RK3399Pro_Android8.1_Software_Development_Guide

(Technical Department, R & D Dept. II)

Status: [ ] Modifying [√] Released	Version:	V1.04
	Author:	Zhou Weixin
	Date:	2018-12-12
	Auditor:	Wu Liangqing
	Date:	2018-12-12

Fuzhou Rockchips Electronics Co., Ltd

Revision History

Version no.	Author	Revision Date	Revision description	Remark
V1.00	Zhou Weixin	2018.11.30	Initial version release	
V1.01	Zhou Weixin	2018.12.12	Add dts compiling for evb v11 board	
V1.02	Wu Liangqing	2019.04.17	Modify uboot part	
V1.03	Zhou Weixin	2019.05.20	Add sdcard configuration, update npu doc path, update rk write tool info	
V1.04	Zhou Weixin	2020.03.19	Add evb v13/v14 software compile	3.2.4

Content

Preface

Overview

This document mainly describes Rockchip RK3399Pro Android8.1 software development guide aiming to help software engineers familiar with RK3399Pro development and debugging quickly.

Product version

Chipset name	kernel version	Android version
RK3399Pro	Linux4.4	Android8.1.0

Object

This document (guide) is mainly suitable for below engineers:

Field application engineers

Software development engineers

1 Support list

1.1 DDR support list

RK3399Pro DDR current AVL supports dual channel DDR3, DDR3L, LPDDR3, L PDDR4.

Table 1-1 RK3399Pro DRAM Support Type

Chip	DRAM Support Type
RK3399Pro	DDR3/DDR3L/LPDDR3/LPDDR4

RK3399Pro DDR component support level refers to 《RK DDR Support List Ver2.34》 in the directory of RKDocs\common\Platform support lists. Only recommend to use the components marked with the symbol √ and T/A as shown in below table.

Table 1-2 RK3399Pro DDR Support Symbol

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

1.2 eMMC support list

RK3399Pro supports eMMC 5.1, SDIO3.0, and can run HS200, HS400 mode. For more details, refer to 《RKeMMCSupportList Ver1.41》 in the directory of RKDocs\common\Platform support lists. Only recommend to use the components marked with the symbol √ and T/A as shown in below table.

Table 1-3 RK3399Pro EMMC Support Symbol

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

1.2.1 High performance eMMC component selection

It is necessary to select high performance EMMC component to improve system performance. Before selecting EMMC component, please refer to our AVL support list, study the corresponding datasheet from vendors, and especially pay attention to the

performance chapter.

Refer to the vendor and read/write rate to do the sorting. Recommend to choose the component with the sequential reading rate >200Mb/s and sequential writing rate >40Mb/s.

Contact with our FAE if you have any questions about the component selection.

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

Picture 1-1 EMMC Performance example

1.3 Wi-Fi/BT support list

RK3399Pro kernel is Linux4.4. RK3399Pro and RK3399 share the same Wi-Fi/BT support list 《 Rockchip_WiFi_Situation_20180403.pdf 》 in the directory of RKDocs\common\Platform support lists. Below table shows the Wi-Fi/BT chipset list currently already verified in RK3399. Recommend to choose the component in the table. If want to debug other Wi-Fi/BT chipset, first need to communicate with Wi-Fi/BT vendor whether they can provide the driver program which can work on Linux4.4 stably and technical support during debugging.

Besides, we may keep upgrading the support list in future. You can contact with our FAE if there is any question or suggestion.

RK3399 Wi-Fi Situation													
WiFi Chip	IFACE	IEEE 802.11 Standard	2.4GHz Band	5.0GHz Band	BT	GPS	NFC	11AC	SDIO3.0	MIMO	BT4.0	BT4.2	Android7.1
AP6330	SDIO	IEEE 802.11A/B/G/N	✓	✓	✓	×	×	×	×	×	✓	×	✓
AP6255	SDIO	IEEE 802.11A/B/G/N/AC	✓	✓	✓	×	×	✓	✓	×	✓	✓	✓
AP6354	SDIO	IEEE 802.11A/B/G/N/AC	✓	✓	✓	×	×	✓	✓	✓	✓	×	✓
1. √：支持 ×：不支持 注：空的表示没调过													
2. 该列表仅适用kernel4.4													

Picture 1-2 RK3399 currently verified Wi-Fi/BT support list

1.4 SDK software package applicable hardware list

This SDK is compatible with RK3399Pro chipset software package based on Google Android8.1 64bit system.

If using Rockchip evb board, refer to 《3399Pro_Evb board instruction》 for details. You can use rk3399pro-evb-v10.dts to config kernel directly.

1.5 Multimedia encoder/decoder support list

RK3399Pro has powerful multimedia which supports 4K VP9 and 4K 10bits H265/H264 video decoder up to 60fps, 1080P multi format video decoder (MWV, MPEG-1/2/4, VP8), 1080P video encoder, H.264, VP8 format, video post processor: de-interleaving, de-noising, edge/detail/color optimization.

For detailed encoder/decoder support list, refer to《RK3399 Multimedia Codec Benchmark v1.0》 in the directory of RKDocs\rk3399Pro.

1.6 NPU development document

NPU development documents refer to the following directory:

/rknn-toolkit

/RKNPUTools

/RKDocs/rk3399pro/RK3399Pro_npu 上电及启动说明_V1.0_20190510.pdf

2 Document/tool index

2.1 Document index

RK3399Pro SDK release documents aim at helping developers familiar with development and debugging quickly. The documents may not cover all the knowledge and issues and the document list is also being updated. Please contact our FAE if you have any question or requirement about the documents.

RK3399Pro SDK includes three kinds of documents in RKDocs directory, android(android related development documents), rk3399Pro(RK3399Pro related release document), and common(common development document). Common directory consists of kernel driver development document, uboot development document, module development document, Platform support lists(support list), RKTools manuals(tool usage document) etc.

```
├─── 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
|    ├─── bt
|    |    └─── ROCKCHIP_ANDROID_8.1_BT 配置说明_V1.0_20180103.pdf
|    ├─── project.config
|    ├─── RK_PCBA_Camera_移植说明_v1.0.pdf
|    ├─── Rockchip Android 8.1 BOX 显示框架配置说明文档 V1.0-20180210.pdf
|    ├─── Rockchip Box 媒体中心使用说明-v1.0.1-20170216.pdf
|    ├─── ROCKCHIP_PCBA 测试工具开发指南_V1.2_20180509.pdf
|    ├─── Rockchip Recovery 用户操作指南 V1.03.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
├── common
|   ├── camera
|   |   ├── Camera 目录文档说明.txt
|   |   ├── CIF_ISP10_Driver_User_Manual_V1.0_20171124.pdf
|   |   ├── CIF_ISP11_Driver_User_Manual_V1.0.pdf
|   |   ├── readme_En.txt
|   |   ├── RK312x_Camera_User_Manual_v1.4(适用 3288&3368).pdf
|   |   ├── RK_ISP10_Camera_User_Manual_v2.2.pdf
|   |   ├── RKISPV1_Camera_Module_AVL_v1.7.pdf
|   |   ├── RKISPV1_Camera_常见问题解决方法 V1.0.pdf
|   |   ├── RKISPV1_Camera_驱动调试方法 V1.0.pdf
|   |   ├── Rockchip_Camera_AVL_v2.0_Package_20180515.7z
|   |   └── Rockchip SOFIA 3G-R_PMB8018(x3_C3230RK)_Camera_Module_
AVL_v1.6_20160226.pdf
|   ├── 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_V1.5_20180830.pdf
|   |   ├── Rockchip 基于 DRM 框架的 HDMI 开发指南 v1.1-20180322.pdf
|   |   └── 基于 DRM 的 Android 显示使用指南_V1.0_20180129.pdf
|   ├── driver
```

```
|   |   ├── RK817_RK809_Codec 开发指南_V1.0_20180228.pdf
|   |   ├── RK 语音通话 3A 算法集成说明及参数调试说明文档_V3.0.pdf
|   |   ├── 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 DEVFreq 开发指南 V1.0-20160701.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 pwm 背光 开发指南-20170220.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_Sensors_开发指南_V1.0_20180605.pdf
|   |   ├── Rockchip Thermal 开发指南 V1.0.1-20170428.pdf
|   |   ├── Rockchip Vendor Storage Application Note.pdf
|   |   ├── Rockchip 以太网 开发指南 V2.3.1-20160708.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
|   ├── hdmi-in
|   |   └── HDMI_IN_开发指南_V1.0_20180726.pdf
```

```
|   ├── mobile-net
|   |   ├── 3G 数据卡 USB 切换文件制作说明_v1.2.pdf
|   |   └── ROCKCHIP_3G_DONGLE_配置说明_V1.0.pdf
|   ├── Platform support lists
|   |   ├── RK3128 BOX  Hardware Design Guide V10-201410.pdf
|   |   ├── RKeMMCSupportList Ver1.41_20181030.pdf
|   |   ├── RKNandFlashSupportList Ver2.72_2016_08_30.pdf
|   |   ├── RK DDR Support List Ver2.34.pdf
|   |   ├── Rockchip_Camera_AVL_v2.0_Package.7z
|   |   ├── Rockchip Kodi 支持程度列表_V2.0_20170715.pdf
|   |   └── Rockchip_WiFi_Situation_20180611.pdf
|   ├── RKTools manuals
|   |   ├── Android 开发工具手册.pdf
|   |   ├── REPO 镜像服务器搭建和管理_V2.2_20131231.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 V2.0.rar
|   |   ├── Rockchip Box 厂测工具操作说明 V2.0.pdf
|   |   ├── Rockchip Keybox Burning Guide V1.2-20180315.pdf
|   |   ├── Rockchip Parameter File Format Ver1.3.pdf
|   |   ├── Rockchip 量产烧录 指南 V1.1-20170214.pdf
|   |   ├── WNpctool 写号工具简要使用说明_V1.1.2.pdf
|   |   ├── 压力测试 Stresstest 文档 forVR_ver3.0.pdf
|   |   ├── 瑞芯微 KeyWrite 使用指南_V1.3_20180508.pdf
|   |   └── 量产工具升级及相关问题处理.pdf
|   ├── security
|   |   ├── Rockchip-Secure-Boot-Application-Note-V1.9.pdf
|   |   └── Rockchip_TEE 安全 SDK 开发手册_V1.1_20170516.pdf
|   ├── u-boot
```

```
|   |   ├── Rockchip-Developer-Guide-Trust.pdf
|   |   ├── Rockchip-Developer-Guide-UBoot-nextdev.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
└── rk3399pro
    ├── RK3399 Multimedia Codec Benchmark v1.0.pdf
    ├── RK3399Pro_EVB 板简介_20181121.pdf
    ├── RK3399_SDK 多媒体性能指标说明文档_V1.0_20180109.pdf
    └── RK3399 USB DTS Configuration Instruction.pdf 工具索引
```

RK3399Pro SDK released tool is used in development debugging stage and MP stage. The tool may upgrade along with new SDK. Please contact with our FAE if there is any question or requirement about the tool.

RK3399Pro SDK contains linux(tool used in Linux operation system environment) and windows(tool used in Windows operation system environment) in RKTools directory.

```
├── linux
|   ├── Linux_AttestationKeyboxPack_Tool.rar
|   ├── Linux_Pack_Firmware
|   |   ├── Linux_rockdev.zip
|   |   └── rockdev
|   |       ├── afptool
|   |       ├── Image
|   |       |   ├── boot.img
|   |       |   ├── kernel.img
|   |       |   ├── MiniLoaderAll.bin
|   |       |   ├── misc.img
```

```
|   |       |   ├── parameter.txt
|   |       |   ├── pcba_small_misc.img
|   |       |   ├── pcba_whole_misc.img
|   |       |   ├── recovery.img
|   |       |   ├── resource.img
|   |       |   ├── system.img
|   |       |   ├── trust.img
|   |       |   └── uboot.img
|   |       ├── mkupdate.sh
|   |       ├── package-file
|   |       ├── readme.txt
|   |       ├── rkImageMaker
|   |       ├── unpack.sh
|   |       └── update.img
|   ├── Linux_SecureBoot
|   |   └── SecureBootConsole_v1.90.rar
|   ├── Linux_TA_Sign_Tool.rar
|   └── Linux_Upgrade_Tool
|       └── Linux_Upgrade_Tool_v1.39.zip
└── windows
    ├── AndroidTool
    |   ├── AndroidTool_Release_v2.54
    |   |   ├── AndroidTool.exe
    |   |   ├── Android 开发工具手册_v1.2.pdf
    |   |   ├── bin
    |   |   |   ├── adb.exe
    |   |   |   ├── AdbWinApi.dll
    |   |   |   ├── AdbWinUsbApi.dll
    |   |   |   ├── AFPTool.exe
    |   |   |   └── RKImageMaker.exe
```

```
|   |   ├── config.cfg
|   |   ├── config.ini
|   |   ├── Language
|   |   |   ├── Chinese.ini
|   |   |   └── English.ini
|   |   ├── Log
|   |   |   ├── Log2018-05-10.txt
|   |   |   └── ScanLog2018-05-10.txt
|   |   └── Readme.txt
|   ├── AndroidTool_Release_v2.54.zip
|   └── rockdev
|       ├── AFPTool.exe
|       ├── backupimage
|       |   ├── backup.img
|       |   └── package-file
|       ├── baseparamer.img
|       ├── mkupdate.bat
|       ├── package-file
|       ├── recover-script
|       ├── RKImageMaker.exe
|       └── update-script
├── Demo 镜像烧写工具包.zip
├── DriverAssitant_v4.5.zip
├── efuse_v1.37.rar
├── FactoryTool_v1.63.zip
├── FWFactoryTool-5.4.rar
├── KeyBoxWrite_v1.53.rar
├── OemTool_v1.3.rar
├── parameter_adjustment_tool.xlsx
├── rk312x-pcba-tools.rar
```

```
├─── RKImageMaker_v1.62.zip
├─── Rockchip  Box 厂测工具 V2.0-M-20170327.zip
├─── Rockchip 平台 DDR 测试工具_V1.35 发布通知.7z
├─── SDDiskTool_v1.56.zip
├─── SecureBootTool_v1.85_foruser.zip
├─── SpiImageTools_v1.41.zip
├─── UpgradeDllTool_v1.35.zip
├─── Windows_TA_Sign_Tool.rar
├─── WNpctool_Setup_V1.2.0.0522.rar
└─── 电池曲线检测工具
    ├─── ADC 电池测试工具_V2.3.pdf
    └─── BatteryArray_V2.4.apk
```

3 SDK compiling/flashing

3.1 How to get SDK

SDK is released through Rockchip code server. Customers apply SDK from Rockchip FAE contact, and will be able to sync code after obtaining the server certificate authorization with SSH public key.

3.1.1 SDK download link

RK3399Pro_ANDROID8.1_SDK download address is as below:

repo init --repo-url=ssh://git@www.rockchip.com.cn:2222/repo-release/tools/repo.git -u ssh://git@www.rockchip.com.cn:2222/Android_oreo_stable/platform/rk3399pro/manifests.git -m Rk3399pro_Android_Oreo_release.xml

3.1.2 repo

repo is a script invoking git developed by Google using Python script, and mainly used to download, manage Android project software lib. The download address is as below:

```
git clone ssh://git@www.rockchip.com.cn/repo/rk/tools/repo
```

3.1.3 SDK code compressed package

Rockchip FAE contact usually will provide the initial compressed package of the corresponding version SDK in order to help customers acquire SDK source code quickly. Developer can acquire the SDK code initial compressed package in this way and unzip it to get the source code. It is the same as the source code downloaded through repo. Take Rk3399Pro_Android8.1_SDK_Beta_V0.1_20181130.tar.gz as an example, you can sync the source code through below command after copy the initial package:

```
mkdir rk3399Pro
tar zxvf Rk3399Pro_Android8.1_SDK_Beta_V0.1_20181130.tar.gz -C rk3399Pro
cd rk3399Pro
.repo/repo/repo sync -l
.repo/repo/repo sync
```

Developers can execute the command ".repo/repo/repo sync" to sync the new code

according to the update notice released by FAE contact periodically in future.

3.2 SDK compiling

3.2.1 JDK installation

Android8.1 system compiling is dependent on JAVA 8. Need to install OpenJDK before compiling.

Install command is as below:

```
sudo apt-get install openjdk-8-jdk
```

Configure JAVA environment variable, for example, if the install path is /usr/lib/jvm/java-8-openjdk-amd64, it is able to execute below command to configure environment variable at the termination.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

SDK contains Open JDK8 configuration script named javaenv.sh in engine root directory.

Directly execute below command to configure JDK:

```
source javaenv.sh
```

3.2.2 Compilation mode

SDK default compiling mode is userdebug.

While using adb, first need to execute adb root, adb disable-verity to close the verity feature of the system partition, then execute adb root, adb remount after reboot, and then execute push operation to debug.

3.2.3 RK3399Pro EVB compilation

uboot compiling:

```
cd u-boot
make rk3399pro_defconfig
./mkv8.sh
Use ./make.sh rk3399pro if update uboot branch of next_dev branch
```

kernel compiling:

```
If for evb_v10 board(green), kernel compiling method is as below:
```

```
cd kernel
make ARCH=arm64 rockchip_defconfig -j8
make ARCH=arm64 rk3399pro-evb-v10.img -j12

If for evb_v11 board(black), kernel compiling method is as below:
cd kernel
make ARCH=arm64 rockchip_defconfig -j8
make ARCH=arm64 rk3399pro-evb-v11.img -j12
```

Android compiling:

```
source build/envsetup.sh
lunch rk3399pro-userdebug
make -j12
./mkimage.sh
```

3.2.4 Image build steps

The complete images package will be generated in rockdev/Image-xxx/(xxx is the detailed product name lunched) directory after executing ./mkimage.sh.

```
rockdev/Image-xxx/
├── boot.img
├── kernel.img
├── MiniLoaderAll.bin
├── misc.img
├── oem.img
├── parameter.txt
├── pcba_small_misc.img
├── pcba_whole_misc.img
├── recovery.img
├── resource.img
├── system.img
├── trust.img
```

RK official development board	Dts	Lunch
RK_EVB_RK3399PRO_XXX_V10	rk3399pro-evb-v10.dts	lunch rk3399pro-userdebug
RK_EVB_RK3399PRO_XXX_V11\V12	rk3399pro-evb-v11.dts	lunch rk3399pro-userdebug
RK_EVB_RK3399PRO_XXX_V13	rk3399pro-evb-v13-multi-cam.dts	lunch rk3399pro_pcie-userdebug
RK_EVB_RK3399PRO_XXX_V14	Pcie： rk3399pro-evb-v13-multi-cam.dts Usb3： rk3399pro-evb-v11.dts	Pcie： Lunch rk3399pro_pcie_v14-userdebug Usb3： Lunch rk3399pro_usb3_v14-userdebug

Note:

V14 evb support npu transfer by usb2+pcie or usb2+usb3 which is switch by hardware

3.2.5 jack-server configuration

Android8.1 system uses jack-server as java code compiler and may meet below errors during compiling:

```
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'
```

In this case it is mainly limited by jack-server compiler itself, one network port number cannot be used by multiple users at the same time.

That means multiple users need to configure different network port numbers separately during co-development in the server while they compile Android8.1 at the same time.

The two configuration files (yhx corresponds to the user name) of jack-server determine its port number:

```
/home/yhx/.jack-server/config.properties
/home/yhx/.jack-settings
```

The two configuration files need to configure two port numbers. One is server port number and the other is client port number. The port numbers in the two configuration files should match.

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

Configuration steps are as below:

1) Confirm the two configuration files existing and set the authority as 0600:

```
chmod 0600 /home/yhx/.jack-server/config.properties
chmod 0600 /home/yhx/.jack-settings
```

2) If the two configuration files not existing, please refer to below to create the two configuration files.

config.properties file example is as below (port number needs to be changed according to the actual):

```
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\=4294967296
jack.server.admin.port=8075
jack.server.config.version=2
jack.server.time-out=7200
```

.jack-settings file example is as below (port number needs to be changed according to the actual):

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

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

3) Change port number, please change service port and admin port as other port numbers and the port numbers in the two configuration files need to match. Example is as below:

```
jack.server.service.port=8023
jack.server.admin.port=8024

SERVER_PORT_SERVICE=8023
SERVER_PORT_ADMIN=8024
```

4) Re-compile Android, if error still occurs, try to modify other port number until compile successfully.

5) If still cannot pass the compilation over 5-time modification, execute the command jack-admin dump-report, unzip the generated compressed package, and analyze the log. If there is below log, re-install 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'
```

3.2.6 Full auto compiling script

As described above, the compilation mainly contains three parts compiling u-boot, kernel and android. In order to improve the compiling efficiency and lower down the possible mistake operation of manual compiling, this SDK integrates the full auto compiling script which is convenient for image compiling and backup.

1) The original file of the full auto compiling script is put in:

```
device/rockchip/RK3399Pro/build.sh
```

2) When repo sync, copy it to project root directory through manifest:

<project path="device/rockchip/rk3399Pro" name="rk/device/rockchip/rk3399Pro" remote="rk" revision="rk33/mid/8.1/develop">

<copyfile src="buildspec.mk" dest="buildspec.mk"/>

<copyfile src="build.sh" dest="build.sh"/>

</project>

3) Modify the specific variable in build.sh script to build out the correspondin g product images.

```
KERNEL_DTS=rk3399pro-evb-v10
```

Modify the variable according to the actual project situation:

KERNEL_DTS variable specifies the product board level configuration for kernel compiling.

Android compiling needs to specify the corresponding lunch option, please execute lunch operation before executing build.sh to make sure to use the correct lunch option. For exemple:

```
lunch rk3399Pro-user
```

4) Execute auto compiling script:

```
source build.sh
```

The script will automatically configure JDK environment variable, compile u-boot, compile kernel, compile Android, then generate images and version information, and package them to be update.img.

5) The script generated contents:

The script will copy the compiled images to:

the directory of IMAGE/RK3399Pro *****_RELEASE_TEST/IMAGES which path is subject to the actual generation. Each compiling will create new directory and save, automatically backup images version during debugging, and keep all the information of images version. Recommend to use this compiling script to generate images for every big version compilation. It includes much version information convenient to locate code

status for debugging issues.

update.img in the directory can be directly used to download and update Android development tool and factory flashing tool.

3.3 Image flashing

Flashing instruction refers to 《Android 开发工具手册.pdf》 in the directory of RKDocs\common\RKTools manuals. SDK provides flashing tools as shown in below picture. After compiling to generate corresponding images, enter flashing mode, it is able to flash images. For the devices with existing images, you can select to re-flash images, or format the device, erase idb, and then flash the images.

Picture 3-1 Android development tool flashing interface

Note:

1) Need to install the latest USB driver before flashing. The driver refers to:

```
RKTools/windows/
├── DriverAssitant_v4.5.zip
```

2) Android8.1 has two additional images vendor.img and oem.img must be flashed,

otherwise the system will fail to boot up.

3.4 MP flashing

Considering the production efficiency and factory work station arrangement during MP, the flashing instruction refers to 《Rockchip 量产烧录指南 V1.1-20170214.pdf》 in the directory of RKDocs\ common\RKTools manuals.

Please contact with our FAE if you have any tool related issues during production.

4 U-Boot development

This chapter simply introduces U-Boot basic concept and compilation notices to help customers understand RK platforms U-Boot framework. For U-Boot development details, you can refer to 《Rockchip-Developer-Guide-UBoot-nextdev.pdf》 in the directory of RKDocs\common\u-boot.

4.1 Rockchip U-Boot brief introduction

Rockchip U-Boots based on the official version of the open source UBoot 2014.10 development:

Support platform:rk3288,rk3036,rk312x,rk3368,rk312x,rk3366,rk3399;

Support firmware startup of Android platform;

Support ROCKUSB and Google Fastboot flashc;

Support secure boot firmware signature encryption protection mechanism;

Supports LVDS, EDP, MIPI, HDMI, CVBS and other display devices;

Support SDCard, Emmc, Nand Flash, U disk and other storage devices;

Support startup logo display, charging animation display, low power management, power management;

Support I2C, SPI, PMIC, CHARGE, GUAGE, USB, GPIO, PWM, DMA, GMAC, EMMC, NAND interrupt drivers;

4.2 Platform configuration

The platform configuration file is in the configs folder under U-Boot root directory. Rockchip related files begin with RK and can be divided into MID and BOX configuration according to the product types.

```
rk3288_defconfig
```

```
rk3126_defconfig
rk3128_defconfig
rk3368_defconfig
rk3399Pro_defconfig

rk3288_box_defconfig
rk3128_box_defconfig
rk3036_box_defconfig
rk3368_box_defconfig
rk322x_box_defconfig
rk3399_box_defconfig
```

4.3 Images generation

Rockchip platform Loader mode is divided into level 1 and level 2. Generate the corresponding Loader image according to different platform configuration. Define level 2 Loader mode through macro CONFIG_SECOND_LEVEL_BOOTLOADER.

4.3.1 Level 1 Loader mode

If U-Boot is as level 1 Loader mode, the image only supports EMMC memory device. The generated mirror after compiling:

```
rk3399pro_loader_v1.15.115.bin
```

V1.15.115 is the released version number.

4.3.2 Level 2 Loader mode

If U-Boot is as level 2 Loader mode, the image supports all the memory devices. In this mode, need MiniLoader support, through macro CONFIG_MERGER_MINILOADER to configure to generate. At the same time, it will generate trust image through macro CONFIG_MERGER_TRUSTIMAGE configuration after introducing Arm Trusted Firmware.

Take the mirror generated by rk3399pro compilation as an example:

```
rk3399pro_loader_v1.15.115.bin
uboot.img
trust.img
```

V1.15.115 is the released version number. U-Boot loader version is defined by

rockchip. 1.15.115 is defined according to the memory version and should not be changed by customers.

uboot.img is the package when U-Boot as level 2 loader.

trust.img is the package when U-Boot as level 2 loader.

RK3036, RK3126, RK3128, RK322x, RK3368, RK3366, RK3399, RK3399Pro etc. use level 2 loader mode.

4.4 U-Boot compilation

RK3399Pro SDK compilation uses below configuration:

./make.sh rk3399pro

After compilation, it will generate trust.img, rk3399pro_loader_v1.15.115.bin, uboot.img the three files.

Currently the compiled rk3399pro_loader_v1.15.115.bin DDR frequency is fixed as 800MHz.

4.5 U-Boot charging related configuration

4.5.1 u-boot charging logo package

Charging images need to be packaged into resource. Img to be read and disp layed by the charging driver.Charging images are not packaged by default when c ompiling the kernel, so you need to package them separately into resource.img.

Package Command:

./pack_resource.sh <input resource.img>

By default, this command will pack the images in./tools/images/ directory into resource. Img as charging images.

Package info:

./pack_resource.sh /home/guest/3399/kernel/resource.img

Pack ./tools/images/ & /home/guest/3399/kernel/resource.img to resource.img ...

Unpacking old image(/home/guest/3399/kernel/resource.img):

rk-kernel.dtb logo.bmp logo_kernel.bmp

Pack to resource.img successed!

Packed resources:

rk-kernel.dtb battery_1.bmp battery_2.bmp battery_3.bmp battery_4.bmp battery_5.bmp

battery_fail.bmp logo.bmp logo_kernel.bmp battery_0.bmp

resource.img is packed ready

4.5.2 Enable charging logo display in dts

The default code already enables this driver by adding and enabling the charge-animation node in DTS.

```
charge-animation {
compatible = "rockchip,uboot-charge";
status = "okay";
rockchip,uboot-charge-on = <0>;  //enable U-Boot charging or not
rockchip,android-charge-on = <1>;  //enable Android charging or not
rockchip,uboot-exit-charge-level = <5>; //Minimum power level allowed to exit uboot charge
rockchip,uboot-exit-charge-voltage = <3650>;  //Minimum voltage allowed to exit uboot charge
rockchip,screen-on-voltage = <3400>; //Minimum voltage allowed to display on
rockchip,uboot-low-power-voltage = <3350>; //The minimum voltage that is forced into charging mode
rockchip,system-suspend = <1>; // into trust for deep sleep
rockchip,auto-off-screen-interval = <20>; //default is 15s if do not define
rockchip,auto-wakeup-interval = <10>; //disable by define 0 or do not define
rockchip,auto-wakeup-screen-invert = <1>; // display on or not
};
```

The purpose of auto sleep/wake:

Some fuel gauges require a timed update algorithm to calibrate the battery.

4.5.3 Low power sleep

After entering the charging process, the system can be turned off by short pressing the power. When the screen is off, the low power standby state is entered. Press the button again to wake up. In the non-low power state, press power to exit the charging process and start up.

4.5.4 Replace the charging picture

1. Replace the pictures in the ./tools/images/ directory. The pictures are in 8-bit or 24-bit bmp format. Use the command "ls |sort" to confirm that the image is sorted from low to high. When packaged with the pack_resource.sh script, all images will be packaged into the resource in this order.

2. Modify the picture and battery relationship information in ./drivers/power/charge_animation.c:

Name: the name of the picture;

Soc: the power corresponding to the picture;

Period: Picture refresh time (unit: ms);

**Note: **The last picture must be a failed picture, and "soc=-1" cannot be changed.

3. Run the pack_resource.sh package command to get the new resource.img.

5 Kernel development

This chapter simply introduces some kernel common configurations changes, mainly for dts configuration, to help customers to do some simple changes easier and more convenient. RK3399Pro kernel version is 4.4 and config files are unified as arch/arm64/configs/ rockchip_defconfig. RK3399Pro serial port baud rate is 1500000. Please make sure the setting correct for debugging.

5.1 DTS description

5.1.1 DTS description

RK3399Pro dts file is in kernel/arch/arm64/boot/dts/rockchip/. Rk3399pro.dtsi is the core configuration file which defines the platform related contents.

RK3399-android.dtsi is the product level configuration file which defines some peripheral devices. The product dts needs to include these two files, e.g. RK3399Pro evb dts file rk3399pro-evb-v10.dts. Configure CPU, GPU, DDR frequency and voltage table in product dts according to the detailed product requirement. Configure io, panel, wifi, bt, sensor, thermal control, backlight, battery, system power configuration etc.

5.1.2 Add a new product DTS

RK3399Pro product dts file must be put in kernel/arch/arm64/boot/dts/rockchip/.

1、Take rk3399pro-evb-v10.dts as reference, copy a dts file and name it as rk3399Pro-product.dts.

2、Modify arch/arm64/boot/dts/rockchip/Makefile file, add the corresponding dtb statement:

```
+rk3399Pro-product.dtb
```

3、Modify the compiling script or command.

4、Re-compile kernel.

5.2 Wi-Fi configuration

```
wireless-wlan {
        compatible = "wlan-platdata";
        rockchip,grf = <&grf>;
        wifi_chip_type = "ap6354";
        sdio_vref = <1800>;
        WIFI,host_wake_irq = <&gpio0 3 GPIO_ACTIVE_HIGH>; /* GPIO0_a3 */
        status = "okay";
    };/
```

The above is the content of Wi-Fi dts configuration, mainly including the power control, interrupt etc. function pins' configuration. The configuration items (generally customers only need to modify the parameters marked in red) function will be explained as below:

wifi_chip_type = " ap6354";

Use to check Wi-Fi chipset. Need to specify the actually used Wi-Fi model here:

sdio_vref = <1800>; //1800mv or 3300mv

This item configures IO reference voltage value of Wi-Fi module, set according to the Wi-Fi module reference voltage input voltage value provided by the actual hardware design. The reference voltage set improperly will cause Wi-Fi communication abnormal, and then lead to Wi-Fi fail to work or work unstably.

WIFI,host_wake_irq = <&gpio0 3 GPIO_ACTIVE_HIGH>;

This item is Wi-Fi interrupt pin configuration. If some Wi-Fi module doesn't have this pin, just comment it out without configuration. Broadcom Wi-Fi such as AP6xxx and RK90x etc. modules all need to configure this GPIO correctly.

For Boardcom wifi AP6xxx system uses this interrupt pin as Wi-Fi data interrupt pin and Wi-Fi cannot work normally if there is problem with the interrupt pin. For other Wi-Fi, such as RTL8723BS, when the device is in sleep mode, the interrupt is used to wake up the device if there is Wi-Fi data coming. So the problem of the interrupt pin will not cause that Wi-Fi cannot work normally.

5.3 BT configuration

```
wireless-bluetooth {
        compatible = "bluetooth-platdata";
        //wifi-bt-power-toggle;
        uart_rts_gpios = <&gpio2 19 GPIO_ACTIVE_LOW>; /* GPIO2_C3 */
        pinctrl-names = "default", "rts_gpio";
        pinctrl-0 = <&uart0_rts>;
        pinctrl-1 = <&uart0_gpios>;
        //BT,power_gpio   = <&gpio3 19 GPIO_ACTIVE_HIGH>; /* GPIOx_xx */
        BT,reset_gpio     = <&gpio0 9 GPIO_ACTIVE_HIGH>; /* GPIO0_B1 */
        BT,wake_gpio      = <&gpio2 26 GPIO_ACTIVE_HIGH>; /* GPIO2_D2 */
        BT,wake_host_irq = <&gpio0 4 GPIO_ACTIVE_HIGH>; /* GPIO0_A4 */
        status = "okay";
    };
```

Above is the BT configuration in dts. Simply introduce some common parts that may need to be modified as below:

BT,reset_gpio = <&gpio0 9 GPIO_ACTIVE_HIGH>;

This configuration item is about BT RESET pin configuration. Not all BT modules have this pin. Refer to the actual schematic.

BT,power_gpio = <&gpio3 19 GPIO_ACTIVE_HIGH>

This configuration item is about BT power control GPIO configuration, high level active, refer to the actual schematic.

BT,wake_gpio = <&gpio2 26 GPIO_ACTIVE_HIGH>;

This configuration item is about BT WAKE pin configuration, corresponding to BT_WAKE pin in the schematic, high level active.

BT,wake_host_irq = <&gpio0 4 GPIO_ACTIVE_HIGH>

This configuration item is about BT interrupt configuration, corresponding to BT_HOST_WAKE pin in the schematic, high level active.

BT uses uart0 interface to connect by default. Uart0 configuration is as below:

```
&uart0 {
    pinctrl-names = "default";
    pinctrl-0 = <&uart0_xfer &uart0_cts>;
    status = "okay";
};
```

5.4 GPIO

RK3399Pro provides 5 groups GPIO(GPIO0~GPIO4) total 122pcs. All GPIO can be used as interrupt. GPIO0/GPIO1 can be used as system wakeup pin. All GPIO can be pull up or down by software configuration. All GPIO default is input and the driver ability can be configured by software.

As for the gpio corresponding relationship between schematic and dts, such as GPIO4c0, the corresponding gpio in dts should be "gpio4 16". As GPIO4A has 8 pins, GPIO4B also has 8 pins, inferring in this way, we can know c0 port is 16, c1 port is 17, and so on.

GPIO usage refers to 《Rockchip Pin-Ctrl 开发指南 V1.0-20160725.pdf》 in the directory of RKDocs\common\driver\.

5.5 ARM, GPU, DDR frequency change

DVFS(Dynamic Voltage and Frequency Scaling) is a real-time voltage and frequency adjusting technology. In current kernel 4.4 modules CPU, GPU, DDR support DVFS.

CPUFreq is a set of framework model supporting dynamically adjusting CPU frequency and voltage defined by kernel developers. It can effectively lower down CPU power consumption and balance CPU performance at the same time.

CPUFreq selects a suitable frequency for CPU through different frequency conversion strategies. Current kernel version provides below strategies:

- interactive: dynamically adjust frequency and voltage according to CPU load.
- conservative: conservative strategy, adjust frequency and voltage step by step.
- ondemand: dynamically adjust frequency and voltage according to CPU load, slower than interactive.
- userspace: user to set voltage and frequency, system doesn't automatically adjust.
- powersave: power consumption first, always set the frequency to the lowest value.
- performance: performance first, always set the frequency to the max value.

The detailed module function and configuration refer to 《Rockchip CPU-Freq 开发指南 V1.0.1-20170213.pdf》and《Rockchip DEVFreq 开发指南 V1.0-20160701.pdf》in the directory of RKDocs/common/driver/.

A53/A72/GPU/DDR all have corresponding debugging interface which can be operated with ADB command. The corresponding interface contents are as below:

A53：/sys/devices/system/cpu/cpu0/cpufreq/

A72：/sys/devices/system/cpu/cpu4/cpufreq/

GPU：/sys/class/devfreq/ff9a0000.gpu/

DDR：/sys/class/devfreq/dmc/

These contents have below similar nodes:

- available_frequencies：display the supported frequency
- available_governors：display the supported frequency conversion strategy
- cur_freq：display current frequency
- Governor：display current frequency conversion strategy

- max_freq：display current supported max frequency
- min_freq：display current supported min frequency

Take GPU as example to do the fixed frequency operation. The process is as below:

- Check the supported frequencies

  cat /sys/class/devfreq/ff9a0000.gpu/available_frequencies

- Switch the frequency conversion strategy

  echo userspace > /sys/class/devfreq/ff9a0000.gpu/governor

- Fix the frequency

  echo 400000000 > /sys/class/devfreq/ff9a0000.gpu/userspace/set_freq

- Check current frequency after setting

  cat /sys/class/devfreq/ff9a0000.gpu/cur_freq

5.6 Thermal control configuration

RK3399Pro chipset ARM core and GPU core have separate thermal control sensors which can real-time monitor CPU and GPU temperature and then control CPU and GPU temperatures through algorithm to control CPU and GPU frequency. Each product's different hardware design and mold correspond to different heat dissipation situation. The following configurations in dts can be used to adjust thermal control parameters to fit the product:

Set the temperature to enable the thermal control:

```
&threshold {
    temperature = <85000>; /* millicelsius */
};
```

Set the upper limit of thermal control temperature:

```
&target {
    temperature = <100000>; /* millicelsius */
};
```

Set the software shutdown temperature:

```
&soc_crit {
    temperature = <105000>; /* millicelsius */
};
```

Configure the hardware shutdown temperature:

```
&tsadc {
    rockchip,hw-tshut-mode = <1>; /* tshut mode 0:CRU 1:GPIO */
    rockchip,hw-tshut-polarity = <1>; /* tshut polarity 0:LOW 1:HIGH */
    rockchip,hw-tshut-temp = <110000>;
    status = "okay";
};
```

The detailed thermal control instruction refers to 《Rockchip Thermal 开发指南 V1.0.1-20170428.pdf》 in the directory of RKDocs\common\driver.

5.7 LPDDR4 configuration

rk3399Pro lpddr4 dts configuration refers to the file: arch/arm64/boot/dts/rockchip/rk3399pro-evb-lp4-v11-avb.dts. Just need to copy the below three nodes in the file to the corresponding product dts:

```
&dfi {
    status = "okay";
};
&dmc {
    status = "okay";
    center-supply = <&vdd_center>;//according to the actual hardware circuit
s
    upthreshold = <40>;
    downdifferential = <20>;
    system-status-freq = <
        /*system status          freq(KHz)*/
        SYS_STATUS_NORMAL        856000
        SYS_STATUS_REBOOT        416000
        SYS_STATUS_SUSPEND       416000
        SYS_STATUS_VIDEO_1080P   416000
        SYS_STATUS_VIDEO_4K      856000
        SYS_STATUS_VIDEO_4K_10B 856000
```

```
        SYS_STATUS_PERFORMANCE   856000
        SYS_STATUS_BOOST         856000
        SYS_STATUS_DUALVIEW      856000
        SYS_STATUS_ISP           856000
    >;
    vop-pn-msch-readlatency = <
    /* plane_number   readlatency */
        0   0
        4   0x20
    >;
    vop-bw-dmc-freq = <
    /* min_bw(MB/s) max_bw(MB/s) freq(KHz) */
        763     1893     416000
        3013    99999    856000
    >;
    auto-min-freq = <416000>;
};

&dmc_opp_table {
    compatible = "operating-points-v2";

    opp-200000000 {
        opp-hz = /bits/ 64 <200000000>;
        opp-microvolt = <900000>;
        status = "disabled";
    };
    opp-300000000 {
        opp-hz = /bits/ 64 <300000000>;
        opp-microvolt = <900000>;
        status = "disabled";
```

```
        };
        opp-400000000 {
            opp-hz = /bits/ 64 <400000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
        opp-416000000 {
            opp-hz = /bits/ 64 <416000000>;
            opp-microvolt = <900000>;
        };
        opp-528000000 {
            opp-hz = /bits/ 64 <528000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
        opp-600000000 {
            opp-hz = /bits/ 64 <600000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
        opp-800000000 {
            opp-hz = /bits/ 64 <800000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
        opp-856000000 {
            opp-hz = /bits/ 64 <856000000>;
            opp-microvolt = <900000>;
        };
        opp-928000000 {
```

```
            opp-hz = /bits/ 64 <928000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
        opp-1056000000 {
            opp-hz = /bits/ 64 <1056000000>;
            opp-microvolt = <900000>;
            status = "disabled";
        };
    };
```

Here we need to notice that: 1) lpddr4 only supports 416MHz and 856MHz, other frequencies are disabled. If customers want to use the same dts to support lpddr4 and other ddr, other ddr will also only support 416MHz and 856MHz. Please pay attention to this. 2) Above configuration enables DDR frequency conversion function by default. Lpddr4 frequency conversion function has some limitation on the audio card number as described below:

If lpddr4 needs frequency conversion function, need to transfer audio buffer to sram. RK3399Pro sram space is limited, available space is 128k, currently pre-allocated space for single audio stream is 32k, so the system can support only 2 audio card at most(32k*2*2, each audio card includes playback and capture). More audio cards cannot be created successfully unless to decrease the single stream pre-allocated size. However, it also relatively decreases the buffer size max supported by bottom layer and will be limited if user layer uses audio card to set a larger buffer. Need to notice that, USB audio card is not subject to the limitation because it doesn't use dma. That means, you can use two audio cards (audio cards with hdmi, spdif, i2s etc. interfaces) and multiple USB audio cards. Therefore, the following description is divided into two cases:

5.7.1 Need lpddr4 frequency conversion

If need lpddr4 frequency conversion function, need to transfer audio buffer to sram,

and now the system only support 2 audio cards at most. Please follow below steps to confirgure:

1. Add sram node in dts

```
/* first 64k(0xff8c0000~0xff8d0000) for ddr and suspend */
iram: sram@ff8d0000 {
  compatible = "mmio-sram";
  reg = <0x0 0xff8d0000 0x0 0x20000>; /* 128k */
};
```

2. Invoke iram node in the corresponding product dts.

```
&dmac_bus {
  iram = <&iram>;
  rockchip,force-iram;
};
```

5.7.2 No need lpddr4 frequency conversion

If need 3 or more audio cards, need to disable lpddr4 frequency conversion function due to the 2 audio cards limitation. That is, to disable the dmc node in the corresponding product dts as shown below:

```
&dmc {
    status = "disabled";
    … …
};
```

Besides, must delete the two kernel configurations described in 5.8.1 chapter.

1. Delete the following configuration in dts:

```
/* first 64k(0xff8c0000~0xff8d0000) for ddr and suspend */
iram: sram@ff8d0000 {
  compatible = "mmio-sram";
  reg = <0x0 0xff8d0000 0x0 0x20000>; /* 128k */
};
```

2. Delete the following configuration in dts:

```
&dmac_bus {
  iram = <&iram>;
  rockchip,force-iram;
};
```

6 SDCard configuration

Sdcard/uartdebug is Multifunction pin,the default is uart debug mode,enable sdcard by default refer to the following:

```
 &fiq_debugger {
+         status = "disabled";
          pinctrl-0 = <&uart2a_xfer>;
 };
&sdmmc {
          sd-uhs-sdr12;
          sd-uhs-sdr25;
          sd-uhs-sdr50;
          sd-uhs-sdr104;
+         status = "okay";
  };
```

7 Android common configuration

7.1 Android product configuration

7.1.1 lunch option description

rk3399pro-userdebug: //rk3399Pro platform product userdebug(64bit)

rk3399pro-user: //rk3399Pro platform product user(64bit)

7.2 Common function configuration description

7.2.1 Common configuration macro description

Macro configuration	Function description
BUILD_WITH_GOOGLE_MARKET	If it is true, integrate GMS package, false not to integrate
BUILD_WITH_GOOGLE_MARKET_ALL	If it is true, integrate full GMS package, false to integrate mini GMS package
BUILD_WITH_GOOGLE_FRP	Enable FRP factory reset protection function
BUILD_WITH_FORCEENCRYPT	Enable default full disk encryption
PRODUCT_SYSTEM_VERITY	Enable Verified boot
BUILD_WITH_GMS_CER	GMS certificate configuration option
BUILD_WITH_WIDEVINE	Integrate Widevine level3 plug-in library
BOARD_NFC_SUPPORT	Enable NFC function
BOARD_SENSOR_ST	Select ST sensor framework
BOARD_SENSOR_MPU	Select MPU sensor framework
BOARD_SENSOR_MPU_VR	Select MPU_VR sensor framework
BOARD_GRAVITY_SENSOR_SUPPORT	Enable G-Sensor
BOARD_COMPASS_SENSOR_SUPPORT	Enable Compass
BOARD_GYROSCOPE_SENSOR_SUPPORT	Enable Gyroscope
BOARD_PROXIMITY_SENSOR_SUPPORT	Enable P-sensor
BOARD_LIGHT_SENSOR_SUPPORT	Enable the light sensor
BOARD_PRESSURE_SENSOR_SUPPORT	Enable the pressure sensor
BOARD_TEMPERATURE_SENSOR_SUPPORT	Enable the temperature sensor
BOARD_ENABLE_3G_DONGLE	Enable 3G Dongle function
TARGET_ROCKCHIP_PCBATEST	Enable PCBA test
BOOT_SHUTDOWN_ANIMATION_RINGING	Enable power on/off animation and

	tones
BOARD_SYSTEMIMAGE_PARTITION_SIZE	System partition maximum capacity

7.2.2 Pre-install APK

Android apk pre-install function means to install the third application prepared in advance into the Android system when configuring the product according to customer requirements. Pre-install can be divided into non-uninstall installation, permanent uninstall installation and automatic installation after factory reset. Please refer to below document in the project directory of RKDocs/android/ for the detailed configuration and usage: 《Android8.0_预安装应用功能说明文档_V1.0_20171109.pdf》.

7.2.3 Power on/off animation and tones

Android8.1 power on tones, power off tones, power on animation, and power off animation customizations refer to the document 《Android8.0_定制开关机动画（铃音）说明_V1.0_20170923.pdf》 in the project directory of RKDocs/android/.

7.3 Parameter description

rk3399Pro Android8.1 platform supports various product types and different product types may need different parameter. For the parameter and partition details, please refer to \RKDocs\common\RKTools manuals\ Rockchip Parameter File Format Ver1.3.pdf.

7.4 New partition configuration

Please refer to \RKDocs\android\《Android 增加一个分区配置指南 V1.00.pdf》.

7.5 OTA upgrade

OTA(over the air) upgrade is the standard software upgrade method provided by Android system. It provides complete upgrading (full package) and incremental upgrading mode (difference package). You can upgrade locally or over the network. For the detailed OTA upgrade and Recovery mode function and configuration, please refer to 《Rockchip Recovery 用户操作指南 V1.03》 in the directory of RKDocs\android.

7 System debug

This chapter mainly introduces the debugging tools and methods used in SDK development and will update and improve continually to help developers familiar with

the basic system debugging quickly and analyze the issues correctly.

7.1 ADB tool

7.1.1 Overview

ADB (Android Debug Bridge) is a tool in Android SDK which can be used to operate and manage Android simulator or the real Android device. The functions mainly include:

- Run the device shell (command line)
- Manage the port mapping of the simulator or the device
- Upload/download files between the computer and the device
- Install the local apk to simulator or Android device

ADB is a "client – server" program. Usually the client is PC and the server is the actual Android device or simulator. The ADB can be divided into two categories according to the way PC connects to the device:

- Network ADB: PC connects to STB device through cable/wireless network.
- USB ADB: PC connects to STB device through USB cable.

7.1.2 USB ADB usage

USB ADB usage has below limitations:

- Only support USB OTG port
- Not support multiple clients at the same time (such as cmd window, eclip se etc.)
- Support host connects to only one device but multiple devices

Connect steps are as below:

1、The device already running Android system, setting -> developer option -> connect to the computer, enable usb debugging switch.

2、PC connects to the device USB OTG port only through USB cable, and the n the computer connects with the device through below command:

```
adb shell
```

3、Execute the command "adb devices" to see if the connection is successful or not. If the device serial number shows up, the connection is succes sful.

7.1.3 Network ADB usage requirement

ADB early versions only support device debugging through USB, and the function of debugging Android devices through tcp/ip is added from adb v1.0.25.

If you need to use network ADB to debug the device, must meet below conditions:

1. The device must have network port, or connect the network through Wi-Fi.
2. The device and PC are already in the local network and the device has IP address.
3. Need to confirm the device and PC can ping each other.
4. PC already installs ADB.
5. Confirm Android device adbd progress (ADB backaground process) is already run. adbd progress will monitor port 5555 to do ADB connection debugging.

7.1.4 SDK network ADB port configuration

SDK doesn't configure network ADB port by default. Need to modify manually to open the configuration.

Modify device/rockchip/rkxxxx/device.mk file, and add below configuration behind PRODUCT_PROPERTY_OVERRIDES:

```
service.adb.tcp.port=5555
```

7.1.5 Network ADB usage

This chapter assumes the device IP is 192.168.1.5. This IP will be used for ADB connection and device debugging in the following context.

1、 Firstly the Android device should boot up, if possible, confirm adbd is started (use ps command to check).

2、 In PC cmd, input:

```
adb connect 192.168.1.5:5555
```

If successful, will prompt relative hints, if fail, execute kill-server command and then retry connection.

```
adb kill-server
```

3、 After connected, you can input ADB relative commands to debug in PC, such as adb shell, it will connect the device through TCP/IP which is the same as USB debugging.

4、 After debugging, input below command to disconnect the connection in PC:

```
adb disconnect 192.168.1.5:5555
```

7.1.6 Manually change the network ADB port number

If SDK doesn't add ADB port number configuration, or want to change ADB port number, you can change through below method:

1. Firstly also connect the device normally through USB, execute adb shell in windows cmd to enter.
2. Set ADB monitor port:

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

3. Look up adbd pid using ps command.
4. Reset adbd.

```
#kill -9<pid>，这个 pid 就是上一步找到那个 pid  This pid is just the one found in last step.
```

After killing adbd, adbd will automatically restart after Android init progress. After adbd restart, if service.adb.tcp.port is set, it will automatically change to monitor network request.

7.1.7 ADB commonly used command elaberation

（1） Check the device situation

Check the Android device or simulator connected to computer:

```
adb devices
```

The return result is the serial number or IP and port number, status of the Android device connected to PC.

（2） Install APK

Install the specific APK file to the device:

```
adb install <apk 文件路径 apk file path>
```

For example:

```
adb install “F:\WishTV\WishTV.apk”
```

Re-install application:

```
adb install –r <apk 文件路径 apk file path>
```

For example:

```
adb install –r “F:\WishTV\WishTV.apk”
```

（3） Uninstall APK

Complete uninstall:

```
adb uninstall <package>
```

For example:

```
adb uninstall com.wishtv
```

（4） Use rm to remove APK file:

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

For example:

```
adb shell
rm “system/app/WishTV.apk”
```

Note: remove WishTV.apk file in the directory of system/app.

（5） Enter shell of the device and simulator

Enthe the shell environment of the device or simulator:

```
adb shell
```

（6） Upload file to the device from computer

Use push command can upload any file or folder from computer to the device. Usually local path means the computer and remote path means the single board device connected with ADB.

```
adb push <local path><remote path>
```

For example:

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

Note: upload local WishTV.apk file to the system/app directory of the Android system.

（7） Download file from the device to computer

Use pull command can download file or folder from the device to local computer.

```
adb pull <remote path><local path>
```

For example:

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

Note: download the file or folder from the system/app directory of Android system to local F:\ directory.

（8） Check bug report

Run adb bugreport command can check all the error message report generate by system. The command will show all dumpsys, dumpstate and logcat information of the Android system.

（9） Check the device system information

The detailed commands to check the device system information in adb shell.

```
adb shell getprop
```

7.2 Logcat tool

Android logcat system provides the function to record and check the system debugging information. The logcats are all recorded from various softwares and some system buffer. The buffer can be checked and used through Logcat. Logcat is the most commonly used function for debugging program. The function shows the program running status mainly by printing logcat. Because the amount of logcat is very large, it needs to be filtered and so on.

7.2.1 Logcat command usage

Use logcat command to check the contents of the system logcat buffer:

The basic format:

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

For example:

```
adb shell
logcat
```

7.2.2 The commonly used logcat filter method

Several ways to control the logcat output:

- Control the logcat output priority.

For example:

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

Note: show the logcat information with priority of warning or higher.

- Control the logcat label and output priority.

For example:

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

Note: support all the logcat information except those with label of ActivityManager and priority of Info above, label of MyApp and priority of Debug above.

- Only output the logcat with the specific label

For example:

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

or

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

Note: only output the logcat with label of WishTV.

- Only output the logcat with the specific priority and label

For example:

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

Note: only output the logcat with priority of I and label of WishTV.

7.2.3 View last log

Add –L parameter can print out the logcat information before last system reset. If the stress test and power down abnormal occur, the command can be used to print out the logcat of last Android running status. The command is as below:

```
adb shell
logcat -L
```

7.3 Procrank tool

Procrank is a debugging tool with Android, running in the shell environment of the device, used to output the memory snapshot of the process, and effectively observe the memory usage status of the process.

Include below memory information:

- VSS：Virtual Set Size The memory size used by virtual (including the memory used by the shared lib)

- RSS：Resident Set Size The actually used physical memory size (including the memory used by the shared lib)
- PSS：Proportional Set Size The actually used physical memory size (allocate the memory used by the shared lib in proportion)
- USS：Unique Set Size The physical memory used exclusively by the process (not including the memory used by the shared lib)

Note:

- USS size represents the memory size only used by the process, and it will be recovered completely after the process is killed.
- VSS/RSS includes the memory used by the shared lib, so it is not helpful to check the memory status of the single process.
- PSS is the shared memory status used by the specific single process after the shared memory is allocated in proportion.

7.3.1 Use procrank

Make sure the terminal has the root authority before executing procrank

```
su
```

The command format:

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

The commonly used command instructions:

- -v： order by VSS
- -r： order by RSS
- -p： order by PSS
- -u： order by USS
- -R： convert to order by increasing[decreasing] method
- -w： only display the statistical count of working set
- -W： reset the statistical count of working set
- -h： help

For example:

- Output the memory snapshot:

```
procrank
```

– Output the memory snapshot by VSS decreasing order:

```
procrank -v
```

Procrank output is ranking by PSS by default.

7.3.2 Search the specific content information

Use below command format to view the memory status of the specific process:

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

cmdline means the target application name, PID means the target application pfocess.

Output the memory status used by systemUI process:

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

Or:

```
procrank | grep 3396
```

7.3.3 Trace the process memory status

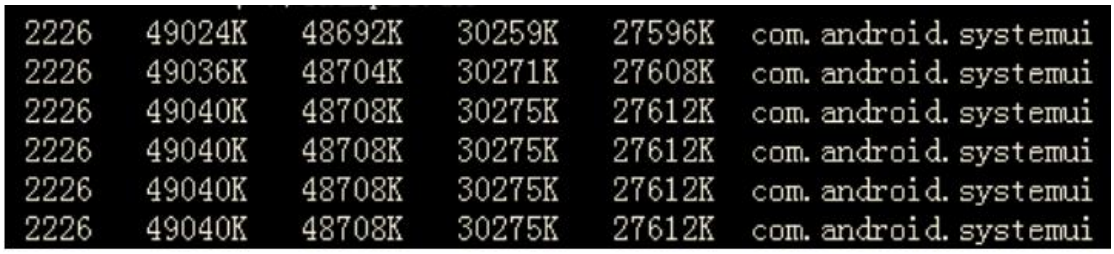
Analyze if there is memory leakage in the process by tracing the memory usage status. Use the script to continuously output the process memory snapshot, and compare with USS segment to see if there is memory leakage in this process.

For example: output the application memory usage of the process named com.android.systemui to see if there is leakage:

1、 Write the script test.sh

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

2、 After connecting to the device by ADB tool, run the script ./test.sh as shown in below picture:



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

Picture 7-1 Trace the process memory status

7.4 Dumpsys tool

Dumpsys tool is a debugging tool in Android system, running in the shell environment of the device, and provides the status information of the running service in the system. The running service means the service process in the Android binder mechanism.

The conditions for dumpsys to output the print:

1、 Only print the services already loaded to ServiceManager.

2、 If the dump function in the service code is not implemented, there will be no information output.

7.4.1 Use Dumpsys

- View Dumpsys help

Function: output dympsys help information.

```
dumpsys  -help
```

- View the service list of Dumpsys

Function: output all the printable service information of dumpsys, developer can pay attention to the service names needed for debugging.

```
dumpsys  -l
```

- Output the specific service information

Function: output the specific service dump information.

Format: dumpsys [servicename]

For example: execute below command can output the service information of SurfaceFlinger

```
dumpsys SurfaceFlinger
```

- Output the specific service and application process information

Function: output the specific service and application process information

Format: dumpsys [servicename] [application name]

For example: execute below command to output the memory information for the service named meminfo and process named com.android.systemui.

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

Note: the service name is case sensitive and must input the full service name.

7.5 Serial port debugging

7.5.1 Serial port configuration

The serial input and output is the most convenient during debugging. Need to note that RK3399 baud rate is set as 1500000. No need to choose RTS/CTS, otherwise the serial port cannot be input.

7.5.2 FIQ mode

FIQ (Fast interrupt request) in ARM is a kind of privilege modes and also one of the abnormal modes.

In RK platforms, input fiq with serial port can enter this mode. At this moment the usage help will pop out and you can do some debugging according to the situation. Usually it is helpful when crash or system die happens.

7.6 Audio codec issue debugging tool and document

Please refer to RKDocs\common\driver\ Rockchip Audio 开发指南 V1.1-20170215-linux4.4.pdf.

7.7 Enable Last log

Add below two nodes in dts file:

```
ramoops_mem: ramoops_mem {
    reg = <0x0 0x110000 0x0 0xf0000>;
    reg-names = "ramoops_mem";
};

ramoops {
    compatible = "ramoops";
    record-size = <0x0 0x20000>;
    console-size = <0x0 0x80000>;
    ftrace-size = <0x0 0x00000>;
    pmsg-size = <0x0 0x50000>;
    memory-region = <&ramoops_mem>;
};
```

- 130|root@rk3399:/sys/fs/pstore # ls

  dmesg-ramoops-0 Log saved after last kernel panic

  pmsg-ramoops-0 Log of last user space, android log

  ftrace-ramoops-0 Print function trace during some period.

  console-ramoops-0 The kernel log for the last boot of last_log, but only save the log with higher priority than default log level

- Usage method

  cat dmesg-ramoops-0

  cat console-ramoops-0

  logcat -L （pmsg-ramoops-0） pull out by logcat and parse

  cat ftrace-ramoops-0

8 Commonly used tool instruction

This chapter simply describes some developing and MP tools usage along with SDK to help the developers familiar with RK platform tool usage. The detailed tool usage refers to the tool related documents in the directories of RKTools and RKDocs\ common\ RKTools manuals.

8.1 StressTest

Use the Stresstest tool to do the stress test for the vasious functions on the target devices to make sure the whole system running stably.

The test items of Stresstest tool mainly include:

Module related

- Camera stress test: include Camera on/off, Camera taking photo and Cam era switch.
- Bluetooth stress test: include Bluetooth on/off.
- WiFi stress test: include WiFi on/off, (plan to add ping test and iperf tes t).

Non module related

- Fly mode on/off test

- Sleep and resume stress test
- Video playing stress test
- Restart stress test
- Recovery stress test
- ARM frequency conversion test
- GPU frequency conversion test
- DDR frequency conversion test

8.2 PCBA test tool

PCBA test tool is used to help quickly identify good and bad product features during production to improve the production efficiency. Current test items include panel (LCD), wireless (WiFi), Bluetooth, DDR/eMMC memory, SD card, USB HOST, key, speaker earphone (Codec).

These test items include automatic test item and manual test item. Wireless network, DDR/eMMC, Ethernet are automatic test items, while key, SD card, USB Host, Codec are manual test items.

For detailed PCBA function configuration and usage, pleaes refer to:

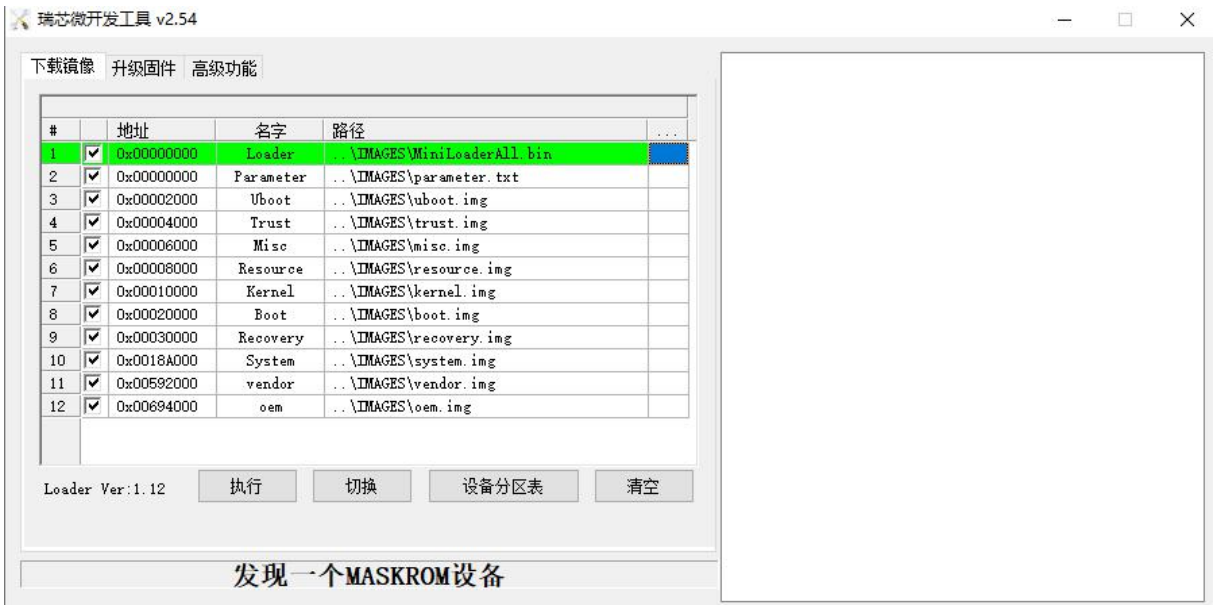
\RKDocs\common\RKTools manuals\Rockchip PCBA 模块开发指南--20170210.pdf

8.3 DDR test tool

Use DDR test tool to do the stability test on the target devices to make sure DDR function normal and stable. Currently DDR test tool of this platform is not relased yet, and it will be updated along with SDK later.

8.4 Android development tool

8.4.1 Download the mirror image



Picture 8-1 Use Android development tool to download the mirror image

1） Connect the developing board to enter the download mode.

Download mode: Firstly press reset key of the developing board, and then long press recovery key around 3-4s to enter.

2） Open the tool, and click “download mirror image” menu. Single click every line end as marked with red arrow, it will pop out file selection box and then choose the img file path of the corresponding partition.

3） Set all the img file paths successively.

4） After configuration, click “execute”. The right information box will display the relative information.

5） Button description

“低格” button: Used to erase the device

“清空” button: Used to clean up the information box

8.4.2 Upgrade image

Picture 8-2 Use Android development tool to upgrade image

1) Prepare the target image (refer to update.img package).
2) Confirm the device is already in the download mode.

   The way to enter the download mode: Firstly press reset key of the developing board, and then long press recovery key around 3-4s to enter.
3) Click “image” button, and choose the target image file update.img.
4) Click “upgrage” button to download. The right information box will display the relative information.

8.4.3 Senior functions

Picture 8-3 Android development tool senior functions

Senior functions description:

1) Boot can only select the packed update.img file or loader file.

2) Image must use the packed update.img.

3) The unpack function can unpack update.img into partial mirror files.

8.5 update.img pack

This platform supports to pack the scattered mirror files into one complete update.img to benefit production flashing and upgrading. The detailed packing steps are as below:

1) Open the rockdev directory under AndroidTool directory. Compile package-file.

2) Configure according to package-file, there are some img mirror put under the directory of Image in package-file. If the directory doesn’t exist, you need to manually create the Image directory and put the needed mirror in the directory. Note that the mirror name must be correct during configuration and bootloader option should change the loader name according to the generated name yourself.

3) Compile mkupdate.bat.

4) Change loader name to be the one actually saved.

5) Click mkupdate.bat to run, and it will generate one update.img in the directory finally.

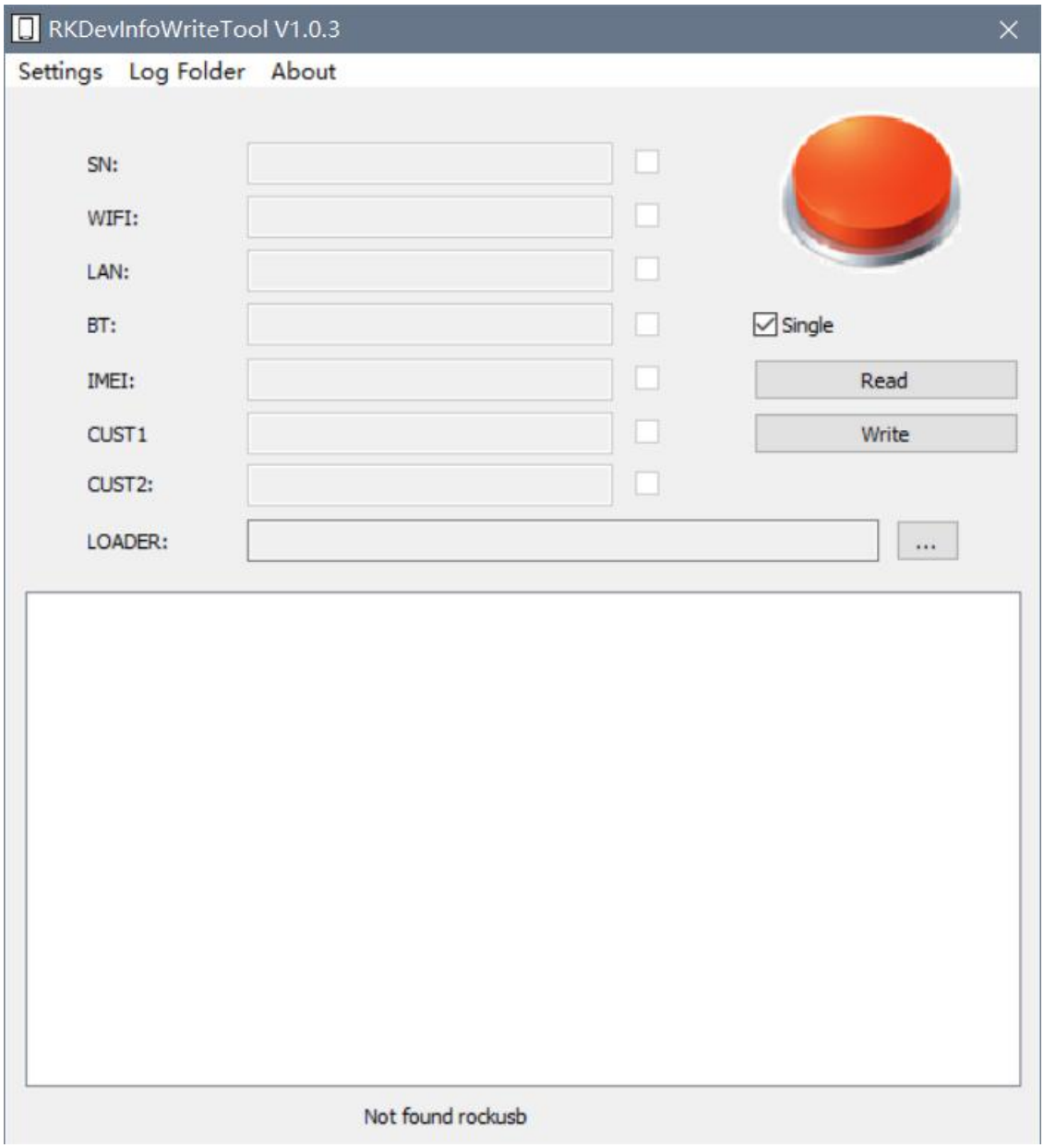
8.6 Image signature tool

Refer to 《Rockchip Secure Boot Application Note》 in RKTools\windows\Secure BootTool_v1.83_foruser.rar.

8.7 SN/Mac/Vendor information flashing-WNpctool tool

This platform uses WNpctool tool to flash SN/Mac/vendor information. Below describe the basic usage.

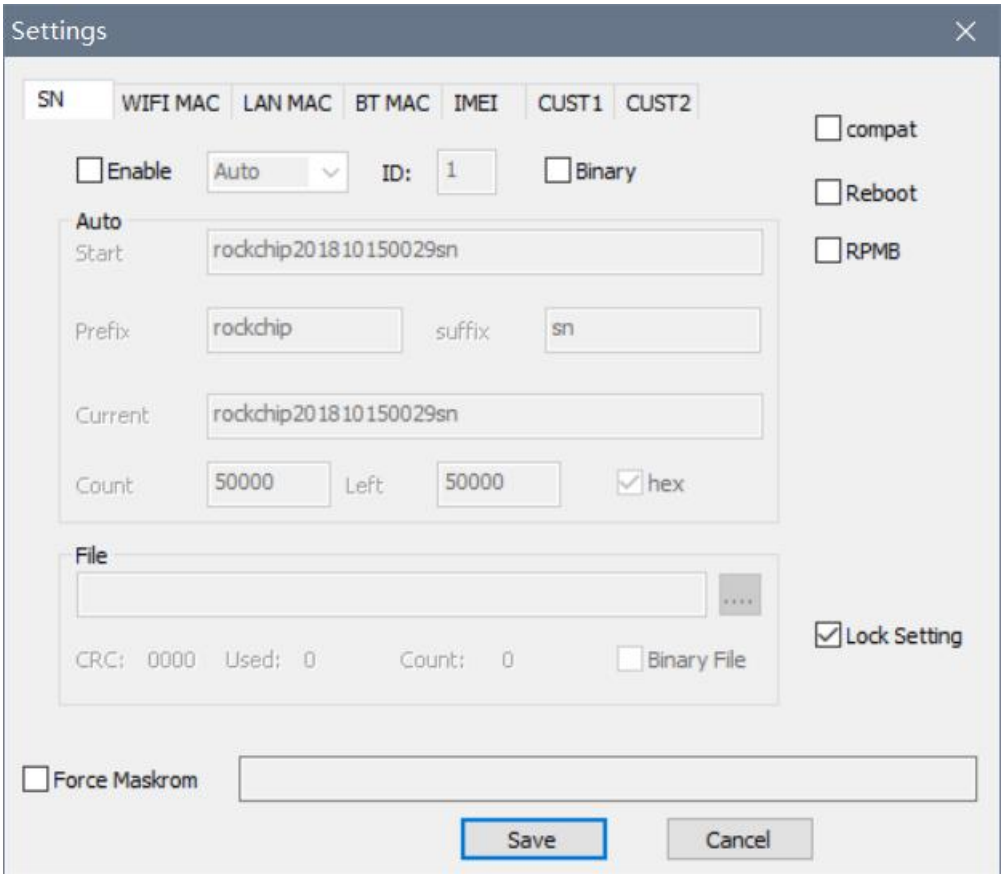
8.7.1 Use RKDevInfoWriteTool to write



Picture 8-4 RKDevInfoWriteTool tool

1) Enter loader mode.

2) Click "setting" menu, click "mode", pop out "settings" window to set SN/WIFI/LAN/BT/IMEI



Picture 8-5 RKDevInfoWriteTool tool mode setting

3) After setting, click "Save" button, close settings window and back to the main window.

4) Click "Write" button.

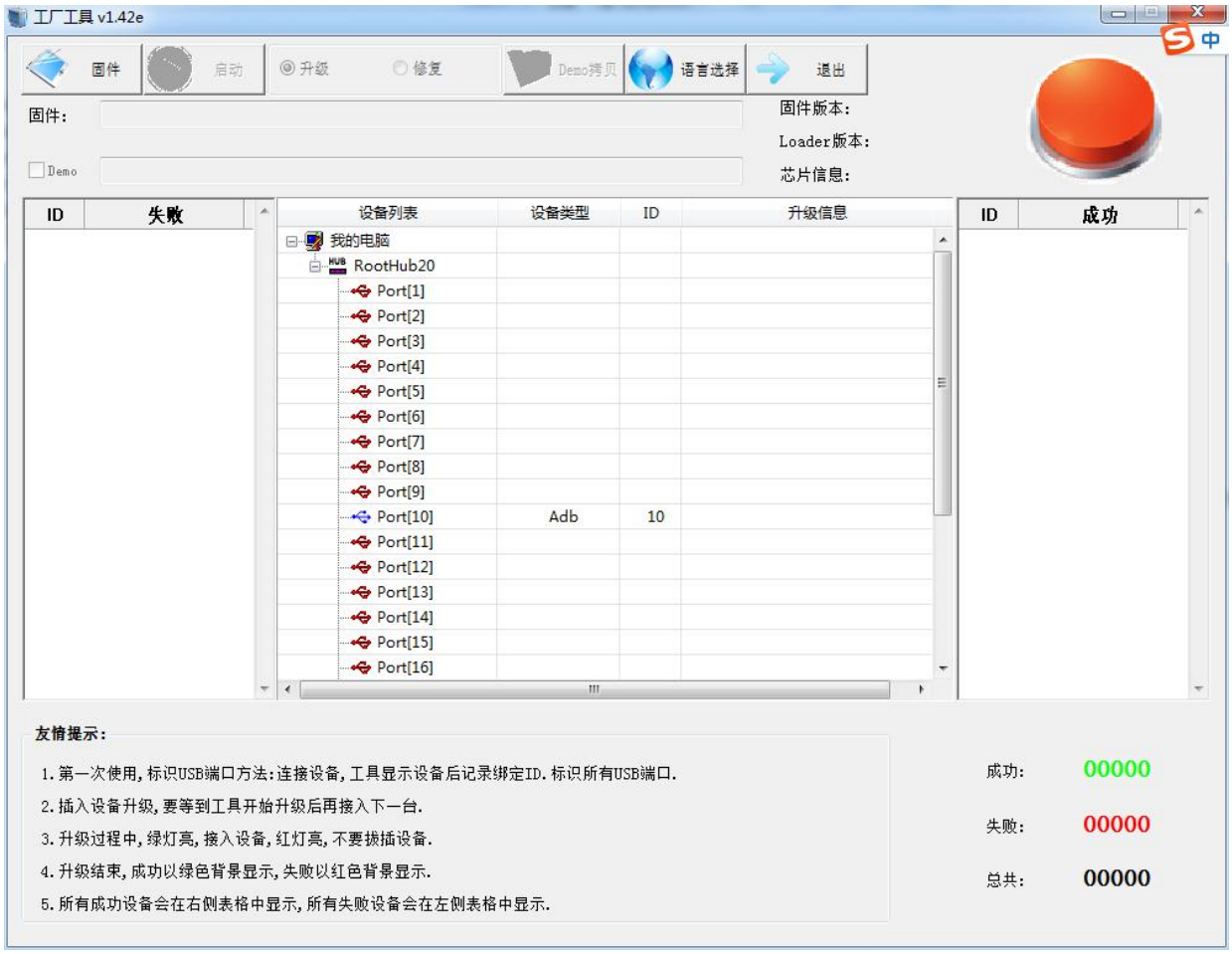
8.7.2 Use RKDevInfoWriteTool to read

1) Enter loader mode.

2) Click "Read" button.

8.8 Production tool usage

8.8.1 Tool download steps



Picture 8-8 Production tool

1) Click image button, select the update.img packed by the package tool, and then wait for unpackage done.
2) Connect the device, make it enter loader or maskrom mode, and the tool will start to download automatically.
3) It is able to connect multiple devices to do the flashing at the same time in order to improve the factory flashing efficiency.