4.4 版本的内核驱动程序。

如果疑问和建议可以与 Rockchip Fae 窗口联系。

1.4 SDK 软件包适用硬件列表

本 SDK 是基于谷歌 Android8.1.0 64bit 系统，适配瑞芯微 RK3328 芯片的软件包，适用于 RK3328 Box Evb 开发板、RK3328 Box 样机板及基于其上所有的开发产品。

若是基于 RK3328 Box Evb 开发板开发，内核配置可参考 rk3328-evb-android.dts 进行改动。

另 SDK 中附带了 RK3328 Box Evb 开发板及 RK3328 Box 样机板的硬件使用说明。

表 1-4 RK3328 硬件说明列表

硬件板	对应文档说明
Evb 开发板	RKDocs\rk3328\RK3328 BOX EVB 用户使用指南 _V1.0_20170223
Box 样机板	RKDocs\rk3328\RK3328 Box 样机硬件设计指南 _V1.0_20170223

1.5 多媒体编解码支持列表

RK3328 多媒体规格，详见表 1-5:

表 1-5 RK3328 多媒体编解码支持列表

多媒体支持	支持 4K VP9 and 4K 10bits H265/H264 视频解码，高达 60fps。
	1080P 多格式视频解码 (VC-1, MPEG-1/2/4, VP8)。支持 JPEG 解码。
	1080P H.264/H.265 格式视频编码。支持 JPEG 编码。
	支持 HDR10,HLG HDR ,支持 SDR 和 HDR 之间的转换。

RK3328 具体的编解码支持列表，详见 RKDocs\rk3328 目录下《RK3328 Multimedia Codec Benchmark v1.0》

2 文档/工具索引

2.1 文档索引

随 RK3328 Box SDK 发布的文档旨在帮助开发者快速上手开发及调试，文档中涉及的内容并不能涵盖所有的开发知识和问题。文档列表也正在不断更新，如有文档上的疑问及需求，请联系我们的 Fae 窗口。

RK3328 SDK 的 RKDocs 目录结构如下所示。

```
RKDocs/
├── android
│   ├── Android8.0_OEM 内容预置功能说明_V1.0_20171122.pdf
│   ├── Android8.0_定制开关机动画（铃音）说明_V1.0_20170923.pdf
│   ├── Android8.0_性能模式使用说明_V1.0_20170923.pdf
│   ├── Android8.0_恢复出厂设置保护功能说明_V1.0_20170923.pdf
│   ├── Android8.0_预安装应用功能说明文档_V1.0_20171109.pdf
│   ├── Android8.0_验证启动功能说明_V1.0_20171109.pdf
│   ├── Android 增加一个分区配置指南 V1.00.pdf
│   ├── Rockchip_android7.1_wifi_配置明 V1.5.pdf
│   ├── Rockchip Android 8.1 BOX 显示框架配置说明文档 V1.0-20180210.pdf
│   ├── ROCKCHIP_PCBA 测试工具开发指南_V1.1_20171222.pdf
│   └── Rockchip Recovery 用户操作指南 V1.03.pdf
├── common
│   ├── camera
│   ├── DDR
│   │   ├── DDR 开发指南.pdf
│   │   └── DDR 问题排查手册.pdf
│   ├── debug
│   │   ├── perf 使用说明.pdf
│   │   ├── RK3399-LOG-EXPLANATION.pdf
│   │   ├── streamline 使用说明.pdf
│   │   └── systrace 使用说明.pdf
│   ├── display
│   │   ├── rockchip_drm_integration_helper-zh.pdf
│   │   ├── Rockchip_DRM_Panel_Porting_Guide_V1.3_20171209.pdf
│   │   └── Rockchip 基于 DRM 框架的 HDMI 开发指南 v1.0-20171214.pdf
│   └── driver
│       ├── Rockchip Audio 开发指南 V1.1-20170215-linux4.4.pdf
│       ├── Rockchip CPU-Freq 开发指南 V1.0.1-20170213.pdf
│       ├── Rockchip-Developer-Guide-linux4.4-PCIe.pdf
│       ├── Rockchip-Developer-Guide-linux4.4-SDMMC-SDIO-eMMC.pdf
│       ├── Rockchip-Developer-Guide-linux4.4-USB.pdf
│       ├── Rockchip-Developer-Guide-MCU.pdf
│       ├── Rockchip-Developer-Guide-SPI.pdf
│       └── Rockchip-Developer-Guide-UART.pdf
```

- | | |—— Rockchip gmac 模块 开发指南 V1.0-20170221.pdf
- | | |—— Rockchip I2C 开发指南 V1.0-20160629.pdf
- | | |—— Rockchip IO-Domain 开发指南 V1.0-20160630.pdf
- | | |—— Rockchip Pin-Ctrl 开发指南 V1.0-20160725.pdf
- | | |—— Rockchip pwm ir 开发指南 V1.00.pdf
- | | |—— Rockchip RK805 开发指南 V1.0-20170217.pdf
- | | |—— Rockchip RK816 开发指南 V1.pdf
- | | |—— Rockchip RK818_6 电量计 开发指南 V2.0-20170525mo.pdf
- | | |—— Rockchip RK818 电量计 开发指南 V1.0-20160725.pdf
- | | |—— Rockchip Thermal 开发指南 V1.0.1-20170428.pdf
- | | |—— Rockchip Vendor Storage Application Note.pdf
- | | |—— Rockchip 休眠唤醒 开发指南 V0.1-20160729.pdf
- | | |—— Rockchip 时钟子模块 开发指南 V1.1-20170210.pdf
- | | |—— Rockchip 电源 独立 DCDC 开发指南 V1.0-20170519.pdf
- | |—— Platform support lists
- | | |—— RK DDR Support List Ver2.29.pdf
- | | |—— RKeMMCSupportList Ver1.38_2018_01_22.pdf
- | | |—— RKISPV11_Camera_Module_AVL_v1.5.pdf
- | | |—— RKISPV1_Camera_Module_AVL_v1.7.pdf
- | | |—— RKNandFlashSupportList Ver2.72_2016_08_30.pdf
- | | |—— Rockchip Kodi 支持程度列表_V2.0_20170715.pdf
- | | |—— Rockchip_WiFi_Situation_20171215.pdf
- | |—— RKTools manuals
- | | |—— Android 开发工具手册.pdf
- | | |—— RKUpgrade_Dll_UserManual.pdf
- | | |—— RK 平台 apache_tomcat_ota 服务器搭建说明.rar
- | | |—— rk 平台量产升级指导文档 V1.1.pdf
- | | |—— Rockchip Box 厂测工具操作说明 V2.0.pdf
- | | |—— Rockchip Parameter File Format Ver1.3.pdf
- | | |—— Rockchip 量产烧录 指南 V1.1-20170214.pdf
- | | |—— WNpctool 简要使用说明_V1.1.0_0920.pdf
- | | |—— 压力测试 Stresstest 文档 forVR_ver3.0.pdf
- | | |—— 量产工具升级及相关问题处理.pdf
- | |—— security
- | | |—— Rockchip_Secure_Boot_Application_Note_V1.2.1_20171128.pdf
- | | |—— Rockchip_TEE 安全 SDK 开发手册_V1.1_20170516.pdf
- | |—— u-boot
- | | |—— Rockchip-Developer-Guide-Trust.pdf
- | | |—— Rockchip U-Boot 开发指南 V3.8-20170214.pdf
- | |—— usb
- | | |—— RK USB Compliance Test Note V1.2.1.pdf
- | | |—— Rockchip-Developer-Guide-linux4.4-USB.pdf
- | | |—— Rockchip-USB-Performance-Analysis-Guide.pdf
- | | |—— Rockchip-USB-SQ-Test-Guide.pdf
- | |—— wifi
- | |—— RealTek wifi 驱动移植说明_V1.1.pdf

- |—— ROCKCHIP_ANDROID_8.1_WIFI 配置说明_V1.2.pdf
- └—— rk3328
 - |—— RK3328 BOX EVB 硬件使用指南 V1.0_20170223.pdf
 - |—— RK3328 Box 样机硬件设计指南_V1.0_20170223.pdf
 - |—— RK3328 Multimedia Codec Benchmark v1.0.pdf
 - └—— Rockchip RK3328 软件开发指南 V1.0-20180123.pdf

2.2 工具索引

随 RK3328 Box SDK 发布的工具，用于开发调试阶段及量产阶段。工具版本会随 SDK 更新不断更新，如有工具上的疑问及需求，请联系我们的 Fae 窗口。

RK3328 SDK 中在 RKTools 目录下附帶了 linux(Linux 操作系统环境下使用工具)、windows (Windows 操作系统环境下使用工具)。

表 2-1 工具索引表格

工具名称	工具说明	工具路径
AndroidTool	分立升级固件及整个 update 升级固件工具	RKTools\windows\AndroidTool_Release_v2.48
FactoryTool	量产升级工具	RKTools\windows\FactoryTool_v1.53
SecureBootTool	固件签名工具	RKTools\windows\SecureBootTool_v1.85_foruser
efuseTool	efuse 烧写工具	RKTools\windows\efuse_v1.37
WNpctool	写号工具	RKTools\windows\WNpctool_Setup_V1.1.9_180118
SD_Firmware_Tool	SD 卡镜像制作	RKTools\windows\SD_Firmware_Tool._v1.46
SpiImageTools	烧录器升级工具	RKTools\windows\SpiImageTools_v1.36
DriverAssitant	驱动安装工具	RKTools\windows\DriverAssitant_v4.5
OemTool	新增分区镜像制作工具	RKTools\windows\OemTool_v1.3
Rockchip 平台 DDR 测试工具	DDR 测试工具	RKTools\windows\Rockchip 平台 DDR 测试工具_V1.35
Rockchip Box 厂测工具	厂测工具	RKTools\windows\Rockchip Box 厂测工具 V2.0-M-20170327
Linux_Pack_Firmware	Linux 打包工具	RKTools\linux\Linux_Pack_Firmware
Linux_SecureBoot	Linux 签名工具	RKTools\linux\Linux_SecureBoot
Linux_Upgrade_Tool	Linux 烧写工具	RKTools\linux\Linux_Upgrade_Tool

3 SDK 编译/烧写

3.1 SDK 获取

SDK 通过瑞芯微代码服务器对外发布。客户向瑞芯微技术窗口申请 SDK，需同步提供 SSH 公钥进行服务器认证授权，获得授权后即可同步代码。关于瑞芯微代码服务器 SSH 公钥授权，请参考《RK3328_ANDROID8.1-BOX-SDK_V1.00_20180210 发布说明.pdf》，该文档与 SDK 一同发布。

3.2 SDK 编译配置

3.2.1 System 分区大小配置

由于 `parameter` 中定义了各个分区的大小，为保证 `parameter` 中的分区配置与系统分区配置关联匹配，编译系统会从 `TARGET_DEVICE_DIR`（不同产品配置不同，可以通过 `source build/envsetup.sh;get_build_var TARGET_DEVICE_DIR` 查看。例如：`device/rockchip/rk3288/rk3328_box`）读取 `parameter` 中的分区大小信息，并设置给 `BOARD_SYSTEMIMAGE_PARTITION_SIZE`。

若客户需要自行配置，可在具体产品目录的 `BoardConfig.mk` 中覆盖自动配置的值。反之，若希望使用系统自动生成的分区配置，删除具体产品中的 `BOARD_SYSTEMIMAGE_PARTITION_SIZE` 配置。

3.2.2 WITH_DEXPREOPT

默认 Android8.1 user-debug 编译方式是不做 odex 预编译的：

- 1) 第一次出厂开机速度较慢
- 2) 恢复出厂设置速度较慢
- 3) `system.img` 大小较小

默认 Android8.1 user 编译方式是做 odex 预编译的：

- 1) 第一次出厂开机速度较快
- 2) 恢复出厂设置速度较快
- 3) `system.img` 大小较大

建议开发调试过程中，都不要做 odex 预编译，原因是 `system.img` 较大、烧录时间较长；调试不方便，需要连 odex 文件一并更新。

以下是 user-debug 编译方式，打开 odex 预编译选项的修改，开发者可根据实际情况自行选择，源文件为 `device/rockchip/common/BoardConfig.mk`：

```
# Enable dex-preoptimization to speed up first boot sequence
ifeq ($(HOST_OS),linux)
  ifeq ($(TARGET_BUILD_VARIANT),user)
    ifeq ($(WITH_DEXPREOPT),)
      WITH_DEXPREOPT ?= true
    endif
  else
    WITH_DEXPREOPT ?= false
  endif
endif
```

3.2.3 jack-server 配置

Android8.1 系统使用 jack-server 作为 java 代码编译器，在编译过程中可能会遇到以下类似的错误：

```
Jack server already installed in "/home/yhx/.jack-server"
Communication error with Jack server (1), try 'jack-diagnose' or see Jack server log
Communication error with Jack server 1. Try 'jack-diagnose'
Communication error with Jack server 1. Try 'jack-diagnose'
```

这种情况主要是由于 jack-server 本身编译器限制，同一个网络端口号不能多个用户同时使用。也就是在服务器上协同开发过程中，多用户同时编译 Android8.1 时，需要配置各自使用不同的网络端口号。

jack-server 的两个配置文件，决定了它所使用的端口号：

```
~/ .jack-server/config.properties
~/ .jack-settings
```

这两个配置文件需要配置两个端口号，分别为服务端端口号，及客户端端口号，两个配置文件中的端口号要匹配。

```
jack.server.service.port=8074
jack.server.admin.port=8075
及
SERVER_PORT_SERVICE=8074
SERVER_PORT_ADMIN=8075
```

配置步骤如下：

- 1) 确保两个配置文件存在，并且权限设置为 0600：

```
chmod 0600 ~/ .jack-server/config.properties
chmod 0600 ~/ .jack-settings
```

- 2) 若两个配置文件不存在，请参照以下文本新建这两个配置文件。

config.properties 文件示例如下（端口号需按实际修改）：

```
jack.server.max-jars-size=104857600
jack.server.max-service=4
jack.server.service.port=8074
jack.server.max-service.by-mem=1\=2147483648\;2\=3221225472\;3\=42949
67296
jack.server.admin.port=8075
jack.server.config.version=2
jack.server.time-out=7200
```

.jack-settings 文件示例如下（端口号需按实际修改）：

```
# Server settings
SERVER_HOST=127.0.0.1
SERVER_PORT_SERVICE=8074
SERVER_PORT_ADMIN=8075

# Internal, do not touch
SETTING_VERSION=4
```

- 3) 修改端口号，请更改 service port 及 admin port 为其他端口号，两个配置文件里的端口号需要匹配。示例如下：

```
jack.server.service.port=8023
```

```
jack.server.admin.port=8024
```

```
SERVER_PORT_SERVICE=8023
```

```
SERVER_PORT_ADMIN=8024
```

- 4) 重新编译 Android, 看是否会报错, 若依然报错, 请尝试更改其他端口号, 直至编译通过。
- 5) 若更改 5 次编译依然无法通过, 可以执行 `jack-admin dump-report` 命令, 解压命令生成的压缩包, 分析 log 日志, 若出现以下 log, 可以重新安装下 libcurl:

```
$ JACK_EXTRA_CURL_OPTIONS=-v jack-admin list server
* Protocol https not supported or disabled in libcurl
* Closing connection -1
Communication error with Jack server 1. Try 'jack-diagnose'
```

3.2.4 全自动编译脚本

为了提高编译的效率, 降低人工编译可能出现的误操作, 该 SDK 中集成了全自动化编译脚本, 方便固件编译、备份。

- 1) 该全自动化编译脚本原始文件存放于:

```
device/rockchip/rk3328/build_box.sh
```

- 2) 在 repo sync 的时候, 通过 manifest 中的 copy 选项拷贝至工程根目录下:

```
<project path="device/rockchip/rk3328" name="rk/device/rockchip/rk3328"
remote="rk" revision="rk33/mid/8.0/develop">
  <copyfile src="buildspec_box.mk" dest="buildspec.mk"/>
  <copyfile src="build_box.sh" dest="build.sh"/>
</project>
```

- 3) 修改 build.sh 脚本中的特定变量以编出对应产品固件。

```
KERNEL_DTS=rk3328-evb-android
```

变量请按实际项目情况, 对应修改:

KERNEL_DTS 变量指定编译 kernel 的产品板极配置, 如使用 RK3328 Box 样机配置, 则可将改变量改为: rk3328-box。

Android 默认编译为 rk3328_box-userdebug 模式, 也可在脚本中对应修改, 可改为 rk3328_box-user 及其它配置:

```
lunch rk3328_box-user
```

4) 指定 update.img 打包用的 loader:

如 RKTools\linux\Linux_Pack_Firmware\rockdev\mkupdate.sh 脚本所示:

```
15 # pause
16 fi
17 ./afptool -pack ./ Image/update.img || pause
18 ./rkImageMaker -RK322H Image/MiniLoaderAll.bin Image/upda
19 echo "Making update.img OK."
```

Windows 打包脚本(RKTools\windows\AndroidTool\rockdev\mkupdate.bat)也是类似, 如下所示:

```
3
4
5 RKImageMaker.exe -RK322H Image\MiniLoaderAll.bin Image\upda
6
7 rem update.img is new format, Image\update.img is old format
8 del Image\update.img
```

update.img 打包用的 loader 被命名位 MiniLoaderAll.bin，由于 SDK 更新兼容 Loader，所以在此通过 u-boot 目录编译生成 rk3328_loader_v1.06.238.bin（拷贝时会重命名为 MiniLoaderAll.bin），需要指定脚本中 loader 文件名。

5) 执行自动编译脚本：

```
source build.sh
```

该脚本会自动配置 JDK 环境变量，编译 u-boot，编译 kernel，编译 Android，继而生成固件，并打包成 update.img。

5) 脚本生成内容：

脚本会将编译生成的固件拷贝至：

IMAGE/RK3328-EVB-ANDROID_8.1.0_*****_RELEASE_TEST/IMAGES 目录下，具体路径以实际生成为准。每次编译都会新建目录保存，自动备份调试开发过程的固件版本，并存放固件版本的各类信息。

该目录下的 update.img 可直接用于 Android 开发工具及工厂烧写工具下载更新。

3.3 量产烧写

量产上考虑到生产效率及工厂工位安排，量产烧写说明详见 RKDocs\common\RKTools manuals 目录下《Rockchip 量产烧录 指南 V1.1-20170214》。

在量产过程中如涉及到工具上的问题，可以联系我们的 Fae 窗口。

4 U-Boot 开发

本节简单介绍 U-Boot 基本概念和编译的注意事项，帮助客户了解 RK 平台 U-Boot 框架，具体 U-Boot 开发细节可参考 RKDocs\common\u-boot 目录下《Rockchip U-Boot 开发指南 V3.8-20170214》。

4.1 Rockchip U-Boot 简介

Rockchip U-Boot 是基于开源的 UBoot 2014.10 正式版进行开发的，主要支持：

- 支持芯片：rk3288、rk3036、rk312x、rk3368、rk3328、rk3366、rk3399 等；
- 支持 Android 平台的固件启动；
- 支持 ROCKUSB 和 Google Fastboot 两种方式烧写；
- 支持 secure boot 固件签名加密保护机制；
- 支持 LVDS、EDP、MIPI、HDMI、CVBS 等显示设备；
- 支持 SDCard、Emmc、Nand Flash、U 盘等存储设备；
- 支持开机 logo 显示、充电动画显示，低电管理、电源管理；
- 支持 I2C、SPI、PMIC、CHARGE、GUAGE、USB、GPIO、PWM、DMA、GMAC、EMMC、NAND、中断等驱动；

4.2 平台配置

平台配置文件位于 U-Boot 根目录下的 configs 文件夹下，其中 Rockchip 相关的以 RK 开头，并根据产品形态分为 MID 和 BOX 两种配置：

```
rk3288_defconfig
rk3126_defconfig
rk3128_defconfig
rk3368_defconfig

rk3288_box_defconfig
rk3128_box_defconfig
rk3036_box_defconfig
rk3368_box_defconfig
rk3328_box_defconfig
```

RK3328 Box 开发调试选用的是 rk3328_box_defconfig 配置。

4.3 固件生成

Rockchip 平台 Loader 分为一级模式和二级模式，根据不同的平台配置生成相应的 Loader 固件。通过宏 CONFIG_SECOND_LEVEL_BOOTLOADER 的定义二级 Loader 模式。

4.3.1 一级 Loader 模式

U-BOOT 作为一级 Loader 模式，那么仅支持 EMMC 存储设备，编译完成后生成的镜像：

```
rk3328_loader_v1.06.238.bin
```

其中 238 是发布的版本号。

4.3.2 二级 Loader 模式

U-Boot 作为二级 Loader 模式，那么固件支持所有的存储设备，该模式下，需要 MiniLoader

支持，通过宏 CONFIG_MERGER_MINILOADER 进行配置生成。同时引入 Arm Trusted Firmware 后会生成 trust image, 这个通过宏 CONFIG_MERGER_TRUSTIMAGE 进行配置生成。

以 rk3328 编译生成的镜像为例：

```
rk3328_loader_v1.06.238.bin
uboot.img
trust.img
```

其中 238 是发布的版本号，rockchip 定义 U-Boot loader 的版本，其中 238 是根据存储版本定义的，客户务必不要修改这个版本。

uboot.img 是 U-Boot 作为二级 loader 的打包。

trust.img 是 trust firmware 的打包镜像。

RK3036、RK3126、RK3128、RK322x、RK3368、RK3366、RK3328 等采用二级 loader 模式。

4.4 U-Boot 编译

RK3328 Box SDK 编译使用的是如下配置：

```
make rk3328_box_defconfig
make ARCHV=aarch64
```

编译完，会生成 trust.img、rk3328_loader_v1.06.238.bin、uboot.img 三个文件。

4.5 U-Boot logo 相关的配置

4.5.1 U-Boot logo 开关配置

Sdk 默认开启 U-Boot logo 功能，以达到更快显示开机 logo 的目的，见 arch/arm64/boot/dts/rockchip/rk3328-android.dtsi 中如下配置：

```
&display_subsystem {
    logo-memory-region = <&drm_logo>;
    status = "okay";

    route {
        route_hdmi: route-hdmi {
            status = "okay";
            logo,uboot = "logo.bmp";
            logo,kernel = "logo_kernel.bmp";
            logo,mode = "fullscreen";
            charge_logo,mode = "center";
            connect = <&vop_out_hdmi>;
        };
    };
};
```

如果需要关闭该功能，请将上述的 dts 文件中改为 status = "disabled"。

4.5.2 U-Boot logo 图片更换

U-boot logo 显示的两张图片是 kernel 根目录下的 logo.bmp 和 logo_kernel.bmp，如果需要更换，用同名的 bmp 替换掉，重新编译 resource.img 即可。

附：logo 替换不一定要两张图片，可以只要一张，如果只有开发者手上只有一张 logo 图片，就保留 logo.bmp 这一张即可。

5 内核开发常见配置

本节简单介绍内核一些常见配置的修改，主要是 dts 的配置，帮助客户更快更方便的进行一些简单的修改。RK3328 kernel 版本是 4.4，config 配置文件统一为 arch/arm64/configs/rockchip_defconfig。RK3328 的串口波特率为 1500000，调试时请保证设置准确。

5.1 DTS 介绍

5.1.1 DTS 说明

RK3328 的 dts 文件在 kernel/arch/arm64/boot/dts/rockchip/下，如 RK3328 evb 评估板的 dts 文件为 rk3328-evb-android.dts。产品的 dts 里需根据具体的产品需求配置 CPU、GPU、DDR 的频率和电压表；配置 io、wifi、bt、温控、电配置等等。

请各位开发者尽量以 SDK 发布的示例产品 dts 文件做参考，进行后期的开发。

5.1.2 新增一个产品 DTS

RK3328 的产品 dts 文件需放在 kernel/arch/arm64/boot/dts/rockchip 下，

1、以 rk3328-evb-android.dts 为参照，拷贝一份 dts 文件命名为 rk3328-product.dts。

2、修改 arch/arm64/boot/dts/Makefile 文件，添加对应 dtb 申明

```
+rk3328-product.dtb
```

3、修改编译脚本或编译命令。

4、重新编译内核。

5.2 关闭 system 和 vendor 分区 verify

调试时方便能否进行 adb remount，可以先关闭 system 分区和 vendor 分区的 verify，请按照如下修改：

```
diff --git a/arch/arm64/boot/dts/rockchip/rk3328-android.dtsi
b/arch/arm64/boot/dts/rockchip/rk3328-android.dtsi
index 3153314..656416b 100644
--- a/arch/arm64/boot/dts/rockchip/rk3328-android.dtsi
+++ b/arch/arm64/boot/dts/rockchip/rk3328-android.dtsi
@@ -31,14 +31,14 @@
                                dev =
"/dev/block/platform/ff520000.dwm mmc/by-name/system";
                                type = "ext4";
                                mnt_flags =
"ro,barrier=1,inode_readahead_blks=8";
-                                fsmgr_flags = "wait,verify";
+                                fsmgr_flags = "wait";
                                };
                                vendor {
                                        compatible = "android,vendor";
                                        dev =
"/dev/block/platform/ff520000.dwm mmc/by-name/vendor";
                                        type = "ext4";
```

```

mnt_flags =
"ro,barrier=1,inode_readahead_blks=8";
-
+
};
};

```

5.3 WiFi&BT 的配置

RK3328 Android 8.1 平台上 WiFi、BT 可做到自动兼容，按照 RK 提供的编译 Android8.1 编译步骤，生成固件后，默认就可以支持相应的 WiFi 模块，并且一套固件可以支持多个 WiFi 模块。目前 rk3328 android 8.1 平台 wifi、bt 模块 android 和 kernel 无需做任何配置。

5.4 GPIO 对应关系注意

关于原理图上的 gpio 跟 dts 里面的 gpio 的对应关系，这边有个需要注意的地方：例如 GPIO4_C0，那么对应的 dts 里面应该是“gpio4 16”。GPIO 分为 4 个端口 PORTA（0-7）、PORTB（8-15）、PORTC（16-23）、PORTD（24-31），每个 PORT 有 8 个 PIN，以此计算可得 C0 是 16，C1 口是 17，以次类推。

GPIO 的使用请参考 RKDocs\common\driver 目录下《Rockchip Pin-Ctrl 开发指南 V1.0-20160725.pdf》

5.5 ARM、GPU、DDR 频率修改

DVFS（Dynamic Voltage and Frequency Scaling）动态电压频率调节，是一种实时的电压和频率调节技术。目前 4.4 内核中支持 DVFS 的模块有 CPU、GPU、DDR。CPU 使用 cpufreq 框架，GPU 和 DDR 使用 devfreq 框架。

CPUFreq 是内核开发者定义的一套支持动态调整 CPU 频率和电压的的框架模型。它能有效的降低 CPU 的功耗，同时兼顾 CPU 的性能。

CPUFreq 通过不同的变频策略，选择一个合适的频率供 CPU 使用，目前的内核版本提供了以下几种策略：

- interactive: 根据 CPU 负载动态调频调压；
- conservative: 保守策略，逐级调整频率和电压；
- ondemand: 根据 CPU 负载动态调频调压，比 interactive 策略反应慢；
- userspace: 用户自己设置电压和频率，系统不会自动调整；
- powersave: 功耗优先，始终将频率设置在最低值；
- performance: 性能优先，始终将频率设置为最高值。

详细的模块功能及配置，请参考 RKDocs\common\driver 目录下《Rockchip CPU-Freq 开发指南 V1.0.1-20170213.pdf》

DEVFreq 是内核开发者定义的一套支持动态调整设备频率和电压的的框架模型。它能有效的降低该设备的功耗，同时兼顾其性能。目前我们的平台，有 GPU 和 DDR 在使用 DEVFreq。DEVFreq 通过不同的变频策略，选择一个合适的频率供设备使用，目前的内核版本提供了以下几种策略：

- Simple Ondemand：根据负载动态调频调压；
- Userspace: 用户自己设置电压和频率，系统不会自动调整；
- Powersave: 功耗优先，始终将频率设置在最低值；
- Performance: 性能优先，始终将频率设置为最高值；
- Dmc Ondemand: 我司实现的 ddr 变频策略，支持负载和场景变频；

GPU 默认使用的是 Simple Ondemand 负载变频，DDR 默认使用 DMC Ondemand 的变频策略是 RK

自己实现的。

5.6 温控配置

在 Linux 内核中，定义一套温控框架 linux Generic Thermal Sysfs Drivers，它可以通过不同的策略控制系统的温度，目前常用的有以下几种策略：

- **power_allocator**: 引入 PID（比例-积分-微分）控制，根据当前温度，动态给各模块分配 power，并将 power 转换为频率，从而达到根据温度限制频率的效果。
- **step_wise** : 根据当前温度，逐级限制频率。
- **userspace**: 不限制频率

详细的模块功能及配置，请参考 RKDocs\common\driver 目录下《Rockchip Thermal 开发指南 V1.0.1-20170428》

5.7 PWM IR 配置

红外遥控的发射电路是采用红外发光二极管来发出经过调制的红外光波；红外接收电路由红外接收二极管、三极管或硅光电池组成，它们将红外发射器发射的红外光转换为相应的电信号，再送后置放大器。鉴于家用电器的品种多样化和用户的使用特点，生产厂家对进行了严格的规范编码，这些编码各不相同，从而形成不同的编码方式，统一称为红外遥控器编码传输协议。目前 RK 平台只支持 NEC 编码的红外协议。

RK3328 平台详细的遥控器适配，键值添加，红外按键定义，及遥控器功能相关调试内容请参考 RKDocs\common\driver 目录下《Rockchip pwm ir 开发指南 V1.00》。

5.8 DDR 频率修改说明

请参考 RKDocs\common\DDR 目录下《DDR 开发指南》如何修改 ddr 频率章节。

6 Android 开发常见配置

本节简单介绍 Android 开发中一些常见配置的修改，RK3328 平台搭载的是最新的 Android8.1.1 系统。

6.1 Android 编译配置

6.1.1 lunch 选项说明

rk3328_box-userdebug: rk3328 平台 box 产品 userdebug (64 位)

rk3328_box-user: rk3328 平台 box 产品 user (64 位)

user 版本开启 odex 预编译, 编译出来的固件偏大, 但会提高开机速度。开发过程中涉及到 apk 及 jar 的更新调试相对麻烦很多。

建议开发调试阶段默认选择 userdebug 编译。

6.1.2 添加一个新的产品

各开发厂商可能有同款芯片不同产品开发的需求, 一套 SDK 需同时编译生成多款产品固件。

RK3328 平台支持 Box 类型各种产品形态, 当需要添加一个新的产品时, 可以基于已有的 rk3328_box 来建立, 如下以建立一个新的平板产品为例进行说明, 具体步骤为:

1) 产品命名规则:

Box 产品名中需带有“box”字样;

请务必遵守以上规则, 否则系统会异常。

2) 新增文件夹 device/rockchip/rk3328/rk3328_box_000, 基于 rk3328_box.mk 创建 rk3328_box_000.mk, 将 rk3328_box 目录下的所有文件拷贝至 rk3328_box_000 目录下。

```
cd device/rockchip/rk3328
mkdir rk3328_box_000
cp rk3328_box.mk ./ rk3328_box_000.mk
cp rk3328_box/* rk3328_box_000/
```

3) 在 device/rockchip/rk3328/ AndroidProducts.mk 中添加:

```
PRODUCT_MAKEFILES := \
    $(LOCAL_DIR)/rk3328.mk \
    $(LOCAL_DIR)/rk3328_box.mk \
    $(LOCAL_DIR)/rk3328_box_000.mk \
```

4) 在 vendorsetup.sh 中添加产品对应的 lunch 选项:

```
add_lunch_combo rk3328_box-eng
add_lunch_combo rk3328_box-userdebug
add_lunch_combo rk3328_box-user
add_lunch_combo rk3328_box_000-userdebug
add_lunch_combo rk3328_box_000-user
```

5) 修改 rk3328_box_000.mk 及 rk3328_box_000 目录下的新产品所需要修改的配置。

6) 修改编译脚本或编译命令, 重新 lunch 产品名称进行新产品编译。

6.2 预置 APK

Android 上的应用预安装功能, 主要是指配置产品时, 根据厂商要求, 将事先准备好的第三方应用预置进 Android 系统。

预安装的 **APK** 应用需要得到对应厂商授权，若因为开发者及客户厂商私自预安装未授权应用进而需要承担法律责任的，**RK** 概不负责。

预安装分为可卸载预安装和不可卸载预安装，本文主要阐述的是可卸载预安装的功能。配置步骤如下：

- 1) 若是希望可卸载预安装，新增文件夹 `device/rockchip/rk3328/rk3328_box/preinstall_del`；若是不可卸载原装，新增文件夹 `device/rockchip/rk3328/rk3328_box/preinstall`。
- 2) 拷贝需要预制的第三方应用到上述文件夹，注意 **apk** 文件名尽量使用英文，避免空格。
- 3) 编译结束后会将预制的文件拷贝至 **system** 固件中。烧录后，系统会自动安装这些应用到 `data/app` 目录。
- 4) 需要注意的是，在 `preinstall` 目录中的应用，即使用户在使用过程中将其卸载，但在恢复出厂设置后，应用又会自动安装。如果希望恢复出厂设置后不再恢复预安装应用，可以将上述文件夹名字改为 `preinstall_del_forever` 即可实现。

6.3 开/关机动画

需要在产品的 `device/rockchip /common/BoardConfig.mk` 中配置 `BOOT_SHUTDOWN_ANIMATION_RINGING := true`，并且准备如下相应资源文件，编译结束后对应的资源文件会拷贝到相应的 `out` 目录下。

将开机动画 复制到 `device/rockchip/common/bootanimation.zip` (源码路径)

将关机动画 复制到 `device/rockchip/common/shutdownanimation.zip` (源码路径)

6.4 Parameter 说明

请参考 `device/rockchip/rk3328/rk3328_box` 目录下 `parameter.txt` 文件来相应修改配置，关于 `parameter` 中各个参数、分区情况细节，请参考 `\RKDocs\common\RKTools manuals` 目录下的《Rockchip Parameter File Format Ver1.3.pdf》文档。

6.5 新增分区配置

请参考 `RKDocs\android\Android 增加一个分区配置指南 V1.00.pdf`

6.6 显示框架配置

请参考 `RKDocs\android\Rockchip Android 8.1 BOX 显示框架配置说明文档 V1.0-20180210.pdf`

6.7 OTA 升级

6.7.1 OTA 介绍

OTA (over the air) 升级是 Android 系统提供的标准软件升级方式。它功能强大，提供了完全升级（完整包）、增量升级模式（差异包），可以通过本地升级，也可以通过网络升级。

详细的 OTA 升级及 Recovery 模块功能及配置，请参考 `RKDocs\android` 目录下《Rockchip Recovery 用户操作指南 V1.00》。

6.7.2 生成完整包

完整包所包含内容：`system.img`、`recovery.img`、`boot.img`

发布一个固件正确的顺序：

- 1、`make -j4`
- 2、`make otapackage -j4`

3、./mkimage.sh ota

发布固件必须使用./mkimage.sh ota,将 boot 与 kernel 打包,不需要单独烧 kernel,如果量产固件是分开的,将会影响后面差异包升级,除非你不需要用差异升级。

在 out/target/product/rkxxxx/目录下会生成 ota 完整包 rkxxxx-ota-eng.root.zip,改成 update.zip 即可拷贝到 T 卡或者内置的 flash 进行升级。

6.7.3 生成差异包

OTA 差异包只有差异内容,包大小比较小,主要用于 OTA 在线升级,也可 T 卡本地升级。OTA 差异包制作需要特殊的编译进行手动制作。

1、首先发布 v1 版本的固件,生成 v1 版本的完整包

2、保存

out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip 为 rkxxxx-target_files-v1.zip,作为 v1 版本的基础素材包。

3、修改 kernel 代码或者 android 代码,发布 v2 版本固件,生成 v2 版本完整包

4、保存

out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip 为 rkxxxx-target_files-v2.zip,作为 v2 版本的基础素材包。

5、生成 v1-v2 的差异升级包:

```
./build/tools/releasetools/ota_from_target_files --block -v -i  
rkxxxx-target_files-v1.zip -p out/host/linux-x86 -k  
build/target/product/security/testkey rkxxxx-target_files-v2.zip  
out/target/product/rk3328/rkxxxx-v1-v2.zip
```

说明: 生成差异包命令格式:

ota_from_target_files

--block

-v -i 用于比较的前一个 target file

-p host 主机编译环境

-k 打包密钥

用于比较的后一个 target file

最后生成的 ota 差异包

6.8 预制 Demo

在开发及样机准备中,多数开发者及厂商有需要集成测试音视频资源、图片资源等,本 SDK 也附带了预置 Demo 资源的功能,详情见 8.8 节 OemTool 打包工具使用。

6.9 DRM Widevine Level 1 配置

待完善

7 系统调试

本节重点介绍 SDK 开发过程中的一些调试工具和调试方法，并会不断补充完善，帮助开发者快速上手基础系统调试，并做出正确的分析。

7.1 ADB 工具

7.1.1 概述

ADB (Android Debug Bridge) 是 Android SDK 里的一个工具，用这个工具可以操作管理 Android 模拟器或真实的 Android 设备。主要功能有：

- 运行设备的 shell (命令行)
- 管理模拟器或设备的端口映射
- 计算机和设备之间上传/下载文件
- 将本地 apk 软件安装至模拟器或 Android 设备

ADB 是一个“客户端—服务器端”程序，其中客户端主要是指 PC，服务器端是 Android 设备的实体机器或者虚拟机。根据 PC 连接 Box 机器的方式不同，ADB 可以分为两类：

- 网络 ADB：主机通过有线/无线网络（同一局域网）连接到 STB 设备
- USB ADB：主机通过 USB 线连接到 STB 设备

7.1.2 USB adb 使用说明

USB adb 使用有以下限制：

- 只支持 USB OTG 口
- 不支持多个客户端同时使用（如 cmd 窗口，eclipse 等）
- 只支持主机连接一个设备，不支持连接多个设备

连接步骤如下：

1、Box 机器已经运行 Android 系统， 设置->开发者选项->已连接到计算机 打开，usb 调试开关打开。

2、PC 主机只通过 USB 线连接到机器 USB otg 口，然后电脑通过如下命令与 Box 机器相连。

```
adb shell
```

3、测试是否连接成功，运“adb devices”命令，如果显示机器的序列号，表示连接成功。

7.1.3 网络 adb 使用要求

adb 早期版本只能通过 USB 来对设备调试，从 adb v1.0.25 开始，增加了对通过 tcp/ip 调试 Android 设备的功能。

如果你需要使用网络 adb 来调试设备，必须要满足如下条件：

- 1、设备上面首先要有网口，或者通过 WiFi 连接网络。
- 2、设备和研发机（PC 机）已经接入局域网，并且设备设有局域网的 IP 地址。
- 3、要确保研发机和设备能够相互 ping 得通。
- 4、研发机已经安装了 adb。
- 5、确保 Android 设备中 adbd 进程（adb 的后台进程）已经运行。adbd 进程将会监听端口 5555 来进行 adb 连接调试。

7.1.4 SDK 网络 adb 端口配置

SDK 默认未开启网络 adb，需要手动在开发者选项中打开。

7.1.5 网络 adb 使用

本节假设设备的 ip 为 192.168.1.5，下文将会用这个 ip 建立 adb 连接，并调试设备。

1、首先 Android 设备需要先启动，如果可以的话，可以确保一下 **adb** 启动(ps 命令查看)。

2、在 PC 机的 cmd 中，输入：

```
adb connect 192.168.1.5:5555
```

如果连接成功会进行相关的提示，如果失败的话，可以先 **kill-server** 命令，然后重试连接。

```
adb kill-server
```

3、如果连接已经建立，在研发机中，可以输入 **adb** 相关的命令进行调试了。比如 **adb shell**，将会通过 **tcp/ip** 连接设备上。和 **USB** 调试是一样的。

4、调试完成之后，在研发机上面输入如下的命令断开连接：

```
adb disconnect 192.168.1.5:5555
```

7.1.6 手动修改网络 adb 端口号

若 SDK 未加入 **adb** 端口号配置，或是想修改 **adb** 端口号，可通过如下方式修改：

1、首先还是正常地通过 **USB** 连接目标机，在 windows cmd 下执行 **adb shell** 进入。

2、设置 **adb** 监听端口：

```
#setprop service.adb.tcp.port 5555
```

3、通过 **ps** 命令查找 **adb** 的 pid

4、重启 **adb**

```
#kill -9<pid>，这个 pid 就是上一步找到那个 pid
```

杀死 **adb** 之后，android 的 **init** 进程后自动重启 **adb**。**adb** 重启后，发现设置了 **service.adb.tcp.port**，就会自动改为监听网络请求。

7.1.7 ADB 常用命令详解

（1）查看设备情况

查看连接到计算机的 Android 设备或者模拟器：

```
adb devices
```

返回的结果为连接至开发机的 Android 设备的序列号或是 IP 和端口号（Port）、状态。

（2）安装 apk

将指定的 **apk** 文件安装到设备上：

```
adb install <apk 文件路径>
```

示例如下：

```
adb install "F:\WishTV\WishTV.apk"
```

重新安装应用：

```
adb install -r <apk 文件路径>
```

示例如下：

```
adb install -r "F:\WishTV\WishTV.apk"
```

（3）卸载 apk

完全卸载：

```
adb uninstall <package>
```

示例如下：

```
adb uninstall com.wishtv
```

（4）使用 rm 移除 apk 文件：

```
adb shell rm <filepath>
```

示例如下：

```
adb shell
rm "system/app/WishTV.apk"
```

示例说明：移除“system/app”目录下的“WishTV.apk”文件。

（5）进入设备和模拟器的 shell

进入设备或模拟器的 shell 环境：

```
adb shell
```

（6）从电脑上传文件到设备

用 push 命令可以把本机电脑上的任意文件或者文件夹上传到设备。本地路径一般指本机电脑；远程路径一般指 adb 连接的单板设备。

```
adb push <本地路径> <远程路径>
```

示例如下：

```
adb push "F:\WishTV\WishTV.apk" "system/app"
```

示例说明：将本地“WishTV.apk”文件上传到 Android 系统的“system/app”目录下。

（7）从设备下载文件到电脑

pull 命令可以把设备上的文件或者文件夹下载到本机电脑中。

```
adb pull <远程路径> <本地路径>
```

示例如下：

```
adb pull system/app/Contacts.apk F:\
```

示例说明：将 Android 系统“system/app”目录下的文件或文件夹下载到本地“F:\”目录下。

（8）查看 bug 报告

需要查看系统生成的所有错误消息报告，可以运行 adb bugreport 指令来实现，该指令会将 Android 系统的 dumphsys、dumpstate 与 logcat 信息都显示出来。

（9）查看设备的系统信息

在 adb shell 下查看设备系统信息的具体命令。

```
adb shell getprop
```

7.2 Logcat 工具

Android 日志系统提供了记录和查看系统调试信息的功能。日志都是从各种软件和一些系统的缓冲区中记录下来的，缓冲区可以通过 Logcat 来查看和使用。Logcat 是调试程序用的最多的功能。该功能主要是通过打印日志来显示程序的运行情况。由于要打印的日志量非常大，需要对其进行过滤等操作。

7.2.1 Logcat 命令使用

用 logcat 命令来查看系统日志缓冲区的内容：

基本格式：

```
[adb] logcat [<option>] [<filter-spec>]
```

示例如下：

```
adb shell
logcat
```

7.2.2 常用的日志过滤方式

控制日志输出的几种方式：

- 控制日志输出优先级。

示例如下：

```
adb shell
logcat *:W
```

示例说明：显示优先级为 **warning** 或更高的日志信息。

- 控制日志标签和输出优先级。

示例如下：

```
adb shell
logcat ActivityManager:I MyApp:D *:S
```

示例说明：支持所有的日志信息，除了那些标签为“ActivityManager”和优先级为“Info”以上的、标签为“MyApp”和优先级为“Debug”以上的。

- 只输出特定标签的日志

示例如下：

```
adb shell
logcat WishTV:* *:S
```

或者

```
adb shell
logcat -s WishTV
```

示例说明：只输出标签为 **WishTV** 的日志。

- 只输出指定优先级和标签的日志

示例如下：

```
adb shell
logcat WishTV:I *:S
```

示例说明：只输出优先级为 **I**，标签为 **WishTV** 的日志。

7.3 Procrank 工具

Procrank 是 Android 自带一款调试工具，运行在设备侧的 **shell** 环境下，用来输出进程的内存快照，便于有效的观察进程的内存占用情况。

包括如下内存信息：

- **VSS**: Virtual Set Size 虚拟耗用内存大小（包含共享库占用的内存）
- **RSS**: Resident Set Size 实际使用物理内存大小（包含共享库占用的内存）
- **PSS**: Proportional Set Size 实际使用的物理内存大小（比例分配共享库占用的内存）
- **USS**: Unique Set Size 进程独自占用的物理内存大小（不包含共享库占用的内存）

注意：

- **USS** 大小代表只属于本进程正在使用的内存大小，进程被杀死后会被完整回收；
- **VSS/RSS** 包含了共享库使用的内存，对查看单一进程内存状态没有参考价值；
- **PSS** 是按照比例将共享内存分割后，某单一进程对共享内存区的占用情况。

7.3.1 使用 procrank

执行 **procrank**，前需要先让终端获取到 **root** 权限

```
su
```

命令格式：

```
procrank [ -W ] [ -v | -r | -p | -u | -h ]
```

常用指令说明：

- **-v**: 按照 **VSS** 排序

- -r: 按照 RSS 排序
- -p: 按照 PSS 排序
- -u: 按照 USS 排序
- -R: 转换为递增[递减]方式排序
- -w: 只显示 working set 的统计计数
- -W: 重置 working set 的统计计数
- -h: 帮助

示例:

- 输出内存快照:

```
procrank
```

- 按照 VSS 降序排列输出内存快照:

```
procrank -v
```

默认 procrank 输出是通过 PSS 排序。

7.3.2 检索指定内容信息

查看指定进程的内存占用状态，命令格式如下：

```
procrank | grep [cmdline | PID]
```

其中 `cmdline` 表示需要查找的应用程序名，`PID` 表示需要查找的应用进程。

输出 `systemUI` 进程的内存占用状态：

```
procrank | grep "com.android.systemui"
```

或者：

```
procrank | grep 3396
```

7.3.3 跟踪进程内存状态

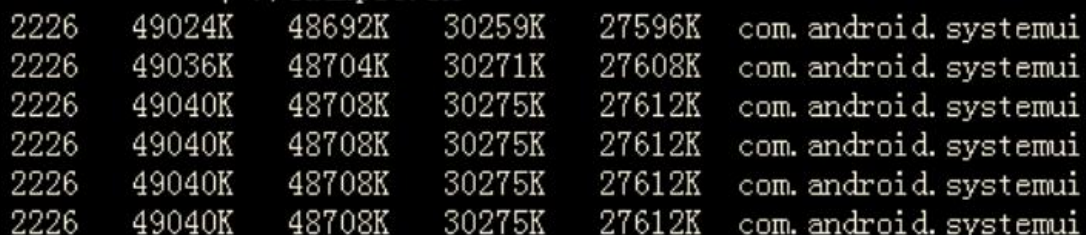
通过跟踪内存的占用状态，进而分析进程中是否存在内存泄露场景。使用编写脚本的方式，连续输出进程的内存快照，通过对比 `USS` 段，可以了解到此进程是否内存泄露。

示例：输出进程名为 `com.android.systemui` 的应用内存占用状态，查看是否有泄露：

1、编写脚本 `test.sh`

```
#!/bin/bash
while true;do
adb shell procrank | grep "com.android.systemui"
sleep 1
done
```

2、通过 `adb` 工具连接到设备后，运行此脚本：`./test.sh`。如图所示。



2226	49024K	48692K	30259K	27596K	com.android.systemui
2226	49036K	48704K	30271K	27608K	com.android.systemui
2226	49040K	48708K	30275K	27612K	com.android.systemui
2226	49040K	48708K	30275K	27612K	com.android.systemui
2226	49040K	48708K	30275K	27612K	com.android.systemui
2226	49040K	48708K	30275K	27612K	com.android.systemui

图 7-1 跟踪进程内存状态

7.4 Dumpsys 工具

`Dumpsys` 工具是 `Android` 系统中自带的一款调试工具，运行在设备侧的 `shell` 环境下，提供系统中正在运行的服务状态信息功能。正在运行的服务是指 `Android binder` 机制中的服务端进

程。

dumpsys 输出打印的条件:

- 1、只能打印已经加载到 ServiceManager 中的服务;
- 2、如果服务端代码中的 dump 函数没有被实现, 则没有信息输出。

7.4.1 使用 Dumpsys

- 查看 Dumpsys 帮助

作用: 输出 dumpsys 帮助信息。

```
dumpsys -help
```

- 查看 Dumpsys 包含服务列表

作用: 输出 dumpsys 所有可打印服务信息, 开发者可以关注需要调试服务的名称。

```
dumpsys -l
```

- 输出指定服务的信息

作用: 输出指定的服务的 dump 信息。

格式: dumpsys [servicename]

示例: 输出服务 SurfaceFlinger 的信息, 可执行命令:

```
dumpsys SurfaceFlinger
```

- 输出指定服务和应有进程的信息

作用: 输出指定服务指定应用进程信息。

格式: dumpsys [servicename] [应用名]

示例: 输出服务名为 meminfo, 进程名为 com.android.systemui 的内存信息, 执行命令:

```
dumpsys meminfo com.android.systemui
```

注意: 服务名称是大小写敏感的, 并且必须输入完整服务名称。

7.5 音视频问题调试工具及文档

RKDocs\common\video 目录下《Rockchip Videodebug_V1.0_20170109》为音视频问题调试工具包, 其中也包含了详细的调试说明文档《Rockchip 音视框架调试文档 V0.4》可帮助开发者对音视频问题做出基本的排查和分析。

8 常用工具说明

本节简单介绍 SDK 附带的一些开发及量产工具的使用说明，方便开发者了解熟悉 RK 平台工具的使用。详细的工具使用说明请见 RKTools 目录下各工具附带文档，及 RKDocs\common\RKTools manuals 目录下工具文档。

8.1 StressTest

设备上使用 Stresstest 工具，对待测设备的各项功能进行压力测试，确保各项整个系统运行的稳定性。SDK 通过打开计算器应用，输入“83991906=”暗码，可启动 StressTest 应用，进行各功能压力测试。

Stresstest 测试工具测试的内容主要包括：

模块相关

- Camera 压力测试：包括 Camera 打开关闭，Camera 拍照以及 Camera 切换。
- Bluetooth 压力测试：包括 Bluetooth 打开关闭。
- Wifi 压力测试：包括 Wifi 打开关闭，（ping 测试以及 iperf 测试待加入）。

非模块相关

- 飞行模式开关测试。
- 休眠唤醒拷机测试。
- 视频拷机测试。
- 重启拷机测试
- 恢复出厂设置拷机测试。
- Arm 变频测试
- Gpu 变频测试
- DDR 变频测试

8.2 PCBA 测试工具

PCBA 测试工具用于帮助在量产的过程中快速地甄别产品功能的好坏，提高生产效率。目前包括屏幕（LCD）、无线（wifi）、蓝牙（bluetooth）、DDR/EMMC 存储、SD 卡（sdcard）、UST HOST、按键（KEY），喇叭耳机（Codec）测试项目。

这些测试项目包括自动测试项和手动测试项，无线网络、DDR/EMMC、以太网为自动测试项，按键、SD 卡、USB HOST、Codec、为手动测试项目。

具体 PCBA 功能配置及使用说明，请参考 RKDocs\android\ROCKCHIP_PCBA 测试工具开发指南_V1.1_20171222.pdf。

8.3 DDR 测试工具

设备上使用 DDR 测试工具，对待测设备的 DDR 进行稳定性测试，确保 DDR 功能正常及稳定。RK3328 DDR 测试工具还未发布，后续会随 SDK 更新。

8.4 Android 开发工具

8.4.1 下载镜像

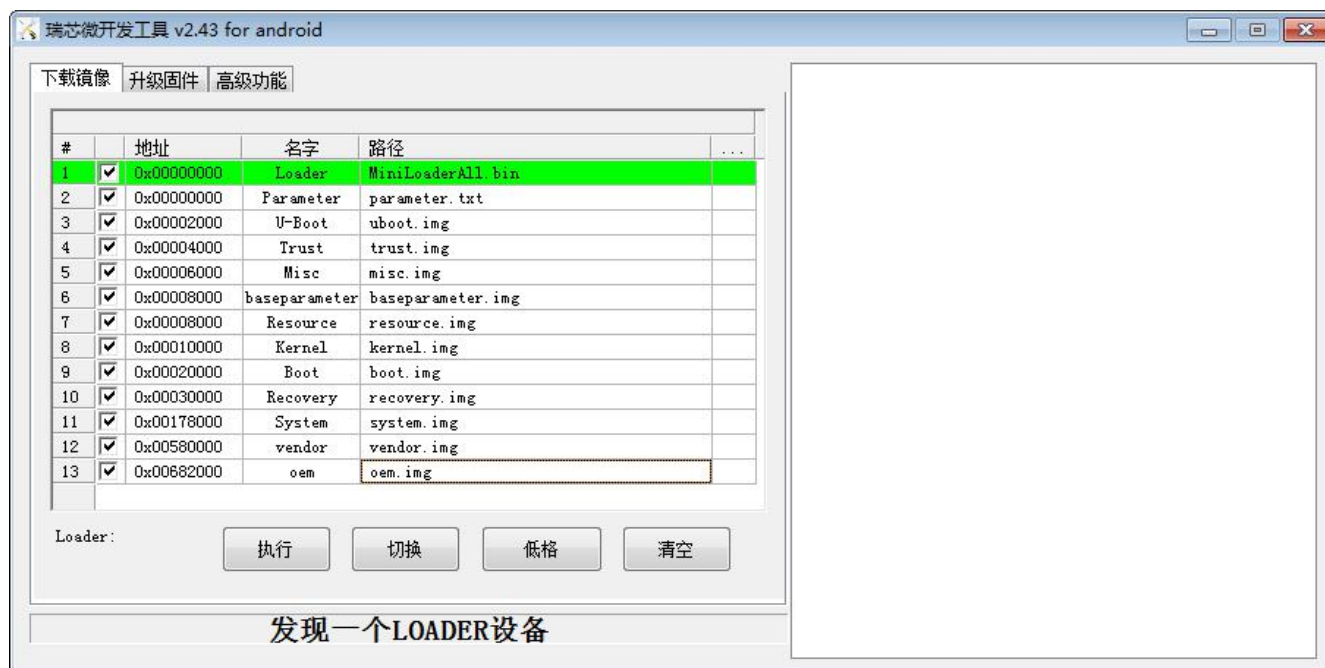


图 8-1 Android 开发工具下载镜像

1) 连接开发板进入下载模式(下载模式先按住开发板 **reset** 按键,再长按 **recover** 按键约 3-4s 时间进入 **loader** 模式)。

2) 打开工具点击下载镜像菜单,点击红色箭头对应列会跳出来一个文件选择框,可以选择对应分区的 **img** 本地地址,其他几项依次配置。

3) 配置完成后,点击执行就可以看到右边空白框进入下载提示。

其中“低格”按钮是用来擦除设备的,“清空”按钮是清空编辑框文本。

8.4.2 升级固件



图 8-2 Android 开发工具升级固件

- 1) 进行打包固件。
- 2) 点击固件选择刚打包好的 `update.img` 文件，并点击升级按钮进行下载。（注意设备必须在下载模式下）。

8.4.3 高级功能



图 8-3 Android 开发工具高级功能

Boot 只能选择打包好的 `update.img` 文件或是 `loader` 的文件；
固件必须使用打包后的 `update.img`；
解包功能可将 `update.img` 拆解为各部分镜像文件。

8.5 update.img 打包

RK3328 平台支持将各零散镜像文件，打包成一个完成的 update.img 形式，方便量产烧写及升级。具体打包步骤如下：

1) 打开 AndroidTool 工具目录底下的 rockdev 目录。编辑 package-file。

按照 package-file 进行配置，package-file 里面有一些 img 镜像放在 Image 目录底下的，如果没有该目录存在，则自己手工新建该 Image 目录，并将需要放到 Image 目录的镜像放进去即可。且注意配置时，镜像名字的准确。其中注意 bootloader 选项，应该根据自己生成的 loader 名称进行修改。

2) 编辑 mkupdate.bat

```
1 Afptool -pack .\backupimage backupimage\backup.img
2 Afptool -pack ./ Image\update.img
3
4
5 RKImageMaker.exe -RK322H RK3328MiniLoaderAll.bin Image\update.img update.img -os_type:android
6
7 rem update.img is new format, Image\update.img is old format, so delete older format
8 del Image\update.img
9
10 pause
```

图 8-4 update.img 打包脚本

需要修改 loader 名称为实际存放的 loader 名称即可。

3) 点击 mkupdate.bat 运行即可，运行完会在该目录生成一个 update.img。

8.6 固件签名工具

选择 chip 类型和加密类型，如果是 RK3328 则选择 efuse。

点击“Generate Key Pairs”按钮，则会生成公私钥对，点击保存。

点击加载密钥，会连续跳出来两次选择密钥文件的界面，第一次为选择私钥文件，第二次为公钥选择文件。

点击“Sign Firmware”按钮，签名 update.img 文件。



图 8-5 固件签名工具

附键盘输入 R+K+Ctrl+Alt 键可打开右侧隐藏功能。

8.7 序列号/Mac/厂商信息烧写-WNpctool 工具

在 RK3328 平台上，序列号/Mac/厂商信息烧写，都是使用 WNpctool 工具进行的。以下说明该工具基本的用法。

8.7.1 序列号获取

在 RK3328 平台上当未用工具烧写过序列号时，默认是读取 WiFi Mac 地址，并依此随机产生一个序列号的。若需要读取工具烧录的序列号值，需要手动修改对应的配置选项。

需修改/system/core/ drmservice/drmservice.c 文件中：

```
#define SERIALNO_FROM_IDB 1 //if 1 read sn from idb3; if 0 generate sn auto  
设为 1 后，默认会从 vendor storage 中读取工具写入的序列号。
```

8.7.2 WNpctool 写入步骤

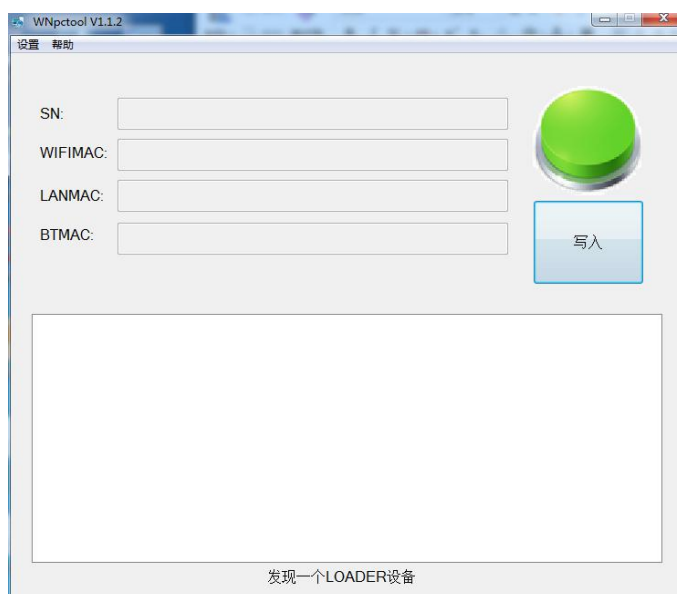


图 8-6 WNpctool 工具

- 1) 进入 loader 模式。
- 2) 点击设置按钮，会有一个下拉框按钮，点击“读取”按钮，用来切换是写入还是读取功能。切换到写入功能。
- 3) 点击模式，出现下列窗口，用来设置 SN/WIFI/LAN/BT

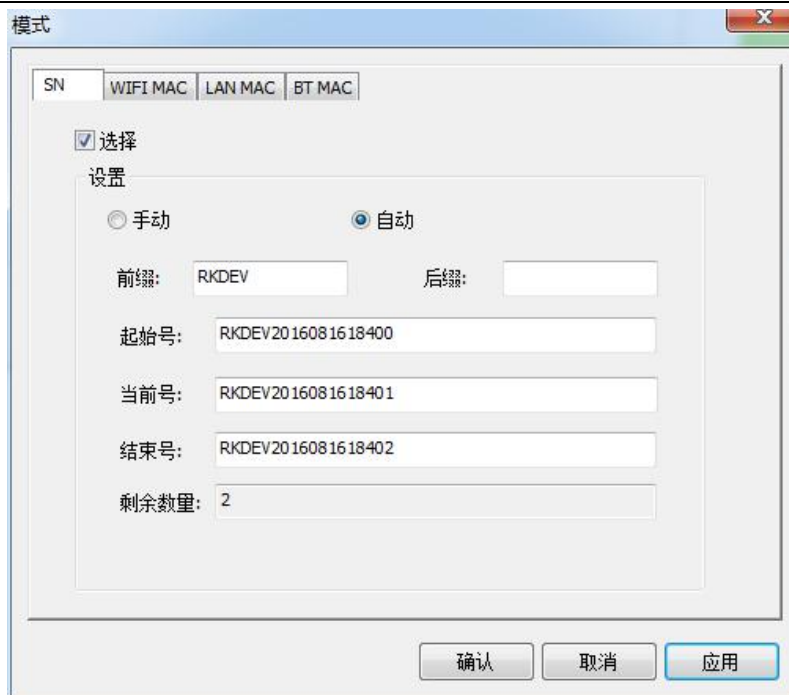


图 8-7 WNpctool 工具模式设置

4) 设置完成后，点击应用按钮，关闭窗口，返回主窗口，点击写入按钮即可。

8.7.3 WNpctool 读取步骤

- 1) 进入 loader 模式。
- 2) 点击设置按钮，会有一个下拉框按钮，点击“读取”按钮，用来切换是写入还是读取功能。切换到读取功能。
- 3) 点击“读取”即可。

8.8 OemTool 打包工具

8.8.1 Oem 打包工具步骤

RK3328 只支持 Ext4 镜像格式，故镜像格式选择 Ext4。下载分区默认 UserData 分区，可直接不填写。



图 8-8 Oem 工具

点击选择按钮选择要打包的数据，数据必须是目录。目录最外围默认为 data 目录，假设你目录为/data/media/0,且 0 有一个文件为 sss.txt(如下图示)。则当你升级完 demo 镜像的时候，会在 RK3328 系统上的 data 目录下有目录 media/0,且在 0 目录下有文件 sss.txt 存在。



图 8-9 Oem 工具镜像制作文件夹路径要求

文件选择成功后，直接点击开始执行，会在 Oem 工具目录生成一个 OemImage.img 镜像。将镜像放在 FactoryTool 工具上下载即可。

8.9 量产工具使用

8.9.1 工具下载步骤

- 1) 点击固件按钮，选择打包工具打包后的 update.img，等待解包成功。
- 2) 如果需要 demo 镜像，则点击 Demo 拷贝按钮，添加由 Oem 工具打包的镜像，并单击 Demo 复选框。
- 3) 连接设备，并让设备进入 loader 或者 maskrom 模式，工具会自动进行下载。
- 4) 可同时连接多台设备，进行一拖多烧写，提高工厂烧写效率。

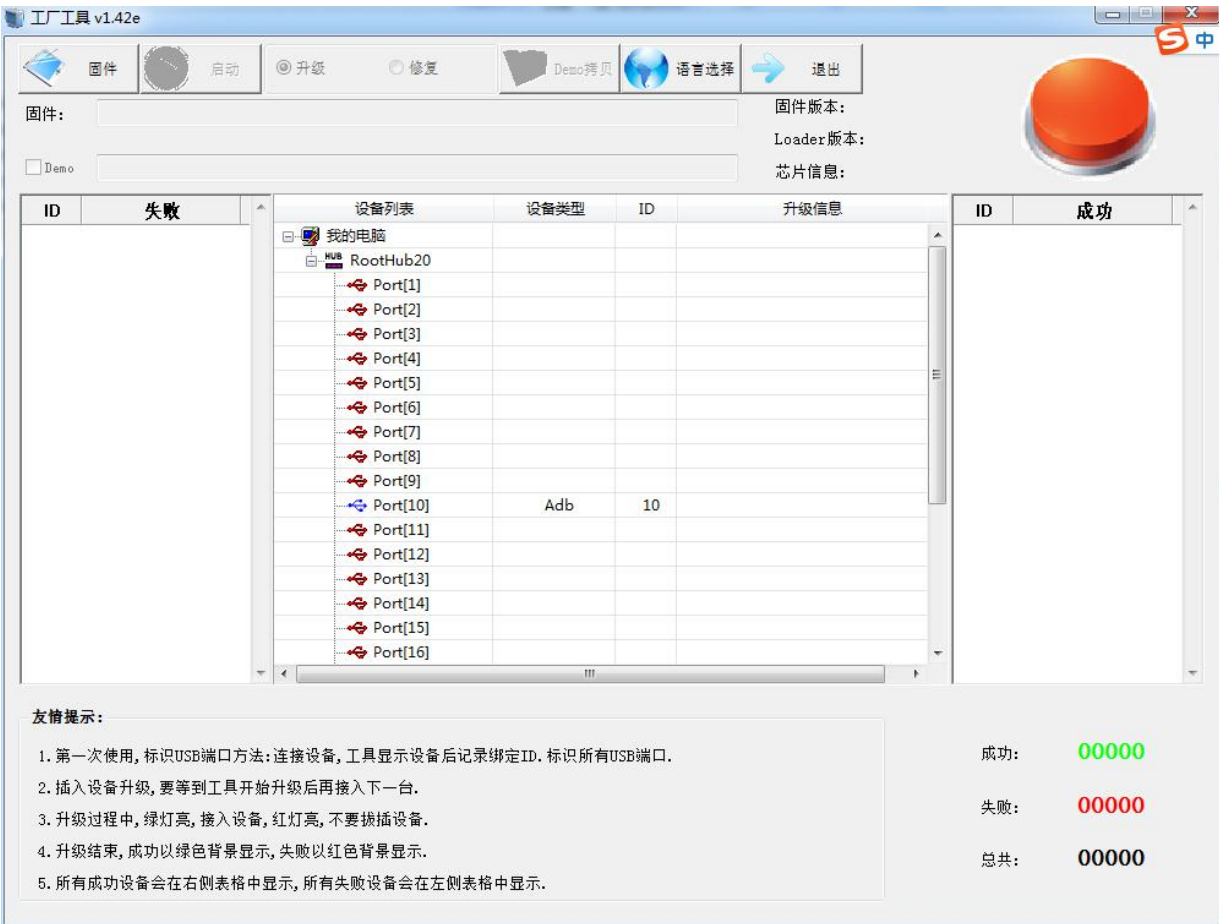


图 8-10 量产工具

8.10 Box 厂测工具

本测试工具用于帮助在量产过程中测试设备的好坏以及长时间运行的老化稳定性测试。测试工具只需用 U 盘或 SDCard 引导启动，方便快捷，提高生产效率。

本测试工具适用于运行完整固件的 PCBA 或整机测试，包含功能测试和老化测试。功能测试主要包含 WiFi、BT、LAN、SD、USB、HDMI、左右声道、按键、LED、CVBS 等。老化测试包含 CPU、VPU、GPU、Memory 的测试。

SDK 默认编译已带有该测试工具，具体操作说明请参考 RKDocs\common\RKTools manuals\Rockchip RK3328 Box 厂测工具操作说明 V2.0.pdf。

配置文件参考请见 RKTools\windows\Rockchip Box 厂测工具 V2.0-M-20170327.zip。



图 8-11 功能测试界面

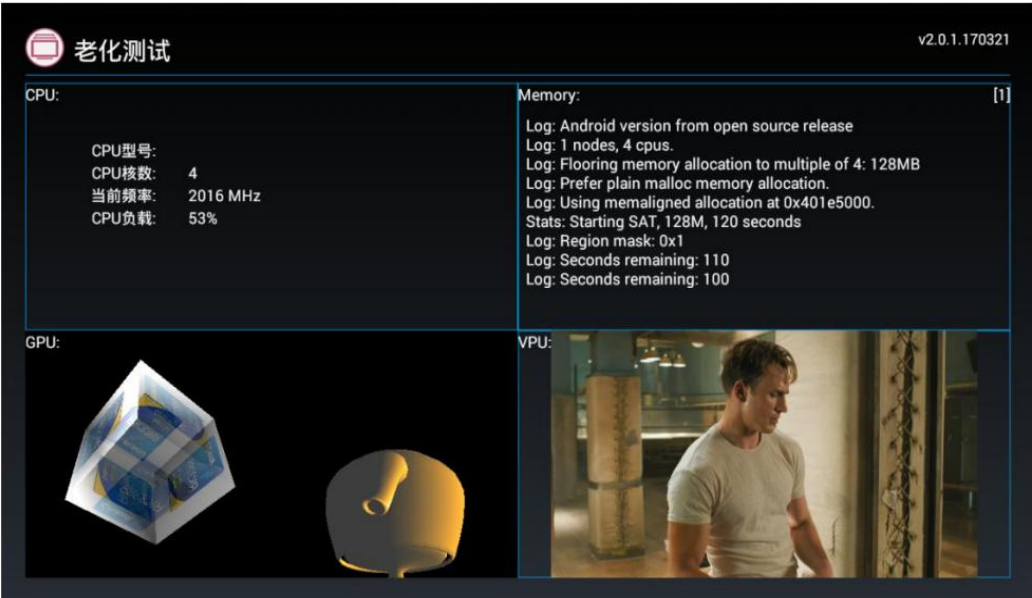


图 8-12 老化测试界面

注：由于默认该工具插外设启动，如果客户是烧机后手动安装的方式，需要重启后再测试，不然由于系统本身限制，apk 服务不会马上启动，引起无法识别测试文件问题。