Security Classification:

Most Confidential(　　) Confidential(　　) Internal(　) Public(√)

RK3328_Linux_Debian-SDK_V1.1 Development Guide

（Technical Department，Product R&D Dept.I）

Mark： [　] Editing [√] Issued	Current Ver.：	V1.00
	Drafted By：	YHX
	Fulfill Date：	2017-07-11
	Checked By：	ZXZ,CW
	Completed Date：	2017-07-11

福州瑞芯微电子股份有限公司

Fuzhou Rockchips Semiconductor Co．, Ltd.

Revision History

Version	Author	Date	Revision Note	Remarks
V1.0	YHX	2017-04-19	Initial version	
V1.1	YHX	2017-07-11	Add English Version	

Content

1 Overview

This SDK is based on Debian 9 system with kernel 4.4. It is applicable to the development of RK3328 EVB and all other linux products based on it.

This SDK supports VPU hardware decoding, GPU 3D, QT and other functions.

2 Main Supported Functions

Functions	**Module Name**
Data Communication	Wi-Fi、Ethernet、USB、SDCARD
Application	Browser、Calculator、Terminal、Image viewer、Explorer、Setting、Video Player

3 SDK obtaining introduction

The SDK is released via Github. The introduction of establishing a build and development environment, and the development references are published through Rockchip's opensource website,as below:

http://opensource.rock-chips.com/wiki_Main_Page

3.1 GitHub Address

All source codes of SDK will be published and updated via Github.

Github:https://github.com/rockchip-linux

3.2 Rockchip's Opensource Website

Rockchip releases referenence documents through the opensource document wiki site, including TRM, Datasheet, Schematic&Layout Guide, establishing a development environment, build configuration and command instructions.

http://opensource.rock-chips.com/wiki_Main_Page

3.3 Repo Installation

Repo is a script written by Google using Python, which is mainly used to download and manage the Android software repository. Its download command is as below:

```
sudo apt-get install repo
```

3.4 SDK Download

The download command of RK3328_Linux_SDK is as below:

```
repo init -u https://github.com/rockchip-linux/manifests -b master
repo sync
```

Later, according to update information from FAE window or information posted on the GitHub, developers can do code synchronization by following command:

```
.repo/repo/repo sync
```

4 Software Development Guide

4.1 Overview of Software Composition

4.1.1 Rootfs

Rockchip provides support for Debian. Developers can add various software packages to the constructed Debian Rootfs. The Rootfs is based on the Linaro build project. It adds support for the graphical display framework and video encode/devode framework, including libmali, xserver, gstreamer-rockchip, etc. Developers can customize the rootfs according to the actual needs of the product.

4.1.2 Xserver

The RK3328 Debian system uses Xserver as the graphical display framework, and use the LXDE as desktop.

Xserver is a common desktop display system for Linux platform. Our RK3328 Debian system uses a customized Xserver that has enabled 2D hardware acceleration. Our Xserver has good compatibility,the software size is larger than other display systems.

For more detailed introduction , see the following link:

http://opensource.rock-chips.com/wiki_Graphics

4.1.3 Gstreamer

RK3328 Debian system uses GStreamer as its multimedia framework. GStreamer is an open source multimedia framework library. With it, you can build a series of media processing modules, including simple Ogg play function to complex audio and video processing.

Applications can use decoding and filtering technology transparently. Developers can call concise and common interfaces to create a simple plug-in to add new decoders or filters.

The GStreamer is divided into the following several parts.

Gstreamer: core library and interface implementation, including the implementation of gstbuffer, gstclock, gstbin and other core components.

gst-plugins-base: the core library and interface implementation of the plug-ins.

gst-plugins-good: a set of high quality plug-ins under the LGPL license, here we need video file parsing plugin-in, such as isomp4.

gst-plugins bad: some plug-ins that require more testing, here the video parser, such as h264parse is used.

gst-libav: a FFmpeg-based plug-in that supports encoding and decoding of audios and videos, many common audios and videos are supported.

For detail, see following link:

http://opensource.rock-chips.com/wiki_Mpp
https://github.com/rockchip-linux/gstreamer-rockchip
https://gstreamer.freedesktop.org/documentation/

4.2 Development guide

RK3288 Linux SDK Kernel version： Linux4.4； Debian version：Debian 9 Stretch.

To help developers get started and familiar with SDK's development and debugging work, we public all development, build and debugging documents through the opensource website , see the following link:

http://opensource.rock-chips.com/wiki_Linux_user_guide

5 SDK Build Introduction

5.1 Build Environment Install

The software packages which is relied on to establish a build environment , their installing command is below:

```
    sudo apt-get install git-core gitk git-gui gcc-arm-linux-gnueabihf u-boot-tools
device-tree-compiler gcc-aarch64-linux-gnu mtools parted libudev-dev
libusb-1.0-0-dev gcc-4.8-multilib-arm-linux-gnueabihf gcc-arm-linux-gnueabihf
libssl-dev gcc-aarch64-linux-gnu
```

5.2 Code Compilation

5.2.1 U-boot Compilation Steps

Execute mk-uboot.sh in the project root directory to get idbloader.img, trust.img, uboot.img, and rk3328_loader_v1.00.238.bin:

RK3328 EVB development board：

```
./build/mk-uboot.sh rk3328-evb
```

The generated files will be copied to out/u-boot which is under the project root directory：

```
yhx@ubuntu:~/RK3328_Debian/out/u-boot$ tree
 ├── idbloader.img
 ├── rk3328_loader_v1.00.238.bin
 ├── trust.img
 └── uboot.img
```

5.2.2 Kernel Compilation Steps

Execute the following command in the project root directory to automatically complete kernel 's compile and package:

RK3328 EVB board：

```
./build/mk-kernel.sh rk3328-evb
```

The generated boot.img after compilation will be copied to the out/ in the project root directory.

```
out/
├── boot.img
├── kernel
    ├── Image
    └── rk3328-evb.dtb
```

Linux SDK needs to pack Image and board level dtb files into boot.img.

5.2.3 Debian Rootfs Compilation

1. Debian system is based on ubuntu-build-service from Linaro

```
sudo apt-get install binfmt-support qemu-user-static live-build
sudo dpkg -i ubuntu-build-service/packages/*
sudo apt-get install -f
ARCH=armhf ./mk-base-debian.sh
```

After compilation, it will generate linaro-stretch-alip-xxxxx-1.tar.gz in rootfs/ dir (xxxxx represents generation timestamp).

2. Building rk-debian rootfs

```
ARCH=armhf ./mk-rootfs.sh
```

3. Create the ext4 image(linaro-rootfs.img)

```
./mk-image.sh
```

Now “rootfs/linaro-rootfs.img” is generated.

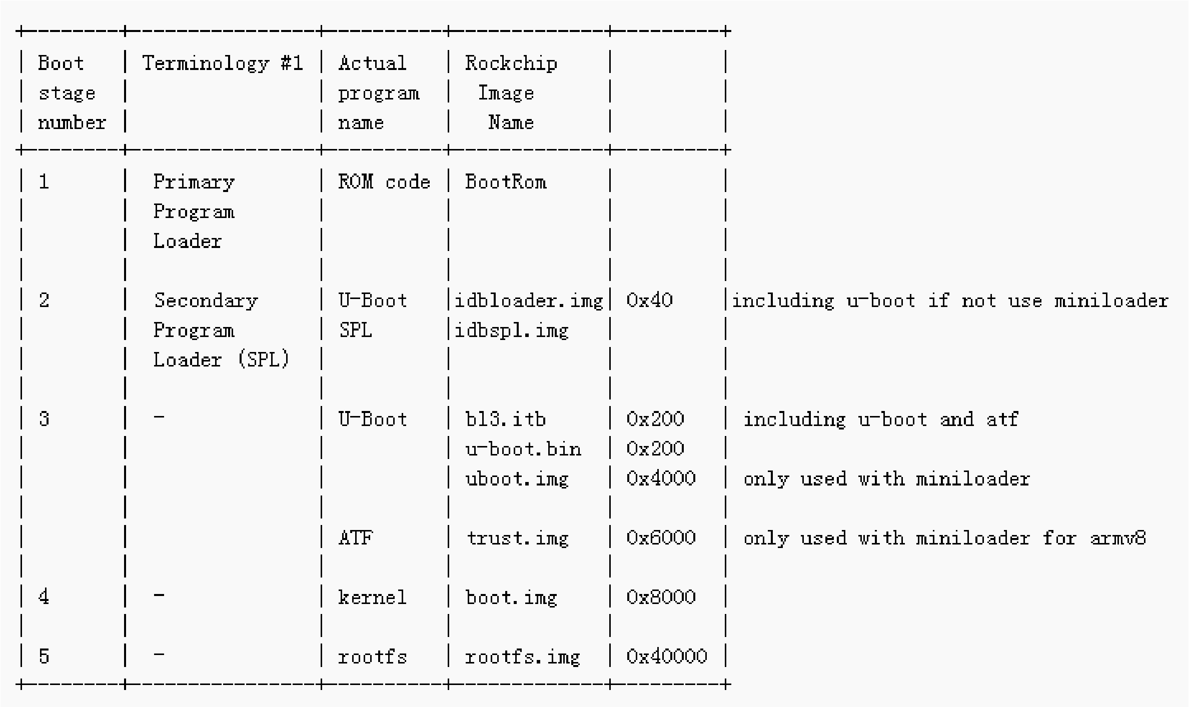
4. linaro-rootfs.img can be downloaded in the following link:

http://opensource.rock-chips.com/wiki_Debian

6 Partition Introduction

For the platform's partition information and the image files' description, see the following link:

http://opensource.rock-chips.com/wiki_Boot_option



```
+--------+----------------+----------+-------------+---------+
| Boot   | Terminology #1 | Actual   | Rockchip    |         |
| stage  |                | program  |  Image      |         |
| number |                | name     |   Name      |         |
+--------+----------------+----------+-------------+---------+
| 1      |  Primary       | ROM code | BootRom     |         |
|        |  Program       |          |             |         |
|        |  Loader        |          |             |         |
|        |                |          |             |         |
| 2      |  Secondary     | U-Boot   |idbloader.img| 0x40    |including u-boot if not use miniloader
|        |  Program       | SPL      |idbspl.img   |         |
|        |  Loader (SPL)  |          |             |         |
|        |                |          |             |         |
| 3      |  -             | U-Boot   | bl3.itb     | 0x200   | including u-boot and atf
|        |                |          | u-boot.bin  | 0x200   |
|        |                |          | uboot.img   | 0x4000  | only used with miniloader
|        |                |          |             |         |
|        |                | ATF      | trust.img   | 0x6000  | only used with miniloader for armv8
|        |                |          |             |         |
| 4      |  -             | kernel   | boot.img    | 0x8000  |
|        |                |          |             |         |
| 5      |  -             | rootfs   | rootfs.img  | 0x40000 |
+--------+----------------+----------+-------------+---------+
```

The platform's partition table file describes the partitioning information, and it's stored in the following path

```
rkbin/tools/
 ├── parameter_gpt.txt
```

You can update gpt partition table by the following commands:

```
rkdeveloptool gpt parameter_gpt.txt
```

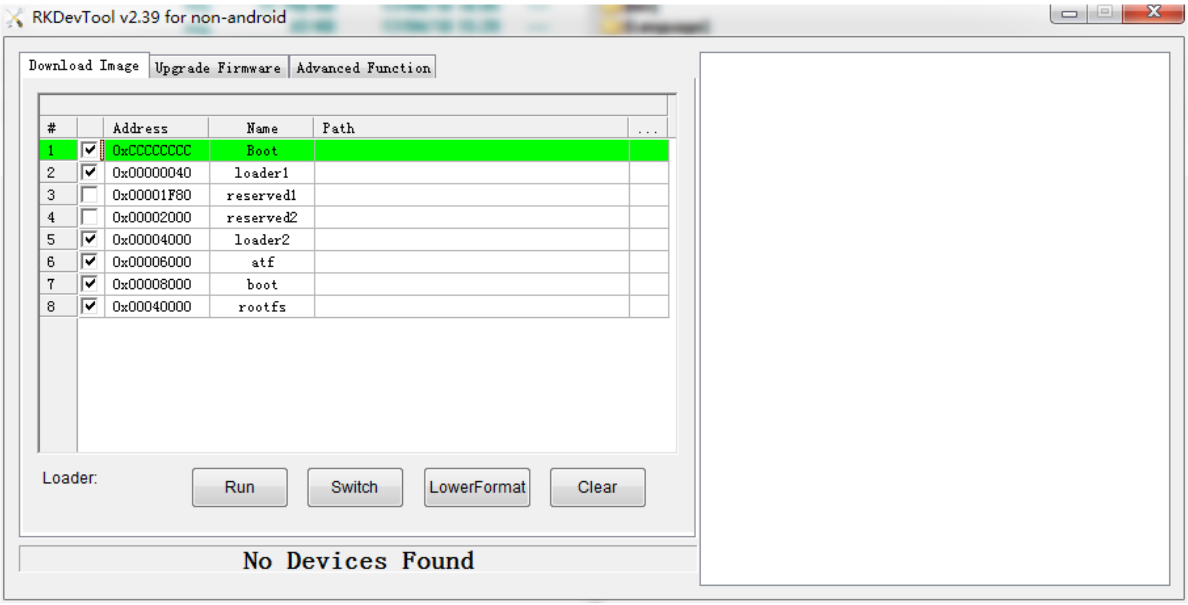
7 Window's Flashing Tool

SDK provides window's flashing tool which is located in the below directory:

```
rkbin/tools/
```

```
├─── AndroidTool_Release_v2.39
```

As shown below, the device needs to enter MASKROM mode for flashing. After connecting to the usb cable, long press the button "Update" and press the reset button "RESET", then release, at last the device will enter the MASKROM mode. Then you should load the paths of the corresponding images and click the "Run" button to start flashing.



RKDevTool v2.39 for non-android

#		Address	Name	Path	...
1	☑	0xCCCCCCCC	Boot		
2	☑	0x00000040	loader1		
3	□	0x00001F80	reserved1		
4	□	0x00002000	reserved2		
5	☑	0x00004000	loader2		
6	☑	0x00006000	atf		
7	☑	0x00008000	boot		
8	☑	0x00040000	rootfs		

No Devices Found

Partition offset and downloading files:

```
Address   Name        Path
          Boot    -- rkbin/rk33/rk3328_loader_v1.00.238.bin
0x40      loader1 -- out/u-boot/idbloader.img
0x4000    loader2 -- out/u-boot/uboot.img
0x6000    atf     -- out/u-boot/trust.img
0x8000    boot    -- out/boot.img
0x40000   rootfs  -- out/linaro.img
```

Please refer to the following URL for more details about the partition information:

```
http://opensource.rock-chips.com/wiki_Partitions
```

Note: Before flashing, you need to install the latest USB driver, which is placed in:

```
rkbin/tools/
├─── DriverAssitant_v4.5
```

After flashing successfully, press the “Enter” key through the serial port to enter command line configuration mode after reboot. Using the following command to write the gpt partition table into the device, and then the system will reboot and load the rootfs.

```
gpt write mmc 0 $partitions
boot
```

8 Linux Flashing Tool

The flashing tool rkdeveloptool is located in rkbin/tools/ directory. Please make sure your developing board is connected to maskrom rockusb. The flashing commands are shown below:

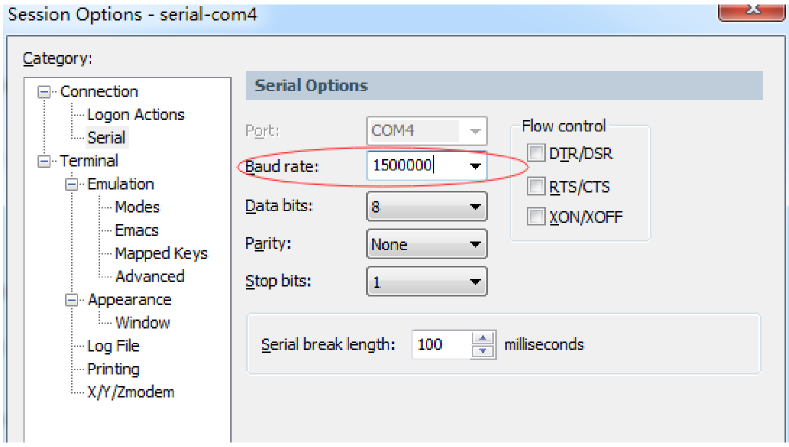
```
rkdeveloptool db rk3328_loader_v1.00.238.bin
rkdeveloptool gpt parameter_gpt.txt
rkdeveloptool wl 0x40 idbloader.img
rkdeveloptool wl 0x4000 uboot.img
rkdeveloptool wl 0x6000 trust.img
rkdeveloptool wl 0x8000 boot.img
rkdeveloptool wl 0x40000 rootfs.img
rkdeveloptool rd
```

After flashing successfully, press the “Enter” key through the serial port to enter the command line configuration mode after reboot. Use the following command to write the gpt partition table into the device, and then the system will reboot and load the rootfs.

```
gpt write mmc 0 $partitions
boot
```

9 Secure CRT Parameter Setting

If you want to use the secure CRT software to print debugging information, you need to set the serial port parameters as below, the baud rate is 1500000:



Session Options - serial-com4

Port: COM4

Baud rate: 1500000

Data bits: 8

Parity: None

Stop bits: 1

Flow control

□ DTR/DSR

□ RTS/CTS

□ XON/XOFF

Serial break length: 100 milliseconds