PET_RK3288_P01 安卓主板
开发板编译烧写手册

一、搭建编译环境

1、 安装 Ubuntu 18.04.4 64 位系统（安装时不要选择自动下载并安装更新）。

2、 安装依赖软件

```
sudo apt -y clean
sudo apt -y update
sudo apt -y upgrade
sudo apt -y install openssh-server git flex bison gperf build-essential libncurses5-dev:i386 libi2c-dev
sudo apt -y install libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-dev tofrodos pv autopoint
sudo apt -y install python-markdown libxml2-utils xsltproc zlib1g-dev:i386 dpkg-dev libsdl1.2-dev
sudo apt -y install git-core gnupg zip curl zlib1g-dev gzip bzip2 perl tar cpio python lzop
sudo apt -y install libc6-dev-i386 lib32ncurses5-dev sed bash gcc g++ g++-multilib gcc-multilib
sudo apt -y install x11proto-core-dev libx11-dev lib32z-dev ccache libgl1-mesa-dev libxml2-utils
sudo apt -y install xsltproc unzip m4 gawk fakeroot u-boot-tools make texinfo clang cmake dos2unix
sudo apt -y install libssl-dev mtools parted libudev-dev libusb-1.0-0-dev lib32gcc-7-dev autoconf
sudo apt -y install autotools-dev libstdc++-7-dev libsigsegv2 m4 intltool libdrm-dev binutils patch
sudo apt -y install rsync file bc wget libncurses5 libglib2.0-dev libgtk2.0-dev libglade2-dev cvs w3m
sudo apt -y install mercurial openssh-client dblatex graphviz python-matplotlib e2fsprogs devscripts
sudo apt -y install gcc-arm-linux-gnueabihf python-linaro-image-tools linaro-image-tools net-tools
sudo apt -y install expect binfmt-support qemu-user-static live-build device-tree-compiler proot
sudo apt -y install patchelf multistrap xutils-dev libtool libwayland-dev openjdk-8-jdk openjdk-8-jre
sudo apt -y autoremove
```

3、 在开发工具目录下有安装好的虚拟机磁盘镜像文件（VMware 15.5.6 及以上版本），虚拟机内存设置最少需要 16G，磁盘镜像文件所在的 windows 磁盘分区剩余容量大于 200G。

4、 虚拟机磁盘镜像文件已经安装好相关软件，不用再运行上面的安装命令，可以直接复制源码后解压编译。

二、编译 Android 安卓系统

1、解压源代码

将源代码压缩文件全部复制到 Ubuntu 系统下，保证所在磁盘剩余空间要大于 200G，使用以下命令解压源代码（注意参数中是大写 J）：

```
tar xvJf PET_RK3288_P01_Android_Source.tar.xz
```

2、编译 Android 全部源码

```
cd PET_RK3288_P01_Android
./build_rk3288_7.1.2.sh -F
编译完成后正确提示如下：
```

```
TARGET_BASE_PARAMETER_IMAGE =
Make image ok!
Make update.img
start to make update.img...
Android Firmware Package Tool v1.62
------ PACKAGE ------
Add file: ./package-file
Add file: ./Image/MiniLoaderAll.bin
Add file: ./Image/parameter.txt
Add file: ./Image/trust.img
Add file: ./Image/uboot.img
Add file: ./Image/misc.img
Add file: ./Image/resource.img
Add file: ./Image/kernel.img
Add file: ./Image/boot.img
Add file: ./Image/recovery.img
Add file: ./Image/system.img
Add file: ./Image/vendor0.img
Add file: ./Image/vendor1.img
Add file: ./Image/gzpeite.img
Add CRC...
Make firmware OK!
------ OK ------
********RKImageMaker ver 1.63********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
Make update image ok!
```

编译完成后会在 rockdev\Image-rk3288 目录下生成 update.img 烧写镜像文件。

如果使用虚拟机内存容量不足，编译安卓系统，默认参数可能会因内存不足引起错误，可以尝试将虚拟机内存设置为 8G，进行以下修改后再重启虚拟机进行编译。

文件 prebuilts\sdk\tools\jack-admin

JACK_SERVER_COMMAND="java -XX:MaxJavaStackTraceDepth=-1 -Djava.io.tmpdir=$TMPDIR $JACK_SERVER_VM_ARGUMENTS -cp $LAUNCHER_JAR $LAUNCHER_NAME"

修改为

JACK_SERVER_COMMAND="java -Xmx8G -XX:MaxJavaStackTraceDepth=-1 -Djava.io.tmpdir=$TMPDIR $JACK_SERVER_VM_ARGUMENTS -cp $LAUNCHER_JAR $LAUNCHER_NAME"

重启虚拟机，再运行编译命令。

首次编译会很耗时，后续修改 uboot、kernel、android 的某个源码后再次使用./build_rk3288_7.1.2.sh -F 编译会快很多。

3、单项编译

单项编译命令必须在完成首次完整编译后才能正常使用。

```
编译 uboot：./build_rk3288_7.1.2.sh -U
编译 kernel：./build_rk3288_7.1.2.sh -K
编译 android：./build_rk3288_7.1.2.sh -A
```

以上命令都会在 rockdev\Image-rk3288 目录下生成更新后的 update.img

4、清理 Android 源码

```
cd PET_RK3288_P01_Android
./build_rk3288_7.1.2.sh -C
会自动清除所有编译过程产生的文件。
```

三、编译 Linux 系统

1、解压源代码

将源代码压缩文件全部复制到 Ubuntu 系统下，使用以下命令解压源代码（注意参数中是大写 J）：

```
tar xvJf PET_RK3288_P01_Linux_Source.tar.xz
```

2、编译 buildroot

```
cd PET_RK3288_P01_Linux
./build.sh
```

```
create boot.img...done.
/root/Work/PET_RK3288_Linux/device/rockchip/rk3288/parameter-buildroot.tx
t
0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00
008000(misc),0x00010000@0x0000a000(boot),0x00010000@0x0001a000(recovery),
0x00010000@0x0002a000(backup),0x00020000@0x0003a000(oem),0x00001000@0x000
5a000(gzpeite),0x00200000@0x0005b000(rootfs),-@0x0025b000(userdata:grow)
 Image: image in rockdev is ready
Make image ok!
Make update.img
start to make update.img...
Android Firmware Package Tool v1.65
------ PACKAGE ------
Add file: ./package-file
Add file: ./Image/MiniLoaderAll.bin
Add file: ./Image/parameter.txt
Add file: ./Image/trust.img
Add file: ./Image/uboot.img
Add file: ./Image/misc.img
Add file: ./Image/boot.img
Add file: ./Image/recovery.img
Add file: ./Image/gzpeite.img
Add file: ./Image/rootfs.img
Add file: ./Image/oem.img
Add file: ./Image/userdata.img
Add CRC...
Make firmware OK!
------ OK ------
********RKImageMaker ver 1.66********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
/root/Work/PET_RK3288_Linux
Make update image ok!
```

编译完成后会在 rockdev 目录下生成 update_buildroot.img 烧写文件

3、编译 debian9

编译 debian 之前，需首先完成过 buildroot 的编译

编译主机需要能连接互联网，如果编译后的文件大小与 SDK 里面的差异较大需要检查联网情况。

如果是新的系统，需要执行一次下面的命令

```
cd PET_RK3288_P01_Linux
sudo dpkg -i debian/ubuntu-build-service/packages/*
sudo apt install -y -f
```

编译 debian9

```
cd PET_RK3288_P01_Linux
./build.sh debianall
```

```
create loader...done.
create boot.img...done.
/root/Work/PET_RK3288_Linux/device/rockchip/rk3288/parameter-debian.txt
0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00
gzpeite),0x00700000@0x0005b000(rootfs),-@0x0075b000(userdata:grow)
 Image: image in rockdev is ready
Make image ok!
Make update.img
start to make update.img...
Android Firmware Package Tool v1.65
------ PACKAGE ------
Add file: ./package-file
Add file: ./Image/MiniLoaderAll.bin
Add file: ./Image/parameter.txt
Add file: ./Image/trust.img
Add file: ./Image/uboot.img
Add file: ./Image/misc.img
Add file: ./Image/boot.img
Add file: ./Image/recovery.img
Add file: ./Image/gzpeite.img
Add file: ./Image/rootfs.img
Add file: ./Image/oem.img
Add file: ./Image/userdata.img
Add CRC...
Make firmware OK!
------ OK ------
********RKImageMaker ver 1.66********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
/root/Work/PET_RK3288_Linux
Make update image ok!
```

编译完成后会在 rockdev 目录下生成 update_debian.img 烧写文件

4、编译 ubuntu 18.04

编译 ubuntu 之前，需首先完成过 buildroot 的编译

编译主机需要能连接互联网，如果编译后的文件大小与 SDK 里面的差异较大需要检查联网情况。

cd PET_RK3288_P01_Linux

./build.sh ubuntuall

```
create boot.img...done.
/root/Work/PET_RK3288_Linux/device/rockchip/rk3288/parameter-ubuntu.txt
0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00
008000(misc),0x00010000@0x0000a000(boot),0x00010000@0x0001a000(recovery),
0x00010000@0x0002a000(backup),0x00020000@0x0003a000(oem),0x00001000@0x000
5a000(gzpeite),0x00A00000@0x0005b000(rootfs),-@0x00A5b000(userdata:grow)
 Image: image in rockdev is ready
Make image ok!
Make update.img
start to make update.img...
Android Firmware Package Tool v1.65
------ PACKAGE ------
Add file: ./package-file
Add file: ./Image/MiniLoaderAll.bin
Add file: ./Image/parameter.txt
Add file: ./Image/trust.img
Add file: ./Image/uboot.img
Add file: ./Image/misc.img
Add file: ./Image/boot.img
Add file: ./Image/recovery.img
Add file: ./Image/gzpeite.img
Add file: ./Image/rootfs.img
Add file: ./Image/oem.img
Add file: ./Image/userdata.img
Add CRC...
Make firmware OK!
------ OK ------
********RKImageMaker ver 1.66********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
/root/Work/PET_RK3288_Linux
Make update image ok!
```

编译完成后会在 rockdev 目录下生成 update_ubuntu.img 烧写文件

5、清理 Linux 源码

```
cd PET_RK3288_P01_ Linux
. /build.sh cleanall
会自动清除所有编译过程产生的文件。
```

四、镜像文件烧写

1、安装驱动并连接硬件

解压开发工具目录下的 DriverAssitant.7z，右击以管理员权限运行 DriverInstall.exe，安装驱动程序。

将主板与 PC 机用 MicroUSB 线连接好，如果出现无法识别的情况可以通过重新连接、更换 PC 机 USB 接口、更换 USB 线、更换 PC 机等方式重试。

解压开发工具目录下的 RKDevTool.7z，右击以管理员权限运行 RKDevTool.exe

2、进入烧写模式

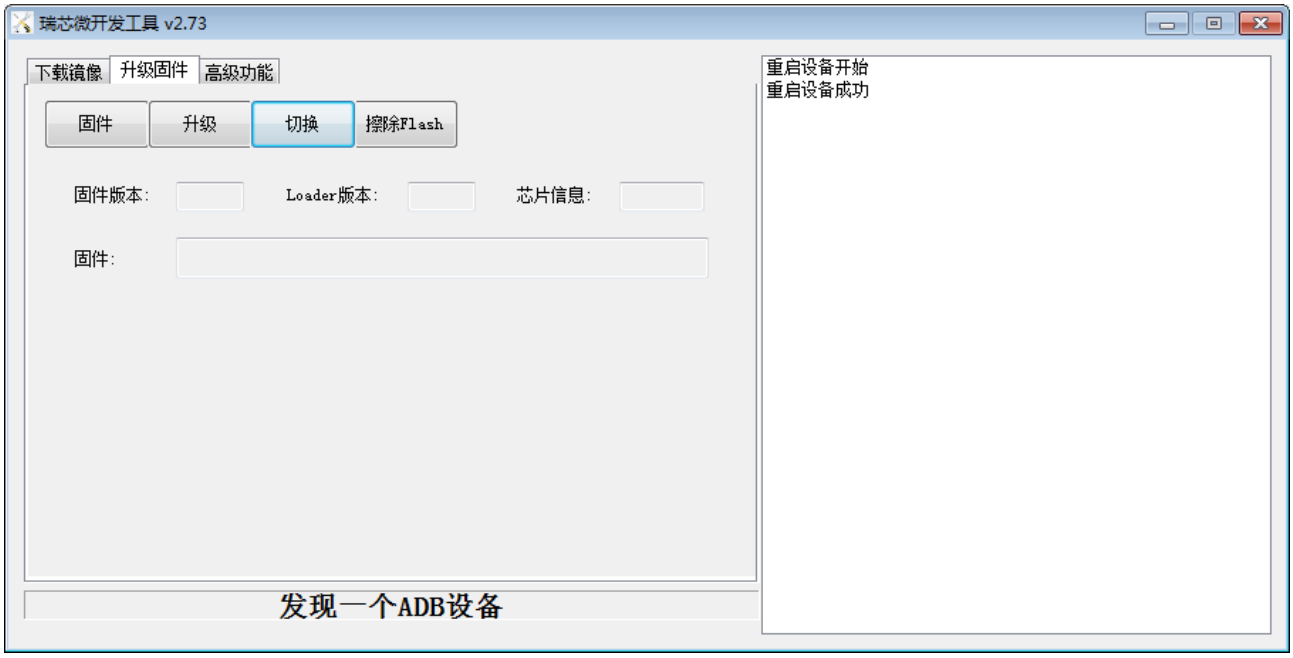
主板处于 Loader 或 Maskrom 模式时可以对系统进行格式化和烧写系统镜像文件操作。

进入 Loader 模式

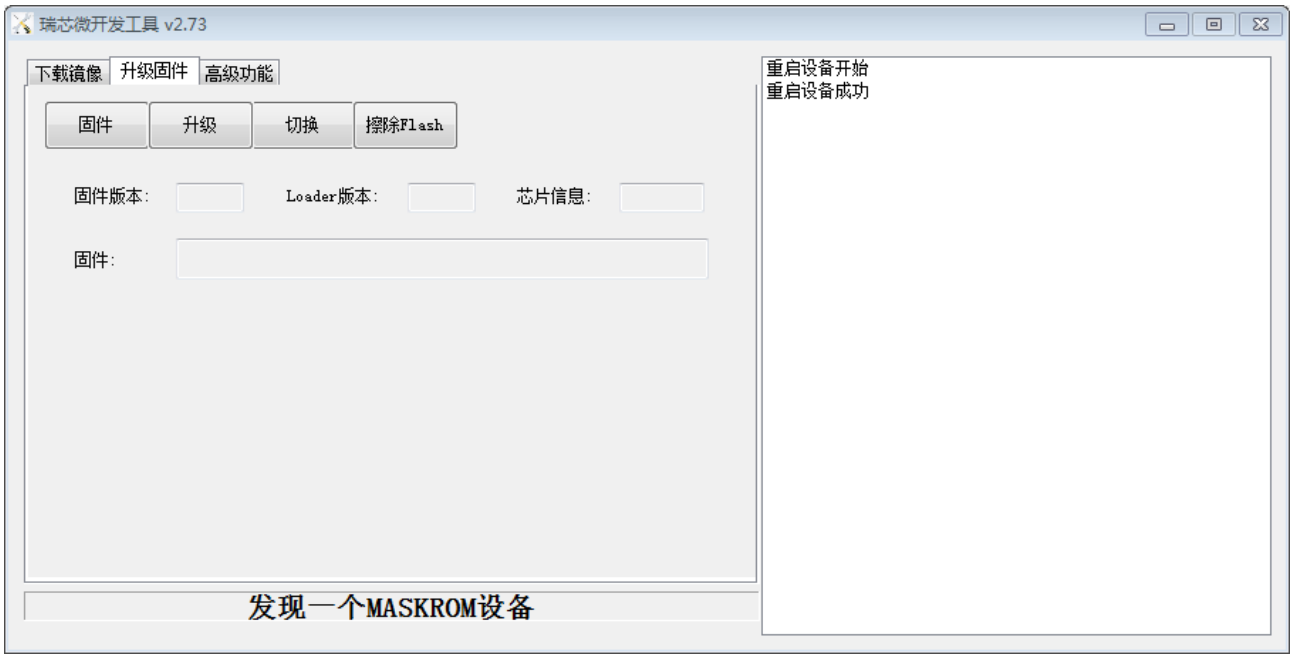
方式一、将主板断电，首先用 MicroUSB 线将主板与 PC 机连接好，按下主板上的 BOOT 键（Power 键旁边，三个按键中间那个），并保持按下状态，然后再上电开机，主板会进入 Loader 操作模式，注意这种方式适用于主板上的 BootLoader 可正常工作的情况。

方式二、将主板接通电源，进入系统，用 MicroUSB 线将主板与 PC 机连接好如下图，RKDevTool 软件会识别到一个 ADB 设备，单击切换按钮，主板会重启进入 Loader 模式

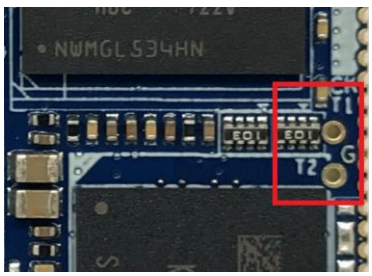
方式三、在调试串口控制台或其他控制终端输入 reboot loader，主板会重启进入 Loader 模式



进入 Maskrom 模式



一般仅在 uboot 损坏，无法进入 Loader 模式时使用。
首先准备好一个金属镊子
将主板断电
用镊子短接右图中的核心板上 T1 和 T2 两个沉金通孔
然后再上电开机，进入 Masktom 模式后取消短接，松开镊子
继续进行檫除 flash、升级固件等操作。



3、烧写系统镜像

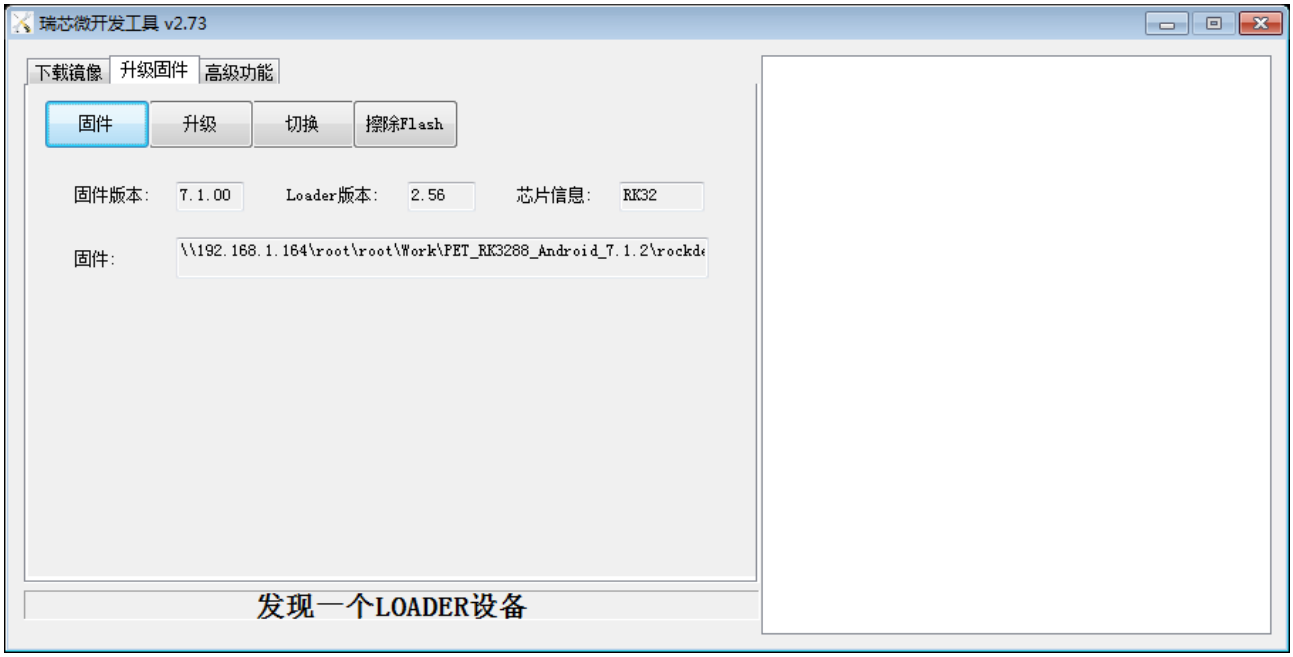
特别注意：

进行源码编译前，首先关闭镜像烧写软件 RKDevTool，同时检查系统是否存在 adb 进程，也同时结束掉（会打开占用编译源码需要更新的镜像文件）

在从 Android 系统切换到 Linux 系统，或 Linux 系统切换到 Android 系统时，进入 Loader 或 Maskrom 模式，选择好需要烧写的固件文件，首先点击 RKDevTool 的擦除 Flash 按键将 Flash 格式化一遍，否则可能会出现烧写失败或烧写成功但系统功能不正常（例如 HDMI 无法使用等等）的情况。

系统烧写流程

首先将主板进入 Loader 或 Maskrom 模式，打开 RKDevTool 软件，点击固件按钮选择需要烧写的镜像文件，然后点击升级按钮，右侧窗口会显示烧写进度。



五、联系方式

地址　　：广州市天河区大观中路新塘大街鑫盛工业园 A1 栋 201
电话　　：020-85625526
传真　　：020-85625526-606
主页　　：http://www.gzpeite.net
淘宝店　：https://shop149045251.taobao.com

商务洽谈：王先生
电话　　：18926288206
电子信箱：18926288206@gzpeite.net
业务 QQ：594190286

技术咨询：杨先生
微信　　：18902281981（请加微信）
电子信箱：18902281981@gzpeite.net
业务 QQ：151988801

广州佩特电子科技有限公司

2020 年 5 月