Rockchip Debian Developer Guide

文件标识：RK-KF-YF-913

发布版本：V1.0.1

日期：2022-03-10

文件密级：□绝密 □秘密 □内部资料 ■公开

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

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

前言

概述

本文档介绍基于Rockchip 的arm平台，如何使用官方Debian发行版来构建和适配相关硬件功能的开发文档。

芯片支持情况

芯片名称	**Debian**版本	**Kernel**版本
RK3588	11	5.10
RK3568	10	4.19
RK3566	10	4.19
RK3399	10	4.4、4.19
RK3399PRO	10	4.4、4.19
RK3326	10	4.4、 4.19
PX30	10	4.4、 4.19
RK3288	10	4.4、4.19
RK3328	10	4.4
RK312X	10	4.4

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
2021-12-30	V1.0.0	Caesar Wang	初始版本
2022-03-10	V1.0.1	Ruby Zhang	文档格式更新

目录

Rockchip Debian Developer Guide

1. Debian介绍
   - 1.1 什么是Debian
   - 1.2 Debian支持的版本
2. Debian快速入门
   - 2.1 环境搭建
   - 2.2 获取源码
   - 2.3 编译
3. Debian目录结构
4. Debian live-build使用指南
   - 4.1 相关命令
   - 4.2 软件源设定
   - 4.3 对系统的软件包进行定制
     - 4.3.1 自定义目录
     - 4.3.2 HOOKS
5. Debian预编包介绍
   - 5.1 mpp
   - 5.2 libv4l
   - 5.3 chromium
   - 5.4 glmark2
   - 5.5 gst-plugins-base1.0
   - 5.6 gst-rkmpp
   - 5.7 libdrm
   - 5.8 libdrm-cursor
   - 5.9 libmali
   - 5.10 rga
   - 5.11 openbox
   - 5.12 pcmanfm
   - 5.13 rkaiq
   - 5.14 rkisp
   - 5.15 rkwifibt
   - 5.16 xserver
6. Debian开发基础能力
   - 6.1 Debian软件包重构
   - 6.2 Debian docker构建
   - 6.3 Debian图形适配方案
     - 6.3.1 显示架构适配方案
     - 6.3.2 窗口管理适配方案
     - 6.3.3 桌面坏境适配方案
     - 6.3.4 Chromium适配方案
     - 6.3.5 Debian Glmark2、Glxgears适配
     - 6.3.6 Debian开机logo/动画适配
     - 6.3.7 Debian Panfrost适配方案
   - 6.4 Debian音视频适配方案
     - 6.4.1 音频Pulseaudio通路适配
     - 6.4.2 MPP、VPU适配
     - 6.4.3 Gstreamer适配
     - 6.4.4 Rockit适配
     - 6.4.5 QT/MPV/QSplayer/Parole播放器适配
   - 6.5 Debian网络适配方案
     - 6.5.1 RKWIFIBT适配
     - 6.5.2 以太网适配
     - 6.5.3 3G/4G/5G 模块适配
     - 6.5.4 网络管理适配
   - 6.6 Debian摄像头适配方案

6.6.1 rkisp适配
6.6.2 rkaiq适配
6.6.3 gstreamer/rockit通路适配
6.6.4 结构光模组适配
6.7 Debian电源管理适配方案
6.7.1 休眠唤醒适配
6.7.2 电池充放电适配
6.7.3 开关机适配
6.7.4 电源键适配
6.8 Debian AI适配方案
6.8.1 RKNPU适配
6.8.2 RKNN 测试Demo适配
6.9 Debian安全升级适配方案
6.9.1 Secureboot安全适配
6.9.2 recovery升级适配
6.9.3 OTA升级适配
6.10 Debian触摸适配方案
6.10.1 触摸屏适配
6.10.2 触摸板适配
6.10.3 鼠标适配
6.11 Debian传感器适配方案
6.12 Debian系统信息
6.13 Debian裁剪
6.14 Debian测试
6.15 Debian调试工具
6.16 Debian性能优化
6.16.1 性能优化
6.16.2 内存优化
6.16.3 开机优化
7. Debian FAQ
7.1 遇到" noexec or nodev"问题
7.2 下载"Base Debian"失败问题
7.3 异常操作导致挂载/dev出错问题
7.4 怎么查看系统相关信息
7.4.1 如何查看系统Debian版本？
7.4.2 如何查看Debian显示用X11还是Wayland？
7.4.3 如何查看系统分区情况
8. Debian 第三方开源软件及许可说明
9. Debian 参考资料

1. Debian介绍

1.1 什么是Debian

Debian 是一种完全自由开放并广泛用于各种设备的 Linux 操作系统。选择Debian原因如下：

- Debian 是自由软件

  Debian 是由自由和开放源代码的软件组成，并将始终保持100%自由。每个人都能自由使用、修改，以及发布。大家可以基于Rockchip构建的Debian系统进行二次开发。

- Debian 是一个基于 Linux稳定且安全的的操作系统。

  Debian 是一个广泛用于各种设备的操作系统，其使用范围包括笔记本计算机，台式机和服务器。自1993年以来，它的稳定性和可靠性就深受用户的喜爱。我们为每个软件包提供合理的默认配置。Debian 开发人员会尽可能在其生命周期内为所有软件包提供安全更新。

- Debian 具有广泛的硬件支持。

  大多数硬件已获得 Linux 内核的支持。当自由软件无法提供足够的支持时，也可使用专用的硬件驱动程序。目前Rockchip RK3588/RK3568/RK3566/RK3399/RK3288等芯片已经适配并支持。

- Debian 提供平滑的更新。

  Debian 以在其发行周期内轻松流畅地进行更新而闻名，不仅如此，还包括轻松升级到下一个大版本。Rockchip目前已从Debian Stretch(9)升级到Debian Buster(10)和 Bullseye(11)版本。

- Debian 是许多其他发行版的种子和基础。

  许多非常受欢迎的 Linux 发行版，例如 Ubuntu、Knoppix、PureOS、SteamOS 以及 Tails，都选择了 Debian 作为它们的软件基础。Debian 提供了所有工具，因此每个人都可以用能满足自己需求的软件包来扩展 Debian 档案库中的软件包。

- Debian 项目是一个社区。

  Debian 不只是一个 Linux 操作系统。该软件由来自世界各地的数百名志愿者共同制作。即使您不是一个程序员或系统管理员，也可以成为 Debian 社区的一员。

1.2 Debian支持的版本

版本	支持架构	计划时间	目前状态
Debian 9 “Stretch”	armhf and arm64	July 6, 2020 to June 30, 2022	不维护
Debian 10 “Buster”	armhf and arm64	July, 2022 to June, 2024	维护
Debian 11 “Bullseye”	armhf and arm64	July, 2024 to June, 2026	正开发

更多Debian长期支持版本时间以官网为主。

2. Debian快速入门

2.1 环境搭建

我们推荐使用 Ubuntu 21.04 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 21.04 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 ncurses-dev \
```

建议使用 Ubuntu21.04 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 获取源码

从瑞芯微代码服务器对外发布SDK中获取源码，位于工程`<SDK>/Debian`目录下。

2.3 编译

进入SDK工程中， 直接编译

```
 ./build.sh debian
```

或进入 debian/ 目录：

```
cd debian/
```

后续的编译和 Debian 固件生成请参考当前目录 readme.md。

(1) Building base Debian system

```
sudo apt-get install binfmt-support qemu-user-static live-build
sudo dpkg -i ubuntu-build-service/packages/*
sudo apt-get install -f
```

3. Debian目录结构

```
debian
├── mk-base-debian.sh ##获取Debian基础包和编译
├── mk-image.sh ##打包生成ext4的固件
├── mk-rootfs-buster/bullseye.sh ##适配Rockchip相关硬件加速包
├── mk-rootfs.sh  ##指向具体Rootfs版本，目前有Buster、Bullseye两个版本。
├── overlay ##适配Rockchip平台共性配置文件
├── overlay-debug ##系统常使用的调试工具
├── overlay-firmware ##一些设备firmware的存放，比如wifibt/dp等
├── packages ## 包含armhf arm64系统适配硬加速使用的预编译的包
├── packages-patches ##预编包，基于官方打上的补丁
├── readme.md ## 文档指引
└── ubuntu-build-service ##从官方获取Debian发行版，可依赖包和定制安装相关包。
```

整个目录结构内容是通过Shell脚本来达到获取构建Linux Debian发行版源码，编译和安装适配Rockchip硬加速包的操作系统。

4. Debian live-build使用指南

live build是一组用于构建实时系统映像的脚本。live build背后的思想是一个工具套件，它使用一个配置目录来完全自动化和定制构建live映像的所有方面。

4.1 相关命令

- lb config

在当前目录下建立auto和config目录和相关配置文件，运行auto/config脚本。

- lb clean

运行auto/clean脚本

- lb build

按照config目录下的各种配置脚本构建系统镜像

4.2 软件源设定

- 方法一

```
$ lb config --mirror-bootstrap http://mirrors.ustc.edu.cn/debian \
--mirror-chroot-security http://mirrors.ustc.edu.cn/debian-security/ \
--mirror-chroot-backports http://mirrors.ustc.edu.cn/debian-backports/
```

chroot mirror：--mirror-chroot, 默认使用—mirror-bootstrap 的值或者创建config/archives/your-repository.list.chroot 文件，内容为源地址。源会被加入到live系统的/etc/apt/sources.list.d/目录。

- 方法二

```
$ lb config --mirror-binary http://mirrors.ustc.edu.cn/debian \
--mirror-binary-security http://mirrors.ustc.edu.cn/debian-security/
```

或者创建config/archives/your-repository.list.binary 文件，内容为源地址。

4.3 对系统的软件包进行定制

- 方法一

将所需的包列表放在customization/package-lists目录下，并命名为XXX.list.chroot或XXX.list.binary即可。

- 方法二

使用--package-lists “XXX”,将使用/usr/share/live/build/package-lists/下的指定包列表。

在执行lb config后会按此脚本中的参数生成config目录下的binary、bootstrap、 chroot、 common四个配置文件。lb build读取这四个配置文件，所以也可以在lb config后可以对这四个文件内的参数做具体修改。

auto/config中为配置参数，例如：

```
set -e

echo "I: create configuration"
export LB_BOOTSTRAP_INCLUDE="apt-transport-https gnupg"
lb config \
 --mirror-bootstrap "http://mirrors.ustc.edu.cn/debian" \
 --mirror-chroot "http://mirrors.ustc.edu.cn/debian" \
 --mirror-chroot-security "http://mirrors.ustc.edu.cn/debian-security" \
 --mirror-binary "http://mirrors.ustc.edu.cn/debian" \
 --mirror-binary-security "http://mirrors.ustc.edu.cn/debian-security" \
 --apt-indices false \
 --apt-recommends false \
 --apt-secure false \
 --architectures arm64 \
 --archive-areas 'main contrib non-free' \
 --backports false \
 --binary-filesystem ext4 \
 --binary-images tar \
 --bootappend-live "hostname=linaro-alip username=linaro" \
 --bootloader "syslinux" \
 --bootstrap-qemu-arch arm64 \
 --bootstrap-qemu-static /usr/bin/qemu-aarch64-static \
 --cache false \
 --chroot-filesystem none \
 --compression gzip \
 --debootstrap-options "--variant=minbase --include=apt-transport-https,gnupg" \
 --distribution bullseye \
 --gzip-options '-9 --rsyncable' \
 --iso-publisher 'Linaro; http://www.linaro.org/; linaro-dev@lists.linaro.org' \
 --iso-volume 'Linaro Bullseye $(date +%Y%m%d-%H:%M)' \
 --linux-flavours none \
```

```
 --linux-packages none \
 --mode debian \
 --security true \
 --system normal \
 --updates true
```

4.3.1 自定义目录

自定义的目录和其中的文件放在config/相应的include目录下即可

config/binary_local-includes（以生成镜像的根目录为根目录）
config/chroot_local-includes（以目标系统的根目录为根目录）

4.3.2 HOOKS

在live-build完成每个阶段的工作后都会运行config/相应hooks中的脚本。
config/binary_local-hooks
config/chroot_local-hooks

live-build新版本会从live/normal两个目录中获取补丁。

```
customization/hooks/live/
├── 0001-setup_user_linaro.binary
├── 0002-add_linaro_to_groups.binary
├── 0003-check_sudoers_for_admin.binary
├── 0021-silence-systemd.binary
├── 0022-disable-systemd-services.binary
├── 0023-lightdm-autologin.binary
└── 0098-resolvconf.binary
```

5. Debian预编包介绍

```
packages
├── armhf/arm64
│   ├── mpp
│   ├── libv4l
│   ├── chromium
│   ├── glmark2
│   ├── gst-plugins-bad1.0
│   ├── gst-plugins-base1.0
│   ├── gst-plugins-good1.0
│   ├── gst-rkmpp
│   ├── libdrm
│   ├── libdrm-cursor
│   ├── libmali
│   ├── openbox
│   ├── pcmanfm
│   ├── rga
│   ├── rkaiq
│   ├── rkisp
```

```
│   ├── rkwifibt
│   └── xserver
```

5.1 mpp

Rockchip提供的媒体处理软件平台(Media Process Platform,简称 MPP)是适用于瑞芯微芯片系列的通用媒体处理软件平台。该平台对应用软件屏蔽了芯片相关的复杂底层处理,其目的是为了屏蔽不同芯片的差异,为使用者提供统一的视频媒体处理接口(Media Process Interface,缩写 MPI)。

MPP提供的功能包括：

- 视频解码

  H.265 / H.264 / H.263 / VP9 / VP8 / MPEG-4 / MPEG-2 / MPEG-1 / VC1 / MJPEG

- 视频编码

  H.264 / VP8 / MJPEG

- 视频处理

  视频拷贝,缩放,色彩空间转换,场视频解交织(Deinterlace)

更多信息参考MPP开发文档。

Debian中集成的预编译包如下：

```
mpp/
├── librockchip-mpp-dev_1.5.0-1_arm64.deb
├── librockchip-mpp1-dbgsym_1.5.0-1_arm64.deb
├── librockchip-mpp1_1.5.0-1_arm64.deb
├── librockchip-vpu0-dbgsym_1.5.0-1_arm64.deb
├── librockchip-vpu0_1.5.0-1_arm64.deb
├── rockchip-mpp-demos-dbgsym_1.5.0-1_arm64.deb
└── rockchip-mpp-demos_1.5.0-1_arm64.deb
```

5.2 libv4l

对接Chromium浏览器和mpp实现硬解的v4l2 plugin。
预编译包：

```
/libv4l/
...
├── libv4l-dev_1.16.3-3_arm64.deb
├── libv4l-rkmpp-dbgsym_1.3.2-1_arm64.deb
├── libv4l-rkmpp_1.3.2-1_arm64.deb
├── libv4l2rds0-dbgsym_1.16.3-3_arm64.deb
├── libv4l2rds0_1.16.3-3_arm64.deb
├── libv4lconvert0-dbgsym_1.16.3-3_arm64.deb
├── libv4lconvert0_1.16.3-3_arm64.deb
├── qv4l2-dbgsym_1.16.3-3_arm64.deb
├── qv4l2_1.16.3-3_arm64.deb
├── v4l-utils-dbgsym_1.16.3-3_arm64.deb
└── v4l-utils_1.16.3-3_arm64.deb
```

5.3 chromium

```
chromium/
├── chromium-x11-dbgsym_91.0.4472.164_arm64.deb
└── chromium-x11_91.0.4472.164_arm64.deb
```

Chromium浏览器上视频支持H264\VP8\VP9等视频格式，但不支持H265. 目前Debian已集成Chromium视频硬解的支持，采用定制chromium+v4l2 plugin+mpp高效硬解。

定制Chromium主要修改如下：

- 修改chromium开启v4l2 vda支持，以及相关补丁
- 添加v4l2 mpp plugin

缺点是：
a、只支持vp8、h264、vp9
b、需要修改编译chromium（调通yocto完整编译流程）
目前我们提供chromium wayland补丁，在yocto、buildroot上支持，有少数客户正在使用

- Chromium版本

root@linaro-alip:~# chromium --version
Chromium 91.0.4472.164 stable

- 测试硬解命令

用如下命令可进行测试：

```
chromium --no-sandbox file:///usr/local/test.mp4
```

具体测试脚本在 /usr/local/bin/test_dec-chromium.sh

```
root@linaro-alip:~# test_dec-chromium.sh
[2588:2588:0214/104846.535688:ERROR:gpu_init.cc(440)] Passthrough is not
supported, GL is egl
...
```

Linux4.19/5.10可以通过如下命令check是否调用硬解

```
echo 0x100 > /sys/module/rk_vcodec/parameters/mpp_dev_debug
```

Linux4.4可以通过如下命令check是否调用硬解

```
echo 0x4 > /sys/module/rk_vcodec/parameters/debug
```

- 如何调试

如果遇到一些问题，开启如下开关获得更多log进行调试。

export mpi_debug=1
export mpp_debug=1
export h264d_debug=1

- 其他

需要更多chromium信息，可以在网址中输入chrome://about获取。

```
List of Chrome URLs
chrome://about
...
chrome://flags
chrome://gcm-internals
chrome://gpu
chrome://help
chrome://histograms
chrome://history
chrome://indexeddb-internals
chrome://inspect
chrome://interstitials
chrome://invalidations
chrome://settings
chrome://version
chrome://webrtc-internals
chrome://webrtc-logs
List of chrome://internals pages
chrome://internals/web-app

For Debug
The following pages are for debugging purposes only. Because they crash or hang
the renderer, they're not linked directly; you can type them into the address bar
if you need them.

chrome://badcastcrash/
chrome://memory-exhaust/
chrome://memory-pressure-critical/
chrome://memory-pressure-moderate/
...
chrome://quit/
chrome://restart/
```

5.4 glmark2

```
glmark2/
├── glmark2-data_2021.02+ds-1_all.deb
├── glmark2-es2-x11-dbgsym_2021.02+ds-1_arm64.deb
└── glmark2-es2-x11_2021.02+ds-1_arm64.deb
```

Glmark2 是开源的对OpenGL 2.0 和 ES 2.0的基准测试程序，一般用来对GPU进行基准测试。

开源代码 Glmark2 具体测试已集成到Debian中

```
usr/local/bin/test_glmark2_*.sh
├── test_glmark2_fullscreen.sh  ### 全屏测试
├── test_glmark2_normal.sh ### 默认显示800x600分辨率测试
├── test_glmark2_offscreen.sh ### 不显示屏幕测试

root@linaro-alip:~# test_glmark2_normal.sh
/usr/local/bin/test_glmark2_normal.sh: line 36: warning: command substitution:
ignored null byte in input
```

```
performance
arm_release_ver of this libmali is 'g2p0-01eac0', rk_so_ver is '4'.
=======================================================
    glmark2 2021.02
=======================================================
    OpenGL Information
    GL_VENDOR:     ARM
    GL_RENDERER:   Mali-G52
    GL_VERSION:    OpenGL ES 3.2 v1.g2p0-01eac0.327c41db9c110a33ae6f67b4cc0581c7
=======================================================
[build] use-vbo=false:
---
```

更多使用方法help查看

```
root@linaro-alip:~# glmark2-es2 --help
A benchmark for Open GL (ES) 2.0

Options:
  -b, --benchmark BENCH  A benchmark or options to run: '(scene)?(:opt1=val1)*'
                         (the option can be used multiple times)
  -f, --benchmark-file F Load benchmarks to run from a file containing a
                         list of benchmark descriptions (one per line)
                         (the option can be used multiple times)
      --validate         Run a quick output validation test instead of
                         running the benchmarks
      --data-path PATH   Path to glmark2 models, shaders and textures
                         Default: /usr/share/glmark2
      --frame-end METHOD How to end a frame [default,none,swap,finish,readpixels]
      --off-screen       Render to an off-screen surface
      --visual-config C  The visual configuration to use for the rendering
                         target: 'red=R:green=G:blue=B:alpha=A:buffer=BUF'.
                         The parameters may be defined in any order, and any
                         omitted parameters assume a default value of '1'
      --reuse-context    Use a single context for all scenes
                         (by default, each scene gets its own context)
  -s, --size WxH         Size of the output window (default: 800x600)
      --fullscreen       Run in fullscreen mode (equivalent to --size -1x-1)
  -l, --list-scenes      Display information about the available scenes
                         and their options
      --show-all-options Show all scene option values used for benchmarks
                         (only explicitly set options are shown by default)
      --run-forever      Run indefinitely, looping from the last benchmark
                         back to the first
      --annotate         Annotate the benchmarks with on-screen information
                         (same as -b :show-fps=true:title=#info#)
  -d, --debug            Display debug messages
  -h, --help             Display help
```

5.5 gst-plugins-base1.0

基于 gstreamer官方的gst-plugins-base 1.14.4和1.18.5版本，增加 dma buffer和rga/gpu图形加速适配的支持。

gst-plugins-base1.0-1.14.4和1.18.5补丁如下:

```
├── 1.14.4
│   ├── 0001-glupload-Only-offer-DMABuf-caps-feature-if-using-EGL.patch
│   ├── 0002-gl-gbm-allow-headless-mode.patch
│   ├── 0003-gst-libs-include-config.h-in-all-source-files.patch
│   ├── 0004-glmemory-Fix-n_wrapped_pointers-usage.patch
│   ├── 0005-gl-egl-Add-gst_egl_image_from_dmabuf_direct-function.patch
│   ├── 0006-glupload-try-to-use-the-last-method-after-reconfigur.patch
│   ├── 0007-glupload-allow-system-memory-for-dmabuf-in-transform.patch
│   ├── 0008-glupload-handle-upload-methods-with-different-caps.patch
│   ├── 0009-gluploadelement-try-to-avoid-dropping-buffers.patch
│   ├── 0010-glupload-Implement-direct-dmabuf-uploader.patch
│   ├── 0011-glupload-calculate-DRM-fourcc-once-for-direct-dmabuf.patch
│   ├── 0012-glupload-debug-output-from-dmabuf-and-dmabuf_direct-.patch
│   ├── 0013-glupload-dmabuf-direct-query-formats-before-modifier.patch
│   ├── 0014-glupload-dmabuf-direct-report-driver-limitations-to-.patch
│   ├── 0015-glupload-Do-prepend-the-preferred-caps.patch
│   ├── 0016-gl-egl-Determine-correct-format-on-dmabuf-import.patch
│   ├── 0017-glupload-dmabuf-be-explicit-about-gl-formats-used.patch
│   ├── 0018-opengl-gbm-Adds-missing-unrefs-for-gl-context-and-dr.patch
│   ├── 0019-gst-gl-Remove-duplicate-declarations.patch
│   ├── 0020-gl-gbm-Improve-logging-output.patch
│   ├── 0021-gl-gbm-Add-GST_GL_GBM_DRM_CONNECTOR-environment-vari.patch
│   ├── 0022-gl-window-gbm-Remove-unneeded-extra-function.patch
│   ├── 0023-gl-window-gbm-Remove-unused-private-class-member.patch
│   ├── 0024-gl-window-gbm-Restore-CRTC-on-close.patch
│   ├── 0025-glupload-dmabuf-use-out_info-to-create-allocation-pa.patch
│   ├── 0026-glupload-Don-t-leak-caps-features.patch
│   ├── 0027-gl-fix-a-few-other-leaks-when-not-getting-to-PAUSED.patch
│   ├── 0028-gl-Don-t-restore-the-viewport-on-function-exit.patch
│   ├── 0029-gluploadelement-Fix-caps-leak.patch
│   ├── 0030-glupload-prevent-segfault-when-updating-caps.patch
│   ├── 0031-glupload-Keep-track-of-cached-EGLImage-texture-forma.patch
│   ├── 0032-eglimage-Fix-memory-leak.patch
│   ├── 0033-glimagesink-fix-upper-left-and-upper-right-rotate-ma.patch
│   ├── 0034-gl-egl-support-direct-dmabuf-import-with-external-oe.patch
│   ├── 0035-gl-gbm-ensure-we-call-the-resize-callback-before-att.patch
│   ├── 0036-glupload-dmabuf-support-direct-upload-into-external-.patch
│   ├── 0037-glupload-fix-transform_caps-NULL-pointer-dereference.patch
│   ├── 0038-glupload-dmabuf-add-DirectDmabufExternal-uploader.patch
│   ├── 0039-glupload-dmabuf-only-accept-uploads-to-external-oes-.patch
│   ├── 0040-video-Add-NV12_10LE40-pixel-format.patch
│   ├── 0041-parsebin-Add-missing-locks-unlocks-of-the-chain-mute.patch
│   ├── 0042-video-Add-gst_video_info_set_interlaced_format.patch
│   ├── 0043-video-format-add-gst_video_format_info_component.patch
│   ├── 0044-video-info-add-gst_video_info_align_full.patch
│   ├── 0045-playbin3-Fix-qt-videoplayer-cannot-change-video-stat.patch
│   ├── 0046-playbin2-Add-preferred-audio-video-sink.patch
│   ├── 0047-HACK-xvimagesink-Support-dma-buffer-rendering.patch
│   ├── 0048-video-converter-Support-rockchip-RGA-2D-accel.patch
```

```
│   ├── 0049-HACK-gl-egl-allow-direct-dmabuf-import-when-unable-t.patch
│   ├── 0050-glupload-dmabuf-prefer-DirectDmabufExternal-uploader.patch
│   ├── 0051-videoconvert-Support-preferred-formats.patch
│   ├── 0052-playbin2-Fix-deadlock-when-hooking-about-to-finish-s.patch
│   ├── 0053-HACK-xvimage-Support-NV12_10-and-NV16-dma-buffer.patch

├── 1.18.5
│   ├── 0001-gst-libs-gst-video-gstvideoaggregator.c-fix-build-wi.patch
│   ├── 0002-playbin3-Fix-qt-videoplayer-cannot-change-video-stat.patch
│   ├── 0003-playbin2-Add-preferred-audio-video-sink.patch
│   ├── 0004-HACK-xvimagesink-Support-dma-buffer-rendering.patch
│   ├── 0005-video-converter-Support-rockchip-RGA-2D-accel.patch
│   ├── 0006-HACK-gl-egl-allow-direct-dmabuf-import-when-unable-t.patch
│   ├── 0007-glupload-dmabuf-prefer-DirectDmabufExternal-uploader.patch
│   ├── 0008-videoconvert-Support-preferred-formats.patch
│   ├── 0009-playbin2-Fix-deadlock-when-hooking-about-to-finish-s.patch
│   ├── 0010-HACK-xvimage-Support-NV12_10-and-NV16-dma-buffer.patch
```

5.6 gst-rkmpp

gstreamer-rockchip是基于gstreamer适配Rockchip平台的音视频编解码中间件，主要对接mpp接口 。

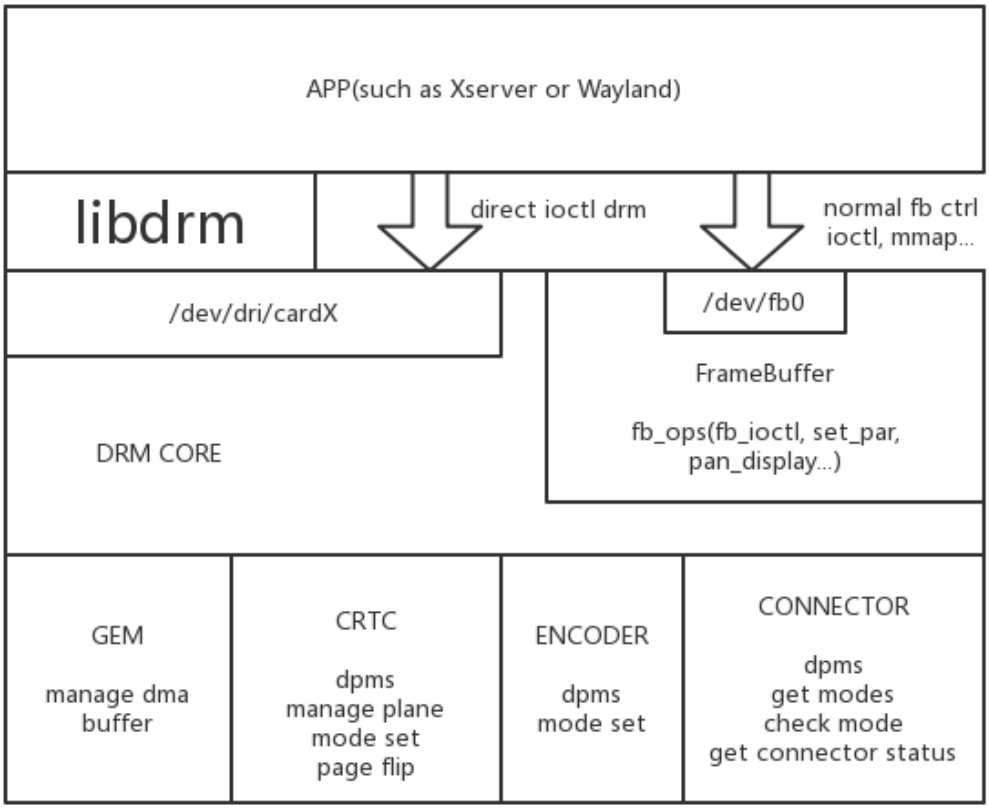
gstreamer-rockchip 预编译包如下：

```
gst-rkmpp/
├── gstreamer1.0-rockchip1-dbgsym_1.14-4_arm64.deb
└── gstreamer1.0-rockchip1_1.14-4_arm64.deb
```

5.7 libdrm

基于官方LIBDRM 2.4.97版本开启kmssink的支持。

LIBDRM是一个跨驱动的中间件,它允许用户空间应用(例如作为Mesa和2D驱动程序)通过DRI与内核通信协议。参考如下DRM结构图:



libdrm 预编译包如下：

```
libdrm
├── libdrm-common_2.4.97-1_all.deb
├── libdrm-dev_2.4.97-1_arm64.deb
├── libdrm2-dbgsym_2.4.97-1_arm64.deb
├── libdrm2_2.4.97-1_arm64.deb
├── libkms1-dbgsym_2.4.97-1_arm64.deb
└── libkms1_2.4.97-1_arm64.deb
```

5.8 libdrm-cursor

这个包主要三个功能：

- vop没有鼠标层，支持把overlay当鼠标层使用
- 支持overlay层AFBC格式的显示
- 限制鼠标超出边界异常的处理

drm-cursor 配置功能如下：

```
cat  /etc/drm-cursor.conf

# Configure file for libdrm-cursor.
#
#debug=
# log-file=
# hide=1 # hide cursors
# atomic=0 # disable atomic drm API
```

```
# max-fps=60
# allow-overlay=1 # allowing overlay planes
# prefer-afbc=0 # prefer plane with AFBC modifier supported
# num-surfaces=8 # num of egl surfaces to avoid edge moving corruption
# prefer-plane=65
# prefer-planes=61,65
# crtc-blocklist=64,83
```

默认 log 在 /var/log/drm-cursor.log

libdrm-cursor 预编译包如下：

```
libdrm-cursor/
├── libdrm-cursor-dbgsym_1.2.3-1_arm64.deb
├── libdrm-cursor-dev_1.2.3-1_arm64.deb
└── libdrm-cursor_1.2.3-1_arm64.deb
```

5.9 libmali

ARM提供的userspace GPU驱动, GPU是提供opengles,egl,opencl API.
Rockchip提供一系列的Mali预编译的deb包。

命名规则: GPU型号-软件版本-硬件版本(如果有的话,比如说r1p0区分RK3288和RK3288w)-编译选项。
要注意编译选项:
不带后缀是x11-gbm,注意GBM是配置DRM使用的memory机制,如果不是3.10的kernel,不要fbdev。GBM是给QT EGLFS程序用的,不依赖X11,Wayland。Wayland/ Wayland-gbm给Wayland使用。

```
libmali/
├── libmali-bifrost-g31-rxp0-x11_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-dummy-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-dummy-gbm-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-dummy-gbm_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-dummy_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-gbm-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-gbm_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-wayland-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-wayland_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-without-cl-dummy-gbm-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-without-cl-dummy-gbm_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-x11-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-g2p0-x11_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-dummy-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-dummy_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-wayland-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-wayland_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-x11-dbgsym_1.9-1_arm64.deb
├── libmali-bifrost-g52-r25p0-x11_1.9-1_arm64.deb
...
```

5.10 rga

Rockchip RGA是一个独立的二维光栅图形加速单元。它加速了二维图形操作,
例如点/线绘制、图像缩放、旋转、位图、图像合成等。

预编包如下：

```
rga/
├── librga-dev_2.1.0-1_arm64.deb
├── librga2-deb
└── librga2_2.1.0-1_arm64.deb
```

5.11 openbox

Openbox是窗口管理器，而不是桌面环境。Openbox只负责维护屏幕上打开的窗口。基于官方v3.6.1版本增加窗口轮廓移动的支持。

```
修改  /home/linaro/.config/openbox/lxde-rc.xml 把
<drawContents>yes</drawContents>  改为 <drawContents>no</drawContents>
```

具体可参考 Openbox

预编包如下：

```
openbox/
└── openbox_3.6.1-8_arm64.deb
```

5.12 pcmanfm

是一款轻量级的文件管理器。基于官方版本1.3.1版本增加轮廓的支持。

预编包如下：

```
pcmanfm/
├── pcmanfm-dbg_1.3.1-1_arm64.deb
└── pcmanfm_1.3.1-1_arm64.deb
```

5.13 rkaiq

全称Rockchip Automatic Image Quality, 自动调整图像信号处理器。主要实现Camera 3A效果,适用于ISP2.x的芯片，比如RK3566、RK3568、RK3588...等。

预编译包如下：

```
rkaiq/
└── camera_engine_rkaiq_v0.1.0_arm64.deb
```

5.14 rkisp

全称Rockchip Image Signal Processor， 图像信号处理器。主要实现Camera 3A效果，适用于ISP1.X的芯片，比如RK3288、RK33999...等。

预编译包如下：

```
rkisp/
└── rkisp-engine-2.2.0_arm64.deb
```

5.15 rkwifibt

基于Rockchip平台调试过WIFI-BT相关模块，里面包含Firmware、工具、配置文件等。

预编译包如下：

```
rkwifibt/
├── rkwifibt-broadcom-firmware_1.0.0-1_arm64.deb
├── rkwifibt-dev-tools-dbgsym_1.0.0-1_arm64.deb
└── rkwifibt-dev-tools_1.0.0-1_arm64.deb
```

5.16 xserver

X server是Linux系统里面图形接口服务器的简称, 常见的Linux界面操作环境有KDE和GNOME，为它们提供系统支持的就是X server。目前Debian使用轻量级的LXDE桌面坏境。Linux目前很多款桌面坏境和窗口管理器对比如下：

桌面环境/窗口管理器	内存使用	CPU使用率	类型
KDE 4.6	363MB	4%	桌面环境
Unity	271MB	14%	桌面环境(shell)
GNOME 3	193MB	10%	桌面环境
GNOME 2.x	191MB	1%	桌面环境
XFCE 4.8	144MB	10%	桌面环境
LXDE	85MB	10%	桌面环境
IceWM	85MB	2%	桌面环境
Enlightenment (E17 Standard)	72MB	1%	窗口管理器
Fluxbox	69MB	1%	窗口管理器
OpenBox	60MB	1%	窗口管理器
JWM	58MB	1%	窗口管理器

Rockchip提供的Xserver有glamor和exa两种加速模式的支持。主要通过 `/etc/X11/xorg.conf.d/20-modesetting.conf` 这个文件进行配置。

具体配置文件说明如下：

```
root@linaro-alip:~# cat /etc/X11/xorg.conf.d/20-modesetting.conf .
Section "Device"
    Identifier  "Rockchip Graphics"
    Driver      "modesetting"

### Use Rockchip RGA 2D HW accel
#    Option      "AccelMethod"    "exa"

### Use GPU HW accel
    Option      "AccelMethod"    "glamor"

    Option      "DRI"            "2"

### Set to "always" to avoid tearing, could lead to up 50% performance loss
    Option      "FlipFB"         "none"

### Limit flip rate and drop frames for "FlipFB" to reduce performance lost
#    Option      "MaxFlipRate"    "25"

    Option      "NoEDID"         "true"
EndSection

Section "Screen"
    Identifier  "Default Screen"
    Device      "Rockchip Graphics"
    Monitor     "Default Monitor"
EndSection

### Valid values for rotation are "normal", "left", "right"
Section "Monitor"
    Identifier  "Default Monitor"
    Option      "Rotate" "normal"
EndSection
```

预编译包如下：

```
Debian10 xserver/
├── xserver-common_1.20.4-1+deb10u3_all.deb
├── xserver-xorg-core_1.20.4-1+deb10u3_arm64.deb
└── xserver-xorg-legacy_1.20.4-1+deb10u3_arm64.deb

Debian11 xserver/
├── xdmx-dbgsym_1.20.11-1_arm64.deb
├── xdmx-tools-dbgsym_1.20.11-1_arm64.deb
├── xdmx-tools_1.20.11-1_arm64.deb
├── xdmx_1.20.11-1_arm64.deb
├── xnest-dbgsym_1.20.11-1_arm64.deb
├── xnest_1.20.11-1_arm64.deb
├── xorg-server-source_1.20.11-1_all.deb
├── xserver-common_1.20.11-1_all.deb
├── xserver-xephyr-dbgsym_1.20.11-1_arm64.deb
├── xserver-xephyr_1.20.11-1_arm64.deb
```

```
├── xserver-xorg-core-dbgsym_1.20.11-1_arm64.deb
├── xserver-xorg-core_1.20.11-1_arm64.deb
├── xserver-xorg-dev_1.20.11-1_arm64.deb
├── xserver-xorg-legacy-dbgsym_1.20.11-1_arm64.deb
├── xserver-xorg-legacy_1.20.11-1_arm64.deb
├── xvfb-dbgsym_1.20.11-1_arm64.deb
├── xvfb_1.20.11-1_arm64.deb
├── xwayland-dbgsym_1.20.11-1_arm64.deb
└── xwayland_1.20.11-1_arm64.deb
```

启动log在 `/var/log/Xorg*` Xserver具体版本，可以通过如下方式确认：

```
root@linaro-alip:~# cat /var/log/Xorg.0.log |grep "X.Org X Server"
X.Org X Server 1.20.4
```

Rockchip修改对应的提交可以通过如下方式确认：

```
root@linaro-alip:~# cat /var/log/Xorg.0.log |grep xorg-server
[    26.786] xorg-server f805fe554 modesetting: Filter out invalid format
modifiers (https://www.debian.org/support)
```

6. Debian开发基础能力

6.1 Debian软件包重构

Debian第三方软件包修改并重新打包步骤如下：

- apt source ##下载源码
- 创建git，打补丁
- apt-get build-dep ##安装编译依赖包
- dpkg-buildpackage -b -uc -us -d ## 编译打包

6.2 Debian docker构建

目前支持通过Docker编译相关源码并打包成deb，便于集成到系统中。

具体参考文档
`<SDK>/docs/Linux/ApplicationNote/Rockchip_Developer_Guide_Debian_Docker_EN.pdf`

6.3 Debian图形适配方案

目前主要支持X11和Wayland的显示架构，主要使用组合如下：

- X11系统默认适配组合：

```
 xfce4/lxde+xserver+lightdm
```

- Wayland系统默认适配组合：

```
gnome + Wayland +gdm3
```

6.3.1 显示架构适配方案

X11/Xserver

Wayland/Weston

6.3.2 窗口管理适配方案

openbox/kwin/xfce

6.3.3 桌面坏境适配方案

lightdm/gnome/kde

6.3.4 Chromium适配方案

6.3.5 Debian Glmark2、Glxgears适配

6.3.6 Debian开机logo/动画适配

6.3.7 Debian Panfrost适配方案

6.4 Debian音视频适配方案

先介绍rockchip平台上视频编解码大概的流程

```
vpu_service  -->  mpp --> gstreamer/rockit/rkmedia/ --> app
vpu_service: 驱动
mpp: rockchip平台的视频编解码中间件,相关说明参考mpp文档
gstreamer/rockit/rkmedia: 对接app的组件
```

目前Debian系统上主要用gstreamer来对接app和编解码组件。

编解码功能，也可以直接通过mpp提供测试接口进行测试 (比如mpi_dec_test\mpi_enc_test...)
mpp源码参考SDK/external/mpp/
测试demo参考:SDK/external/mpp/test
`具体参考SDK文档Rockchip_Developer_Guide_MPP_CN.pdf`

6.4.1 音频**Pulseaudio**通路适配

6.4.2 MPP、**VPU**适配

6.4.3 Gstreamer适配

6.4.4 Rockit适配

6.4.5 QT/MPV/QSplayer/Parole播放器适配

6.5 Debian网络适配方案

6.5.1 RKWIFIBT适配

6.5.2 以太网适配

6.5.3 3G/4G/5G 模块适配

6.5.4 网络管理适配

6.6 Debian摄像头适配方案

6.6.1 rkisp适配

6.6.2 rkaiq适配

6.6.3 gstreamer/rockit通路适配

6.6.4 结构光模组适配

6.7 Debian电源管理适配方案

6.7.1 休眠唤醒适配

6.7.2 电池充放电适配

6.7.3 开关机适配

6.7.4 电源键适配

6.8 Debian AI适配方案

6.8.1 RKNPU适配

6.8.2 RKNN 测试Demo适配

6.9 Debian安全升级适配方案

6.9.1 Secureboot安全适配

6.9.2 recovery升级适配

6.9.3 OTA升级适配

6.10 Debian触摸适配方案

6.10.1 触摸屏适配

6.10.2 触摸板适配

6.10.3 鼠标适配

6.11 Debian传感器适配方案

gsensor/lsensor..

6.12 Debian系统信息

6.13 Debian裁剪

6.14 Debian测试

rockchip_test集成功能、压力、和性能相关测试

6.15 Debian调试工具

6.16 Debian性能优化

6.16.1 性能优化

6.16.2 内存优化

6.16.3 开机优化

7. Debian FAQ

7.1 遇到" noexec or nodev"问题

```
noexec or nodev issue /usr/share/debootstrap/functions: line 1450:
..../rootfs/ubuntu-build-service/buster-desktop-arm64/chroot/test-dev-null:
Permission denied E: Cannot install into target '/rootfs/ubuntu-build-
service/buster-desktop-arm64/chroot' mounted with noexec or nodev
```

解决方法：

```
mount -o remount,exec,dev xxx
（其中xxx 是工程目录路径，然后重新编译）
```

另外如果还有遇到其他编译异常，先排除使用的编译系统是 ext2/ext4 的系统类型。

7.2 下载"Base Debian"失败问题

- 由于编译 Base Debian 需要访问国外网站，而国内网络访问国外网站时，经常出现下载失败的情况:

Debian 使用 live build,镜像源改为国内可以这样配置：

```
32位系统：

+++ b/ubuntu-build-service/{buster/bullseye}-desktop-armhf/configure
@@ -11,6 +11,11 @@ set -e
 echo "I: create configuration"
 export LB_BOOTSTRAP_INCLUDE="apt-transport-https gnupg"
 lb config \
+ --mirror-bootstrap "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-chroot "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-chroot-security "http://mirrors.ustc.edu.cn/debian-security" \
+ --mirror-binary "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-binary-security "http://mirrors.ustc.edu.cn/debian-security" \
```

```
  --apt-indices false \
  --apt-recommends false \
  --apt-secure false \

  64位系统：
  --- a/ubuntu-build-service/{buster/bullseye}-desktop-arm64/configure
+++ b/ubuntu-build-service/{buster/bullseye}-desktop-arm64/configure
@@ -11,6 +11,11 @@ set -e
 echo "I: create configuration"
 export LB_BOOTSTRAP_INCLUDE="apt-transport-https gnupg"
 lb config \
+ --mirror-bootstrap "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-chroot "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-chroot-security "http://mirrors.ustc.edu.cn/debian-security" \
+ --mirror-binary "http://mirrors.ustc.edu.cn/debian" \
+ --mirror-binary-security "http://mirrors.ustc.edu.cn/debian-security" \
  --apt-indices false \
  --apt-recommends false \
  --apt-secure false \
```

如果其他网络原因不能下载包，有预编生成的包分享在[百度云网盘]，放在当前目录直接执行下一步操作。

7.3 异常操作导致挂载/dev出错问题

比如出现这种 askpass command or cannot use one

引起原因可能是编译过程频繁异常操作（CTRL+C），导致上面出错的，可以通过如下方式修复：

```
 sudo -S  umount  /dev
```

7.4 怎么查看系统相关信息

7.4.1 如何查看系统Debian版本？

```
root@linaro-alip:~# cat /etc/debian_version
11.1
```

7.4.2 如何查看Debian显示用X11还是Wayland？

在X11系统上：

```
$ echo $XDG_SESSION_TYPE
x11
```

在Wayland系统上：

```
$ echo $XDG_SESSION_TYPE
wayland
```

7.4.3 如何查看系统分区情况

```
root@linaro-alip:~# parted -l

Model: MMC BJTD4R (sd/mmc)
Disk /dev/mmcblk0: 31.3GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name      Flags
 1      8389kB  12.6MB  4194kB               uboot
 2      12.6MB  16.8MB  4194kB               misc
 3      16.8MB  83.9MB  67.1MB               boot
 4      83.9MB  218MB   134MB                recovery
 5      218MB   252MB   33.6MB               backup
 6      252MB   15.3GB  15.0GB  ext4         rootfs
 7      15.3GB  15.4GB  134MB   ext2         oem
 8      15.6GB  31.3GB  15.6GB  ext2         userdata
```

8. Debian 第三方开源软件及许可说明

Debian 开源相关信息说明，参考官方网站legal

9. Debian 参考资料

参考Debian官方文档debian-docs