

# Rockchip RK3358 Linux SDK 快速入门

---

文档标识: RK-JC-YF-360

发布版本: V1.0.0

日期: 2021-12-30

文件密级: ☐绝密 ☐秘密 ☐内部资料 ☒公开

## 免责声明

本文档按“现状”提供, 瑞芯微电子股份有限公司(“本公司”, 下同)不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因, 本文档将可能在未经任何通知的情况下, 不定期进行更新或修改。

## 商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标, 归本公司所有。

本文档可能提及的其他所有注册商标或商标, 由其各自拥有者所有。

版权所有© 2021 瑞芯微电子股份有限公司

超越合理使用范畴, 非经本公司书面许可, 任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部, 并不得以任何形式传播。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址: 福建省福州市铜盘路软件园A区18号

网址: [www.rock-chips.com](http://www.rock-chips.com)

客户服务电话: +86-4007-700-590

客户服务传真: +86-591-83951833

客户服务邮箱: [fae@rock-chips.com](mailto:fae@rock-chips.com)

前言

概述

本文主要介绍 Rockchip RK3358 Linux SDK 的基本使用方法，旨在帮助开发者快速了解并使用 RK3358 Linux SDK 开发包。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

产品版本

芯片名称	Buildroot版本	Kernel版本
RK3358	2018.02-rc3	4.19

修订记录

日期	版本	作者	修改说明
2021-12-30	V1.0.0	王建龙	初始版本。

## Rockchip RK3358 Linux SDK 快速入门

1. SDK 工程目录介绍
2. SDK 编译说明
  - 2.1 SDK 开发环境搭建
  - 2.2 SDK 板级配置
  - 2.3 查看编译命令
  - 2.4 自动编译
  - 2.5 各模块编译及打包
    - 2.5.1 U-Boot 编译
    - 2.5.2 Kernel 编译
    - 2.5.3 Recovery 编译
    - 2.5.4 Buildroot 编译
    - 2.5.5 Buildroot 的交叉编译
      - 2.5.5.1 Buildroot 中模块编译
    - 2.5.6 固件的打包
3. 刷机说明
  - 3.1 Windows 刷机说明
  - 3.2 Linux 刷机说明
  - 3.3 系统分区说明
4. Buildroot 应用
  - 4.1 桌面应用
    - 4.1.1 多路视频应用
    - 4.1.2 文件浏览应用
    - 4.1.3 多功能播放器应用
    - 4.1.4 摄像头应用
    - 4.1.5 设置应用
  - 4.2 测试 Demo
  - 4.3 用户和密码
5. 快速启动
6. OTP

## 1. SDK 工程目录介绍

---

SDK目录包含有 buildroot、recovery、app、kernel、u-boot、device、docs、external 等目录。每个目录或其子目录会对应一个 git 工程，提交需要在各自的目录下进行。

- app: 存放上层应用 APP，主要是 qcamera/qfm/qplayer/qsetting 等一些应用程序。
- buildroot: 基于 Buildroot (2018.02-rc3) 开发的根文件系统。
- device/rockchip: 存放各芯片板级配置以及一些编译和打包固件的脚本和预备文件。
- docs: 存放开发指导文件、平台支持列表、工具使用文档、Linux 开发指南等。
- IMAGE: 存放每次生成编译时间、XML、补丁和固件目录。
- external: 存放第三方相关仓库，包括音频、视频、网络、recovery 等。
- kernel: 存放 Kernel 4.19 开发的代码。
- prebuilts: 存放交叉编译工具链。
- rkbin: 存放 Rockchip 相关 Binary 和工具。
- rockdev: 存放编译输出固件。
- tools: 存放 Linux 和 Window 操作系统下常用工具。
- u-boot: 存放基于 v2017.09 版本进行开发的 U-Boot 代码。

## 2. SDK 编译说明

### 2.1 SDK 开发环境搭建

本 SDK 开发环境是在 Ubuntu 系统上开发测试。我们推荐使用 Ubuntu 18.04 的系统进行编译。其他的 Linux 版本可能需要对软件包做相应调整。除了系统要求外，还有其他软硬件方面的要求。

硬件要求：64 位系统，硬盘空间大于 40G。如果您进行多个构建，将需要更大的硬盘空间。

软件要求：Ubuntu 18.04 系统：

编译 SDK 环境搭建所依赖的软件包安装命令如下：

```
sudo apt-get install repo git ssh make gcc libssl-dev liblz4-tool \
expect g++ patchelf chrpath gawk texinfo chrpath diffstat binfmt-support \
qemu-user-static live-build bison flex fakeroot cmake gcc-multilib g++-multilib
unzip \
device-tree-compiler python-pip ncurses-dev pyelftools \
```

建议使用 Ubuntu18.04 系统或更高版本开发，若编译遇到报错，可以视报错信息，安装对应的软件包。

### 2.2 SDK 板级配置

进入工程/device/rockchip/rk3358 目录：

板级配置	说明
BoardConfig-rk3358-evb-ddr3-v10.mk	适用于 RK3358 EVB V10 开发板

方法1

`./build.sh` 后面加上板级配置文件, 例如：

选择**RK3358 EVB V10**的板级配置：

```
./build.sh device/rockchip/rk3358/BoardConfig-rk3358-evb-ddr3-v10.mk
```

方法2

```
./build.sh lunch
processing option: lunch

You're building on Linux
Lunch menu...pick a combo:

0. default BoardConfig.mk
1. BoardConfig-rk3358-evb-ddr3-v10.mk
2. BoardConfig.mk
Which would you like? [0]:
```

## 2.3 查看编译命令

在根目录执行命令: `./build.sh -h|help`

```
rk3358$ ./build.sh -h
Usage: build.sh [OPTIONS]
Available options:
BoardConfig*.mk    -switch to specified board config
lunch               -list current SDK boards and switch to specified board config
uboot               -build uboot
spl                 -build spl
loader              -build loader
kernel              -build kernel
modules             -build kernel modules
toolchain           -build toolchain
rootfs              -build default rootfs, currently build buildroot as default
buildroot           -build buildroot rootfs
ramboot             -build ramboot image
multi-npu_boot      -build boot image for multi-npu board
yocto               -build yocto rootfs
debian              -build debian rootfs
pcba                -build pcba
recovery            -build recovery
all                 -build uboot, kernel, rootfs, recovery image
cleanall            -clean uboot, kernel, rootfs, recovery
firmware            -pack all the image we need to boot up system
updateimg           -pack update image
otapackage          -pack ab update otapackage image
save                -save images, patches, commands used to debug
allsave             -build all & firmware & updateimg & save
info                -see the current board building information
app/<pkg>            -build packages in the dir of app/*
external/<pkg>       -build packages in the dir of external/*

Default option is 'allsave'.
```

查看部分模块详细编译命令, 例如: `./build.sh -h kernel`

```
rk3358$ ./build.sh -h kernel
###Current SDK Default [ kernel ] Build Command###
cd kernel
make ARCH=arm64 rockchip_linux_defconfig
make ARCH=arm64 rk3358-evb-ddr3-v10-linux.img -j12
```

## 2.4 自动编译

进入工程根目录执行以下命令自动完成所有的编译:

```
./build.sh all # 只编译模块代码 (u-Boot, kernel, Rootfs, Recovery)
               # 需要再执行 ./mkfirmware.sh 进行固件打包

./build.sh     # 在 ./build.sh all 基础上
               # 1. 增加固件打包 ./mkfirmware.sh
               # 2. update.img 打包
               # 3. 复制 rockdev 目录下的固件到 IMAGE/***_RELEASE_TEST/IMAGES 目录
               # 4. 保存各个模块的补丁到 IMAGE/***_RELEASE_TEST/PATCHES 目录
               # 注: ./build.sh 和 ./build.sh allsave 命令一样
```

## 2.5 各模块编译及打包

### 2.5.1 U-Boot 编译

```
### U-Boot 编译命令
./build.sh uboot

### 查看 U-Boot 详细编译命令
./build.sh -h uboot
```

### 2.5.2 Kernel 编译

```
### Kernel 编译命令
./build.sh kernel

### 查看 Kernel 详细编译命令
./build.sh -h kernel
```

### 2.5.3 Recovery 编译

```
### Recovery 编译命令
./build.sh recovery

### 查看 Recovery 详细编译命令
./build.sh -h recovery
```

### 2.5.4 Buildroot 编译

进入工程目录根目录执行以下命令自动完成 Rootfs 的编译及打包：

```
./build.sh rootfs
```

编译后在 Buildroot 目录 output/rockchip\_rk3358/images 下生成 rootfs.ext4。

## 2.5.5 Buildroot 的交叉编译

若需要编译单个模块或者第三方应用，需对交叉编译环境进行配置。交叉编译工具位于 `buildroot/output/rockchip_rk3358/host/usr` 目录下，需要将工具的 `bin/` 目录和 `aarch64-buildroot-linux-gnu/bin/` 目录设为环境变量，在顶层目录执行自动配置环境变量的脚本（只对当前控制台有效）：

```
source envsetup.sh
```

输入命令查看：

```
cd buildroot/output_rk3358/host/usr/bin  
./aarch64-linux-gcc --version
```

此时会打印如下信息：

```
gcc version 9.3.0 (Buildroot 2018.02-rc3-02723-gd3fbc6ae13)
```

### 2.5.5.1 Buildroot 中模块编译

比如 `qplayer` 模块，常用相关编译命令如下：

- 编译 `qplayer`

```
SDK$make qplayer
```

- 重编 `qplayer`

```
SDK$make qplayer-rebuild
```

- 删除 `qplayer`

```
SDK$make qplayer-dirclean  
或者  
SDK$rm -rf /buildroot/output/rockchip_rk3358/build/qplayer-1.0
```

## 2.5.6 固件的打包

上面 `Kernel/U-Boot/Recovery/Rootfs` 各个部分的编译后，进入工程目录根目录执行以下命令自动完成所有固件打包到 `rockdev` 目录下：

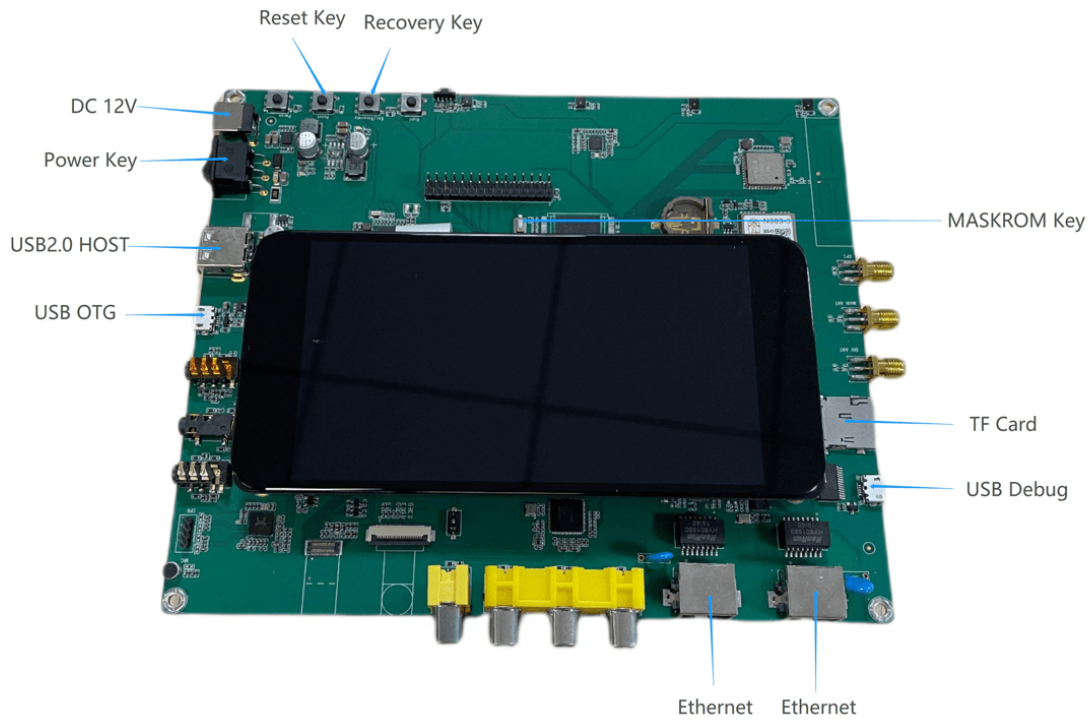
固件生成：

```
./mkfirmware.sh
```

## 3. 刷机说明

---

RK3358 EVB V10 接口分布图如下：

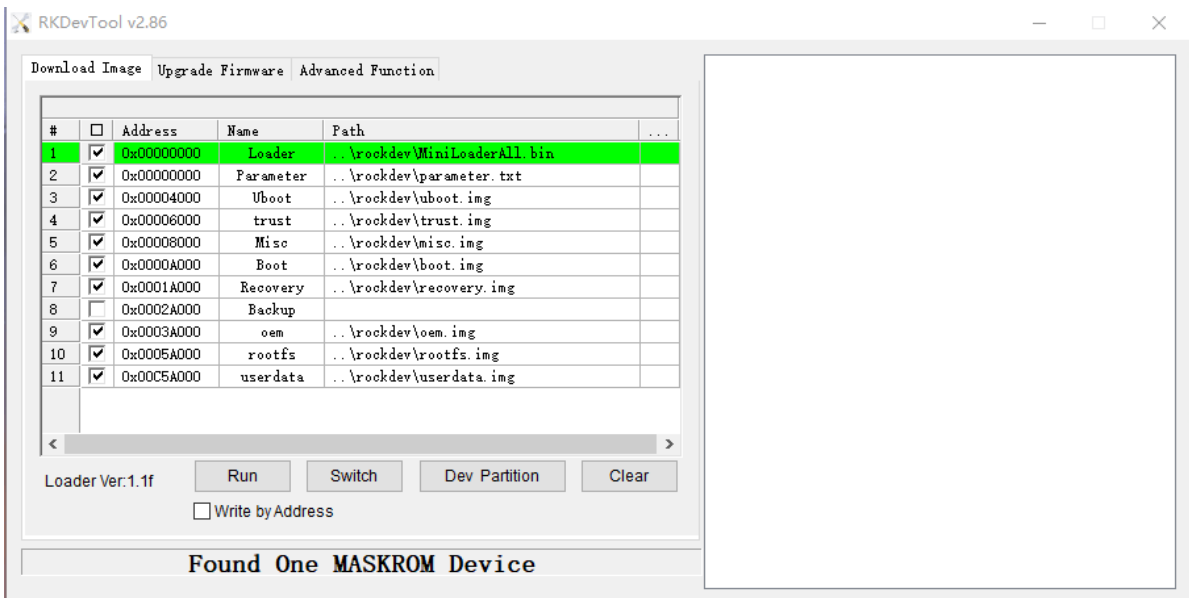


### 3.1 Windows 刷机说明

SDK 提供 Windows 烧写工具(工具版本需要 V2.86 或以上)，工具位于工程根目录：

```
tools/  
└─ windows/RKDevTool
```

如下图，编译生成相应的固件后，设备烧写需要进入 MASKROM 或 BootROM 烧写模式，连接好 USB 下载线后，按住按键“MASKROM”不放并按下复位键“RST”后松手，就能进入 MASKROM 模式，加载编译生成固件的相应路径后，点击“执行”进行烧写，也可以按 “recovery” 按键不放并按下复位键 “RST” 后松手进入 loader 模式进行烧写，下面是 MASKROM 模式的分区偏移及烧写文件。(注意： Windows PC 需要在管理员权限运行工具才可执行)



注：烧写前，需安装最新 USB 驱动，驱动详见：



```
<SDK>/tools/windows/DriverAssitant_v5.11.zip
```

## 3.2 Linux 刷机说明

Linux 下的烧写工具位于 tools/linux 目录下(Linux\_Upgrade\_Tool 工具版本需要 V1.65 或以上), 请确认你的板子连接到 MASKROM/loader rockusb。比如编译生成的固件在 rockdev 目录下, 升级命令如下:

```
sudo ./upgrade_tool ul rockdev/MiniLoaderAll.bin
sudo ./upgrade_tool di -p rockdev/parameter.txt
sudo ./upgrade_tool di -u rockdev/uboot.img
sudo ./upgrade_tool di -t rockdev/trust.img
sudo ./upgrade_tool di -misc rockdev/misc.img
sudo ./upgrade_tool di -b rockdev/boot.img
sudo ./upgrade_tool di -recovery rockdev/recovery.img
sudo ./upgrade_tool di -oem rockdev/oem.img
sudo ./upgrade_tool di -rootfs rockdev/rootfs.img
sudo ./upgrade_tool di -userdata rockdev/userdata.img
sudo ./upgrade_tool rd
```

或升级打包后的完整固件:

```
sudo ./upgrade_tool uf rockdev/update.img
```

或在根目录, 机器在 MASKROM 状态运行如下升级:

```
./rkflash.sh
```

## 3.3 系统分区说明

默认分区说明 (下面是 RK3358 EVB 分区参考)

Number	Start (sector)	End (sector)	Size	Name
1	16384	24575	4096K	uboot
2	24576	32767	4096K	trust
3	32768	40959	4096K	misc
4	40960	106495	32M	boot
5	106496	303104	32M	recovery
6	172032	237567	32M	bakcup
7	237568	368639	64M	oem
8	368640	12951551	6144M	rootfs
9	12951552	30535646	8585M	userdata

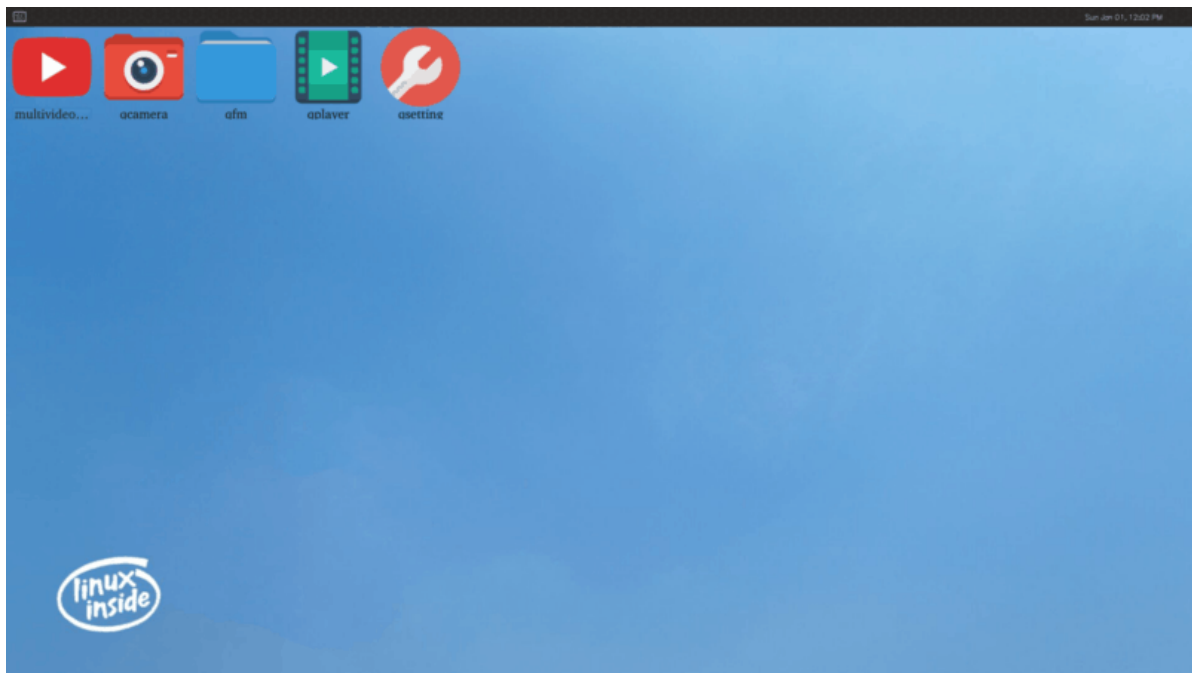
- uboot 分区：供 uboot 编译出来的 uboot.img。
- trust 分区：供 uboot 编译出来的 trust.img。
- misc 分区：供 misc.img，给 recovery 使用。
- boot 分区：供 kernel 编译出来的 boot.img。
- recovery 分区：供 recovery 编译出的 recovery.img。
- backup 分区：预留，暂时没有用，后续跟 Android 一样作为 recovery 的 backup 使用。
- oem 分区：给厂家使用，存放厂家的 APP 或数据。挂载在 /oem 目录。
- rootfs 分区：供 buildroot、debian 或 yocto 编出来的 rootfs.img。
- userdata 分区：供 APP 临时生成文件或给最终用户使用，挂载在 /userdata 目录下。

## 4. Buildroot 应用

---

### 4.1 桌面应用

Buildroot 默认支持 Wayland 桌面环境以及一些 Qt 应用。如下图：



这些 Qt 应用提供了一些基础功能，如，摄像头预览，文件管理器，多媒体播放器，WiFi 连接等。

#### 4.1.1 多路视频应用

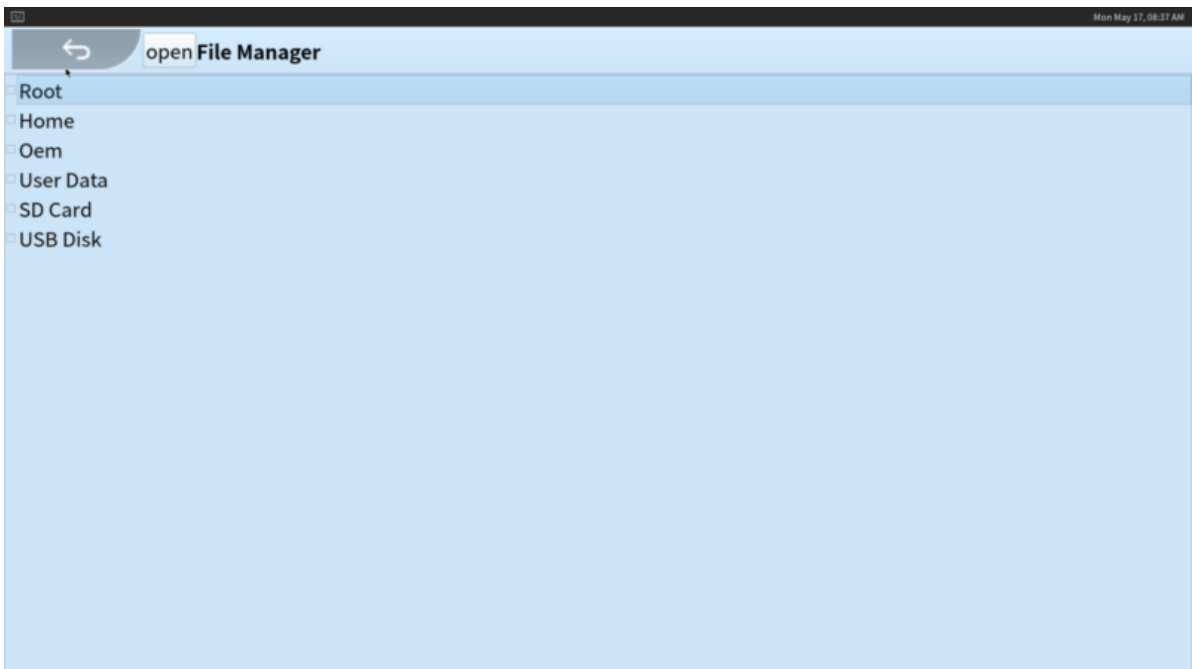


多路视频播放器用于测试设备的多路视频播放能力、显示能力以及硬解码能力，点击启动应用后会自动循环播放 9 路视频 demo。

双击画面可以全屏播放。

#### 4.1.2 文件浏览应用

qfm 是一个文件浏览应用。



通过 qfm 可以浏览设备文件，点击目录名进入下一级目录，点击左上角返回上一级。

点击文件可以调用系统对应的默认应用打开文件，如点击 mp4 文件将调用 qplayer 应用打开视频。

### 4.1.3 多功能播放器应用

qplayer 是一个多功能播放器，可以播放视频、音频和浏览图片。点击启动后将自动播放视频 demo。

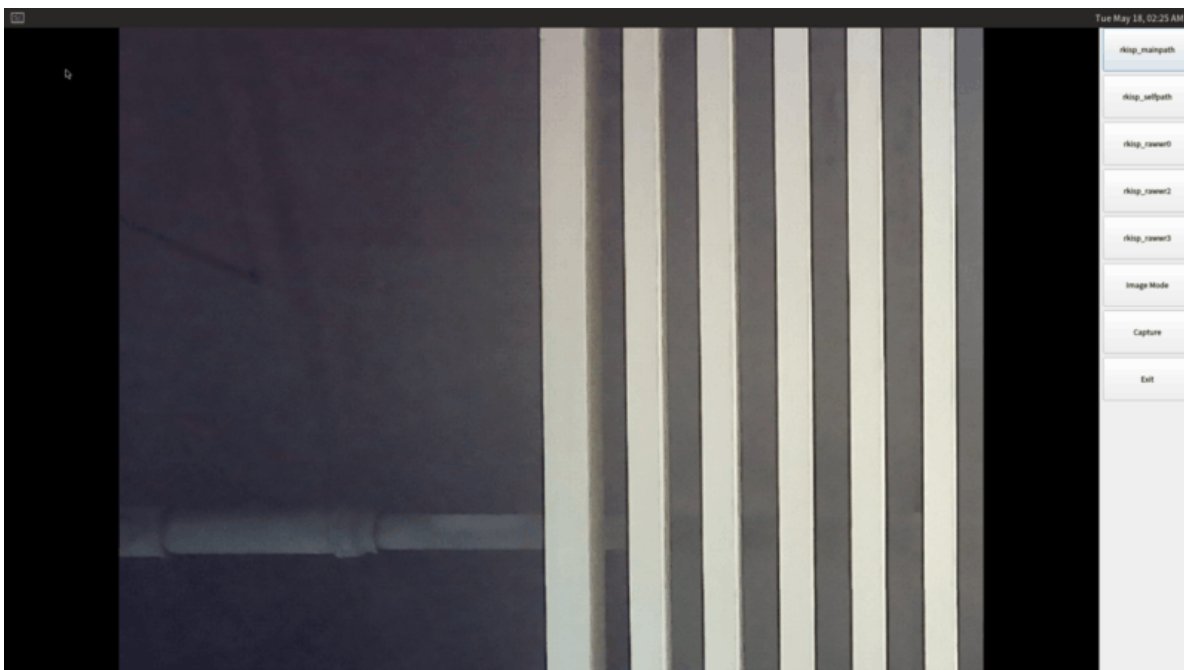


也可以通过在 qfm 中点击指定文件来使用 qpalyer 打开。

双击画面可以全屏播放。

### 4.1.4 摄像头应用

qcamera 是一款相机应用，可以进行拍摄和录像。



设备连接摄像头的情况下启动 qcamera 将自动显示摄像头画面，右侧按钮：

Image Mode: 照相模式，点击可切换为 Video Mode 视频录制模式。

Capture: 捕捉图像，在 Video Mode 下会变为 Record 录制按钮。

Exit: 退出。

## 4.1.5 设置应用

qsetting 是系统设置工具，其中可以设置 WiFi 连接，蓝牙连接，实现恢复出厂设置以及固件升级。



## 4.2 测试 Demo

更新完固件, 可以运行 rockchip\_test.sh 进行相关功能测试。

```
[root@rk3358:/]# /rockchip_test/rockchip_test.sh
*****
ddr test :                1 (memtester & stressapptest)
cpufreq test:             2 (cpufreq stresstest)
flash stress test:       3
bluetooth test:          4 (bluetooth on&off test)
audio test:              5
recovery test:           6 (default wipe all)
suspend_resume test:     7 (suspend & resume)
wifi test:               8
ethernet test:           9
auto reboot test:       10
ddr freq scaling test   11
npu test                12
npu2 test               13 (rk356x)
camera test             14 (use rkisp_demo)
video test              15 (use gstreamer-wayland and app_demo)
gpu test                16 (use glmark2)
chromium test           17 (chromium with video hardware acceleration)
nand power lost test:   18
*****
please input your test moudle:
```

或者直接运行glmark2-es2-wayland --fullscreen --annotate 进行测试

```
[root@rk3358:/]# glmark2-es2-wayland --fullscreen --annotate
arm_release_ver of this libmali is 'g2p0-01eac0', rk_so_ver is '7'.
=====
glmark2 2020.04
=====
OpenGL Information
GL_VENDOR:      ARM
GL_RENDERER:    Mali-G31
GL_VERSION:     OpenGL ES 3.2 v1.g2p0-01eac0.7abca4a459ce7997f20ab707543ff422
=====
[build] use-vbo=false:

...
```

## 4.3 用户和密码

用户: root

密码: rockchip

## 5. 快速启动

常规启动流程大多经过以下过程: miniloader -> uboot -> kernel -> rootfs -> app, 故只需要对上述的各个流程的启动进行优化, 便能够实现快速启动。本章主要介绍 RK3358 快速启动的调优方法和注意事项。

典型的优化方式如下:

- kernel 和 rootfs 通过 SPL 方式加载, 裁减掉 uboot;
- kernel 配置进行针对性的裁减, 减少无用配置, 缩小 kernel 体积;
- rootfs 根据应用场景需要进行裁减, 缩小体积;

对于 kernel, 可以开启 init\_debug, 即 CONFIG\_PRINTK\_TIME 和 CONFIG\_KALLSYMS 配置, 通过 kernel/scripts/bootgraph.pl, 分析 kernel 启动时间的分布, 针对性地优化耗时的模块, 耗时长驱动可以通过模块的方式在 rootfs 启动后加载。

对于 rootfs, 需要根据应用的场景进行构建, 可以从工具链、C库、GUI 等方向进行选择, 如:

- 更高版本的 gcc 和 binutils 往往会有更好的优化功能;
- 标准且最全面的 glibc 可以根据需要替换成可配置且体积小的 uClibc、musl;
- rootfs 默认集成的 wayland 和 QT, 可以根据需要替换成 MiniGui、Unity等其他框架;
- 裁减 rootfs 体积, 去除无用的启动服务和后台服务;

## 6. OTP

OTP (One Time Programmable), 即只可编程一次的非易失性存储。作为对比, FLASH 存储可多次擦写。RK3358 的 OTP SIZE 为 64 bytes。OTP 相关开发的详细说明可以参考文档/docs/Linux/Security/Rockchip\_Developer\_Guide\_TEE\_SDK\_CN.pdf。

