# Rockchip

## PCBA 测试工具开发指南

发布版本:**0.02**

日期:**2017.02**

# 前言

## 概述

本文档主要介绍 Rockchip PCBA 测试工具的使用方法和开发指南。通过本文档可快速了解 PCBA 工具的使用，以及 PCBA 测试功能的扩展。

## 产品版本

| 芯片名称 | 内核版本 |
| --- | --- |
| RK3399 | Linux4.4 |
| | |

## 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 | 修改说明 |
| --- | --- | --- | --- |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# 目录

# 1 PCBA

## 1.1 概述

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。

工具通过配置文件 test_config.cfg 对测试项进行配置，并可根据需求增加新测试项，具体的配置说明请参考本文“配置说明”。

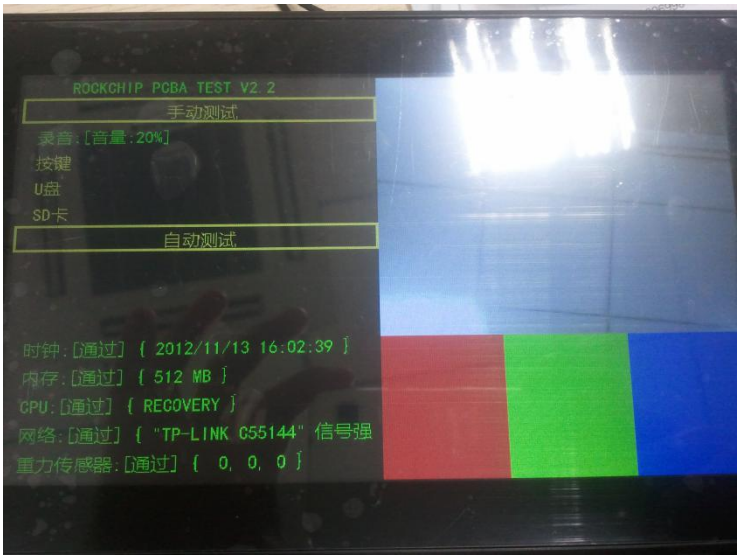



图 1-1 PCBA 测试界面

# 1.2 测试项说明

测试项分为“自动测试项” 和 “手动测试项”。

自动测试项由系统自动进行测试并判断测试结果,如：网络、存储等。手动测试项需要由人工配合完成或者配合判断测试结果。如：录音，按键，U 盘，SD 卡等。

测试项分别有 “红”，“黄”，“绿” 三种颜色表示不同的测试状态

黄色：未测试项或者正在测试的项

绿色：测试通过项

红色：测试未通过项

具体测试项说明如下。

## 1.2.1 自动测试项

- DDR/EMMC 容量测

DDR/EMMC 容量检测为自动测试项,自动显示当前设备 DDR 容量和 EMMC 容量大小,显示单位为 GB,常见 DDR 容量为 1GB、2GB、4GB 等，常见 EMMC 容量为 2GB、4GB、8GB、16GB、32GB、64GB、128GB 等。测试结果示例如下： “系统存储：[通过] { DDR：2GB，EMMC：16GB }”

- 无线网络（wifi 测试）

Wifi 为自动测试项，会自动扫描周边的 AP，并显示信号最强的那个 AP 名字及信号强度。信号强度根据 AP 强度显示 0 到 4 格。测试结果如下： “网络：[通过] { “AP WIFI” 信号强度 4 格 }”

- 以太网测试

有两种测试方式，默认以 ping 的方式测试以太网的通信功能，发 5 个包收 5 个包，0 包丢失。本机网址和 ping 的目标网址在 external/rk-pcba-test/res/test_cong.cfg 中修改。

另一中测试方法，是以检测网卡是否启动来判断，网卡启动时会有一个 inet6 的网址，如:“inet6 addr: fe80::c8eb:eaff:fe6d:730d/64 Scope:Link”。

检测到有 inet6 网址，则认为以太网是正常的。（这种方式不能确定网络的通信是否正常）。如需使用此方式测试以太网须修改 external/rk-pcba-test/lan_test.c，将 c 文件中的“use_ping = 1;”注释掉即可。

- 耳机喇叭（codec）测试

Codec 测试有两种模式：边录边放、先录后放。先放后录模式，测试效率相对低，使用喇叭时不会有啸叫，可在使用喇叭时选择此模式。边录边放模式，测试效率高，使用喇叭时会有啸叫，可在使用耳机时选择此模式。测试后的录音音量根据实际输入变化，范围从 0-100%：“录音音量：[25%]”（模式配置见本文

配置说明）。

- 蓝牙（bluetooth）测试

蓝牙测试为自动测试项，开启检测后系统自动进行检测，根据检测结果显示成功或失败。

## 1.2.2手动测试项

- 按键（KEY）测试

打开按键测试项后屏幕显示按键测试栏，手动点击设备按键后，屏幕会显示相应按键信息，如"ESC、POWER、MENU、HOME 等"。测试人员根据所按按键判断屏幕所显示按键信息是否正确判断按键是否正常。

- 屏幕坏点（LCD）测试

LCD 坏点测试为手动测试项，打开测试选项后屏幕的右下方显示红、绿、蓝三原色的方块。测试人员连续“**N 次**”按“**启动按键**”后进入坏点测试模式，坏点测试整屏依次显示“红、绿、蓝、黑、白”画面并通过“**切换按键**”进行切换。测试人员根据显示画面判断 LCD 显示是否正常、是否有坏点。

“启动按键”及“N 次”以及“切换按键”的设定见“配置文件”中的 lcd 配置。

- SD 卡（sdcard）测试

插入 SD 卡，系统会进行自动检测，检测后屏幕显示 SD 卡是否通过，如通过，则会显示 SD 卡容量信息。

SD card 必须为 FAT32 格式，不支持其他格式！整个卡只能包含一个分区。如果不符合要求，请通过格式化来格式成标准格式。

- USB HOST 测试

USB HOST 常用 U 盘进行测试，插入 U 盘后会自动检测，检测结束后显示是否通过，并显示 U 盘容量信息。

与 SD 卡类似，U 盘必须为 FAT32 格式，不支持其他格式！整个卡只能包含一个分区。如果不符合要求，请通过格式化来格式成标准格式。

# 1.3 配置文件说明

PCBA 所有的测试项目通过配置脚本 test_config.cfg 来配置，位于”Android src”/external/rk-pcba-test/res/test_config.cfg，用户可以根据项目的硬件配置来配置 test_config.cfg 文件，决定要对哪些模块进行测试，以及给自己的测试程序传递相关的参数。

该脚本使用 ini 文件格式，由段、键和值三者组成，通常一个段表示一个模块配置。目前要求该配置文件使用 UTF-8 编码，其他编译格式可能会导致未知错误。

## 1.3.1 配置文件模板

```
[example]
display_name= "Example"
activated = 1
program = "example.sh"
category = 0
```

**[example]**

Example 表示一个配置模块的名称，如果是 cfg 文件中自带的模块名称，则不能改动，否则会导致某个测试项不被测试系统启动。

**display_name**

display_name 表示该测试模块在屏幕上显示的名称，可以根据自己的需要修改。该名称最长为 64 字节，如果为空，则测试程序不会运行。

**activated**

activated 表示是否测试该模块，0：不测试该模块 1：测试该模块

**program**

该键值目前没用到，可以不用配置

**category**

category 表示测试方式，0：自动测试 1：手动测试

示例：

```
[Key]                          //按键测试
display_name= "Key"
activated    = 1               //测试该项目
program      = "keytester"
category     = 1               //手动测试
```

## 1.3.2 部分测试项配置说明

**屏幕测试**

```
[Lcd]
display_name= "lcd"
activated    = 1         //测试该项
program      = "lcd test"
category     = 1         //手动测试
run_type     = 1
start_key    = "KEY_BACK"   //启动测试的按键
key_times    = 3            //连续按启动键的次数
all_key_change = 1          //进入测试后是否全部按键可切换画面
```

连续按 key_times 次 start_key 按键将进入测试模式，进入测试模式后通过 start_key 进行画面切换，若需要通过任意键进行画面切换则令 all_key_change=1，否则 all_key_change 设为 0；测试结束后回到主界面，显示“屏幕：[已测试]”。

start_key 常用按键: "KEY_BACK"，"KEY_VOLUMEUP"，"KEY_VOLUMEDOWN""KEY_HOME""KEY_MENU"，"KEY_ENTER"，"KEY_ALL" 等。(start_key 设为 KEY_ALL 或无效键值则任一键连续按 key_times 次进入测试模式)

**音频测试**

```
[Codec]
display_name= "Codec"
activated    = 1             //测试该项目
program      = "case1"    //case1, case2
category     = 1             //手动测试
run_type     = 1
delay        = 5
volume       = 40
```

case1：
先放后录模式，测试效率相对低，使用喇叭时不会有啸叫，可在使用喇叭时选择此模式
case2：
边录边放模式，测试效率高，使用喇叭时会有啸叫，可在使用耳机时选择此模式

**蓝牙测试**

```
[bluetooth]
display_name= "bluetooth"
activated    = 1
program      =
category     =
run_type     = 1
chip_type    = "" ; rk903, mt6622, rda587x, rda5990,rtk8723as
```

chip_type 为选择相应的 BT 芯片型号，默认为空，也就是不测试 BT，Android 5.0 后不需要选择，系统会自动识别。

**传感器测试**

```
[allsensor]
display_name= "allsensor"
activated    = 0
program      = ""
category     = 0
run_type     = 1
sensor_type = 39
```

sensor_type 会选择相应的传感器种类。1：GSENSOR,2：GYROSCOPE,4：COMPASS，8：LSENSOR,16：PSENSOR,32：GSENSOR_CALIBRATE。如果需要测试多种传感器，只需把传感器种类对应的数值相加即可。默认的 sensor_type 是 39,即 GSENSOR_CALIBRATE(32)+COMPASS(4)+GYROSCOPE(2)+GSENSOR(1)。

**Camera 测试**

```
[camera]
display_name = "camera"
activated    = 1
category     = 0
program      = ""
number       = 2
```

number 表示测试的 camera 个数，最大支持测试 2 个 camera.

如需自行添加测试模组，请参考文档《RK_PCBA_Camera_移植说明_v1.0.doc》

### 1.3.3 配置脚本参数扩展

test_config.cfg 配置脚本中各测试项的参数可以进一步扩展，如果某个模块需要通过配置脚本传递相关参数，可进行如下扩展：

```
[example]
display_name= "Example"
activated = 1
program = "example.sh"
module_args = “xxx”
```

在具体的测试程序中，可以通过 script_fetch api 获得设置的相关参数值：

```
int script_fetch(char *main_name, char *sub_name, int value[], int count)
```

main_name：测试模块的名称，在 test_config.cfg 文件中测试项，示例中的“example”
sub_name:键值，比如 activated、display_name、module_args 等等。
取值示例：

```
if(script_fetch("example", "module_args", (int *)des, 8) == 0)
{
```

```
    printf("module_args value is: %s\n",des);
}
```

## 1.4 测试项扩展

该 PCBA 程序允许用户扩展自己的测试样例。如果因为项目需要，用到了该测试程序中目前还未支持到的模块，可以自己添加测试程序，然后集成到测试框架中。

集成方法如下：

（1）先写好自己的测试程序和头文件。测试程序要封装成 void * xxxx_test(void *argv)格式的接口。

（2）确定该测试项为手动测试项或者是自动测试项，并在 test_config.cfg 里面加入想要的配置。

（3）在 pretest.c 的中注册自己的测试代码。

以 Lcd 为例：

Pretest.c 头部包含头文件并定义线程：

```
#include "screen_test.h"
pthread_t screen_tid;
```

Pretest.c 文件中 start_test_pthread()函数新增创建线程代码：

```
if (!strcmp(tc_info->base_info->name, "Lcd")) {
    err = pthread_create(&screen_tid, NULL, screen_test, tc_info);
    if (err != 0) {
        printf("create screen test thread error: %s/n",
                strerror(err));
        return -1;
    }
}
```

其中 screen_test 为模块中的测试函数。

## 1.5 字体配置

说明：PCBA 2.0 以后的版本增加了对中文的支持，并可以支持多种字体大小的配置，包括 18*18, 20*20, 24*24, 28*28, 32*32, 36*36,可以通过修改 minuitwrp/graphics.c 的头文件来包含修改使用不同大小的字库。(输出到屏幕的中文必须是 UTF-8 编码格式)

## 1.6 屏幕旋转配置

PCBA 测试支持屏幕旋转功能，根据需求旋转 0°、90°、180°、270°以支持不同摆放的横/竖屏显示，其中 0°、180°用于支持竖屏，90°、270°用于支持横屏，默认为 0°。

旋转屏幕的配置在 rk-pcba-test 根目录下的 Android.mk 中，打开相应角度的配置后进行编译，如已经编译过需强制重编。mmm external/rk-pcba-test/ -B

```
#=========  rotate screen  0, 90, 180, 270  degree===========
# warning:  If changed please force rebuid the target -B
#ROTATE_SCREEN := 0
#ROTATE_SCREEN := 90
#ROTATE_SCREEN := 180
ROTATE_SCREEN := 270
```

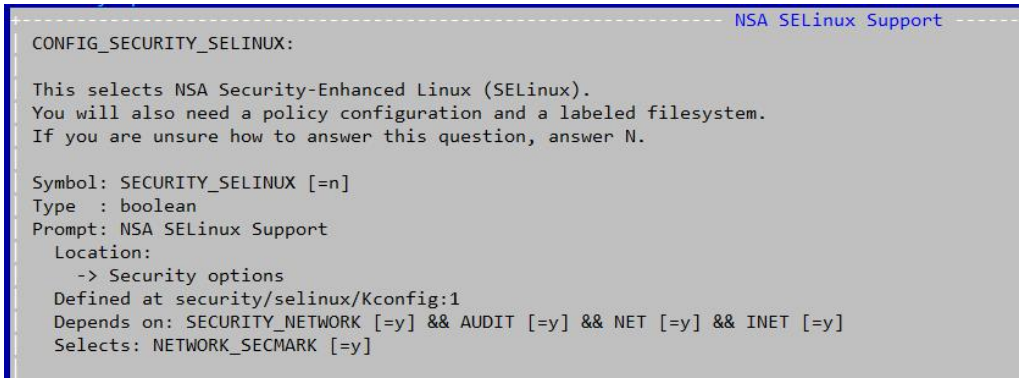
# 1.7 固件编译配置打包

PCBA 测试程序位于 Android 源码/extenal/rk-pcba-test 目录下，编译会生成 pcba_core 可执行文件。pcba_core 和 rk-pcab-test/res 下的相关文件在编译的时候会被自动拷贝到 recovery 的 sbin 目录下。

PCBA 程序运行于 Recovery 系统中，具体测试流程为：开机进入 Recovery，启动 PCBA 测试程序进行各项功能测试。

**1、内核配置(7.1 系统跳过此步骤)**

PCBA 固件无需 selinux，请关闭，对于正常固件请打开：




重新编译 kernel，然后更新 recovery 的 kernel。执行下面命令即可：cp kernel/arch/arm64/boot/Image out/target/product/rk3399/kernel （红色的目录需要换成你自己的编译路径。）

**2、打开 device/rockchip/common/BoardConfig.mk 文件中的：**

修改 BoardConfig.mk 文件中的 TARGET_ROCKCHIP_PCBATEST?=true

make installclean

make

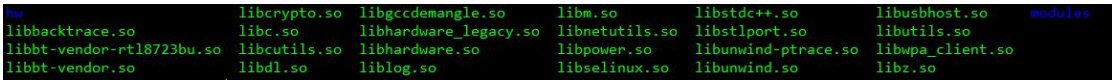
rm -rf out/target/product/rk3399/root/ 删除 out 下的 root 目录

make recoveryimage

**3、编译**

执行 make 命令，编译完后请确保下面目录是否有一下文件(注意:红色目录是你们自己的目录)：lsout/target/product/rk3399/recovery/root/sbin/目录:


ls out/target/product/rk3399/recovery/root/system/lib




（如果没有的话请手动在 external/rk-pcba-test/目录执行 mm –B ,并重新 make。）