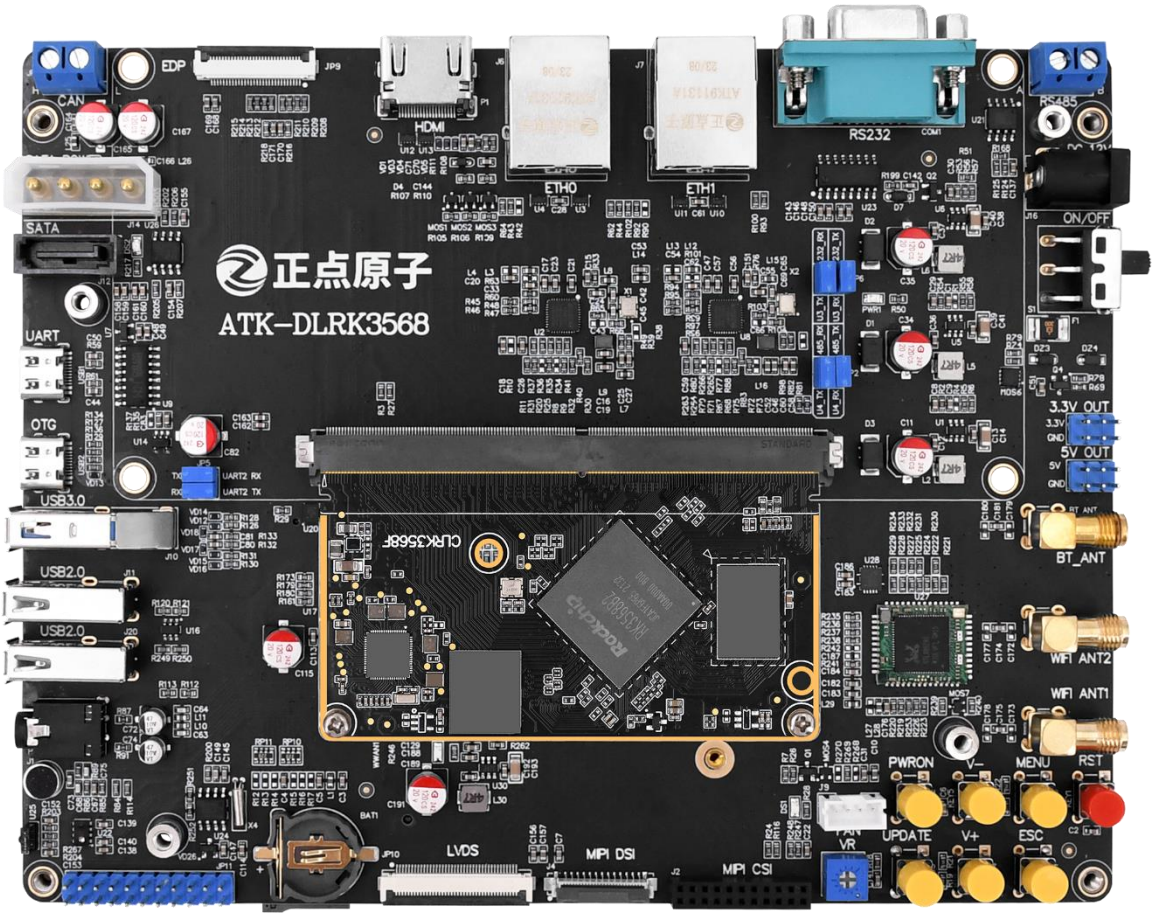
Debian 系统开发手册 V1.0

-正点原子 ATK-DLRK3568



正点原子公司名称 ：广州市星翼电子科技有限公司
原子哥在线教学平台：www.yuanzige.com
开源电子网 / 论坛 ：http:#www.openedv.com/forum.php
正点原子淘宝店铺 ：https:#openedv.taobao.com
正点原子官方网站 ：www.alientek.com
正点原子 B 站视频 ：https:#space.bilibili.com/394620890

电话：020-38271790 传真：020-36773971

请关注正点原子公众号，资料发布更新我们会通知。

请下载原子哥 APP，数千讲视频免费学习，更快更流畅。

扫码关注正点原子公众号

扫码下载“原子哥”APP

文档更新说明

版本	版本更新说明	负责人	校审	发布日期
V1.0	初稿：	正点原子 linux 团队	正点原子 linux 团队	2023.07.03

目录

前言

本文档所使用的环境：

- Windows 10 64bits，（Window11 没有测试，我们都是用 Window 10 开发，如果你使用的 Window11，理论上也适用。）。文档都是以 64 位操作系统做介绍。
- Ubuntu20.04，Ubuntu 务必使用 20.04，过新或者过旧的 Ubuntu 否则安装环境不一样导致出错，请自行解决！
- 要求读者会使用 FileZilla、WinSCP 及 Windows Git 进行 Ubuntu 与 Windows 间互传文件的方法。
- 本文档是针对正点原子 ATK-DLRK3568 Debian 系统进行编写这份文档，若用户使用了其他文件系统，有可能文因件系统没有相应的指令或者模块，导致实验进行不下去的，请注意。

免责声明

为了得到最新版本的产品信息，请用户定时访问正点原子资料下载中心或者与淘宝正点原子旗舰店客服联系索取。感谢您的包容与支持。

第一章 ATK-DLRK3568 开发板使用前准备

在这里我们将介绍在使用开发板需要注意的事项。请大家耐心的看。因为如果不注意这些事项，可能导致您的实验不成功，或者导致开发板损坏情况等，所以务必注意。

1.1 上电前需要注意事项

- 首先检查电源是否插上，电源插上了需要检查，插板开关是打开和是否正常供电。
- 出厂前整个板子是由亚克力板和屏幕保护的，但是某些情况下，用户可能将亚克力板或者屏幕卸下，这时需要注意观察底板不要放至杂物台上，需要在底板下放保护膜，否则容易接触到金属异物，将底板下的某两个触点短路。在使用久了的情况下也要注意观察有没有异物落在开发板的上面，以防短路，注意防水，防潮，防尘等。

1.2 串口软件安装

这里简单的介绍 CH340 USB 串口驱动安装及 MobaXterm 终端的安装，不再详细写安装教程，比较基础。

- 安装 CH340USB 串口驱动（PC 机（电脑）要与开发板的串口通信，我们需要安装此驱动）

在开发板光盘 A-基础资料→4、软件->CH340 驱动(USB 串口驱动)_XP_WIN7 共用 （实际上 Win10 也适用，如果你已经有安装过了，可以不用安装）文件夹下找到 SETUP.EXE，双击运行，然后弹出的窗口，直接点击安装，等待预安装或者安装成功窗口出现即可。如安装失败请看安装失败解决办法文件夹。注意：开发板默认的串口波特率为 1500000（1.5M）。

- 安装 MobaXterm 终端软件（或安装 Xshell，SecureCRT 等终端软件）本文以 MobaXterm 为例。网盘路径：开发板光盘 A-基础资料->3、软件->MobaXterm_Installer_v12.3.zip 。双击打开这个压缩包，等待解压后，直接双击 MobaXterm_installer_12.3.msi 安装即可。MobaXterm 安装程序将会引导您安装。

1.3 开发板与 PC 机连接

开发过程中我们与开发板信息交互，一般是通过串口，当然可以通过网络。网络得先知道开发板的 ip。所以此时串口十分重要。

ATK-DLRK3568 开发板默认配置两根 USB Type-C 连接线，在开发板找到 USB Type-C 座子，找到“UART”丝印字样的座子，然后接上 USB Type-C 连接线，另一头接电脑主机 USB 接口。

1.4 烧写 Debian Linux 系统

ATK-DLRK3568 开发板出厂默认烧写的系统为 Buildroot Linux 系统。你需要烧写 Debian 系统才能到第三章进行测试。烧写 Debian 系统请参考 10、用户手册->03、辅助文档->03【正点原子】ATK-DLRK3568 出厂镜像烧录指导第四章。

1.5 SSH 登录密码

正点原子 ATK-DLRK3568 Debian 系统默认有两个用户，一个是 root 用户，另一个是 linaro 用户。SSH 用户 linaro 登录的密码为 linaro，开机默认登录的用户为 root，默认开机免输入密码。root 用户自动登录默认无密码。

在 Debian 系统中，root 用户默认是不能通过 SSH 登录的。如果您想为 root 用户添加 SSH 密码，可以按照以下步骤操作。在使用 ssh，root 用户作用很大，可以读写任何目录，如果你只使用 linaro 用户，那么只能读写/home/linaro 目录。

编辑/etc/ssh/sshd_config 文件。

将#PermitRootLogin prohibit-password 改为如下。

```
PermitRootLogin yes
```

保存后，然后重启 ssh 服务。

```
systemctl restart sshd
```

因为默认是 root 用户登录的，且无密码，所以我们需要为 root 用户设置一个密码。

输入下面指令修改密码，输入新的密码，按回车，再输入一次按回车即可。密码是不显示的。

```
passwd root
```

```
root@ATK-DLRK3568-Debian:~# passwd root
新的 密码：
重新输入新的 密码：
passwd：已成功更新密码
root@ATK-DLRK3568-Debian:~#
```

1.6 安全登录

默认 Debian 配置是 root 用户登录的且无密码，这样任何一个用户连接到你的串口都可以无限制的修改你的系统。

可以编辑/lib/systemd/system/serial-getty\@.service 文件。

找到以下行

```
ExecStart=-/sbin/agetty --autologin root --noclear %I $TER
#ExecStart=-/sbin/agetty -o '-p -- \\u' --keep-baud 115200,38400,9600 %I $TERM
```

默认是 root 登录，我们修改如下。这样就可以使用 linaro 用户登录，默认密码是 linaro

```
#ExecStart=-/sbin/agetty --autologin root --noclear %I $TER
ExecStart=-/sbin/agetty -o '-p -- \\u' --keep-baud 115200,38400,9600 %I $TERM
```

第二章 构建 Debian Linux 系统

请先熟悉 10、用户手册->02、开发文档->01《【正点原子】ATK-DLRK3568 嵌入式 Linux 系统开发手册》，提前安装 Ubuntu，Ubuntu 需要使用教程指定版本，不要使用自己的以前的 Ubuntu，免得环境错乱导致编译出现错误！提前安装依赖包，如《【正点原子】ATK-DLRK3568 嵌入式 Linux 系统开发手册》第 4.1.1 小节安装依赖包。我们需要按《【正点原子】ATK-DLRK3568 嵌入式 Linux 系统开发手册》第 4.3 小节全自动编译一次，默认是编译 Buildroot 系统，也会编译 uboot 和内核，buildroot 某些软件包依赖内核，所以我们必须编译内核再编译 Buildroot。同理 Debian 也需要从 Buildroot 编译后的产物，拷贝相关软件到 Debian 中，一般是一些驱动模块。所以它们的编译关系是不可分开的。

2.1 Debian 安装 qemu

在 SDK 顶层目录，进入 debian 路径下

```
cd debian/
```

```
alientek@ubuntu:~/rk3568_linux_sdk$ cd debian/
alientek@ubuntu:~/rk3568_linux_sdk/debian$ ls
mk-base-debian.sh    mk-rootfs.sh    overlay-firmware  post-build.sh
mk-image.sh          overlay         packages          readme.md
mk-rootfs-buster.sh  overlay-debug   packages-patches   ubuntu-build-service
alientek@ubuntu:~/rk3568_linux_sdk/debian$
```

执行下面的指令安装 qemu 及需要的一些包。

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

前面两条指令信息太长，下图只截下最后一条指令的信息。

```
alientek@ubuntu:~/rk3568_linux_sdk/debian$ sudo apt-get install -f
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
正在修复依赖关系... 完成
下列软件包将被【卸载】：
  linaro-image-tools python-linaro-image-tools
升级了 0 个软件包，新安装了 0 个软件包，要卸载 2 个软件包，有 385 个软件包未被升级。
有 2 个软件包没有被完全安装或卸载。
解压缩后将会空出 918 kB 的空间。
您希望继续执行吗？ [Y/n] y
(正在读取数据库 ... 系统当前共安装有 183368 个文件和目录。)
正在卸载 linaro-image-tools (2012.12-0ubuntu1~linaro1) ...
正在卸载 python-linaro-image-tools (2012.12-0ubuntu1~linaro1) ...
alientek@ubuntu:~/rk3568_linux_sdk/debian$
```

2.2 构建 Debian 系统

执行下面指令开始构建 Debian 系统。注意，如果你没有先编译 buildroot 和内核，虽然不会报错，完整的系统会从 buildroot 和内核中拷贝一些驱动或者文件到 Debian 系统中，拷贝的文件是 Wifi 驱动和蓝牙驱动。（注意构建的时候 Ubuntu 必须能上网，因为 Debian 会从网上下载构建所需要的资源！）

```
cd ..                 #返回 SDK 顶层目录
./build.sh debian     #开始构建 Debian
```

```
alientek@ubuntu:~/rk3568_linux_sdk/debian$ cd ..
alientek@ubuntu:~/rk3568_linux_sdk$ ./build.sh debian
processing option: debian
=========Start building debian for arm64=========
 Staring Download......
sudo lb clean --purge
[2023-07-07 14:23:18] lb clean --purge
P: Cleaning chroot
rm -f linaro-buster-alip-*
rm -rf config
I: create configuration
[2023-07-07 14:23:18] 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 --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 buster --gzip-options -9 --rsyncable --iso-publisher Linaro; http://www.linaro.org/; linaro-dev@lists.linaro.org --iso-volume Linaro Buster $(date +%Y%m%d-%H:%M) --linux-flavours none --linux-packages none --mode debian --security true --system normal --updates true
```

上图构建过程中，首先会下载 Debian 系统，下载完 Debian 系统后，开始解压 Debian 系统，此时需要用户权限，请输入你的用户密码，提升为 sudo 权限再继续编译。（注意千万不要为了省事直接以 root 用户直接构建！避免出现离奇的错误！）

```
er-alip-`date +%Y%m%d`-1.tar.gz > linaro-buster-alip-`date +%Y%m%d`-1.sha1sums.txt; \
fi
[sudo] password for alientek:
 Building for arm64
 Extract image
 Change root.....................
root@ubuntu:/#
root@ubuntu:/# apt-get update
Hit:1 http://mirrors.ustc.edu.cn/debian buster InRelease
Hit:2 http://mirrors.ustc.edu.cn/debian-security buster/updates InRelease
Hit:3 http://mirrors.ustc.edu.cn/debian buster-updates InRelease
Reading package lists... Done
```

输入你的用户密码

构建成功如下图。

```
Executing post-build.sh...
Adding build-info to /etc/os-release...
Fixing up /etc/fstab...
Fixing up rootfs type: auto
Adding dirs and links...
0+0 records in
0+0 records out
0 bytes copied, 0.000551601 s, 0.0 kB/s
mke2fs 1.45.5 (07-Jan-2020)
Discarding device blocks: done
Creating filesystem with 750592 4k blocks and 187680 inodes
Filesystem UUID: fcb4dbd7-f0c9-4946-b0ff-a512a14a19b3
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Copying files into the device: done
Writing superblocks and filesystem accounting information: done

Rootfs Image: linaro-rootfs.img
Running build_debian succeeded.
Running build_rootfs succeeded.
alientek@ubuntu:~/rk3568_linux_sdk$
```

构建成功，生成的映像文件

生成的 Debian 系统镜像位于 SDK 目录/debian 下，如下图。

```
alientek@ubuntu:~/rk3568_linux_sdk$ cd debian/
alientek@ubuntu:~/rk3568_linux_sdk/debian$ ls
binary                              mk-base-debian.sh    overlay           packages-patches
linaro-buster-alip-20230707-1.tar.gz  mk-image.sh          overlay-debug     post-build.sh
linaro-buster-arm64.tar.gz          mk-rootfs-buster.sh  overlay-firmware  readme.md
linaro-rootfs.img                   mk-rootfs.sh         packages          ubuntu-build-service
alientek@ubuntu:~/rk3568_linux_sdk/debian$
```

2.3 如何使用编译出来的 linaro-rootfs.img

请参考《【正点原子】ATK-DLRK3568 嵌入式 Linux 系统开发手册》第五章 SDK 镜像烧录。因为上面编译出来的只是文件系统，无法生成 update.img，如果你要生成 update.img，按下面的操作。

注意，在编译 debian 之前，你必须编译 buildroot，也就是《【正点原子】ATK-DLRK3568 嵌入式 Linux 系统开发手册》的第 4.3 小节全自动编译一次再做下面的操作，因为生成 update.img 需要依赖 buildroot 里编译后的产物。

```
export RK_ROOTFS_SYSTEM=debian     # 构建系统类型为 debian
./build.sh                         # 执行./build.sh 会生成 update.img
```

```
alientek@ubuntu:~/rk3568_linux_sdk$ export RK_ROOTFS_SYSTEM=debian
alientek@ubuntu:~/rk3568_linux_sdk$ ./build.sh
processing option: allsave
============================================
TARGET_ARCH=arm64
TARGET_PLATFORM=rk356x
TARGET_UBOOT_CONFIG=rk3568
TARGET_SPL_CONFIG=
TARGET_KERNEL_CONFIG=rockchip_linux_defconfig
TARGET_KERNEL_DTS=rk3568-atk-evb1-ddr4-v10-linux
TARGET_TOOLCHAIN_CONFIG=
TARGET_BUILDROOT_CONFIG=rockchip_rk3568
TARGET_RECOVERY_CONFIG=rockchip_rk356x_recovery
TARGET_PCBA_CONFIG=
TARGET_RAMBOOT_CONFIG=
============================================
```

构建的过程中会编译 Uboot、内核等，等等漫长的时间后，最后会将 Uboot、内核和文件系统等打包生成 update.img。

生成的 update.img 在 SDK/rockdev 下。

第三章 ATK-DLRK3568Debian 系统开发

本章将介绍如何测试ATK-DLRK3568开发板在Debian系统下如何使用或者测试部分硬件，实际上 Buildroot 或 Yocto 与 Debian 系统都是使用 Linux 内核，内核向系统提供的接口相同，关于如何测试板子硬件请参考我们另一份文档 10、用户手册->01、测试文档->01【正点原子】ATK-DLRK3568 开发板 Buildroot 系统快速体验手册。本章即向大家演示 Debian 系统开发的重要部分。

3.1 Debian 简介

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 是自由软件并且保持 100%自由。每个人都能自由使用、修改，以及发布。大家都可以基于 Rockchip 构建的 Debian 系统进行二次开发。

Debian 是一个基于 Linux 稳定且安全的操作系统，其使用范围包括笔记本计算机台式机和服务器等。它的稳定性和可靠性就深受用户的喜爱。

Debian 还是许多其他发行版的种子和基础，例如 Ubuntu、Knoppix、PureOS 等，由世界各种的数百名志愿者共同制作。

目前 Rockchip RK3568 等多款芯片已经适配并支持。

3.2 Debian 版本

输入下面的指令查看 Debian 版本，如下可以看到是 Debian10 版本。

```
cat /etc/issue
```

```
root@ATK-DLRK3568-Debian:~# cat /etc/issue
Debian GNU/Linux 10 \n \l

root@ATK-DLRK3568-Debian:~#
```

Debian10 桌面如下。

3.3 如何更新源

所谓的源，就是源头，我们常见的源有阿里云源（mirrors.aliyun.com），中科大源（mirrors.ustc.edu.cn），南京大学（mirror.nju.edu.cn）源，清华大学源（mirrors.tuna.tsinghua.edu.cn），

还有 Debian 官方源（deb.debian.org）等都提供了对 Debian 源的服务。在 Rockchip 中默认使用的是中科大源（mirrors.ustc.edu.cn）。实际上 Rockchip 已经为我们由 Debian 官方源切换为中科大源，只要是国内的源，下载速度都是较快的。我们可以无需再换源。如果我们还是想试试换其他源，可以按以下的方法来更换其它源。

查看默认的源。

```
cat /etc/apt/sources.list
```

```
root@ATK-DLRK3568-Debian:~# cat /etc/apt/sources.list
deb http://mirrors.ustc.edu.cn/debian buster main contrib non-free
deb-src http://mirrors.ustc.edu.cn/debian buster main contrib non-free
deb http://mirrors.ustc.edu.cn/debian-security buster/updates main contrib non-free
deb-src http://mirrors.ustc.edu.cn/debian-security buster/updates main contrib non-free
deb http://mirrors.ustc.edu.cn/debian buster-updates main contrib non-free
deb-src http://mirrors.ustc.edu.cn/debian buster-updates main contrib non-free
root@ATK-DLRK3568-Debian:~#
```

修改为其他源，例如阿里云源，将原来的中科大源地址改为阿里云源，如下，替换完成保存。如需要替换成其它源，请自行尝试，笔者只测试阿里云源。

编辑/etc/apt/sources.list 文件。

```
vi /etc/apt/sources.list
```

替换成如下内容。

```
deb http://mirrors.aliyun.com/debian buster main contrib non-free
deb-src http://mirrors.aliyun.com/debian buster main contrib non-free
deb http://mirrors.aliyun.com/debian-security buster/updates main contrib non-free
deb-src http://mirrors.aliyun.com/debian-security buster/updates main contrib non-free
deb http://mirrors.aliyun.com/debian buster-updates main contrib non-free
deb-src http://mirrors.aliyun.com/debian buster-updates main contrib non-free
```

执行下面指令更新本地仓库，也就是重定向软件包列表。可以看到我们源已经指向阿里云源了。

更新软件源，记得插网线！并且网线可以上网的！

```
apt-get update
```

```
root@ATK-DLRK3568-Debian:~# apt-get update
获取:1 http://mirrors.aliyun.com/debian buster InRelease [122 kB]
获取:2 http://mirrors.aliyun.com/debian-security buster/updates InRelease [34.8 kB]
获取:3 http://mirrors.aliyun.com/debian buster-updates InRelease [56.6 kB]
获取:4 http://mirrors.aliyun.com/debian buster/contrib Sources [42.5 kB]
获取:5 http://mirrors.aliyun.com/debian buster/main Sources [7,852 kB]
获取:6 http://mirrors.aliyun.com/debian buster/non-free Sources [85.9 kB]
获取:7 http://mirrors.aliyun.com/debian buster/main arm64 Packages [7,737 kB]
获取:8 http://mirrors.aliyun.com/debian buster/main Translation-en [5,969 kB]
获取:9 http://mirrors.aliyun.com/debian buster/main Translation-zh [1,526 B]
获取:10 http://mirrors.aliyun.com/debian buster/main Translation-zh_CN [126 kB]
获取:11 http://mirrors.aliyun.com/debian buster/contrib arm64 Packages [38.4 kB]
获取:12 http://mirrors.aliyun.com/debian buster/contrib Translation-en [44.2 kB]
获取:13 http://mirrors.aliyun.com/debian buster/non-free arm64 Packages [53.9 kB]
获取:14 http://mirrors.aliyun.com/debian buster/non-free Translation-en [88.9 kB]
获取:15 http://mirrors.aliyun.com/debian-security buster/updates/main Sources [344 kB]
获取:16 http://mirrors.aliyun.com/debian-security buster/updates/non-free Sources [2,680 B]
获取:17 http://mirrors.aliyun.com/debian-security buster/updates/main arm64 Packages [480 kB]
获取:18 http://mirrors.aliyun.com/debian-security buster/updates/main Translation-en [266 kB]
获取:19 http://mirrors.aliyun.com/debian-security buster/updates/non-free arm64 Packages [3,812 B]
获取:20 http://mirrors.aliyun.com/debian-security buster/updates/non-free Translation-en [23.7 kB]
获取:21 http://mirrors.aliyun.com/debian buster-updates/main Sources [4,616 B]
获取:22 http://mirrors.aliyun.com/debian buster-updates/main arm64 Packages [8,780 B]
获取:23 http://mirrors.aliyun.com/debian buster-updates/main Translation-en [6,915 B]
正在读取软件包列表... 完成
root@ATK-DLRK3568-Debian:~#
```

3.4 查看系统已经安装的软件

```
apt list --installed
```

```
root@ATK-DLRK3568-Debian:~# apt list --installed
正在列表... 完成
acpi-support-base/oldstable,now 0.142-8 all [已安装]
acpid/oldstable,now 1:2.0.31-1 arm64 [已安装，自动]
adduser/oldstable,now 3.118 all [已安装]
adwaita-icon-theme/oldstable,now 3.30.1-1 all [已安装，自动]
alsa-utils/oldstable,now 1.1.8-2 arm64 [已安装]
anacron/oldstable,now 2.3-28 arm64 [已安装]
apt-transport-https/oldstable,oldstable-updates,now 1.8.2.3 all [已安装]
apt-utils/oldstable,oldstable-updates,now 1.8.2.3 arm64 [已安装]
apt/oldstable,oldstable-updates,now 1.8.2.3 arm64 [已安装]
aspell-en/oldstable,now 2018.04.16-0-1 all [已安装，自动]
aspell/oldstable,oldstable,now 0.60.7~20110707-6+deb10u1 arm64 [已安装，自动]
base-files/oldstable,now 10.3+deb10u13 arm64 [已安装]
base-passwd/oldstable,now 3.5.46 arm64 [已安装]
bash/oldstable,now 5.0-4 arm64 [已安装]
binfmt-support/oldstable,now 2.2.0-2 arm64 [已安装]
blueman/oldstable,oldstable,now 2.0.8-1+deb10u1 arm64 [已安装]
bluez-obexd/oldstable,now 5.50-1.2~deb10u3 arm64 [已安装，自动]
bluez/oldstable,now 5.50-1.2~deb10u3 arm64 [已安装]
bsdmainutils/oldstable,now 11.1.2+b1 arm64 [已安装]
bsdutils/oldstable,now 1:2.33.1-0.1 arm64 [已安装]
bubblewrap/oldstable,now 0.3.1-4 arm64 [已安装，自动]
busybox/oldstable,now 1:1.30.1-4 arm64 [已安装]
bzip2/oldstable,now 1.0.6-9.2~deb10u2 arm64 [已安装]
ca-certificates/oldstable,oldstable-updates,now 20200601~deb10u2 all [已安装]
camera-engine-rkaiq/now 2.0x60.1 arm64 [已安装，本地]
can-utils/oldstable,now 2018.02.0-1 arm64 [已安装]
cheese-common/oldstable,now 3.31.90-1 all [已安装，自动]
cheese/oldstable,now 3.31.90-1 arm64 [已安装]
chromium-x11-dbgsym/now 91.0.4472.164 arm64 [已安装，本地]
chromium-x11/now 91.0.4472.164 arm64 [已安装，本地]
cmake-data/oldstable,now 3.13.4-1 all [已安装，自动]
cmake/oldstable,now 3.13.4-1 arm64 [已安装]
console-setup-linux/oldstable,now 1.193~deb10u1 all [已安装，自动]
console-setup/oldstable,now 1.193~deb10u1 all [已安装]
coreutils/oldstable,now 8.30-3 arm64 [已安装]
cpio/oldstable,now 2.12+dfsg-9 arm64 [已安装，自动]
cpp-8/oldstable,now 8.3.0-6 arm64 [已安装，自动]
cpp/oldstable,now 4:8.3.0-1 arm64 [已安装，自动]
cpufrequtils/oldstable,now 008-1.1 arm64 [已安装]
crda/oldstable,now 3.18-1 arm64 [已安装]
curl/oldstable,now 7.64.0-4+deb10u6 arm64 [已安装]
dash/oldstable,now 0.5.10.2-5 arm64 [已安装]
dbus-x11/oldstable,now 1.12.24-0+deb10u1 arm64 [已安装]
dbus/oldstable,now 1.12.24-0+deb10u1 arm64 [已安装，自动]
```

3.5 查看可安装的软件

执行下面的指令查看可安装的软件，若软件已经安装会显示已安装。

```
apt list
```

```
root@ATK-DLRK3568-Debian:~# apt list
正在列表... 完成
0ad-data-common/oldstable 0.0.23.1-1 all
0ad-data/oldstable 0.0.23.1-1 all
0install-core/oldstable 2.12.3-2 arm64
0install/oldstable 2.12.3-2 arm64
0xffff/oldstable 0.8-1 arm64
2048-qt/oldstable 0.1.6-1+b1 arm64
2ping/oldstable 4.3-1 all
2to3/oldstable 3.7.3-1 all
2vcard/oldstable 0.6-1 all
3270-common/oldstable 3.6ga4-3+b1 arm64
389-ds-base-dev/oldstable 1.4.0.21-1+deb10u1 arm64
389-ds-base-legacy-tools/oldstable 1.4.0.21-1+deb10u1 arm64
389-ds-base-libs/oldstable 1.4.0.21-1+deb10u1 arm64
389-ds-base/oldstable 1.4.0.21-1+deb10u1 arm64
389-ds/oldstable 1.4.0.21-1+deb10u1 all
3dchess/oldstable 0.8.1-20 arm64
3depict/oldstable 0.0.21-1 arm64
4digits/oldstable 1.1.4-1+b1 arm64
4g8/oldstable 1.0-3.2 arm64
4pane/oldstable 5.0-2 arm64
4store/oldstable 1.1.6+20151109-2+b3 arm64
4ti2-doc/oldstable 1.6.9+ds-1 all
4ti2/oldstable 1.6.9+ds-1 arm64
6tunnel/oldstable 1:0.12-1 arm64
7kaa-data/oldstable 2.15.1+dfsg-1 all
7kaa/oldstable 2.15.1+dfsg-1 arm64
9base/oldstable 1:6-7+b1 arm64
9menu/oldstable 1.9-2 arm64
9mount/oldstable 1.3+hg20170412-1 arm64
9wm/oldstable 1.4.1-1 arm64
a2jmidid/oldstable 8~dfsg0-3 arm64
a2ps/oldstable 1:4.14-4 arm64
a56/oldstable 1.3+dfsg-9 arm64
a7xpg-data/oldstable 0.11.dfsg1-10 all
aa3d/oldstable 1.0-8+b2 arm64
aac-enc/oldstable 0.1.6-1 arm64
aajm/oldstable 0.4-9+b2 arm64
aaphoto/oldstable 0.45-1 arm64
aapt/oldstable 1:8.1.0+r23-3 arm64
abacas/oldstable 1.3.1-5 all
abcde/oldstable 2.9.3-1 all
abci/oldstable 0.0~git20170124.0.f94ae5e-2+b33 arm64
abcm2ps/oldstable 8.14.2-0.2 arm64
abcmidi/oldstable 20190101-1 arm64
abe-data/oldstable 1.1+dfsg-3 all
abe/oldstable 1.1+dfsg-3 arm64
abgate/oldstable 1.1.9-1 arm64
abi-compliance-checker/oldstable 2.3-0.2 all
abi-dumper/oldstable 1.1-1 all
abi-monitor/oldstable 1.12-2 all
```

3.6 卸载软件

使用 apt remove 来卸载软件。apt 会解决和安装模块的依赖问题,并会咨询软件仓库，但不会安装本地的 deb 文件，apt 是建立在 dpkg 之上的软件管理工具。所以我们一般用 apt 管理软件。

例如我们卸载 vim 软件。首先我们查看 vim 是否已经安装。可以执行下面指令。

```
apt list --installed | grep vim
```

```
root@ATK-DLRK3568-Debian:~# apt list --installed | grep vim

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

vim-common/oldstable,now 2:8.1.0875-5+deb10u4 all [已安装，自动]
vim-runtime/oldstable,now 2:8.1.0875-5+deb10u4 all [已安装，自动]
vim-tiny/oldstable,now 2:8.1.0875-5+deb10u4 arm64 [已安装]
vim/oldstable,now 2:8.1.0875-5+deb10u4 arm64 [已安装]
root@ATK-DLRK3568-Debian:~#
```

上图可以看到 vim 已经安装，那么我们将它卸载。询问是否卸载时，我们输入“y”确认卸载。

```
apt remove vim
```

```
root@ATK-DLRK3568-Debian:~# apt remove vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了：
  liba52-0.7.4 libdca0 libdrm-freedreno1 libdrm-tegra0 vim-runtime
使用'apt autoremove'来卸载它(它们)。
下列软件包将被【卸载】：
  vim
升级了 0 个软件包，新安装了 0 个软件包，要卸载 1 个软件包，有 45 个软件包未被升级。
解压缩后将会空出 2,940 kB 的空间。
您希望继续执行吗？ [Y/n] y
(正在读取数据库 ... 系统当前共安装有 74641 个文件和目录。)
正在卸载 vim (2:8.1.0875-5+deb10u4) ...
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/bin/vi (vi)
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/bin/view (view)
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/bin/ex (ex)
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/bin/editor (editor)
update-alternatives: 使用 /usr/bin/vim.tiny 来在自动模式中提供 /usr/bin/rview (rview)
root@ATK-DLRK3568-Debian:~#
```

3.7 安装软件

在 3.6 小节我们已经卸载了 vim 软件，那么我们这次将它安装，做一次安装软件的演示。同样我们使用 apt 来安装软件，执行下面的指令安装 vim。

```
apt install vim
```

```
root@ATK-DLRK3568-Debian:~# apt install vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了：
  liba52-0.7.4 libdca0 libdrm-freedreno1 libdrm-tegra0
使用'apt autoremove'来卸载它(它们)。
建议安装：
  ctags vim-doc vim-scripts
下列【新】软件包将被安装：
  vim
升级了 0 个软件包，新安装了 1 个软件包，要卸载 0 个软件包，有 45 个软件包未被升级。
需要下载 1,192 kB 的归档。
解压缩后会消耗 2,940 kB 的额外空间。
获取:1 http://mirrors.aliyun.com/debian-security buster/updates/main arm64 vim arm64 2:8.1.0875-5+deb10u4 [1,192 kB]
已下载 1,192 kB，耗时 3秒 (408 kB/s)
正在选中未选择的软件包 vim。
(正在读取数据库 ... 系统当前共安装有 74631 个文件和目录。)
准备解压 .../vim_2%3a8.1.0875-5+deb10u4_arm64.deb  ...
正在解压 vim (2:8.1.0875-5+deb10u4) ...
正在设置 vim (2:8.1.0875-5+deb10u4) ...
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/vim (vim)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/vimdiff (vimdiff)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/rvim (rvim)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/rview (rview)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/vi (vi)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/view (view)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/ex (ex)
update-alternatives: 使用 /usr/bin/vim.basic 来在自动模式中提供 /usr/bin/editor (editor)
root@ATK-DLRK3568-Debian:~#
```

3.8 如何创建自启动程序

本小节将介绍如何在 debian 系统开机启动自己的程序。

3.8.1 /etc/rc.local 方法

通过将自启动程序放到/etc/rc.local 里，开机就会自启动。当然你也可以写到/etc/init.d 下面的某个文件里。

笔者以/etc/rc.local 文件为例。如下图，笔者打印“111111…”，以此观察系统启动时，是否会自动执行。注意我们的程序不能写到 exit0 后面了！

```
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Generate the SSH keys if non-existent

if [ ! -f /etc/ssh/ssh_host_rsa_key ]
then
        # else ssh service start in dpkg-reconfigure will fail
        systemctl stop ssh.socket||true
        dpkg-reconfigure openssh-server
fi
echo "11111111111111111111111111111111111111111111111"

exit 0
"/etc/rc.local" 24L, 568C                                     24,1          底端
```

保存后重启开发板，可以看到启动后系统执行了这句打印！

```
[  OK  ] Reached target Host and Network Name Lookups.
[  OK  ] Started OpenVPN service.
[    5.264327] rc.local[562]: 11111111111111111111111111111111111111111111111
[  OK  ] Started /etc/rc.local Compatibility.
[  OK  ] Started Daemon for power management.
[  OK  ] Started Permit User Sessions.
[  OK  ] Started Serial Getty on ttyFIQ0.
[  OK  ] Started Getty on tty1.
[  OK  ] Reached target Login Prompts.
[    5.306620] rk-pcie 3c0800000.pcie: PCIe Linking... LTSSM is 0x0
[  OK  ] Started LSB: layer 2 tunelling protocol daemon.
[  OK  ] Started Network Time Service.
```

3.8.2 systemd 方法

我们先创建一个要启动的脚本。这个脚本就是打印“22222……”在/root 目录下编辑 auto_run_script.sh。

vi auto_run_script.sh

```
#!/bin/bash
echo “22222222222222222222222222” > /dev/ttyFIQ0
```

```
root@ATK-DLRK3568-Debian:~# vi auto_run_script.sh
#!/bin/bash
echo “22222222222222222222222222” > /dev/ttyFIQ0

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"auto_run_script.sh" 3L, 68C                                  2,48-44       全部
```

赋予脚本可执行权限，执行下面的指令。

```
chmod +x auto_run_script.sh
```

- /lib/systemd/system 是系统范围的目录，这些文件通常是由发行版的维护者创建的。
- /etc/systemd/system：这个目录也包含 systemd 服务单元文件，但是它是用于本地管理员（系统管理员）自定义的服务配置。不会被/lib/systemd/system 服务覆盖。

所以我们需要进入/etc/systemd/system 目录下，创建一个自启动服务 auto_run_script.service。

```
cd /etc/systemd/system
vi auto_run_script.service
```

在 auto_run_script.service 里添加以下内容。

```
[Unit]
Description=Run a Custom Script at Startup
After=network.target
[Service]
ExecStart=/root/auto_run_script.sh
[Install]
WantedBy=default.target
```

然后我们使用 systemctl daemon-reload 重新加载 systemd 守护进程配置，并使用 systemctl enable 建立符号链接关系。

```
systemctl daemon-reload
systemctl enable auto_run_script.service
systemctl start auto_run_script.service                    #马上启动此服务
```

```
root@ATK-DLRK3568-Debian:/etc/systemd/system# systemctl daemon-reload
[  285.643161] systemd-fstab-generator[992]: Ignoring "noauto" for root device
root@ATK-DLRK3568-Debian:/etc/systemd/system# systemctl enable auto_run_script.service
Created symlink /etc/systemd/system/default.target.wants/auto_run_script.service → /etc/systemd/system/auto_run_script.service.
[  292.160152] systemd-fstab-generator[1009]: Ignoring "noauto" for root device
root@ATK-DLRK3568-Debian:/etc/systemd/system# systemctl start auto_run_script.service
[  OK  ] Started Run a Custom Script at Startup.
root@ATK-DLRK3568-Debian:/etc/systemd/system# "22222222222222222222222222"
root@ATK-DLRK3568-Debian:/etc/systemd/system#
```

启动成功

重启后，我们看看打印信息，查看这个服务是否已经启动，看到下图已经启动了这个服务。说明自启动成功。

```
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Network.
         Starting Permit User Sessions...
[  OK  ] Started Run a Custom Script at Startup.
"22222222222222222222222222"
         Starting Network Time Service...
         Starting OpenVPN service...
[  OK  ] Reached target Network is Online.
[  OK  ] Started strongSwan IPsec I…IKEv2 daemon using ipsec.conf.
```

注如需禁止自启动，输入 systemctl disable auto_run_script.service，这个就会取消链接，下次开机时不会自启动。如需要完全移除，删除这个 auto_run_script.service 以及它的脚本。

3.9 音频开发

Debian 系统使用 PulseAudio 控制音量。可以在 Debian 桌面左下角“小飞机”，类似 windows 的开始菜单一样，在影音》PluseAudio 音量控制，弹出界面后自行设置音量即可或者指定到任意声卡设备，系统有什么声卡？看下文。

3.9.1 查看声卡

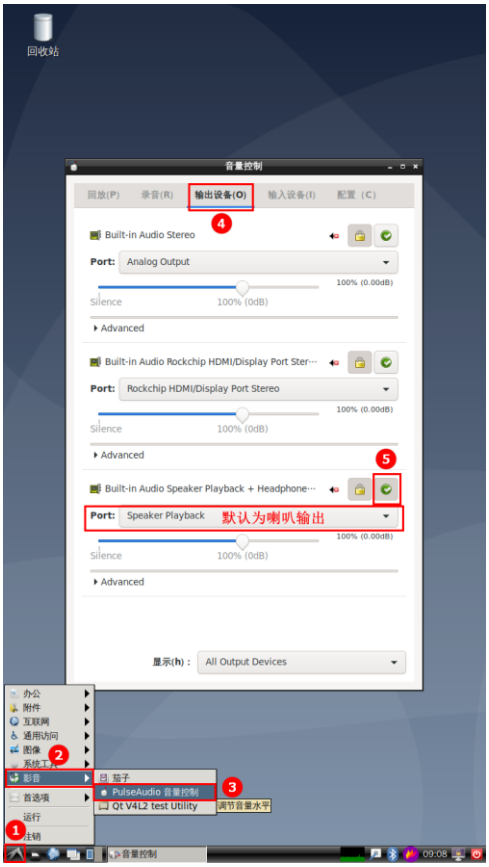
可以使用下面的指令查看系统的声卡。

```
cat /sys/class/sound/card*/id
```

```
root@ATK-DLRK3568-Debian:~#
root@ATK-DLRK3568-Debian:~# cat /sys/class/sound/card*/id
rockchiprk809co
rockchipbt
rockchiphdmi
root@ATK-DLRK3568-Debian:~#
```

如上图，系统有三个声卡，第一个为核心板上 rk809 声卡，第二个为蓝牙音频，第三个为 hdmi 音频。

如果你打开过音量控制界面，我们可以看到默认选上的为下标为 2 的 rk809 音频输出可以指定音频为 headphone(耳机)或者 speaker(喇叭)。默认为 speaker 喇叭输出。可以知道 Debian 的上层也没有做耳机自动切换，需要我们手动设置输出。Pluse 音量控制界面上识别到三个声卡。如下图，第一个为 Analog output 也就是软件模拟输出，就是蓝牙设备。第二个为 hdmi 声卡，第三个就是 rk809 声卡。如有特殊情况，可能是你修改了驱动或者没有此设备时，结果就会与下图的不一样。一切与正点原子默认出厂状态为说明。



3.9.2 修改默认声卡

若需要修改默认的音频输出设备，需要修改系统路径下/etc/pulse/default.pa 文件。找到最后部分，如下图。sink 代表输出设备，source 代表输入设备。可以看到默认为 2，也就是说索引值

为 2 的就是 rk809 声卡设备，根据 pulseaudio 音量控制界面，我们可以知道在输出设备中 hdmi 声卡是索引值为 1。在输入设备中，hdmi 的索引值为 2。

编辑/etc/pulse/default.pa。

```
vi /etc/pulse/default.pa
```

查看默认输出/输入设备。

```
### Make some devices default
set-default-sink 2
set-default-source 3
```

若需要修改为 hdmi 输入输出，那么修改成如下。

```
### Make some devices default
set-default-sink 1
set-default-source 2
```

3.9.3 设置声卡增益

查看 pactl 信息

```
pactl info
```

```
root@ATK-DLRK3568-Debian:~# pactl info
Server String: unix:/tmp/pulse-socket
Library Protocol Version: 34
Server Protocol Version: 34
Is Local: yes
Client Index: 1
Tile Size: 65472
User Name: linaro
Host Name: ATK-DLRK3568-Debian
Server Name: pulseaudio
Server Version: 13.99.3
Default Sample Specification: s16le 2ch 44100Hz
Default Channel Map: front-left,front-right
Default Sink: alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink
Default Source: alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink.monitor
Cookie: b267:bdfe
root@ATK-DLRK3568-Debian:~#
```

设置输出音量。当值为 100%时，即为 0dB，0 增益，输入等输出。那么 150%即约 11dB 增益。为负数时如-50%则是负增益。

```
pactl set-sink-volume alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink 150%
```

```
root@ATK-DLRK3568-Debian:~# pactl set-sink-volume alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink 150%
root@ATK-DLRK3568-Debian:~#
```

设置捕获音量。

```
pactl set-source-volume alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink.monitor 150%
```

```
root@ATK-DLRK3568-Debian:~# pactl set-source-volume alsa_output.platform-rk809-sound.HiFi__hw_rockchiprk809co__sink.monitor 150%
root@ATK-DLRK3568-Debian:~#
```

3.9.4 录音

默认输入设备是 rk809，底板上有个 mic，可以直接录音。执行下面指令进行录音。当然你也可以用 debian 相关录音软件进行录音。本次仅演示命令行方式。

```
arecord -r 44100 -f S16_LE -d 10 record.wav          # 开始录音
```

arecord 命令参数解释：

- -f S16_LE：以 S16_LE 格式采样。
- -r 44100：采样率 44.1K

- -d 10：录音长度 10s
- record.wav：录音存生成的音频文件

播放上面录制的音频文件，备注：生成的文件大小与用户设置的格式及录制的长度有关。

```
aplay record.wav   # 播放录音
```

可以听到底板上的喇叭正在播放录制的音频。

```
root@ATK-DLRK3568-Debian:~# arecord -r 44100 -f S16_LE -d 10 record.wav
Recording WAVE 'record.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Mono
root@ATK-DLRK3568-Debian:~# aplay record.wav
Playing WAVE 'record.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Mono
root@ATK-DLRK3568-Debian:~#
```

3.9.5 播放 mp3

可以用 debian 相关播放器播放 mp3。本次仅演示命令行方式。/oem 目录下存放了 mp3 文件。使用 gst 指令播放，可以听到喇叭播放音乐。

```
gst-play-1.0 /oem/piano2-CoolEdit.mp3
```

```
root@ATK-DLRK3568-Debian:/# gst-play-1.0 ./oem/piano2-CoolEdit.mp3
按“k”键来显示键盘快捷键列表。
正在播放 /oem/piano2-CoolEdit.mp3
Redistribute latency...
0:00:02.1 / 0:00:06.3
root@ATK-DLRK3568-Debian:/#
```

3.10 视频开发

可以用 debian 相关播放器播放视频。本次仅演示命令行方式。/usr/local/目录下存放了 mp4 文件，使用 gst 指令播放。

```
gst-play-1.0 /usr/local/test.mp4
```

注意播放时会报错误，但是不影响播放，gst 自动使用合适的 sink 进行输出播放。因为 debian 适配时可能是使用 buildroot 那套，buildroot 使用了 wayland 显示框架，默认 debian 是使用 xfce4/lxde+xserver+lightdm 框架，不使用 wayland。如果我们不配置环境变量的话从指令看到默认使用的是 wayland sink 这应该是 rk 适配 debian 留下的小问题。我们应该可以通过环境变量指定默认的 sink 即可解决，这里笔者不和大家讲解了，可以自己去研究。

```
root@ATK-DLRK3568-Debian:~# gst-play-1.0 /usr/local/test.mp4
按“k”键来显示键盘快捷键列表。
正在播放 /usr/local/test.mp4
WARNING Could not initialise Wayland output
WARNING debug information: gstwaylandsink.c(502): gst_wayland_sink_find_display (): /GstWaylandSink:waylandsink0:
Failed to create GstWlDisplay: 'Failed to connect to the wayland display '(default)''
Redistribute latency...
0:00:09.9 / 0:00:10.0
到达播放列表结尾。

root@ATK-DLRK3568-Debian:~#
```

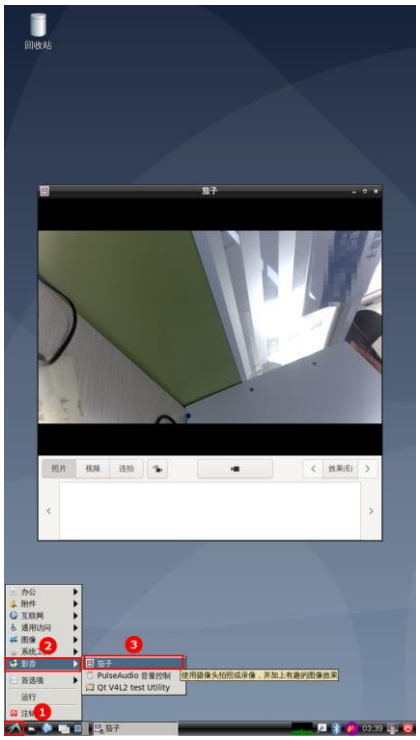
3.11 摄像头开发

在 debian 系统中，已经有自带的软件可以打开摄像头，如茄子软件。为了能够自主开发摄像头，本小节将使用 python 简单进行摄像头开发，包括环境搭建，代码示例等。

3.11.1 图形界面开启摄像头

你可以使用 Debian 桌面上开始菜单上影音》茄子（cheese）软件来打开摄像头，这里笔者只讲我们正点原子店铺售卖的 imx415(800W)4K 和 imx335(500W)2K 摄像头。这两摄像头分辨

率高，效果好，并且 rk 提供了 ISP 开发，注 USB 摄像头不支持 ISP 开发，因为 USB 摄像头不用经过 ISP 出图，且一般的 USB 摄像头的分辨率较低。下图为 imx415 摄像头的显示效果。



3.11.2 搭建 opencv-python 开发环境

更新软件源，记得插网线！并且网线可以上网的！

```
apt update
```

```
root@ATK-DLRK3568-Debian:~#
root@ATK-DLRK3568-Debian:~# apt update
命中:1 http://mirrors.aliyun.com/debian buster InRelease
命中:2 http://mirrors.aliyun.com/debian-security buster/updates InRelease
命中:3 http://mirrors.aliyun.com/debian buster-updates InRelease
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
有 45 个软件包可以升级。请执行 'apt list --upgradable' 来查看它们。
root@ATK-DLRK3568-Debian:~#
```

首先我们需要安装 pyhotn3-pip

```
apt-get install python3-pip
```

```
root@ATK-DLRK3568-Debian:~# apt-get install python3-pip
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了：
  liba52-0.7.4 libdca0 libdrm-freedreno1 libdrm-tegra0
使用'apt autoremove'来卸载它(它们)。
将会同时安装下列软件：
  python-pip-whl
推荐安装：
  build-essential python3-dev python3-setuptools python3-wheel
下列【新】软件包将被安装：
  python-pip-whl python3-pip
升级了 0 个软件包，新安装了 2 个软件包，要卸载 0 个软件包，有 45 个软件包未被升级。
需要下载 1,761 kB 的归档。
解压缩后会消耗 2,686 kB 的额外空间。
您希望继续执行吗？ [Y/n] y
获取:1 http://mirrors.aliyun.com/debian buster/main arm64 python-pip-whl all 18.1-5 [1,591 kB]
获取:2 http://mirrors.aliyun.com/debian buster/main arm64 python3-pip all 18.1-5 [171 kB]
已下载 1,761 kB，耗时 5秒 (339 kB/s)
正在选中未选择的软件包 python-pip-whl。
(正在读取数据库 ... 系统当前共安装有 74641 个文件和目录。)
准备解压 .../python-pip-whl_18.1-5_all.deb  ...
正在解压 python-pip-whl (18.1-5) ...
正在选中未选择的软件包 python3-pip。
准备解压 .../python3-pip_18.1-5_all.deb  ...
正在解压 python3-pip (18.1-5) ...
正在设置 python-pip-whl (18.1-5) ...
正在设置 python3-pip (18.1-5) ...
root@ATK-DLRK3568-Debian:~#
```

安装 pip，使用 pip3 更新。更新前会卸载 pip，由于没有装 pip，所以不用卸载。

```
pip3 install --upgrade pip
```

```
root@ATK-DLRK3568-Debian:~# pip3 install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/08/e3/57d4c24a050aa0bcca46b2920bff40847db79535dc78141eb83581a52eb8/pip-23.1.2-py3-none-any.whl (2.1MB)
    100% |████████████████████████████████| 2.1MB 137kB/s
Installing collected packages: pip
  Found existing installation: pip 18.1
    Not uninstalling pip at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'pip'. No files were found to uninstall.
Successfully installed pip-23.1.2
root@ATK-DLRK3568-Debian:~#
```

使用 pip 安装软件包时，默认从 https://pypi.python.org/simple/ 检索并获取包，容易受网络影响，下载速度比较慢且容易导致网络连接超时，所以我们可以将源换为国内的，pip 常用的国内源有：

```
https://pypi.tuna.tsinghua.edu.cn/simple/    /* 清华大学源 */
https://mirrors.aliyun.com/pypi/simple/      /* 阿里云 */
https://pypi.douban.com/simple/              /* 豆瓣源 */
https://pypi.mirrors.ustc.edu.cn/simple/     /* 中国科学技术大学源 */
```

笔者现在处于 root 用户，所以在 root 用户目录下创建一个.pip 目录，（默认笔者已经提前创建了 pip.conf 放到/root 目录下，你可不用亲自编辑，可直接查看有没有.pip/pip.conf 即可）

```
mkdir .pip          /* 在用户家目录（/root）创建.pip 目录 */
vi .pip/pip.conf    /* 在.pip 目录下新建 pip.conf 文件 */
```

pip.conf 配置文件的内容可修改如下，添加了一个 URL 地址，编辑后保存。

```
[global]
index-url = http://pypi.douban.com/simple
[install]
use-mirrors =true
mirrors =http://pypi.douban.com/simple/
trusted-host =pypi.douban.com
```

```
root@ATK-DLRK3568-Debian:~# pwd
/root
root@ATK-DLRK3568-Debian:~# mkdir .pip
root@ATK-DLRK3568-Debian:~# vi .pip/pip.conf
[global]
index-url = http://pypi.douban.com/simple
[install]
use-mirrors =true
mirrors =http://pypi.douban.com/simple/
trusted-host =pypi.douban.com

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
".pip/pip.conf" [新] 7L, 150C 已写入
root@ATK-DLRK3568-Debian:~#
```

可以看到 root 用户目录下有.pip 隐藏文件夹。

```
root@ATK-DLRK3568-Debian:~# pwd
/root
root@ATK-DLRK3568-Debian:~# ls -a
.   .bash_history  .cache   .gnupg  .profile    .viminfo
..  .bashrc        .config  .pip    record.wav
root@ATK-DLRK3568-Debian:~#
```

接着安装 opencv-python，安装 opencv 的时候，pip 还会解决依赖，同时安装了 numpy，下载源的地方也从 pypi.doubanio.com 处下载。

```
python3 -m pip install opencv-python
```

```
root@ATK-DLRK3568-Debian:~# python3 -m pip install opencv-python
Looking in indexes: http://pypi.douban.com/simple
Collecting opencv-python
  Downloading http://pypi.doubanio.com/packages/9e/b9/0d479c9ce987ba3c894c08785a4a3c38651e8cb14800fbe788d3ed8d3c11/opencv_python-4.7.0.72-cp37-abi3-manylinux_2_17_aarch64.manylinux2014_aarch64.whl (40.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.4/40.4 MB 9.1 MB/s eta 0:00:00
Collecting numpy>=1.19.3 (from opencv-python)
  Downloading http://pypi.doubanio.com/packages/b7/0d/86662f93102e42545cdf031da4fddf0ace9030ec67478932a628afc5973b/numpy-1.21.6-cp37-cp37m-manylinux_2_17_aarch64.manylinux2014_aarch64.whl (13.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.0/13.0 MB 9.0 MB/s eta 0:00:00
Installing collected packages: numpy, opencv-python
Successfully installed numpy-1.21.6 opencv-python-4.7.0.72
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
root@ATK-DLRK3568-Debian:~#
```

3.11.3 查看具有捕获能力的摄像头节点

在 MIPI CSI 接口处插上 imx415 或者 imx335 摄像头，注意是不可热插拔的，开发板上电前插好，注意方向，镜头不朝向核心板。

执行下面指令查看摄像头支持采集的节点，可以看到 video0 和 video1 是支持采集的，并且可以同时采集。其他 video2、video3…不支持采集，有特殊功能，这里笔者不详解。

```
media-ctl -p -d /dev/media0 | grep -E -A 2    'rkisp_mainpath|rkisp_selfpath'
```

```
root@ATK-DLRK3568-Debian:~# media-ctl -p -d /dev/media0 | grep -E -A 2  'rkisp_mainpath|rkisp_selfpath'
                -> "rkisp_mainpath":0 [ENABLED]
                -> "rkisp_selfpath":0 [ENABLED]
        pad3: Source
                -> "rkisp-statistics":0 [ENABLED]
--
- entity 13: rkisp_mainpath (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video0
--
- entity 19: rkisp_selfpath (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video1
root@ATK-DLRK3568-Debian:~#
```

3.11.4 查看节点采集格式

查看 video0 支持采集的格式，以 video0 为例。

```
v4l2-ctl -d /dev/video0 --list-formats
```

```
root@ATK-DLRK3568-Debian:~# v4l2-ctl -d /dev/video0 --list-formats
ioctl: VIDIOC_ENUM_FMT
        Type: Video Capture Multiplanar

        [0]: 'UYVY' (UYVY 4:2:2)
        [1]: '422P' (Planar YUV 4:2:2)
        [2]: 'NV16' (Y/CbCr 4:2:2)
        [3]: 'NV61' (Y/CrCb 4:2:2)
        [4]: 'YM16' (Planar YUV 4:2:2 (N-C))
        [5]: 'NV21' (Y/CrCb 4:2:0)
        [6]: 'NV12' (Y/CbCr 4:2:0)
        [7]: 'NM21' (Y/CrCb 4:2:0 (N-C))
        [8]: 'NM12' (Y/CbCr 4:2:0 (N-C))
        [9]: 'YU12' (Planar YUV 4:2:0)
        [10]: 'YM24' (Planar YUV 4:4:4 (N-C))
        [11]: 'RGGB' (8-bit Bayer RGRG/GBGB)
        [12]: 'GRBG' (8-bit Bayer GRGR/BGBG)
        [13]: 'GBRG' (8-bit Bayer GBGB/RGRG)
        [14]: 'BA81' (8-bit Bayer BGBG/GRGR)
        [15]: 'RG10' (10-bit Bayer RGRG/GBGB)
        [16]: 'BA10' (10-bit Bayer GRGR/BGBG)
        [17]: 'GB10' (10-bit Bayer GBGB/RGRG)
        [18]: 'BG10' (10-bit Bayer BGBG/GRGR)
        [19]: 'RG12' (12-bit Bayer RGRG/GBGB)
        [20]: 'BA12' (12-bit Bayer GRGR/BGBG)
        [21]: 'GB12' (12-bit Bayer GBGB/RGRG)
        [22]: 'BG12' (12-bit Bayer BGBG/GRGR)
root@ATK-DLRK3568-Debian:~#
```

3.11.5 查看节点支持的分辨率

查看 video0 支持的分辨率，video0 是 MP(rkisp_mainpath)节点，video1 是 SP(rkisp_selfpath)节点，其中 SP 节点最大支持 1920x1080 采集，MP 节点最大支持 senor 支持的最高输出分辨率，简单的来说就是你的摄像头支持最大采集像素多大，它就支持多大。下图为 imx415 摄像头支持 4K(3840x2160)的查询结果。

```
v4l2-ctl -d /dev/video0 --list-formats-ext
```

```
root@ATK-DLRK3568-Debian:~# v4l2-ctl -d /dev/video0 --list-formats-ext
ioctl: VIDIOC_ENUM_FMT
        Type: Video Capture Multiplanar

        [0]: 'UYVY' (UYVY 4:2:2)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [1]: '422P' (Planar YUV 4:2:2)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [2]: 'NV16' (Y/CbCr 4:2:2)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [3]: 'NV61' (Y/CrCb 4:2:2)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [4]: 'YM16' (Planar YUV 4:2:2 (N-C))
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [5]: 'NV21' (Y/CrCb 4:2:0)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [6]: 'NV12' (Y/CbCr 4:2:0)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [7]: 'NM21' (Y/CrCb 4:2:0 (N-C))
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [8]: 'NM12' (Y/CbCr 4:2:0 (N-C))
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [9]: 'YU12' (Planar YUV 4:2:0)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [10]: 'YM24' (Planar YUV 4:4:4 (N-C))
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [11]: 'RGGB' (8-bit Bayer RGRG/GBGB)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [12]: 'GRBG' (8-bit Bayer GRGR/BGBG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [13]: 'GBRG' (8-bit Bayer GBGB/RGRG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [14]: 'BA81' (8-bit Bayer BGBG/GRGR)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [15]: 'RG10' (10-bit Bayer RGRG/GBGB)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [16]: 'BA10' (10-bit Bayer GRGR/BGBG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [17]: 'GB10' (10-bit Bayer GBGB/RGRG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [18]: 'BG10' (10-bit Bayer BGBG/GRGR)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [19]: 'RG12' (12-bit Bayer RGRG/GBGB)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [20]: 'BA12' (12-bit Bayer GRGR/BGBG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [21]: 'GB12' (12-bit Bayer GBGB/RGRG)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
        [22]: 'BG12' (12-bit Bayer BGBG/GRGR)
                Size: Stepwise 32x16 - 3840x2160 with step 8/8
root@ATK-DLRK3568-Debian:~#
```

3.11.6 运行 python 摄像头代码

python 摄像头测试源码路径位于/root/camera_cap.py。内容如下。

camera_cap.py 的内容如下：

```
import cv2
import time

# 从 3.11.3 小节中获取摄像头节点
cap = cv2.VideoCapture(1)
# 指定格式采集，如不指定默认为 YU12
cap.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc('Y', 'U', '1', '2'))
time.sleep(2.0)                    # 延时 2 秒，等待相机启动
if(cap.isOpened()):
    cap.set(5, 30)                 # 设置帧率为 30fps
    cap.set(3, 640)                # 设置帧的宽度为 640
    cap.set(4, 480)                # 设置帧的高度为 480
    cap.set(10, 50)                # 设置帧的亮度为 50（可根据实际情况调整）
    fps = cap.get(5)               # 获取帧率
    w = cap.get(3)                 # 获取帧的宽度
```

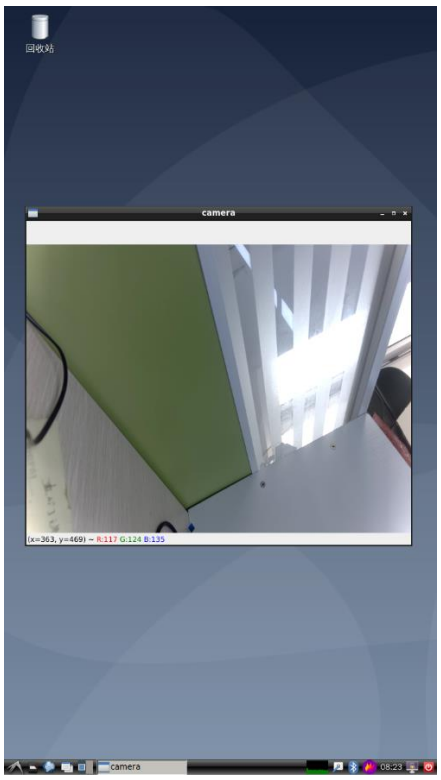
```
        h = cap.get(4)                  # 获取帧的高度
        # 打印初始帧信息
        print("fps=", fps, "weight=", w, "height=", h)
    while(cap.isOpened()):
        ret, frame = cap.read()         # 从摄像头捕获视频
        if not ret:                         # 如果未捕获成功，则 break
            break
        cv2.imshow("camera", frame)   # 显示帧
        key = cv2.waitKey(1) & 0xFF   # 参数不能设置为 0，否则无法显示视频
        # 如果开发板接了键盘，选中图像后再按下键盘的 q 键即可退出程序
        if key == ord("q"):
            break
    cap.release()                           # 释放摄像头
```

如下图，在/root 目录，使用 python3 执行，停止按“Ctrl +c”。

```
python3 camera_cap.py
```

```
root@ATK-DLRK3568-Debian:~# pwd
/root
root@ATK-DLRK3568-Debian:~# ls
camera_cap.py
root@ATK-DLRK3568-Debian:~# python3 camera_cap.py
[ 2122.797156] rkisp-vir0: nonsupport pixelformat:BGR3
[ 2122.797267] rkisp-vir0: nonsupport pixelformat:RGB3
[ 2122.797279] rkisp-vir0: nonsupport pixelformat:YV12
[ WARN:0@2.072] global cap_v4l.cpp:2060 getProperty VIDEOIO(V4L2:/dev/video1): Unable to get camera FPS
fps= -1.0 weight= 640.0 height= 480.0
[ 2124.877531] rkisp rkisp-vir0: first params buf queue
[ 2124.879153] rockchip-csi2-dphy0: dphy0, data_rate_mbps 892
[ 2124.879205] rockchip-csi2-dphy csi2-dphy0: csi2_dphy_s_stream stream on:1, dphy0
[ 2125.020307] rkisp-vir0: tx stream:4 lose frame:0, isp state:0x201 frame:158
```

Debain 桌面上显示的画面如下图。



在上面的 python 代码中，我们使用的是 video1 节点，你也可以改为使用 video0，将这段 cap = cv2.VideoCapture(1)改为 cap = cv2.VideoCapture(0)即可。

查看 video1 节点的所有有效信息，包括摄像头正在使用的视频格式和分辨率等。

```
v4l2-ctl -d /dev/video1 --all
```

```
root@ATK-DLRK3568-Debian:~# v4l2-ctl -d /dev/video1 --all
Driver Info:
        Driver name      : rkisp_v5
        Card type        : rkisp_selfpath
        Bus info         : platform:rkisp-vir0
        Driver version   : 1.8.0
        Capabilities     : 0x84201000
                Video Capture Multiplanar
                Streaming
                Extended Pix Format
                Device Capabilities
        Device Caps      : 0x04201000
                Video Capture Multiplanar
                Streaming
                Extended Pix Format
Media Driver Info:
        Driver name      : rkisp-vir0
        Model            : rkisp0
        Serial           :
        Bus info         :
        Media version    : 4.19.255
        Hardware revision: 0x00000000 (0)
        Driver version   : 4.19.255
Interface Info:
        ID               : 0x03000014
        Type             : V4L Video
Entity Info:
        ID               : 0x00000013 (19)
        Name             : rkisp_selfpath
        Function         : V4L2 I/O
        Pad 0x01000016   : 0: Sink
          Link 0x02000017: from remote pad 0x1000004 of entity 'rkisp-isp-subdev': Data, Enabled
Priority: 2
Format Video Capture Multiplanar:
        Width/Height      : 640/480
        Pixel Format      : 'YU12' (Planar YUV 4:2:0)
        Field             : None
        Number of planes  : 1
        Flags             :
        Colorspace        : Default
        Transfer Function : Default
        YCbCr/HSV Encoding: Default
        Quantization      : Full Range
        Plane 0           :
           Bytes per Line : 640
           Size Image     : 460800
Crop: Left 0, Top 0, Width 3840, Height 2160
Selection: crop, Left 0, Top 0, Width 3840, Height 2160, Flags:
Selection: crop_bounds, Left 0, Top 0, Width 3840, Height 2160, Flags:
Selection: crop, Left 0, Top 0, Width 3840, Height 2160, Flags:
Selection: crop_bounds, Left 0, Top 0, Width 3840, Height 2160, Flags:

User Controls

                     exposure 0x00980911 (int)    : min=4 max=2242 step=1 default=2242 value=1350
                horizontal_flip 0x00980914 (bool)   : default=0 value=0
```

可以看到采集的格式就是我们 python 代码中设置的分辨率及采集格式。注意 OpenCV 中默认就是设置 YU12 格式采集的，VideoCapture 调用系统默认的解码器来解码视频数据会将其解码为 BGR 格式，read()函数读取的数据是 BGR 格式的帧，BGR 不是颜色空间，只是不同颜色通道顺序的约定。如果你使用了其它格式采集，如 NV12 格式，那么很可能解码颜色偏色，需要通过 cvtColor()函数进行转换。

3.12 wifi 开发

3.12.1 安装 WIFI 驱动

Debian 系统路径下就有板载的 wifi 驱动模块/system/lib/modules/8852bs.ko。我们执行下面的指令进行安装。

笔者已经适配系统，上电后已经用脚本安装了这个 8852bs.ko 驱动。

输入 ifconfig

```
ifconfig
```

```
root@ATK-DLRK3568-Debian:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 06:92:40:12:cc:cd  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 38

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.6.88  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::505a:ae1f:f84a:3c33  prefixlen 64  scopeid 0x20<link>
        ether 02:92:40:12:cc:cd  txqueuelen 1000  (Ethernet)
        RX packets 271  bytes 40872 (39.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 155  bytes 13556 (13.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 50

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 32  bytes 2144 (2.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 2144 (2.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether b0:ab:ec:0d:18:22  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@ATK-DLRK3568-Debian:~#
```

3.12.2 nmcli 简介

nmcli（networkmangager command line tool）基于会话的网络管理。网络管理器(NetworManager)是检测网络、自动连接网络的程序。无论是无线还是有线连接，它都可以令您轻松管理。对于无线网络,网络管理器优先连接已知的网络并可以自动切换到最可靠的无线网络。利用网络管理器的程序可以自由切换在线和离线模式。网络管理器会相对无线网络优先选择有线网络，支持 VPN。网络管理器最初由 Redhat 公司开发，现在由 GNOME 管理。

NetworkManager 由一个管理系统网络连接、并且将其状态通过 D-BUS（是一个提供简单的应用程序互相通讯的途径的自由软件项目，它是作为 freedesktoporg 项目的一部分来开发的。）进行报告的后台服务，以及一个允许用户管理网络连接的客户端程序。

同时我们还熟悉另一个网络管理（ConnMan）连接管理器是一个连接管理守护进程（connmand），用于管理运行 Linux 操作系统的设备中的 Internet 连接。它提供低内存消耗，并对网络条件变化进行快速，连贯，同步的反应，ConnMan 也是与 D-Bus 进行通信。

3.12.3 图形界面连接 WIFI

不管是 ConnMan 还是 nmcli 它们的底层都是 wpa_supplicant。恰好，Debian10 系统使用的是 nmcli 进行网络托管。本次就以 nmcli 进行网络开发说明，当然你可以点击桌面上右下角的网络管理进行连接 wifi，如下图。这种图形连接 wifi 的过程，笔者就不多说了。

3.12.4 显示管理网络接口状态

显示和管理网络接口，status：打印设备状态

```
nmcli device status
```

如下图，wlan0 就是我们被 nmcli 托管的 wifi。默认是加载了驱动就是打开的状态，如果没有打开，可以输入 nmcli radio wifi on|off 打开|关闭 wlan0。

```
root@ATK-DLRK3568-Debian:~# nmcli device status
DEVICE  TYPE      STATE      CONNECTION
eth1    ethernet  已连接     有线连接 2
wlan0   wifi      已断开     --
eth0    ethernet  不可用     --
can0    can       未托管     --
lo      loopback  未托管     --
root@ATK-DLRK3568-Debian:~#
```

3.12.5 nmcli 获取 wifi 扫描结果

获取 wifi 扫描结果，执行下面指令，注意 nmcli 网络管理会自动扫描 wifi，nmcli 只是获取扫描的结果，多次获取将会获取上次扫描的结果，等待下一次扫描开始，一般不超过 30s 就会自动扫描一次。

```
nmcli device wifi
```

```
root@ATK-DLRK3568-Debian:~# nmcli device wifi
IN-USE  SSID                            MODE   CHAN  RATE        SIGNAL  BARS  SECU
        Linux-WIFI6                     红外   1     270 Mbit/s  100     ▂▄▆█  WPA2
        Linux-WIFI6_5G                  红外   48    270 Mbit/s  100     ▂▄▆█  WPA2
        GZXH HUAWEI_Wi-Fi5              红外   1     130 Mbit/s  80      ▂▄▆_  WPA2
        ALIENTEK-Linux                  红外   6     270 Mbit/s  79      ▂▄▆_  WPA2
        --                              红外   8     270 Mbit/s  75      ▂▄▆_  WPA1
        RICH901                         红外   8     270 Mbit/s  75      ▂▄▆_  WPA1
        ALIENTEK-Linux                  红外   36    270 Mbit/s  72      ▂▄▆_  WPA2
        ALIENTEK-Linux_Wi-Fi5           红外   36    270 Mbit/s  72      ▂▄▆_  WPA2
        --                              红外   6     130 Mbit/s  70      ▂▄▆_  WPA2
        ALIENTEK-WX                     红外   6     130 Mbit/s  69      ▂▄▆_  WPA2
        --                              红外   6     130 Mbit/s  69      ▂▄▆_  WPA2
        ChinaNet-ymyk                   红外   9     130 Mbit/s  65      ▂▄▆_  WPA1
        ATK-HUAWEI                      红外   11    130 Mbit/s  62      ▂▄▆_  WPA2
        ALIENTEK-7机房                  红外   6     130 Mbit/s  60      ▂▄▆_  WPA2
        --                              红外   6     130 Mbit/s  59      ▂▄▆_  WPA2
        GZXH HUAWEI                     红外   1     130 Mbit/s  57      ▂▄▆_  WPA2
        SSID-2.4G                       红外   11    130 Mbit/s  57      ▂▄▆_  --
        ChinaNet-kss2                   红外   11    130 Mbit/s  50      ▂▄__  WPA1
        --                              红外   149   270 Mbit/s  49      ▂▄__  WPA2
        ALIENTEK-SYS                    红外   11    130 Mbit/s  47      ▂▄__  WPA2
        ChinaNet-ymyk-5G                红外   60    270 Mbit/s  47      ▂▄__  WPA1
        RICH901-5G                      红外   36    270 Mbit/s  45      ▂▄__  WPA1
        --                              红外   149   270 Mbit/s  45      ▂▄__  WPA2
        ALIENTEK-WX                     红外   149   270 Mbit/s  45      ▂▄__  WPA2
        DIRECT-HCDESKTOP-7A885MVmsJN    红外   1     130 Mbit/s  44      ▂▄__  WPA2
        --                              红外   11    130 Mbit/s  44      ▂▄__  WPA2
        --                              红外   36    270 Mbit/s  42      ▂▄__  WPA1
        ALIENTEK-HYS                    红外   7     130 Mbit/s  39      ▂▄__  WPA1
        GZXH HUAWEI                     红外   44    270 Mbit/s  32      ▂▄__  WPA2
        GZXH HUAWEI_Wi-Fi5              红外   44    270 Mbit/s  30      ▂___  WPA2
        --                              红外   149   270 Mbit/s  30      ▂___  WPA2
        --                              红外   149   270 Mbit/s  30      ▂___  WPA2
        ALIENTEK-XHYS                   红外   149   270 Mbit/s  29      ▂___  WPA2
        Tenda_480398                    红外   40    135 Mbit/s  27      ▂___  WPA2
        --                              红外   149   270 Mbit/s  27      ▂___  WPA2
        --                              红外   149   270 Mbit/s  25      ▂___  WPA2
        ALIENTEK-SYS_5G                 红外   149   270 Mbit/s  25      ▂___  WPA2
        酷博网络-5G                     红外   157   270 Mbit/s  15      ▂___  WPA1
        --                              红外   157   270 Mbit/s  15      ▂___  WPA1
        --                              红外   44    270 Mbit/s  14      ▂___  WPA2
        ATK-HUAWEI                      红外   149   270 Mbit/s  14      ▂___  WPA2
root@ATK-DLRK3568-Debian:~#
```

3.12.6 nmcli 重新扫描 wifi

若你需要立刻重新扫描并显示结果，你可以执行下面的指令。

```
nmcli device wifi list --rescan yes
```

或者执行 nmcli device wifi rescan，但是指令不会显示结果，只是触发 wifi 扫描而已。

```
nmcli device wifi rescan
```

```
root@ATK-DLRK3568-Debian:~# nmcli device wifi list --rescan yes
IN-USE  SSID                          MODE   CHAN  RATE        SIGNAL  BARS  SECU
        Linux-WIFI6                   红外   1     270 Mbit/s  100     ▂▄▆█  WPA2
        Linux-WIFI6_5G                红外   48    270 Mbit/s  100     ▂▄▆█  WPA2
        GZXH HUAWEI                   红外   1     130 Mbit/s  77      ▂▄▆_  WPA2
        ALIENTEK-Linux                红外   6     270 Mbit/s  72      ▂▄▆_  WPA2
        ALIENTEK-Linux                红外   36    270 Mbit/s  72      ▂▄▆_  WPA2
        RICH901                       红外   8     270 Mbit/s  70      ▂▄▆_  WPA1
        ALIENTEK-Linux_Wi-Fi5         红外   36    270 Mbit/s  70      ▂▄▆_  WPA2
        --                            红外   6     130 Mbit/s  69      ▂▄▆_  WPA2
        --                            红外   8     270 Mbit/s  69      ▂▄▆_  WPA1
        ALIENTEK-WX                   红外   6     130 Mbit/s  65      ▂▄▆_  WPA2
        ChinaNet-ymyk                 红外   8     130 Mbit/s  65      ▂▄▆_  WPA1
        ATK-HUAWEI                    红外   11    130 Mbit/s  64      ▂▄▆_  WPA2
        ChinaNet-kss2                 红外   11    130 Mbit/s  62      ▂▄▆_  WPA1
        DIRECT-HCDESKTOP-7A885MVmsJN  红外   1     130 Mbit/s  60      ▂▄▆_  WPA2
        ALIENTEK-7机房                红外   6     130 Mbit/s  60      ▂▄▆_  WPA2
        --                            红外   6     130 Mbit/s  60      ▂▄▆_  WPA2
        GZXH HUAWEI_Wi-Fi5            红外   1     130 Mbit/s  59      ▂▄▆_  WPA2
        --                            红外   6     130 Mbit/s  49      ▂▄__  WPA2
        ChinaNet-ymyk-5G              红外   52    270 Mbit/s  49      ▂▄__  WPA1
        --                            红外   149   270 Mbit/s  47      ▂▄__  WPA2
        --                            红外   149   270 Mbit/s  45      ▂▄__  WPA2
        RICH901-5G                    红外   36    270 Mbit/s  44      ▂▄__  WPA1
        --                            红外   36    270 Mbit/s  44      ▂▄__  WPA1
        ALIENTEK-WX                   红外   149   270 Mbit/s  44      ▂▄__  WPA2
        SSID-2.4G                     红外   11    130 Mbit/s  42      ▂▄__  --
        ALIENTEK-7办公                红外   6     270 Mbit/s  40      ▂▄__  WPA2
        ALIENTEK-SYS                  红外   11    130 Mbit/s  39      ▂▄__  WPA2
        DB                            红外   10    130 Mbit/s  32      ▂▄__  WPA2
        GZXH HUAWEI                   红外   44    270 Mbit/s  32      ▂▄__  WPA2
        --                            红外   11    130 Mbit/s  30      ▂___  WPA2
        GZXH HUAWEI_Wi-Fi5            红外   44    270 Mbit/s  30      ▂___  WPA2
        --                            红外   149   270 Mbit/s  30      ▂___  WPA2
        ALIENTEK-XHYS                 红外   149   270 Mbit/s  30      ▂___  WPA2
        Tenda_480398                  红外   40    135 Mbit/s  27      ▂___  WPA2
        --                            红外   149   270 Mbit/s  27      ▂___  WPA2
        Tenda_0430A8                  红外   11    270 Mbit/s  25      ▂___  WPA2
        --                            红外   149   270 Mbit/s  25      ▂___  WPA2
        qml506                        红外   44    270 Mbit/s  17      ▂___  WPA2
        --                            红外   157   270 Mbit/s  15      ▂___  WPA1
        酷博网络-5G                   红外   157   270 Mbit/s  15      ▂___  WPA1
        --                            红外   44    270 Mbit/s  14      ▂___  WPA2
        ATK-HUAWEI                    红外   149   270 Mbit/s  14      ▂___  WPA2
        ALIENTEK-7机房                红外   149   270 Mbit/s  12      ▂___  WPA2
        ALIENTEK-YF-SYS               红外   153   270 Mbit/s  12      ▂___  WPA2
root@ATK-DLRK3568-Debian:~#
```

3.12.7 nmcli 连接 wifi

连接 wifi 格式如下，尽量用英文双引名，防止有空格的热点名称。

```
nmcli device wifi connect "热点名称"  password  "热点密码"
```

如下，笔者连接公司的 Wifi 热点名称为"ALIENTEK-Linux"，那么连接指令如下。

```
nmcli device wifi connect "ALIENTEK-Linux"  password  "1590202***3"
```

成功连接如下图。

```
root@ATK-DLRK3568-Debian:~# nmcli device wifi connect "ALIENTEK-Linux"  password  "159020    3"
[ 3755.432288] IPv6: ADDRCONF(NETDEV_CHANGE): wlan0: link becomes ready
成功用 "57903d94-4173-4243-bc36-b723f373da7d" 激活了设备 "wlan0"。
root@ATK-DLRK3568-Debian:~#
```

再次查看显示和管理网络接口，status：打印设备状态

```
nmcli device status
```

```
root@ATK-DLRK3568-Debian:~# nmcli device status
DEVICE  TYPE      STATE     CONNECTION
eth1    ethernet  已连接    有线连接 2
wlan0   wifi      已连接    ALIENTEK-Linux
eth0    ethernet  不可用    --
can0    can       未托管    --
lo      loopback  未托管    --
root@ATK-DLRK3568-Debian:~#
```

3.12.8 nmcli 断开 wifi

既然我们连接上了 wifi 那么如何断开呢？执行下面的指令。

```
nmcli connection down "ALIENTEK-Linux"
```

```
root@ATK-DLRK3568-Debian:~# nmcli connection down "ALIENTEK-Linux"
成功停用连接 "ALIENTEK-Linux"（D-Bus 活动路径：/org/freedesktop/NetworkManager/ActiveConnection/2）

root@ATK-DLRK3568-Debian:~#
```

再次连接上，可以使用下面指令。

```
nmcli connection up "ALIENTEK-Linux"
```

```
root@ATK-DLRK3568-Debian:~# nmcli connection up "ALIENTEK-Linux"
连接已成功激活（D-Bus 活动路径：/org/freedesktop/NetworkManager/ActiveConnection/3）
root@ATK-DLRK3568-Debian:~#
```

3.12.9 nmcli 删除 wifi 连接

如果你不需要连接了，可以删除这个热点名，同时也会删除密码。

```
nmcli connection delete "ALIENTEK-Linux"
```

```
root@ATK-DLRK3568-Debian:~# nmcli connection delete "ALIENTEK-Linux"
成功删除连接 "ALIENTEK-Linux" (57903d94-4173-4243-bc36-b723f373da7d)。
root@ATK-DLRK3568-Debian:~#
```

3.12.10 Wifi 开启 AP 模式

Wifi 开启 AP 模式简单来说就是开启热点，你的手机或者其他平板电脑设备可以连接到这个热点。

注意，需要确保当前没有热点连接，如果有连接我们需要关闭连接。注意，如果你在上一步连接过 wifi，但是没有关闭或者删除，下一次开机加载驱动后会自动连接 wifi 热点，请执行下面的指令，确保你的 WIFI 状态没有连接。

```
nmcli device status
```

```
root@ATK-DLRK3568-Debian:~# nmcli device status
DEVICE  TYPE      STATE     CONNECTION
eth1    ethernet  已连接    有线连接 2
wlan0   wifi      已断开    --
eth0    ethernet  不可用    --
can0    can       未托管    --
lo      loopback  未托管    --
root@ATK-DLRK3568-Debian:~#
```

3.12.10.1 安装 hostapd 工具

首先我们需要连接有线网络下载 hostapd，这个是开启热点的工具。

```
apt-get install hostapd
```

```
root@ATK-DLRK3568-Debian:~# apt-get install hostapd
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了：
  liba52-0.7.4 libdca0 libdrm-freedreno1 libdrm-tegra0
使用'apt autoremove'来卸载它(它们)。
下列【新】软件包将被安装：
  hostapd
升级了 0 个软件包，新安装了 1 个软件包，要卸载 0 个软件包，有 45 个软件包未被升级。
需要下载 724 kB 的归档。
解压缩后会消耗 2,021 kB 的额外空间。
获取:1 http://mirrors.aliyun.com/debian buster/main arm64 hostapd arm64 2:2.7+git20190128+0c1e29f-6+deb10u3 [724 kB]
已下载 724 kB，耗时 0秒 (1,813 kB/s)
正在选中未选择的软件包 hostapd。
(正在读取数据库 ... 系统当前共安装有 74813 个文件和目录。)
准备解压 .../hostapd_2%3a2.7+git20190128+0c1e29f-6+deb10u3_arm64.deb  ...
正在解压 hostapd (2:2.7+git20190128+0c1e29f-6+deb10u3) ...
正在设置 hostapd (2:2.7+git20190128+0c1e29f-6+deb10u3) ...
insserv: script S99_auto_reboot: service reboot already provided!
insserv: script rkwifibt.sh: service rockchip already provided!
[ 6808.851716] systemd-fstab-generator[2520]: Ignoring "noauto" for root device
[ 6809.572926] systemd-fstab-generator[2542]: Ignoring "noauto" for root device
Created symlink /etc/systemd/system/multi-user.target.wants/hostapd.service → /lib/systemd/system/hostapd.service.
[ 6810.873516] systemd-fstab-generator[2563]: Ignoring "noauto" for root device
[ 6811.424398] systemd-fstab-generator[2580]: Ignoring "noauto" for root device
         Starting Advanced IEEE 802…/WPA/WPA2/EAP Authenticator...
                                                [FAILED] Failed to start Advanced I…1X/WPA/WPA2/EAP Authenticator.
                                                                                                See 'systemctl status hostapd.service' for deta

or hostapd.service failed because the control process exited with error code.
See "systemctl status hostapd.service" and "journalctl -xe" for details.
Created symlink /etc/systemd/system/hostapd.service → /dev/null.
[ 6812.103707] systemd-fstab-generator[2605]: Ignoring "noauto" for root device
正在处理用于 systemd (241-7~deb10u9) 的触发器 ...
[ 6812.679386] systemd-fstab-generator[2623]: Ignoring "noauto" for root device
root@ATK-DLRK3568-Debian:~#
```

3.12.10.2 编辑 dnsmasq.conf

在 root 路径下编辑一个 dnsmasq.conf 文件（默认笔者已经提前放到/root 目录下，你可不用亲自编辑），这个文件是用于分配给连接上的设置分配 ip。

vi dnsmasq.conf

```
root@ATK-DLRK3568-Debian:~# vi dnsmasq.conf
```

内容如下：

```
interface=wlan0
dhcp-range=192.168.4.2,192.168.4.254,255.255.255.0,24h
port=111
```

3.12.10.3 编辑 hostapd.conf

在 root 路径下创建一个 hostapd.conf 文件（默认笔者已经提前放到/root 目录下，你可不用亲自编辑），用于设置无线密码，这里我们不去修改系统的 hostapd.conf。

vi hostapd.conf

```
root@ATK-DLRK3568-Debian:~# vi hostapd.conf
```

内容如下：

```
interface=wlan0
ssid=alientek_softap
driver=nl80211
channel=6
hw_mode=g
ignore_broadcast_ssid=0
auth_algs=1
wpa=3
wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
```

```
rsn_pairwise=CCMP
```

3.12.10.4 设置 wlan0 的 ip

设置 wlan0 的 ip，执行下面的指令。然后使用 ifconfig 查看是否设置成功。

```
ifconfig wlan0 192.168.4.3
ifconfig
```

```
root@ATK-DLRK3568-Debian:~# ifconfig wlan0 192.168.4.3
root@ATK-DLRK3568-Debian:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 06:92:40:12:cc:cd  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 38

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.6.88  netmask 255.255.255.0  broadcast 192.168.6.255
        inet6 fe80::505a:ae1f:f84a:3c33  prefixlen 64  scopeid 0x20<link>
        ether 02:92:40:12:cc:cd  txqueuelen 1000  (Ethernet)
        RX packets 2673  bytes 311645 (304.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 284  bytes 23486 (22.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 50

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 32  bytes 2144 (2.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 2144 (2.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.4.3  netmask 255.255.255.0  broadcast 192.168.4.255
        ether b0:ab:ec:0d:18:22  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 103822 (101.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 1934 (1.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@ATK-DLRK3568-Debian:~#
```

3.12.10.5 开启 AP 热点

执行下面指令，开启热点。成功开启热点如下图。

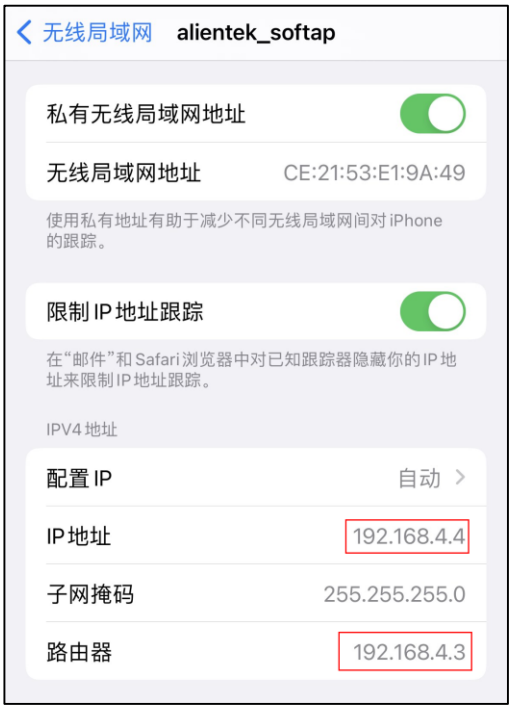
```
dnsmasq -C dnsmasq.conf     #配置配置 DNS 和 DHCP
hostapd ./hostapd.conf -B   #开启热点，-B 是后台运行的意思
```

```
root@ATK-DLRK3568-Debian:~# dnsmasq -C dnsmasq.conf
root@ATK-DLRK3568-Debian:~# hostapd ./hostapd.conf -B
Configuration file: ./hostapd.conf
Using interface wlan0 with hwaddr b0:ab:ec:0d:18:22 and ssid "alientek_softap"
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
root@ATK-DLRK3568-Debian:~#
```

3.12.10.6 热点连接测试

用手机连接测试热点名称“alientek_softap”即可，密码是 hostapd.conf 文件里配置的“12345678”。如下图，笔者用手机连接上了板子发出的热点。

可以看到，板子的 ip 是是我们 3.12.10 小节设置的 wlan0 ip 192.168.4.3，充当了网关即路由器。并且手机也从开发板分配得到了一个 ip 为 192.168.4.4。注意这个热点能连接，但是不是给你上网用的，本次演示仅限局域网内连接，可以编写网络通信程序进行信息收发，热点的用途很大，比如你想附近一个设备与你的板子通信都是可以通过此热点进行信息交互，详细请看 Qt 或者 C 应用教程的网络编程部分。



3.13 蓝牙开发

Debian 的蓝牙开发依赖 Buildroot，蓝牙驱动模块 hci_uart.ko 等文件都是 Buildroot 配置选择正确的 Wi-Fi/BT 模组的前提下才会生成。所以你若自己编译 debian 系统那么你需要先编译 Buildroot。

蓝牙协议有非常多，如下。

- L2CAP: 逻辑链路控制和适配协议，用于建立和维护逻辑链路连接。
- RFCOMM: 远程串口协议，将串口设备的数据传输映射到蓝牙上。
- SDP: 服务发现协议，用于查找和描述可用的蓝牙服务。
- GAP: 通用访问协议，定义了设备之间的基本连接和通信规则。
- GATT: 通用属性协议，用于建立和管理设备之间的属性数据交换。
- ATT: 属性协议，用于在蓝牙设备之间传输属性值。
- HFP: 手机蓝牙耳机协议，用于手机和蓝牙耳机之间的音频通信。
- A2DP: 高级音频分发协议，用于在蓝牙设备之间传输高质量音频数据。
- AVRCP: 音频/视频远程控制协议，用于在蓝牙设备之间传输媒体控制命令。

笔者水平有限，蓝牙是一个很大的知识点，本次仅以演示蓝牙 RFCOMM 远程串口使用方法，其他协议可以自行研究。

3.13.1 蓝牙初始化

Debian 系统路径下就有板载的蓝牙驱动模块/system/lib/modules/hci_uart.ko。

蓝牙初始化 bt_init.sh 脚本笔者已经提前放在/usr/bin/路径下，bt_init.sh 的内容如下。

```
#!/bin/bash
# 杀死进程，防止占用串口
if [[ "$(pidof rtk_hciattach)" != "" ]]
then
        kill -9 $(pidof rtk_hciattach)
fi
echo 0 > /sys/class/rfkill/rfkill0/state #下电
echo 0 > /proc/bluetooth/sleep/btwrite
sleep 1
echo 1 > /sys/class/rfkill/rfkill0/state #上电
echo 1 > /proc/bluetooth/sleep/btwrite
sleep 1
! test -d /sys/module/hci_uart && insmod /system/lib/modules/hci_uart.ko

rtk_hciattach -n -s 115200 /dev/ttyS8 rtk_h5 &
```

上面的脚本中，蓝牙使用的是串口通信，使用的是串口是 ttyS8，使用 rtk_hciattach 进行蓝牙初始化。

直接在串口终端输入 bt_init.sh 进行蓝牙初始化。看到下图红色框框内容后，说明蓝牙初始化成功。

```
bt_init.sh
```

```
root@ATK-DLRK3568-Debian:~# bt_init.sh
[   24.903391] [BT_RFKILL]: bt shut off power
[   25.944429] [BT_RFKILL]: rfkill_rk_set_power: set bt wake_host high!
[   26.000060] [BT_RFKILL]: ENABLE UART_RTS
[   26.106613] [BT_RFKILL]: DISABLE UART_RTS
[   26.106825] [BT_RFKILL]: bt turn on power
[   26.106928] [BT_RFKILL]: Request irq for bt wakeup host
[   26.107061] [BT_RFKILL]: ** disable irq
[   27.147838] hci_uart: loading out-of-tree module taints kernel.
[   27.150936] Bluetooth: HCI UART driver ver 2.2.fdfa5dd.20220110-123627
[   27.151004] Bluetooth: HCI H4 protocol initialized
[   27.151016] Bluetooth: HCI Realtek H5 protocol initialized
[   27.151027] rtk_btcoex: rtk_btcoex_init: version: 1.2
[   27.151034] rtk_btcoex: create workqueue
[   27.151461] rtk_btcoex: alloc buffers 1792, 2432 for ev and l2
root@ATK-DLRK3568-Debian:~# [    27.163679] of_dma_request_slave_channel: dma-names property of node '/serial@fe6c0000' missing or empty
[   27.163748] ttyS8 - failed to request DMA, use interrupt mode
Realtek Bluetooth :Realtek Bluetooth init uart with init speed:115200, type:HCI UART H5
Realtek Bluetooth :Realtek hciattach version 3.1.2c27de6.20220110-123628

Realtek Bluetooth :Use epoll
Realtek Bluetooth :[SYNC] Get SYNC Resp Pkt
Realtek Bluetooth :[CONFIG] Get SYNC pkt
Realtek Bluetooth :[CONFIG] Get CONFG pkt
Realtek Bluetooth :[CONFIG] Get CONFG resp pkt
Realtek Bluetooth :dic is 1, cfg field 0x14
Realtek Bluetooth :H5 init finished

Realtek Bluetooth :Realtek H5 IC
Realtek Bluetooth :Receive cmd complete event of command: 1001
Realtek Bluetooth :HCI Version 0x0b
Realtek Bluetooth :HCI Revision 0x000b
Realtek Bluetooth :LMP Subversion 0x8852
Realtek Bluetooth :Receive cmd complete event of command: fc6d
Realtek Bluetooth :Read ROM version 01
Realtek Bluetooth :LMP Subversion 0x8852
Realtek Bluetooth :EVersion 1
Realtek Bluetooth :IC: RTL8852BS
Realtek Bluetooth :Firmware/config: rtl8852bs_fw, rtl8852bs_config
Realtek Bluetooth :Couldnt open extra config /opt/rtk_btconfig.txt, No such file or directory
Realtek Bluetooth :Couldnt access customer BT MAC file /opt/bdaddr
Realtek Bluetooth :Origin cfg len 33
Realtek Bluetooth :55 ab 23 87 1b 00 0c 00 10 02 80 92 04 50 c5 ea
Realtek Bluetooth :19 e1 1b fd af 5f 01 a4 0b 8d 00 01 fa 8f 00 01
Realtek Bluetooth :bf
Realtek Bluetooth :Config baudrate: 04928002
Realtek Bluetooth :uart flow ctrl: 1
Realtek Bluetooth :Vendor baud from Config file: 04928002
Realtek Bluetooth :New cfg len 33
Realtek Bluetooth :55 ab 23 87 1b 00 0c 00 10 02 80 92 04 50 c5 ea
Realtek Bluetooth :19 e1 1b fd af 5f 01 a4 0b 8d 00 01 fa 8f 00 01
Realtek Bluetooth :bf
Realtek Bluetooth :Load FW /lib/firmware/rtlbt/rtl8852bs_fw OK, size 49852
Realtek Bluetooth :rtb_get_fw_project_id: opcode 0, len 1, data 20
Realtek Bluetooth :FW version 0xddb743b1, Patch num 2
Realtek Bluetooth :Chip id 0x0001
Realtek Bluetooth :Chip id 0x0002
Realtek Bluetooth :Patch length 0x83b5
Realtek Bluetooth :Start offset 0x00003ec0
Realtek Bluetooth :Svn version: 3985952765
Realtek Bluetooth :Coexistence: BTCOEX_20210331-0606

Realtek Bluetooth :FW  exists, Config file  exists
Realtek Bluetooth :Total len 33750 for fwc
Realtek Bluetooth :baudrate in change speed command: 0x02 0x80 0x92 0x04
Realtek Bluetooth :Receive cmd complete event of command: fc17
Realtek Bluetooth :Received cc of vendor change baud
Realtek Bluetooth :Final speed 1500000
Realtek Bluetooth :end_idx: 133, lp_len: 234, additional pkts: 6

Realtek Bluetooth :Start downloading...
Realtek Bluetooth :Send additional packet 7
Realtek Bluetooth :Send additional packet 8
Realtek Bluetooth :Send additional packet 9
Realtek Bluetooth :Send additional packet 10
Realtek Bluetooth :Send additional packet 11
Realtek Bluetooth :Last packet 140
Realtek Bluetooth :Send last pkt
Realtek Bluetooth :Enable host hw flow control
Realtek Bluetooth :h5_hci_reset: Issue hci reset cmd
Realtek Bluetooth :Receive cmd complete event of command: 0c03
Realtek Bluetooth :Received cc of hci reset cmd
Realtek Bluetooth :Init Process finished
[   27.773555] Bluetooth: h5_open
[   27.773657] Bluetooth: hci_uart_register_dev
Realtek Bluetooth :Realtek Bluetooth post process
Realtek Bluetooth :Device setup complete
[   27.775243] rtk_btcoex: Open BTCOEX
[   27.783441] rtk_btcoex: BTCOEX hci_rev 0xddb7
[   27.783549] rtk_btcoex: BTCOEX lmp_subver 0x43b1
[   33.766717] vcc5v0_otg: disabling
[   33.766832] vcc3v3_pcie: disabling
```

3.13.2 蓝牙 rfcomm

蓝牙 RFCOMM 是一种蓝牙协议，它提供了一种串行数据传输的方式，可以在蓝牙设备之间进行数据传输。RFCOMM 可以将串行数据转换为蓝牙的数据包，从而实现无线传输。

RFCOMM 提供了一种虚拟串口的概念，使得蓝牙设备之间的数据传输可以像串口一样进行。这种方式广泛应用于蓝牙耳机、蓝牙打印机等设备上。

在使用 RFCOMM 进行数据传输时，需要先建立蓝牙连接，并且指定 RFCOMM 通道。然后，在连接两端的设备中，可以通过 RFCOMM 发送和接收数据。这种方式适用于需要进行串口通信的应用场景，例如无线控制器、无线传感器等。

笔者编写了开启蓝牙 rfcomm 建立 rfcomm 通道的脚本，位于/usr/bin/rfcomm_init.sh。脚本的内容如下。

```
if [[ "$(pidof rfcomm)" != "" ]]
then
        kill -9 $(pidof rfcomm)
fi
hciconfig hci0 piscan
hciconfig hci0 noauth
sleep 1
sdptool add SP
sleep 1
rfcomm watch hci0 &
echo "rfcomm init finished! please use bluetoothctl cmd to pair your device and then connect it!"
```

执行 rfcomm_init.sh 脚本，可以看到已经创建了通道 1，等待连接。

```
rfcomm_init.sh
```

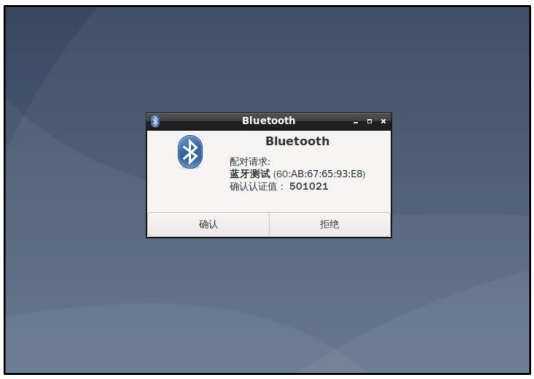
```
root@ATK-DLRK3568-Debian:~# rfcomm_init.sh
[ 2536.925257] Bluetooth: hu 00000000f1b8e418 retransmitting 1 pkts
Serial Port service registered
rfcomm init finished! please use bluetoothctl cmd to pair your device and then connect it!
root@ATK-DLRK3568-Debian:~# Waiting for connection on channel 1

root@ATK-DLRK3568-Debian:~#
```

此时我们需要进行蓝牙配对，然后使用安卓手机“蓝牙调试宝”App 进行连接。苹果手机没有类似软件。

3.13.3 图形界面配对

图形界面配对蓝牙，在安卓手机蓝牙页面处点击“ATK-DLRK3568-Debian”蓝牙设备进行配对，此时 Debian 系统弹出一个配对的弹窗，点击配对即可。

Debian 桌面上弹出的配对弹窗，点击配对即可。



3.13.4 命令行界面配对

考虑到没有屏（显示器）的用户，可以用命令行进行配对。

输入下面指令，进入 bluetoothctl 交互模式。

```
bluetoothctl
```

依次执行下面指令。

```
power on                # 打开电源
agent on                # 开启代理
discoverable on         # 开启可被检测
scan on                 # 开启扫描设备
```

```
root@ATK-DLRK3568-Debian:~# bluetoothctl
Agent registered
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# agent on
Agent is already registered
[bluetooth]# discoverable on
Changing discoverable on succeeded
[bluetooth]# scan on
[bluetooth]# [  909.521190] Bluetooth: hu 00000000dc63c9a8 retransmitting 1 pkts
[  909.525808] rtk_btcoex: hci (periodic)inq start
Discovery started
[CHG] Controller 48:8F:4C:10:2E:BD Discovering: yes
[bluetooth]# [  909.588059] Bluetooth: Unknown advertising packet type: 0x100
[  909.588162] Bluetooth: hci0: advertising data len corrected
[NEW] Device 68:DB:B7:43:DF:D7 68-DB-B7-43-DF-D7
[bluetooth]# [  909.627639] Bluetooth: Unknown advertising packet type: 0x100
[  909.668153] Bluetooth: Unknown advertising packet type: 0x100
[NEW] Device 43:76:F4:B3:26:7E 43-76-F4-B3-26-7E
[NEW] Device 43:32:ED:80:11:D7 43-32-ED-80-11-D7
```

等到扫描到自己的蓝牙设备，输入 scan off 停止扫描。

```
scan off
```

如下图，笔者扫描到安卓手机蓝牙 MAC 地址，确认后就可以进行配对了。

```
[bluetooth]# [  992.739140] rtk_btcoex: hci (periodic)inq cancel/exit
Discovery stopped
[CHG] Controller 48:8F:4C:10:2E:BD Discovering: no
[CHG] Device 55:EF:13:8A:AC:5C RSSI is nil
[CHG] Device D2:32:E0:5D:64:3F RSSI is nil
[CHG] Device 6C:B3:EC:CF:F8:1D TxPower is nil
[CHG] Device 6C:B3:EC:CF:F8:1D RSSI is nil
[CHG] Device 75:A5:9E:4A:4F:01 TxPower is nil
[CHG] Device 75:A5:9E:4A:4F:01 RSSI is nil
[CHG] Device 73:BE:5F:23:0A:E8 TxPower is nil
[CHG] Device 73:BE:5F:23:0A:E8 RSSI is nil
[CHG] Device E9:E6:5E:39:31:13 RSSI is nil
[CHG] Device 4C:D8:BC:5F:3E:4D TxPower is nil
[CHG] Device 4C:D8:BC:5F:3E:4D RSSI is nil
[CHG] Device 7E:AA:B1:1E:11:CE TxPower is nil
[CHG] Device 7E:AA:B1:1E:11:CE RSSI is nil
[CHG] Device E7:E8:E6:3F:18:81 RSSI is nil
[CHG] Device 0C:AE:B0:FB:5B:0E RSSI is nil
[CHG] Device 44:A9:D1:ED:86:82 RSSI is nil
[CHG] Device AC:D6:18:5C:C2:28 RSSI is nil
[CHG] Device 74:D2:85:88:BD:59 RSSI is nil
[CHG] Device 64:68:76:9D:A5:71 RSSI is nil
[CHG] Device 22:56:BD:A8:CB:63 RSSI is nil
[CHG] Device 5F:31:3F:31:78:9D RSSI is nil
[CHG] Device 53:DE:A9:F4:3A:10 TxPower is nil
[CHG] Device 53:DE:A9:F4:3A:10 RSSI is nil
[CHG] Device 56:D2:E0:53:93:43 TxPower is nil
[CHG] Device 56:D2:E0:53:93:43 RSSI is nil
[CHG] Device 5A:78:8A:55:04:7F TxPower is nil
[CHG] Device 5A:78:8A:55:04:7F RSSI is nil
[CHG] Device 7C:2A:DB:03:DF:77 RSSI is nil
[CHG] Device 00:13:EF:A0:04:F1 TxPower is nil
[CHG] Device 00:13:EF:A0:04:F1 RSSI is nil
[CHG] Device 6F:2D:97:F5:77:30 RSSI is nil
[CHG] Device 5C:D5:55:04:3A:B0 TxPower is nil
[CHG] Device 5C:D5:55:04:3A:B0 RSSI is nil
[CHG] Device 42:42:D1:55:64:5D TxPower is nil
[CHG] Device 42:42:D1:55:64:5D RSSI is nil
[CHG] Device 60:AB:67:65:93:E8 RSSI is nil
[CHG] Device 43:32:ED:80:11:D7 TxPower is nil
```

输入下面指令进行配对。

pair 60:AB:67:65:93:E8 # 请填写你自己的蓝牙 MAC

```
[bluetooth]# pair 60:AB:67:65:93:E8
Attempting to pair with 60:AB:67:65:93:E8
[bluetooth]# [ 1414.120919] Bluetooth: hu 00000000dc63c9a8 retransmitting 1 pkts
[ 1414.122829] rtk_btcoex: hci create connection, start paging
[ 1414.470207] rtk_btcoex: connected, handle 000b, status 0x00
[ 1414.470304] rtk_btcoex: Page success
[ 1414.500259] rtk_btcoex: io capability request
[CHG] Device 60:AB:67:65:93:E8 Connected: yes
Request confirmation
[agent] Confirm passkey 289260 (yes/no): yes
[蓝牙测试]# [ 1431.287034] rtk_btcoex: link key notify
[ 1431.327388] rtk_btcoex: l2cap op 2, len 16, out 1
[ 1431.327516] rtk_btcoex: TX l2cap conn req, hndl 0x000b, PSM 0x0001, scid 0x0040
[ 1431.327543] rtk_btcoex: PSM(0x0001) do not need parse
[ 1431.343600] rtk_btcoex: l2cap op 3, len 20, out 0
[ 1431.343816] rtk_btcoex: RX l2cap conn rsp, hndl 0x000b, dcid 0x0042, scid 0x0040, result 0x0000
[ 1431.352224] rtk_btcoex: l2cap op 2, len 16, out 0
[ 1431.352416] rtk_btcoex: RX l2cap conn req, hndl 0x000b, PSM 0x0001, scid 0x0043
[ 1431.352457] rtk_btcoex: PSM(0x0001) do not need parse
[ 1431.352546] rtk_btcoex: l2cap op 3, len 20, out 1
[ 1431.352702] rtk_btcoex: TX l2cap conn rsp, hndl 0x000b, dcid 0x0041, scid 0x0043, result 0x0000
[ 1431.385879] rtk_btcoex: l2cap op 6, len 16, out 0
[ 1431.386048] rtk_btcoex: RX l2cap disconn req, hndl 0x000b, dcid 0x0041, scid 0x0043
[ 1431.386087] rtk_btcoex: handle_l2cap_discon_req: handle 0x000b, dcid 0x0041, scid 0x0043, dir 0
[ 1431.394660] rtk_btcoex: l2cap op 2, len 16, out 0
[ 1431.394857] rtk_btcoex: RX l2cap conn req, hndl 0x000b, PSM 0x0001, scid 0x0044
[ 1431.394926] rtk_btcoex: PSM(0x0001) do not need parse
[ 1431.395036] rtk_btcoex: l2cap op 3, len 20, out 1
[ 1431.395372] rtk_btcoex: TX l2cap conn rsp, hndl 0x000b, dcid 0x0041, scid 0x0044, result 0x0000
[CHG] Device 60:AB:67:65:93:E8 Modalias: bluetooth:v038Fp1200d1436
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001105-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001112-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001115-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001132-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001800-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device 60:AB:67:65:93:E8 UUIDs: 98b97136-36a2-11ea-8467-484d7e99a198
[CHG] Device 60:AB:67:65:93:E8 ServicesResolved: yes
[CHG] Device 60:AB:67:65:93:E8 Paired: yes
Pairing successful
[蓝牙测试]# [ 1431.447184] rtk_btcoex: l2cap op 6, len 16, out 0
[ 1431.447339] rtk_btcoex: RX l2cap disconn req, hndl 0x000b, dcid 0x0041, scid 0x0044
[ 1431.447363] rtk_btcoex: handle_l2cap_discon_req: handle 0x000b, dcid 0x0041, scid 0x0044, dir 0
[ 1433.990448] rtk_btcoex: l2cap op 6, len 16, out 1
[ 1433.990655] rtk_btcoex: TX l2cap disconn req, hndl 0x000b, dcid 0x0042, scid 0x0040
[ 1433.990693] rtk_btcoex: handle_l2cap_discon_req: handle 0x000b, dcid 0x0042, scid 0x0040, dir 1
[ 1436.200336] rtk_btcoex: HCI Disconnect, handle 000b, reason 0x13
[ 1436.269667] rtk_btcoex: disconn cmpl evt: status 0x00, handle 000b, reason 0x16
[ 1436.269766] rtk_btcoex: process disconn complete event.
[CHG] Device 60:AB:67:65:93:E8 ServicesResolved: no
[CHG] Device 60:AB:67:65:93:E8 Connected: no
[bluetooth]#
```

输入yes

手机端也弹出配对码。

输入下面的指令信任 MAC。

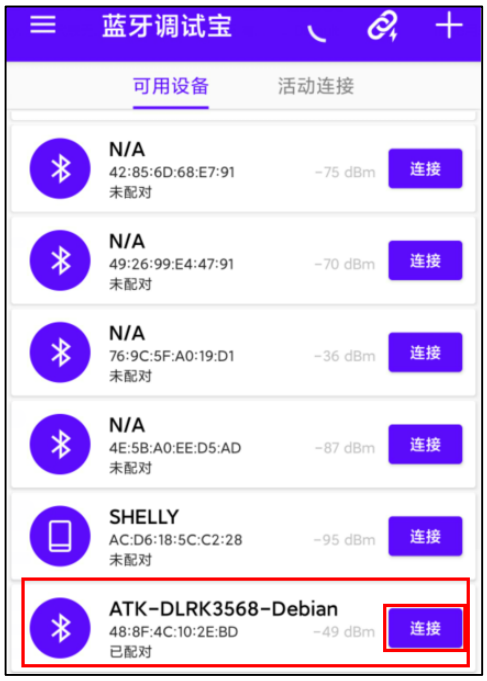
```
trust 60:AB:67:65:93:E8                # 请填写你自己的蓝牙 MAC
```

```
[bluetooth]# trust 60:AB:67:65:93:E8
[CHG] Device 60:AB:67:65:93:E8 Trusted: yes
Changing 60:AB:67:65:93:E8 trust succeeded
[bluetooth]#
```

3.13.5 蓝牙通信测试

在 3.13.2 小节蓝牙 rfcomm 开启后，需要在 3.13.3 或者 3.13.4 小节进行蓝牙配对，然后到了这一步，我们开始通信测试。

安卓手机安装蓝牙调试宝，打开软件后，已经配对和未配对的蓝牙会显示在列表里，如下图。



点击连接，默认 uuid 即可，可以看到下图，已经连接上了。可以进行数据收发了。

板子接收数据可以使用 cat 指令进行收数据。

cat /dev/rfcomm0

```
root@ATK-DLRK3568-Debian:~# cat /dev/rfcomm0
```

手机向开发板蓝牙发“123”。

开发板收到数据。

```
root@ATK-DLRK3568-Debian:~# cat /dev/rfcomm0
123[ 3388.260919] rtk_btcoex: count_pan_packet_timeout: pan_packet_count 4
[ 339[illegible]336879] rtk_btcoex: update_hid_active_state: handle 0x0001, interval 798
[ 3392.336977] rtk_btcoex: HID not connected, nothing to be down
```

同理，我们向手机蓝牙发数据。可以使用 echo 指令。

echo 456 > /dev/rfcomm0

```
root@ATK-DLRK3568-Debian:~# echo 456 > /dev/rfcomm0
root@ATK-DLRK3568-Debian:~# [ 3670.975617] rtk_btcoex: count_pan_packet_timeout: pan_packet_count 2
[ 3671.059706] rtk_btcoex: update_hid_active_state: handle 0x0001, interval 0
[ 3671.059800] rtk_btcoex: HID not connected, nothing to be down

root@ATK-DLRK3568-Debian:~#
```

手机收到结果如下。