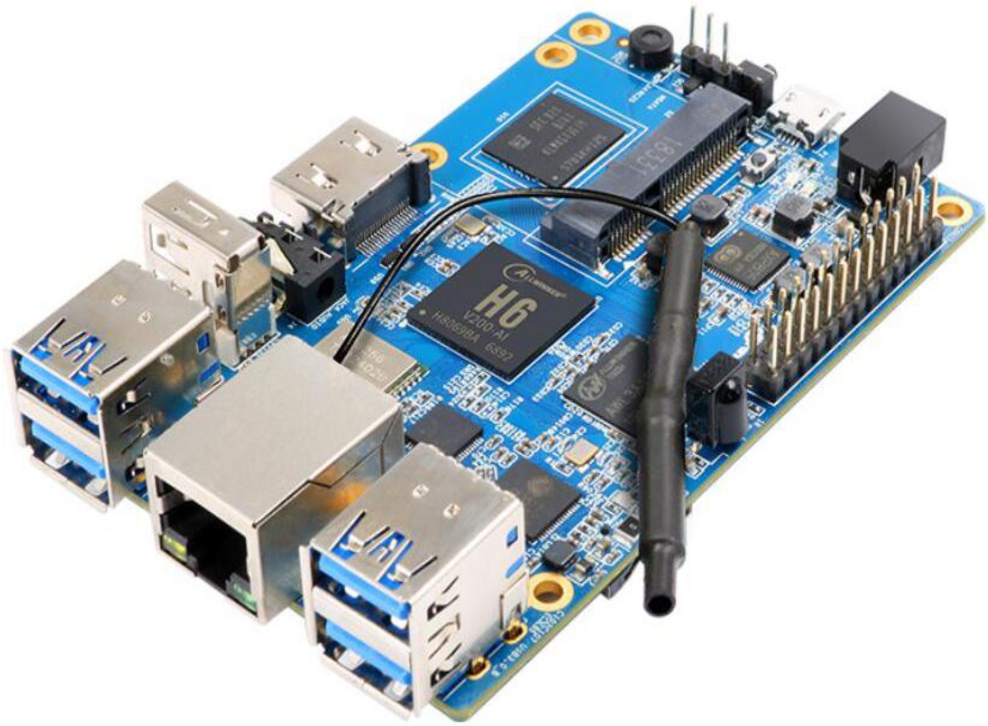
Orange Pi 3
User Manual



History

Ver	Data	Author	Brief	Publish	Memo
1.0	2019-01-24	Leeboby	Creat Document		

Contents

I. Orange Pi One Plus Introduction

1. What is Orange Pi 3?

It’s an open-source single-board computer, new arm64 development board. It can run Android 7.0, Ubuntu, Debian, etc. It uses AllWinner H6 SOC and has 1GB or 2GB LPDDR3 SDRAM.

2. What can I do with Orange Pi 3?

You can use it to build…

- A computer
- A wireless server
- Games
- Music and sounds
- HD video
- A speaker
- Android
- Scratch
- ......

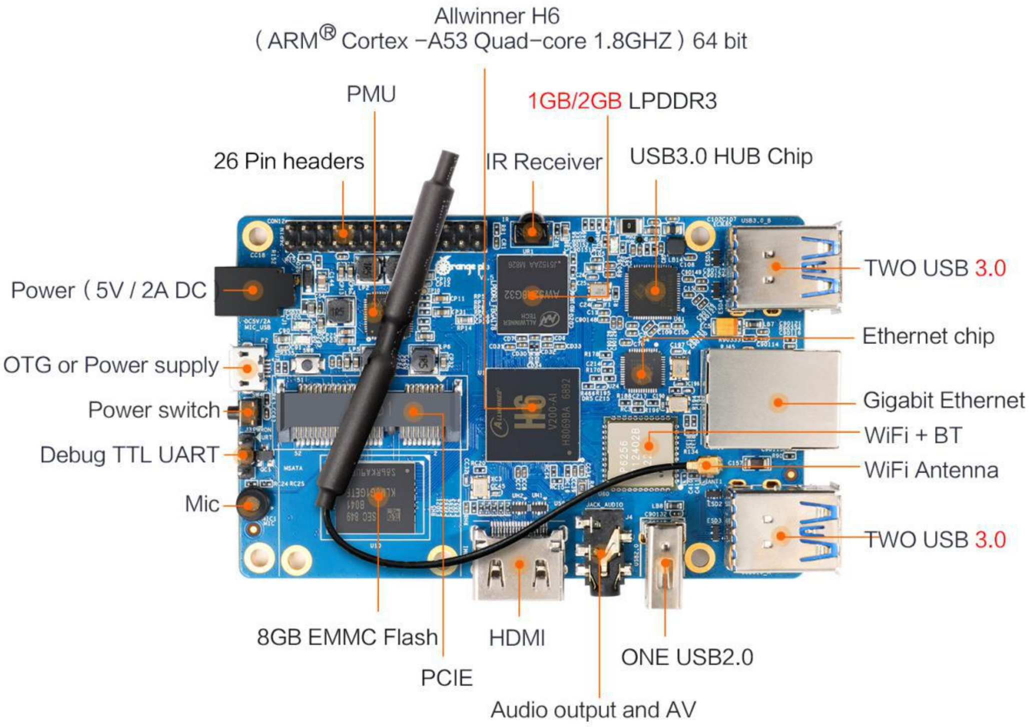
Pretty much anything else, because Orange Pi 3 is open source

3. Whom is it for?

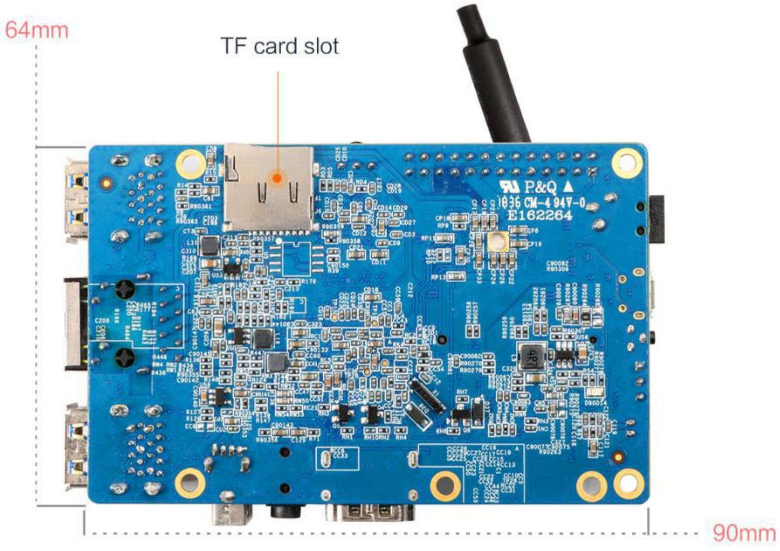
Orange Pi 3 is for anyone who wants to create with technology– not just consuming. It's a simple, fun, useful tool and you can use it to take control of the world around you.

Orange Pi 3: 1GB/2GB LPDDR3 with 8GB EMMC Flash on board

Top view

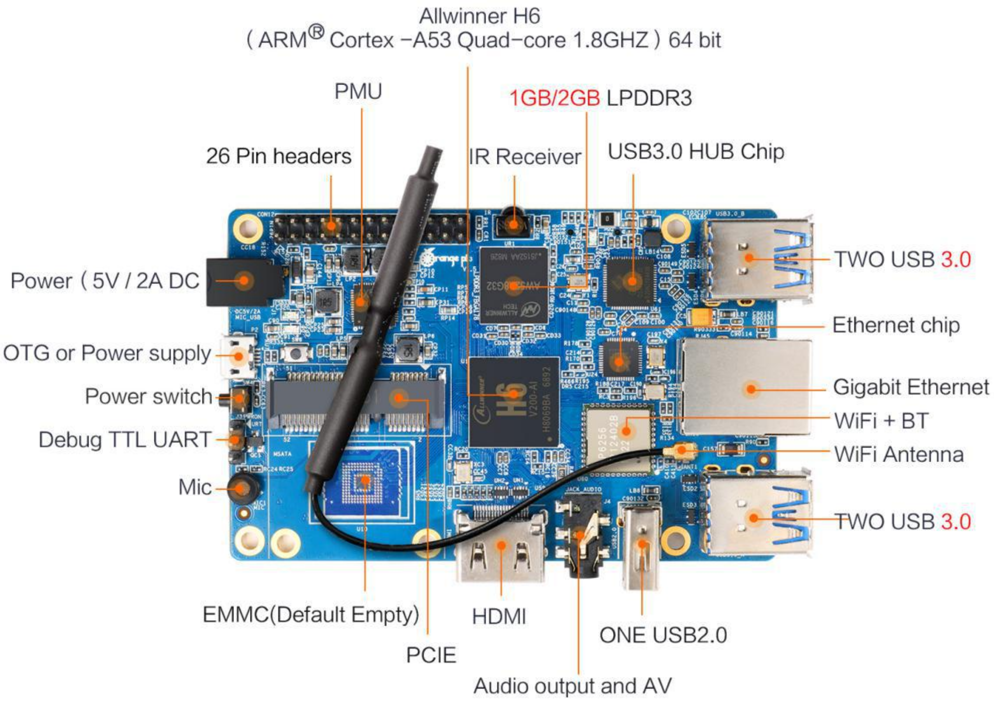


Bottom view

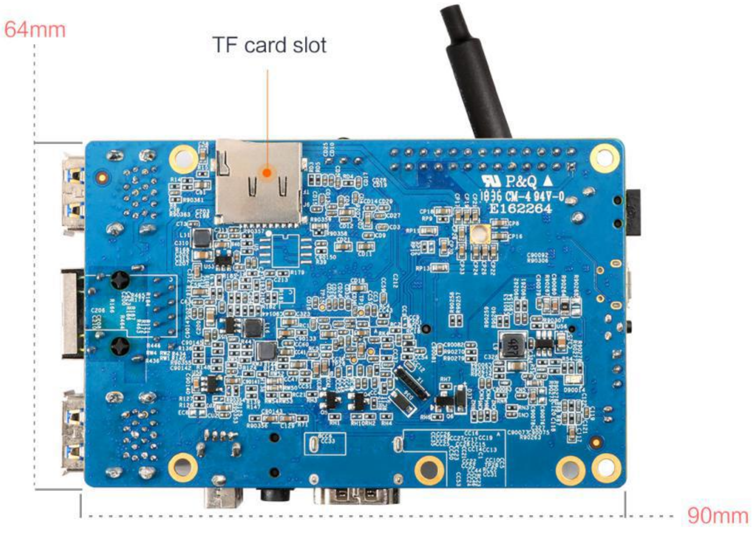


Orange Pi 3: 1GB/2GB LPDDR3 without 8GB EMMC Flash on board

Top view



Bottom view



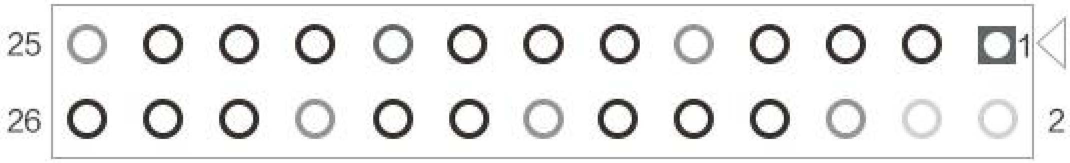
4. Hardware specification of Orange Pi 3

● **Hardware specification**	
CPU	H6 Quad-core 64-bit **1.8GHZ** ARM Cortex™-A53
GPU	• High-performance multi-core GPU Mali T720 • OpenGL ES3.1/3.0/2.0/1.1 • Microsoft DirectX 11 FL9_3 • ASTC(Adaptive Scalable Texture Compression) • Floating point operation greater than 70 GFLOPS
Memory+Onboard Storage	Four Types: 1GB LPDDR3 (shared with GPU)+EMMC(Default Empty) 2GB LPDDR3(shared with GPU)+EMMC(Default Empty) 1GB LPDDR3 (shared with GPU)+8GB EMMC Flash 2GB LPDDR3(shared with GPU)+8GB EMMC Flash
WIFI+BT	AP6256, IEEE 802.11 a/b/g/n/ac, BT5.0
Onboard Network	10/100M/1000M , ethernet RJ45
Network Chip	RTL8211
Audio Input	MIC
Audio Output	HDMI 2.0a and 3.5 mm AV Jack
Video Output	HDMI 2.0a and CVBS
Video Decoding	• H265/HEVC Main/Main10 profile@Level5.2 High-tier ;4K@60fps, up to 6Kx4K@30fps • H264/AVC BP/MP/HP@level5.1, MVC, 4K@30fps

	• VP9，Profile 0/2, 4K@30fps • AVS+/AVS JIZHUN profile@level 6.0, 1080P@60fps
PCIE	• Supports RC mode • Supports x1 Gen2(5.0Gbps) lane • Complies with PCI Express Base 2.0 Specification
Power Source	DC input，MicroUSB (OTG)
PMU	AXP805
USB 2.0 Ports	1*USB 2.0 Host, 1*USB OTG 2.0
USB 3.0 Ports	4*USB 3.0 Host
Low-level peripherals	26 Pin
GPIO(1x3) pin	UART, ground.
LED	Power LED、Status LED and USB3.0 LED
IR	YES
Key	Power(SW4)
Supported OS	Android7.0, Ubuntu, Debian
● **Interface definition**	
Product size	90mm*64mm
Weight	75g
Orange Pi$^{\text{TM}}$ is a trademark of the Shenzhen Xunlong Software CO., Limited	

5. GPIO Specifications

The picture below is GPIO pin definition of Orange Pi 3：



Orange Pi 3 GPIO definition		
CON12-P01	VCC-3.3V	VCC-IO
CON12-P02	VCC-5V	DCIN
CON12-P03	TWI0-SDA	PD26
CON12-P04	VCC-5V	DCIN
CON12-P05	TWI0-SCK	PD25
CON12-P06	GND	GND
CON12-P07	PWM0	PD22
CON12-P08	S-UART-TX	PL02
CON12-P09	GND	GND
CON12-P10	S-UART-RX	PL03
CON12-P11	UART3-RX	PD24
CON12-P12	PD18	PD18
CON12-P13	UART3-TX	PD23
CON12-P14	GND	GND
CON12-P15	PL10	PL10
CON12-P16	PD15	PD15
CON12-P17	VCC-3.3V	VCC-IO
CON12-P18	PD16	PD16
CON12-P19	SPI1_MOSI	PH05
CON12-P20	GND	GND
CON12-P21	SPI1_MISO	PH06
CON12-P22	PD21	PD21
CON12-P23	SPI1_CLK	PH04
CON12-P24	SPI1_CS	PH03
CON12-P25	GND	GND
CON12-P26	PL08	PL08

II. Using Method Introduction

1. Hardware and Software Requirement

Hardware Requirements:

- Orange Pi 3
- TF card in min 8GB and class 10, recommend using a famous brand TF card, such as Sandisk 16GB TF card
- A PC for compilation with following specs:
  64 bit CPU
  Up to 8GB RAM
  Up to 100GB spare disk space
  Operation system at **Ubuntu14.04** would be better

Software Requirements:

- Orange Pi 3 SDK
- Orange Pi 3 Firmware
- Android and Linux flash tool

Below files could be downloaded from Github and Mega, more details pls visit our website:

https://github.com/orangepi-xunlong
http://www.orangepi.org/downloadresources/

2. Power Supply Requirement

There are two methods for power supply:

- DC (5V 2A) in for power
- Micro USB(5V 2A)OTG in for power

III. Android Compilation Environment Construction

The following operations are based on Ubuntu 14.04, there might be some difference if you are working with Ubuntu or Linux distro。

1. Download SDK compression package

After download Android SDK, you need to first packed several compressed files into one package, then unzip the package just packed.

```
$ mkdir OrangePi_3
$ cat H6-2018-1-2.tar.gza* > OrangePi_3.tar
$ tar xf OrangePi_3.tar -C OrangePi_3
```

2. Construct Compilation Environment

● Install JDK

It could only works on version openjdk8 for Android 7.0 compilation, it would cause failure if the version not openjdk8. Commands for installing Openjdk-8 are as follows:

```
$ sudo add-apt-repository ppa:openjdk-r/ppa
$ sudo apt-get update
$ sudo apt-get install openjdk-8-jdk
```

● Configure environment variable of JAVA

If the installation path like this: **/usr/lib/jvm/java-8-openjdk-amd64**, then you could operate the following command in the terminal:

```
$ export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
$ export PATH=$JAVA_HOME/bin:$PATH
$ export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

● Install Software Package

For Ubuntu14.04：

```
$ sudo apt-get update
$ sudo apt-get install git gnupg flex bison gperf build-essential \
zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 \
lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z1-dev ccache \
libgl1-mesa-dev libxml2-utils xsltproc unzip

$ sudo apt-get install u-boot-tools
```

3. Compilation of SDK Source Code

There would be android and lichee two directories after unzipped SDK package, contents on lichee are as following

```
lichee/brandy/u-boot-2014.07                        #uboot code directory
lichee/bootloader/uboot_2014_sunxi_spl              #boot0 code directory
lichee/linux-3.10                        #kernel code
lichee/tools                                   #Hardware specs, packing tools, etc.
```

● Kernel compile steps

Input the following commands on lichee directory：

```
$ cd OrangePi_3/lichee
$ ./build.sh config

Welcome to mkscript setup progress
All available chips:
    0. sun50iw1p1
    1. sun50iw2p1
    2. sun50iw6p1
    3. sun8iw11p1
    4. sun8iw12p1
    5. sun8iw6p1
    6. sun8iw7p1
    7. sun8iw8p1
    8. sun9iw1p1
Choice: 2
All available platforms:
    0. android
    1. dragonboard
    2. linux
    3. eyeseelinux
Choice: 0
All available business:
    0. 5.1
    1. 4.4
    2. 7.x
Choice: 2
```

After the above compilation, you will get the following output：

```
regenerate rootfs cpio
15757 blocks
17099 blocks
build_ramfs
Copy boot.img to output directory ...
Copy modules to target ...

sun50iw6p1 compile Kernel successful

INFO: build kernel OK.

INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: build rootfs OK.
----------------------------------------
build sun50iw6p1 android 7.x lichee OK
----------------------------------------
```

Kernel code is on the directory of lichee/linux-3.10.The system would copy the configure file from lichee/linux-3.10/arch/arm64/configs/sun50iw6p1smp_android_7.x_defconfig to lichee/linux-3.10/.config as default configure when you input the above compilation command.

In the next compilation, you could run ./build.sh on lichee directory and go on with previous .config configure.

- **uboot/boot0 Configure Steps(Optional)**

Usually there is no need to re-compile uboot, if you want to make some custom modification, then please refer to the following：

```
cd lichee/brandy/u-boot-2014.07
make distclean && make sun50iw6p1_config && make -j5   #编译 uboot

cd lichee/brandy/u-boot-2014.07
make distclean && make sun50iw6p1_config && make spl   #编译 boot0
```

If you do not compile uboot/boot0, it is default running with lichee/tools/pack/chips/sun50iw6p1/bin. It would be replaced the default command if recompiled with the above command.

- **Building Android from source code**

```
$ cd android
$ source ./build/envsetup.sh
$ lunch petrel_fvd_p1-eng
$ extract-bsp
$ make -j8 && pack
```

The pack command is to pack it into firmware, if it is packed successfully, then there would be the following information：

```
Dragon execute image.cfg SUCCESS !
----------image is at----------

OrangePi_3/lichee/tools/pack/sun50iw6p1_android_petrel-p1_uart0.img

pack finish
```

According to the above prompt, you could check the generated firmware of sun50iw6p1_android_petrel-p1_uart0.img on the directory of OrangePi_3/lichee/tools/pack/. About Android image flashing, you could refer to the section of Android image flashing

IV. Linux Environment Construction

1. Download SDK compression package

● Orange Pi Linux Source Downloader

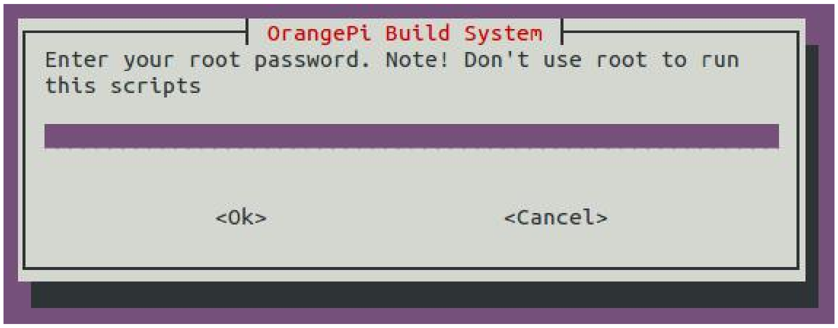
Orange Pi 3 Linux Source Code have been uploaded to GitHub, the kernel version is **Linux 3.10.** We could use the OrangePi Linux source-specific downloader for download as follow:

```
$ sudo apt-get install git
$ git clone https://github.com/orangepi-xunlong/OrangePi_Build.git
$ cd OrangePi_Build
$ ls
Build_OrangePi.sh    lib    README.md
```

● Run Downloader

```
$ ./Build_OrangePi.sh
```

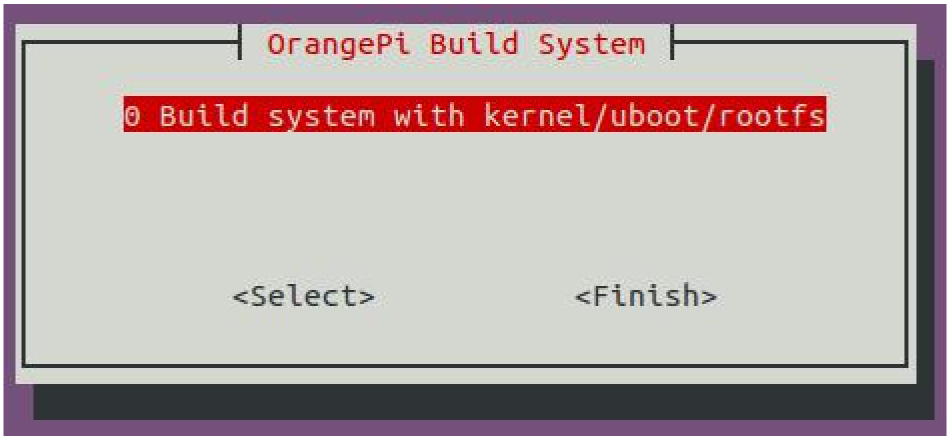
Input password of root, then enter to next step



OrangePi Build System

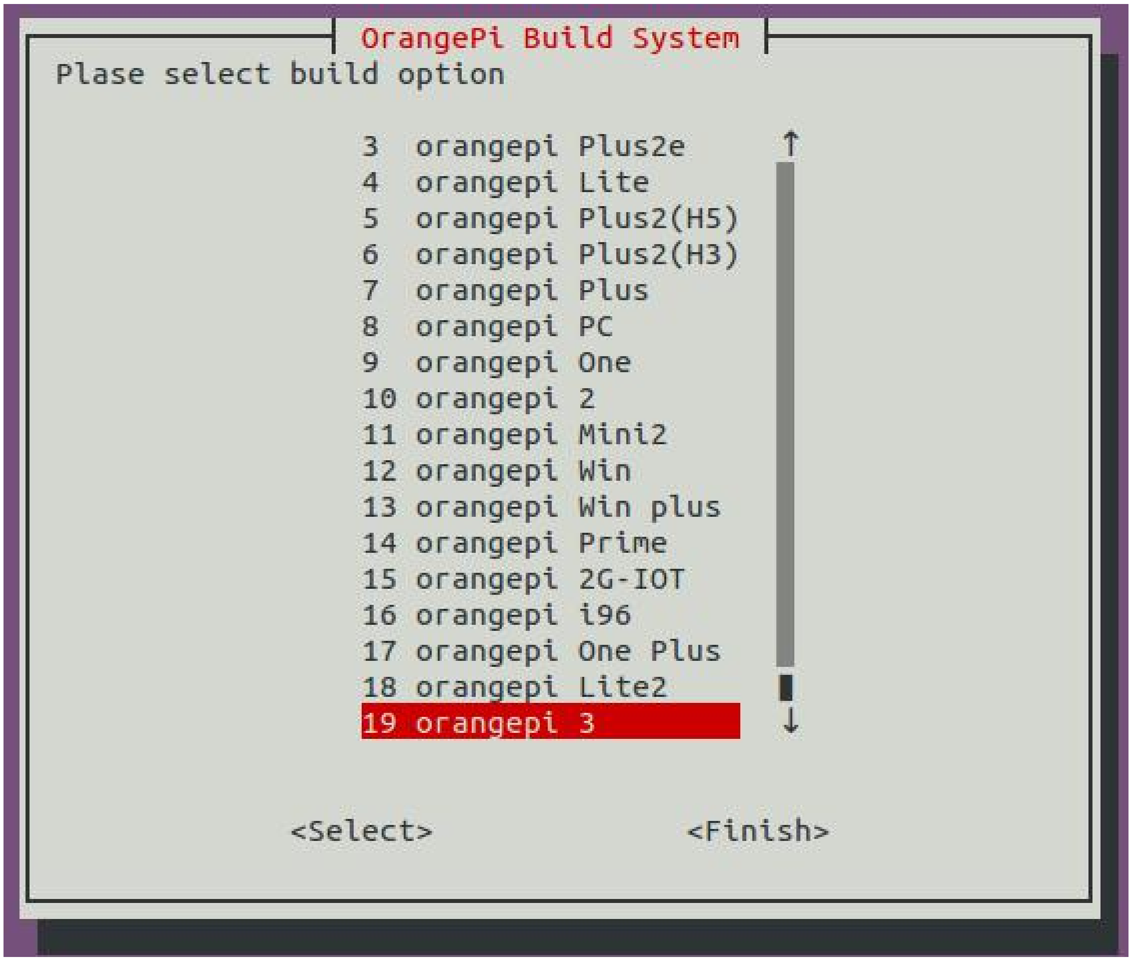
Enter your root password. Note! Don't use root to run this scripts

Choose **0 Build system with kernel/uboot/rootfs** enter the interface of model selection of development board



```
OrangePi Build System

0 Build system with kernel/uboot/rootfs

<Select>          <Finish>
```

Choose **19 orangepi 3, click enter, then it will be started to download the linux SDK for Orange Pi 3**



```
OrangePi Build System
Plase select build option

3  orangepi Plus2e
4  orangepi Lite
5  orangepi Plus2(H5)
6  orangepi Plus2(H3)
7  orangepi Plus
8  orangepi PC
9  orangepi One
10 orangepi 2
11 orangepi Mini2
12 orangepi Win
13 orangepi Win plus
14 orangepi Prime
15 orangepi 2G-IOT
16 orangepi i96
17 orangepi One Plus
18 orangepi Lite2
19 orangepi 3

<Select>          <Finish>
```

The downloaded source code will be stored in OrangePi_Build's peer directory

```
$ ls ../OrangePi_Build
 OrangePi_Build OrangePiH6
```

2. Construct Compilation Environment

Linux directory construction of OrangePi H6

```
$ cd OrangePiH6
$ tree -L 1
.
├── build.sh -> scripts/build.s     Compile boot scripts
├── external                                  Store external configure file
├── kernel                                    Linux kernel source code
├── output                                    Store output files
├── scripts                                   Scripts file used in the compilation
├── toolchain                   Cross compiler tool chain used by the kernel and u-boot
└── uboot                                     Store source code of boot0 and u-boot

6 directories, 1 file
```

The directory structure of cross compile tool chain is shown below. If the source code is different from it, or it is empty in the directory of toolchain,which means there are errors during the download, please download the source code again with OrangePi_Build

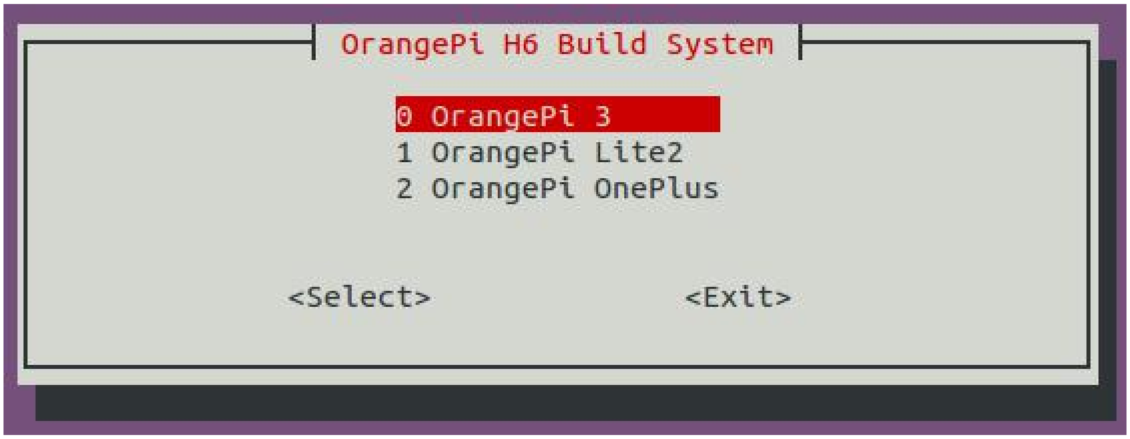
```
$ cd toolchain
$ tree -L 2
.
└── gcc-linaro-aarch
    ├── aarch64-linux-gnu
    ├── bin
    ├── gcc-linaro                     Store tool train for u-boot compile
    ├── gcc-linaro-4.9-4.9-2015.01-3-2015.02-3-manifest.txt
    ├── include
    ├── lib
    ├── libexec
    └── share

8 directories, 1 file
```

3. Configure Linux and U-boot Source Code

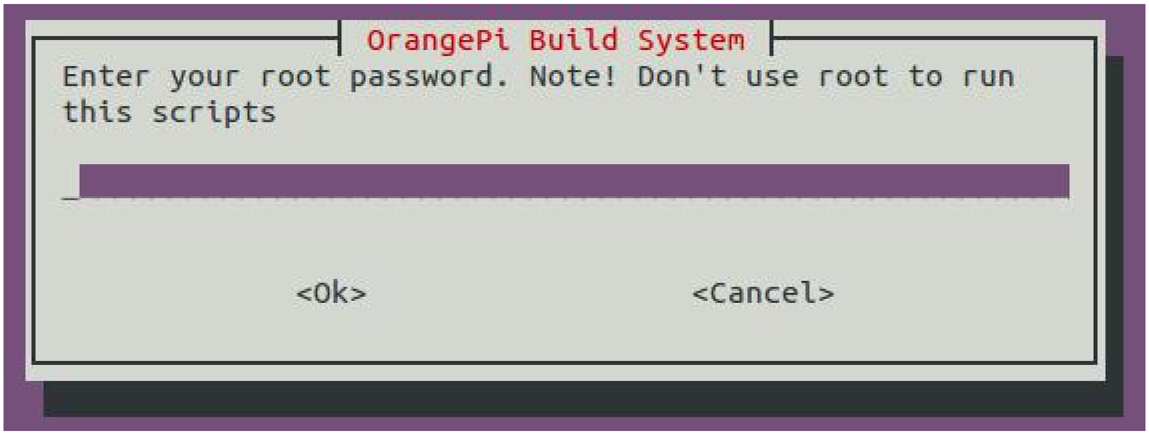
Execute compile-booting scripts

```
$ cd OrangePiH6
$ ./build.sh
```

Choose **0 OrangePi 3 and enter**



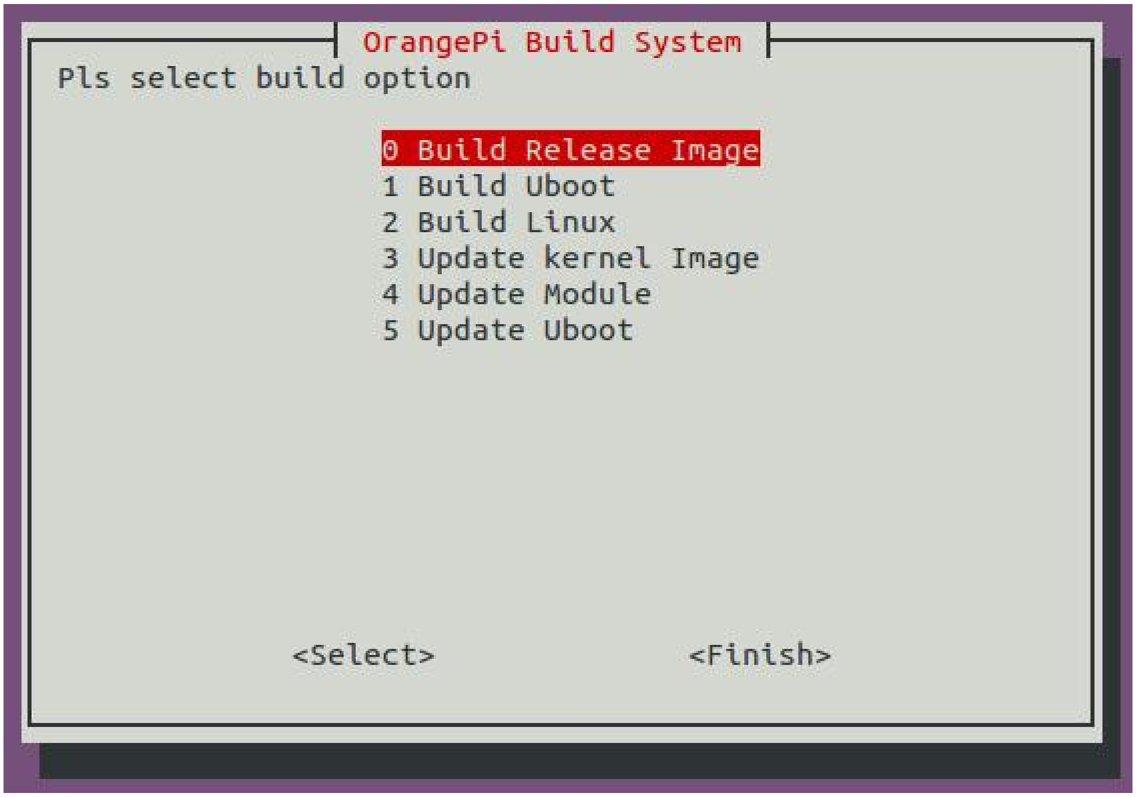
Input password of root and Enter, the select the function you want to run



Enter your root password. Note! Don't use root to run this scripts

Here are explanations and functions of the selections：

- **0 Build Release Image —— Compile the release ubuntu or debian image**
- **1 Build Uboot —— Compile U-boot and boot0 source code**
- **2 Build Linux —— Compile linux kernel source code**
- **3 Update kernel Image —— Update kernel of Linux on SD Card and OrangePiH6.dtb file**
- **4 Update Module —— Update kernel module of Linux on SD card**
- **5 Update Uboot —— Update boot0 and U-boo of Linux on SD card**



You need to first compile kernel source code before compile U-boot, otherwise the dtc program maybe cannot find.

The generated file will store on output directory

```
$ cd output
$ tree -L 2
.
├── boot0.bin
├── initrd.img
├── lib
│   └── modules
├── OrangePiH6.dtb
├── pack
│   └── out
├── uboot.bin
├── uEnv.txt
└── uImage

4 directories, 6 files
```

4. Linux SDK Usage Sample

We will make an example of Linux SDK usage with adding **rtl8812AU** USB WIFI kernel module on kernel source code.

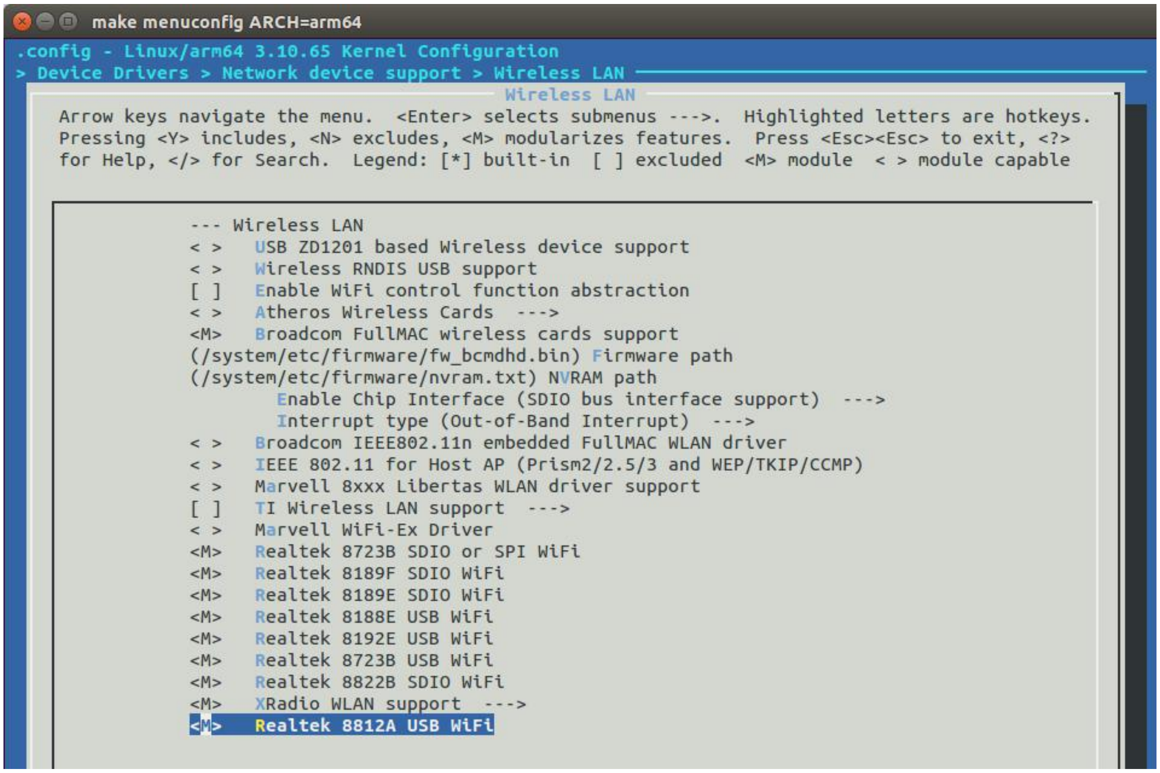
- **Download source code of rtl8812AU from github**

```
$ cd OrangePiH6/kernel/drivers/net/wireless
$ git clone https://github.com/diederikdehaas/rtl8812AU.git
Cloning into 'rtl8812AU'...
remote: Counting objects: 2347, done.
Receiving objects: 100% (2347/2347), 7.87 MiB | 22.00 KiB/s, done.
Resolving deltas: 100% (1292/1292), done.
Checking connectivity... done.
```

- **Add rtl8812AU compilation**

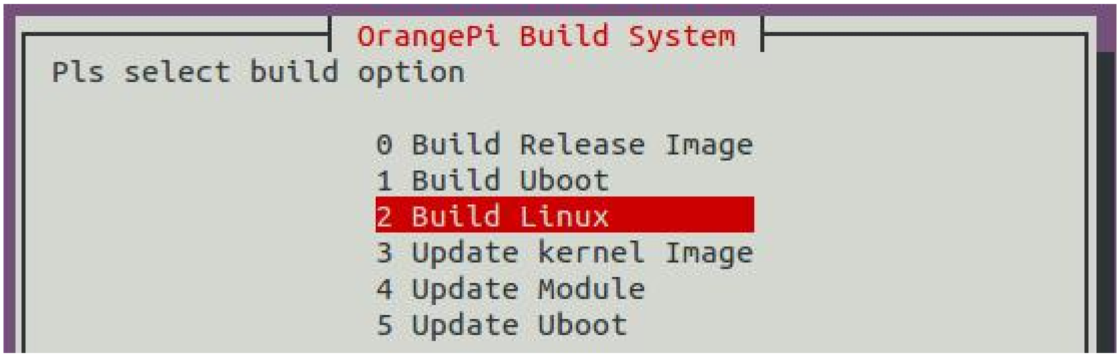
```
$ cd OrangePiH6/kernel/drivers/net/wireless
$ git diff .
diff --git a/drivers/net/wireless/Kconfig b/drivers/net/wireless/Kconfig
index 373666b..b7ebd5c 100755
--- a/drivers/net/wireless/Kconfig
+++ b/drivers/net/wireless/Kconfig
@@ -294,4 +294,5 @@ source "drivers/net/wireless/rtl8192eu/Kconfig"
+source "drivers/net/wireless/rtl8812AU/Kconfig"
  endif # WLAN

diff --git a/drivers/net/wireless/Makefile b/drivers/net/wireless/Makefile
index fd8a466..3aef800 100755
--- a/drivers/net/wireless/Makefile
+++ b/drivers/net/wireless/Makefile
@@ -66,3 +66,4 @@ obj-$(CONFIG_XR_WLAN) += xradio/
+obj-$(CONFIG_RTL8812AU)              += rtl8812AU/
```

- **Select Realtek 8812A USB WiFi on kernel configure, and compile into kernel module**



```
make menuconfig ARCH=arm64
.config - Linux/arm64 3.10.65 Kernel Configuration
> Device Drivers > Network device support > Wireless LAN ──────────────────────
  ┌──────────────────────────── Wireless LAN ─────────────────────────────┐
  │ Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.
  │ Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?>
  │ for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  │
  │ ┌─────────────────────────────────────────────────────────────────────┐
  │ │            --- Wireless LAN
  │ │            < >   USB ZD1201 based Wireless device support
  │ │            < >   Wireless RNDIS USB support
  │ │            [ ]   Enable WiFi control function abstraction
  │ │            < >   Atheros Wireless Cards  --->
  │ │            <M>   Broadcom FullMAC wireless cards support
  │ │            (/system/etc/firmware/fw_bcmdhd.bin) Firmware path
  │ │            (/system/etc/firmware/nvram.txt) NVRAM path
  │ │                    Enable Chip Interface (SDIO bus interface support)  --->
  │ │                    Interrupt type (Out-of-Band Interrupt)  --->
  │ │            < >   Broadcom IEEE802.11n embedded FullMAC WLAN driver
  │ │            < >   IEEE 802.11 for Host AP (Prism2/2.5/3 and WEP/TKIP/CCMP)
  │ │            < >   Marvell 8xxx Libertas WLAN driver support
  │ │            [ ]   TI Wireless LAN support  --->
  │ │            < >   Marvell WiFi-Ex Driver
  │ │            <M>   Realtek 8723B SDIO or SPI WiFi
  │ │            <M>   Realtek 8189F SDIO WiFi
  │ │            <M>   Realtek 8189E SDIO WiFi
  │ │            <M>   Realtek 8188E USB WiFi
  │ │            <M>   Realtek 8192E USB WiFi
  │ │            <M>   Realtek 8723B USB WiFi
  │ │            <M>   Realtek 8822B SDIO WiFi
  │ │            <M>   XRadio WLAN support  --->
  │ │            <M>   Realtek 8812A USB WiFi
```

- **You could recompile kernel according to the section of configure Linux and U-boot source code on Linux Environment Construction**



```
 ┌─────────────┤ OrangePi Build System ├─────────────┐
 │ Pls select build option
 │
 │                   0 Build Release Image
 │                   1 Build Uboot
 │                   2 Build Linux
 │                   3 Update kernel Image
 │                   4 Update Module
 │                   5 Update Uboot
```

Part of Log would show like the following：

```
Start Compile.....
Start Compile Module
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_cmd.o
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_security.o
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_debug.o
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_io.o
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_ioctl_query.o
  CC [M]    drivers/net/wireless/rtl8812AU/core/rtw_ioctl_set.o
```

The compiled module would be listed on **output/lib/modules/3.10.65+/kernel/drivers/net/wireless/rtl8812AU** after compiled.

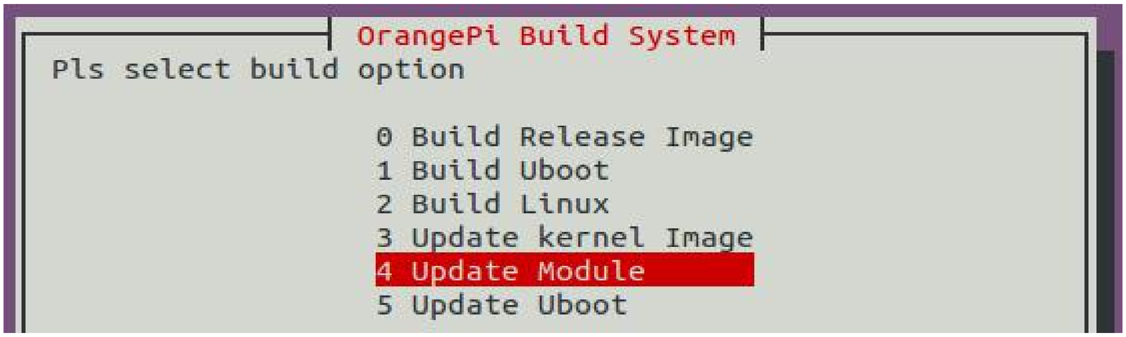
```
$ cd output/lib/modules/3.10.65+/kernel/drivers/net/wireless/rtl8812AU
$ ls
8812au.ko
```

- ## Update kernel module

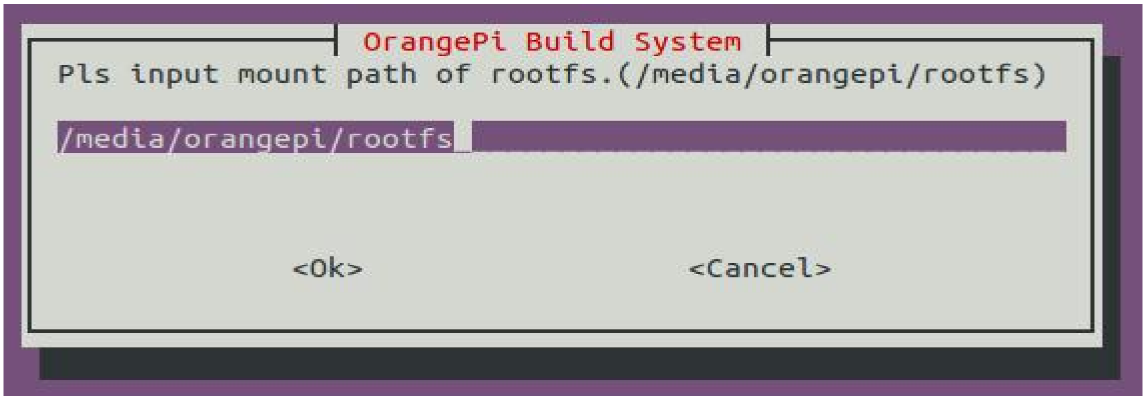
Insert SD card with Linux firmware into PC(installed Ubuntu 14.04 virtual or virtual PC), when the system recognized and mounted SD card, you could check corresponding partition name on **/media/$LOGNAME**

```
$ cd /media/$LOGNAME
$ ls
BOOT                    Store kernel and OrangePiH6.dtb file
rootfs        Rootfs file system
```

Refer to the section of compile Linux and U-boot on Linux environment construction, then select **4 Update Module** to update kernel module



Enter path of root file partition, and Enter, the scripts will copy kernel module into SD card auto.



Pls input mount path of rootfs.(/media/orangepi/rootfs)

/media/orangepi/rootfs

Boot the board with SD card with USB WIFI card driver of 8812au.ko kernel module.

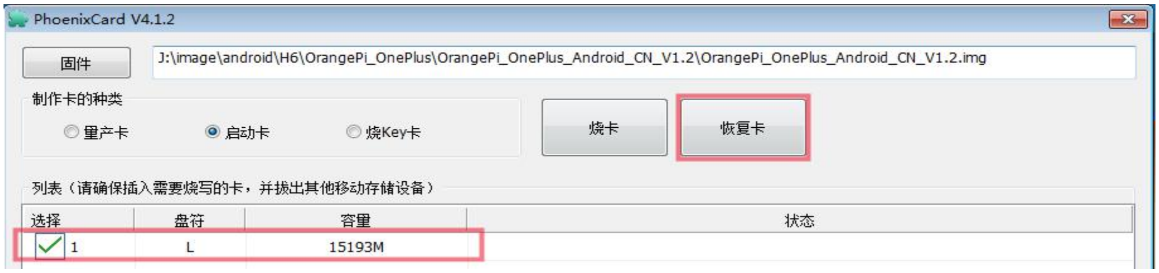
V． Android Firmware Flash

Android firmware **cannot** be flashed into SD card with dd command or writing with Win32 Diskimager on Windows. PhoenixCard card could be used for Android firmware flash. The latest version of PhoenixCard is **PhoenixCard V4.1.2**

1． Steps for Android Firmware Flashing(Boot from SD Card)

● Format TF Card

Check whether card disk as same for TF card and selected disk, click restore to format SD card

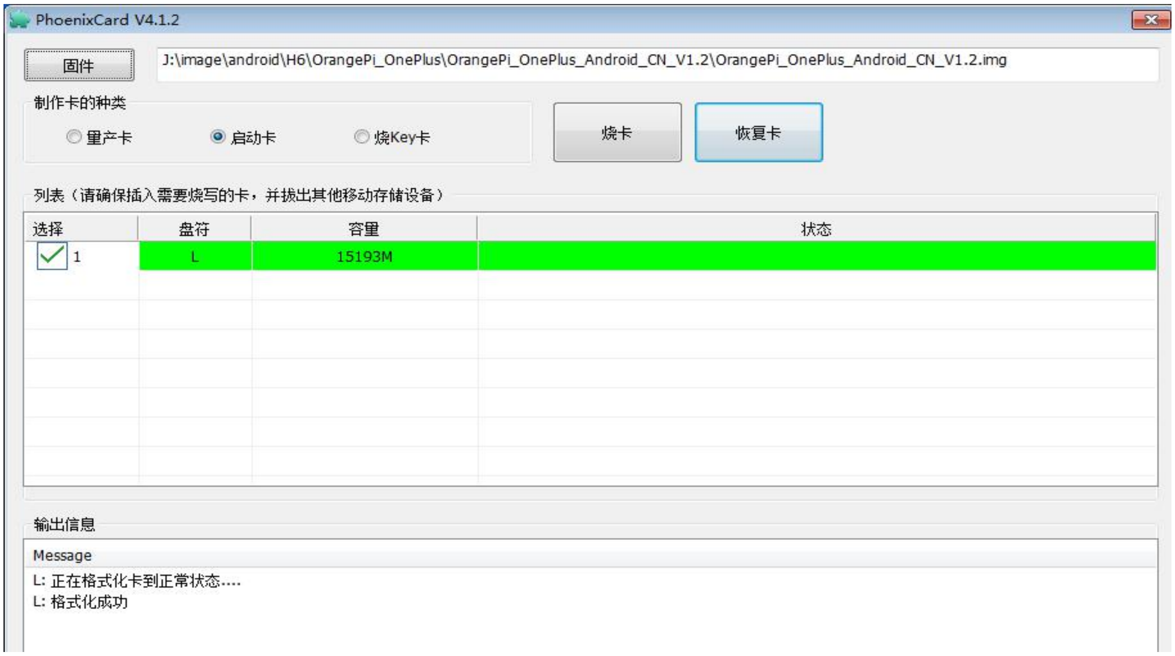


固件: J:\image\android\H6\OrangePi_OnePlus\OrangePi_OnePlus_Android_CN_V1.2\OrangePi_OnePlus_Android_CN_V1.2.img

制作卡的种类: ○ 量产卡 ● 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	L	15193M	



固件: J:\image\android\H6\OrangePi_OnePlus\OrangePi_OnePlus_Android_CN_V1.2\OrangePi_OnePlus_Android_CN_V1.2.img

制作卡的种类: ○ 量产卡 ● 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

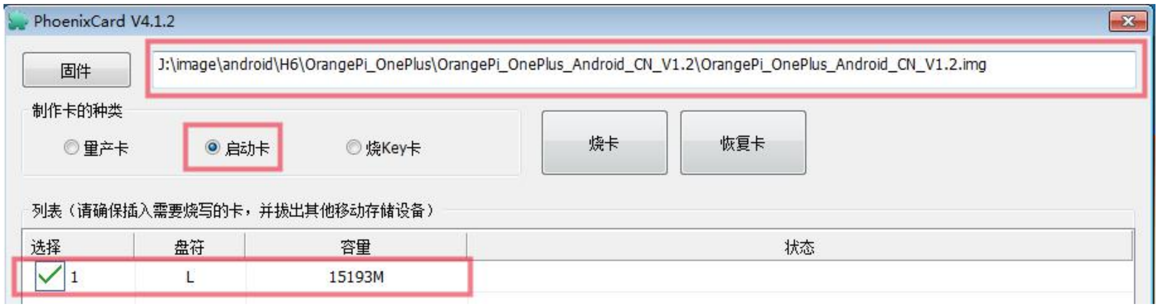
选择	盘符	容量	状态
☑ 1	L	15193M	🟩

输出信息

```
Message
L: 正在格式化卡到正常状态....
L: 格式化成功
```

● Choose Firmware and select Start up

Please note the following picture in RED notes



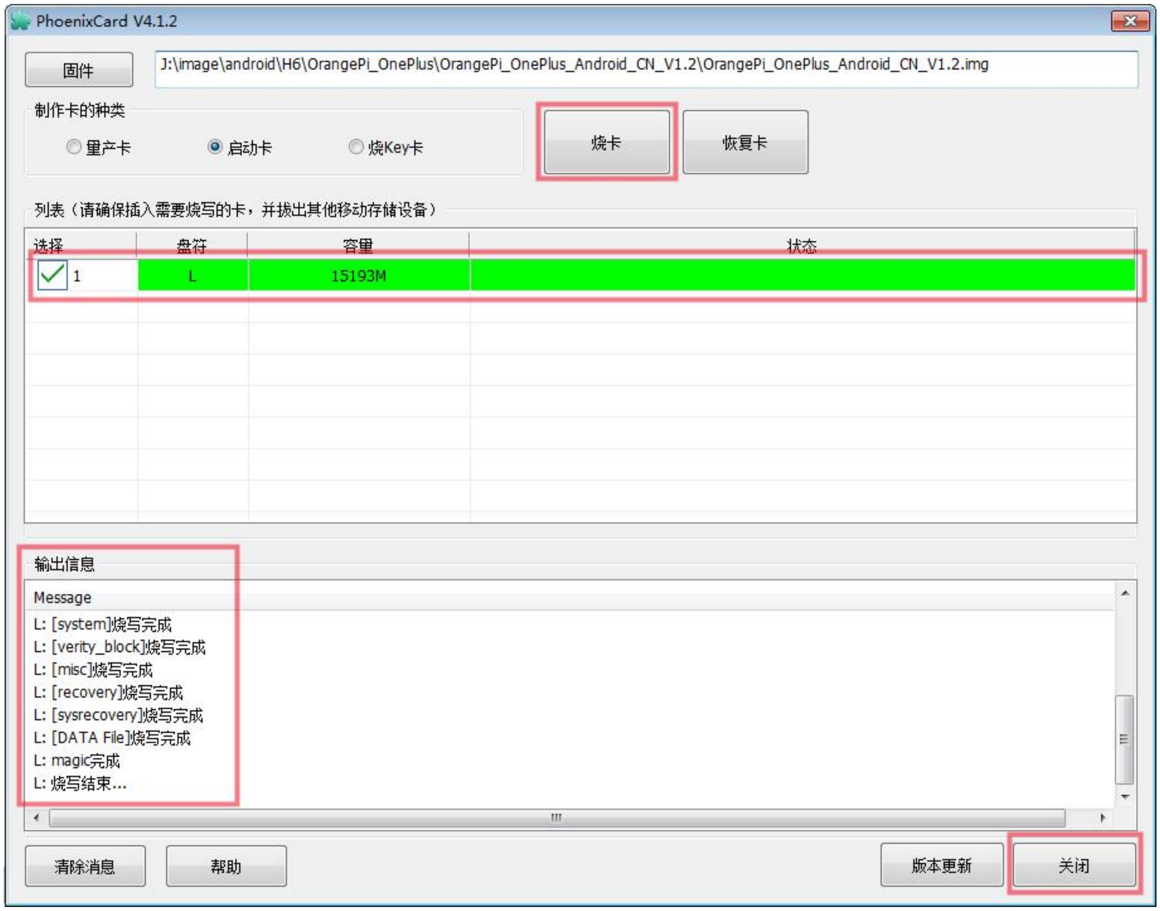
J:\image\android\H6\OrangePi_OnePlus\OrangePi_OnePlus_Android_CN_V1.2\OrangePi_OnePlus_Android_CN_V1.2.img

○ 量产卡 ● 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	L	15193M	

- **Click ‘burn-烧卡’ enter into image flashing into SD card**



J:\image\android\H6\OrangePi_OnePlus\OrangePi_OnePlus_Android_CN_V1.2\OrangePi_OnePlus_Android_CN_V1.2.img

○ 量产卡 ● 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	L	15193M	

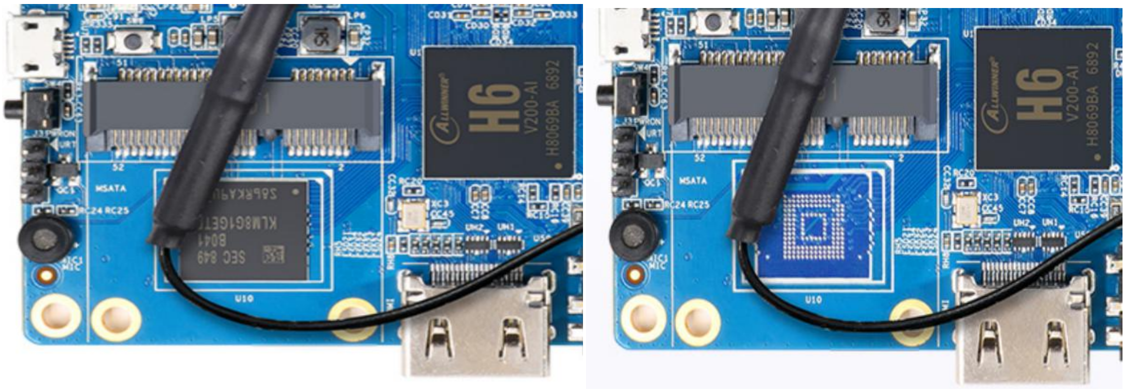
```
Message
L: [system]烧写完成
L: [verity_block]烧写完成
L: [misc]烧写完成
L: [recovery]烧写完成
L: [sysrecovery]烧写完成
L: [DATA File]烧写完成
L: magic完成
L: 烧写结束...
```

After finished Android flash, click close and you could use this SD card with written image to boot.

2. Steps for Android Firmware Flashing(Boot from EMMC Flash with SD Card)

If you bought the orange pi 3 with EMMC flash slodered on board, then you could flash the Android image into EMMC with SD Card, and then boot it from EMMC. If you bought the orange pi 3 without EMMC Flash soldered on board, then you could boot it from sd card only.

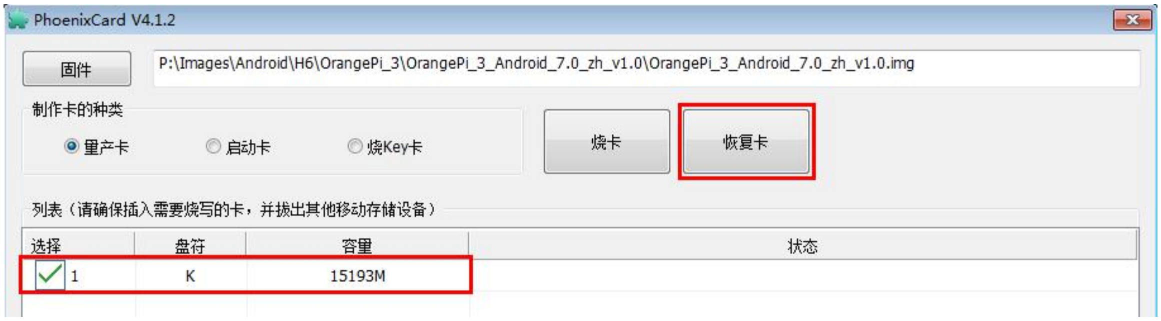
As shown in the figure below, the Orange Pi 3 development board with EMMC Flash chip is on the left, but not on the right



The Steps of flashing image into EMMC flash with SD card are as follow：

- **Format TF Card**

Check whether card disk as same for TF card and selected disk, click restore to format SD card



PhoenixCard V4.1.2

固件 P:\Images\Android\H6\OrangePi_3\OrangePi_3_Android_7.0_zh_v1.0\OrangePi_3_Android_7.0_zh_v1.0.img

制作卡的种类

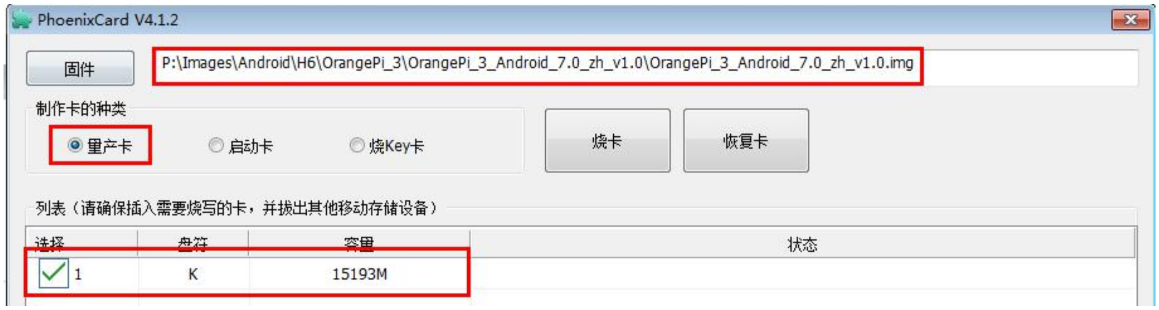
◉ 量产卡 ○ 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	K	15193M	

- **Choose Android image of Orange Pi 3, and then click the ‘量产卡-Product’**

Note the red marking in the following picture:



PhoenixCard V4.1.2

固件 P:\Images\Android\H6\OrangePi_3\OrangePi_3_Android_7.0_zh_v1.0\OrangePi_3_Android_7.0_zh_v1.0.img

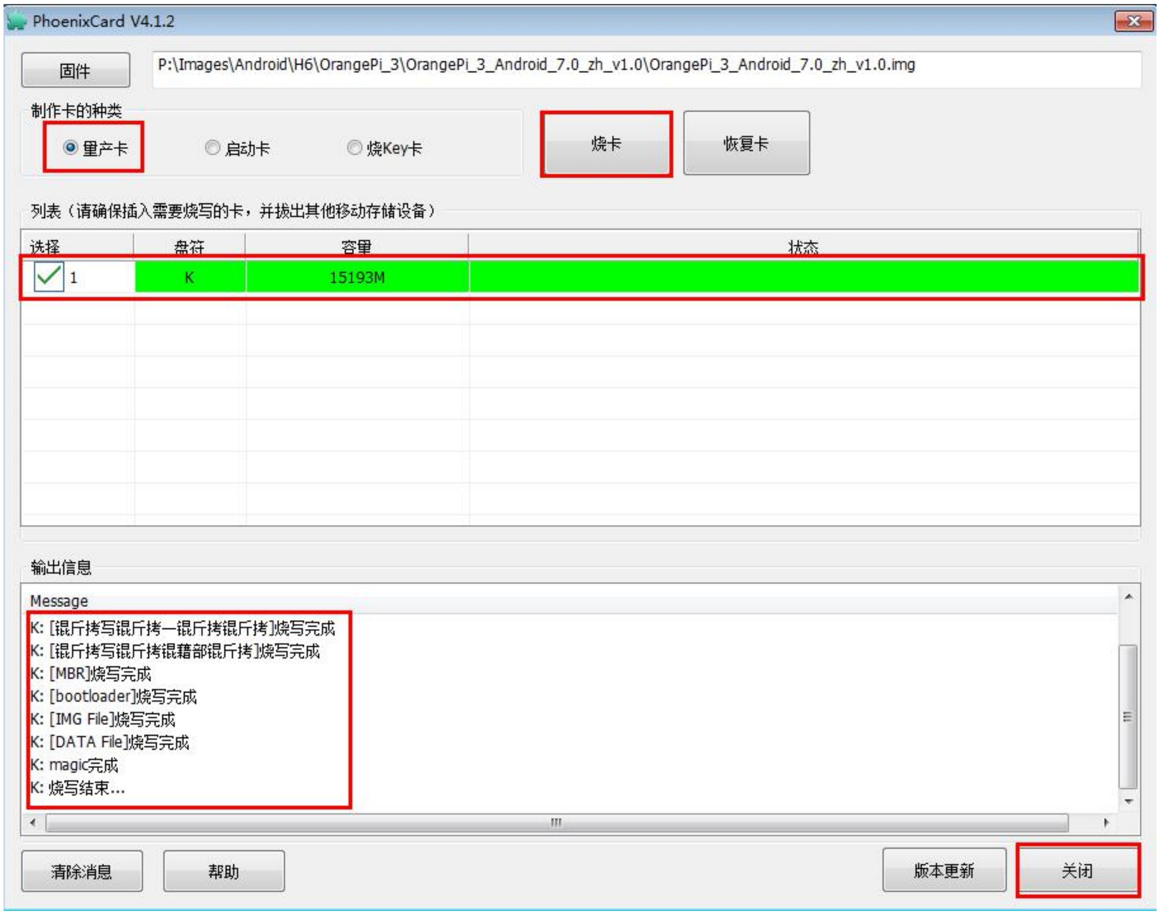
制作卡的种类

● 量产卡 ○ 启动卡 ○ 烧Key卡

列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	K	15193M	

- **Click burn-烧卡 enter into image flashing into SD card**
-



PhoenixCard V4.1.2

固件 P:\Images\Android\H6\OrangePi_3\OrangePi_3_Android_7.0_zh_v1.0\OrangePi_3_Android_7.0_zh_v1.0.img

制作卡的种类

● 量产卡 ○ 启动卡 ○ 烧Key卡

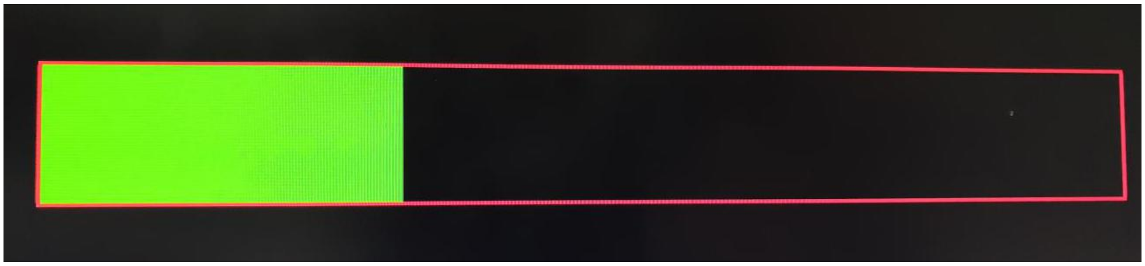
列表（请确保插入需要烧写的卡，并拔出其他移动存储设备）

选择	盘符	容量	状态
☑ 1	K	15193M	🟩

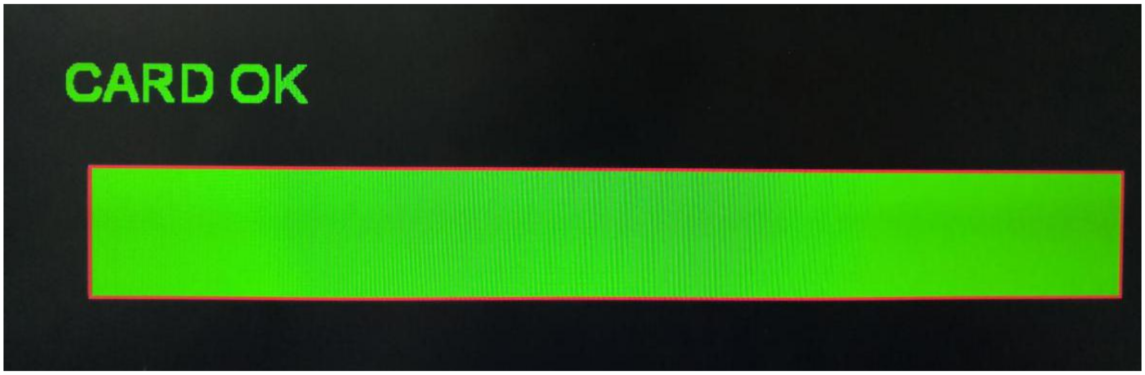
输出信息

```
Message
K: [锟斤拷写锟斤拷一锟斤拷锟斤拷]烧写完成
K: [锟斤拷写锟斤拷锟藉部锟斤拷]烧写完成
K: [MBR]烧写完成
K: [bootloader]烧写完成
K: [IMG File]烧写完成
K: [DATA File]烧写完成
K: magic完成
K: 烧写结束...
```

After finished Android flash, click close-关闭 and you could use this SD card with written image to boot. When you power the board, it will auto. transfer the image from sd card into emmc, the red led onboard will be twinkled until the end. You could check the following buring interface via the HDMI



Screen displayed during burning process



Screen displayed after burning

When it finished, unplug the power supply and sd card, the board with boot from emmc.

VI. Linux Firmware Flash

We could use **Etcher** to write Linux Firmware into TF card and boot the Orange Pi 3 with TF card. If you bought the orange pi 3 without EMMC Flash, then you could only boot from sd card. Etcher supports following computer system:

- Linux（Most distro versions, such as Ubuntu）
- MacOS 10.9 or higher version
- Windows 7 or higher version

Etcher package could be downloaded from their website(**https://etcher.io/**) or Orange Pi official website.

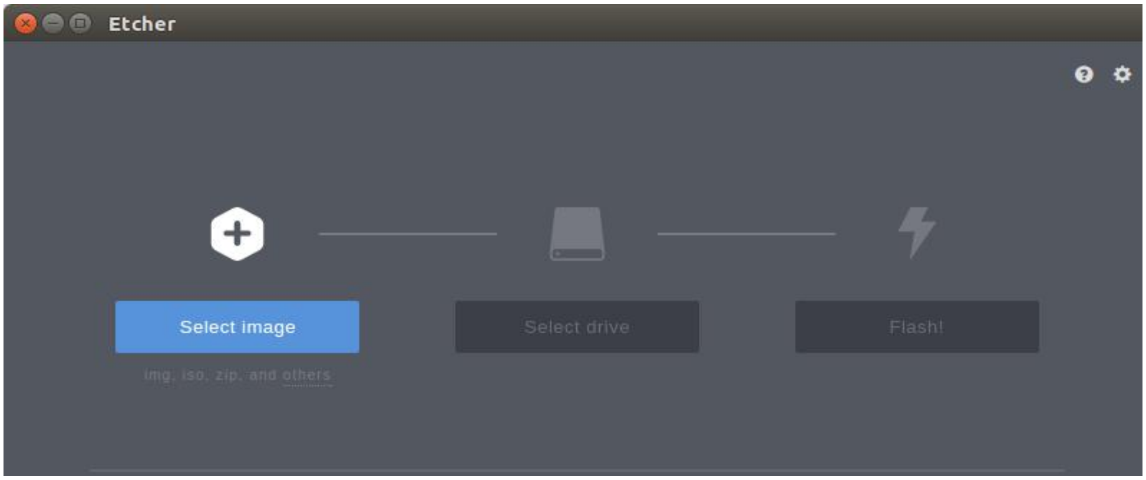
1. Install Etcher

- Install Etcher on Windows OS is same like installing other software(such as PhoneixCard)
- Install Etcher on Ubuntu and Debian OS:

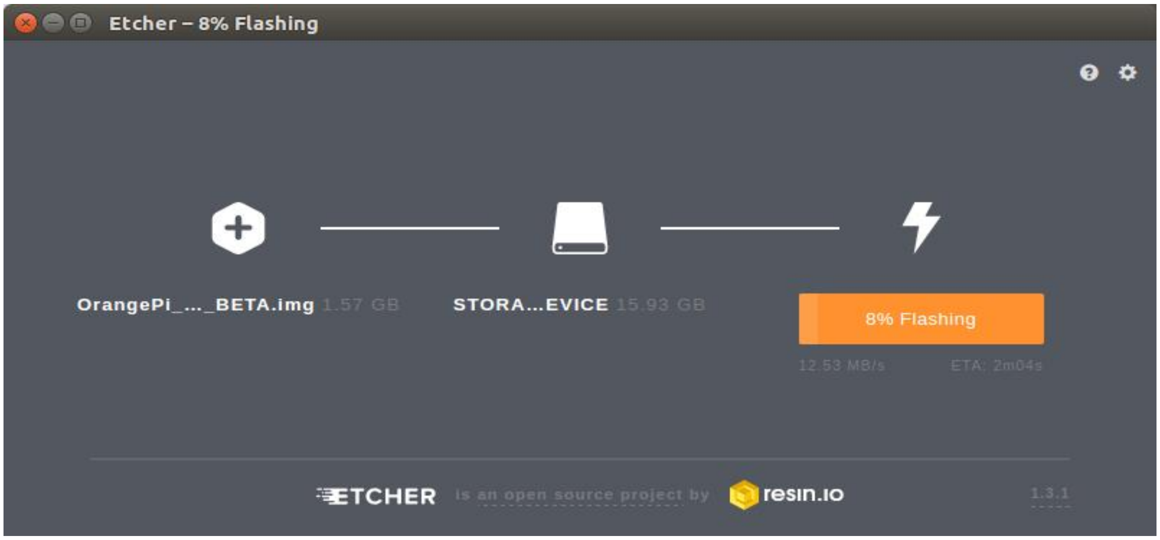
```
1.  Add Etcher Debian library：

$ echo "deb https://dl.bintray.com/resin-io/debian stable etcher" | sudo tee
/etc/apt/sources.list.d/etcher.list

2.  Download key

$ sudo apt-key adv --keyserver hkp://pgp.mit.edu:80 --recv-keys 379CE192D401AB61

3.  Update and install
$ sudo apt-get update && sudo apt-get install etcher-electron

4.  Upload
$ sudo apt-get remove etcher-electron
$ sudo rm /etc/apt/sources.list.d/etcher.list && sudo apt-get update
```

2. Flash Linux Firmware via Etcher

- Open Etcher like the following interface



- "Select image", select Linux firmware you are going to flash
- Insert TF card, Etcher will recognize corresponding driver auto
- Click "Flash！ " , after finished, boot the board with inserting the TF card



3. Flash linux Firmware into EMMC Flash with Scripts

If you bought the Orange Pi 3 with EMMC soldered on board, you could boot it from sd card as well as emmc flash with linux os, you could flash the linux image into emmc flash with **OrangePi_install2EMMC.sh** script.

Enter the command OrangePi_install2EMMC.sh in linux terminal, and input y according to the reference, then linux image will be flashed into emmc auto.. After flash end, turn off the power, unplug the sd card, re-power the board, you could boot the board fromm emmc.

```
root@OrangePi:~# OrangePi_install2EMMC.sh

WARNING: EMMC WILL BE ERASED !, Continue (y/N)?   y
Erasing EMMC ...
Creating new filesystem on EMMC ...
   New filesystem created on /dev/mmcblk0.
Partitioning EMMC ...
   Creating boot & linux partitions
   OK.
Formating fat partition ...
   fat partition formated.
Formating linux partition (ext4), please wait ...
   linux partition formated.

Instaling u-boot to EMMC ...

Mounting EMMC partitions...
FAT partitions mounted to /tmp/_fatdir
linux partition mounted to /tmp/_extdir

Copying file system to EMMC ...

   Creating "fstab"

*******************************
Linux system installed to EMMC.
*******************************
```

VII. Linux System Usage

1. Reflect when booting with Linux

- Once the board boot, the RED LED would light up first, then RED LED off, then YELLOW LED keep lighting

2. Login account and password

- User name: root, password: orangepi
- User name: orangepi, password:orangepi

3. Expand rootfs partition

You could expand roots partition once you made SD card with firmware. It could enhance the performance of the system.

We could use the built-in scripts of resize_rootfs.sh to expansion after enter into system

The size of the available space before expansion of the system

```
root@OrangePi:~# df -h
Filesystem          Size   Used Avail Use% Mounted on
/dev/mmcblk1p2    1.1G   981M    28M   98% /
devtmpfs            985M      0   985M    0% /dev
tmpfs               994M      0   994M    0% /dev/shm
tmpfs               994M   8.9M   985M    1% /run
tmpfs               5.0M   4.0K   5.0M    1% /run/lock
tmpfs               994M      0   994M    0% /sys/fs/cgroup
/dev/mmcblk1p1       50M    15M    36M   30% /boot
tmpfs               199M      0   199M    0% /run/user/0
root@OrangePi:~#
```

Run the system built-in expansion script

```
root@OrangePi:~# resize_rootfs.sh
```

The size of the available space after expansion

```
root@OrangePi:~# df -h
```

```
Filesystem          Size    Used Avail Use% Mounted on
/dev/mmcblk1p2       15G    982M    13G    7% /
devtmpfs            985M       0   985M    0% /dev
tmpfs               994M       0   994M    0% /dev/shm
tmpfs               994M    8.9M   985M    1% /run
tmpfs               5.0M    4.0K   5.0M    1% /run/lock
tmpfs               994M       0   994M    0% /sys/fs/cgroup
/dev/mmcblk1p1       50M     15M    36M   30% /boot
tmpfs               199M       0   199M    0% /run/user/0
```

4. Record and Play Sound

On platform of H6, it is default use AHUB with not standard driver of ALSA. At present test, we only test with tinyalsa for record and play sound function with orange pi 3 linux image. Source code of tinyalsa tool already uploaded on GitHub. You could refer to the following:

- **Download tinyalsa source code**

Download source code of tinyalsa test tool from Github before enter into Linux system on Orange Pi 3

```
$ sudo apt-get udpate
$ sudo apt-get install -y git make gcc
$ git clone https://github.com/orangepi-xunlong/OrangePiH6_Tinyalsa.git
Cloning into 'OrangePiH6_Tinyalsa'...
remote: Counting objects: 21, done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 21 (delta 3), reused 21 (delta 3), pack-reused 0
Unpacking objects: 100% (21/21), done.
Checking connectivity... done.
```

- **Compile tinyalsa tool**

```
$ cd OrangePiH6_Tinyalsa
$ ./build.sh

编译完的可执行文件放在 out/tinyalsa-arm64 中
$ cd out/tinyalsa-arm64
$ ls
libtinyalsa.so    tinycap_ahub    tinypcminfo    tinyplay_ahub
tinycap           tinymix         tinyplay
```

Export path of tinyalsa Shared library

```
$ cd out/tinyalsa-arm64
$ export LD_LIBRARY_PATH=`pwd`
```

Tinyalsa tool usage

Check device node

```
$ cat /proc/asound/cards
 0 [sndahub            ]: sndahub - sndahub
                          sndahub
 1 [sndhdmi            ]: sndhdmi - sndhdmi
                          sndhdmi
 2 [snddaudio2         ]: snddaudio2 - snddaudio2
                          snddaudio2
 3 [sndacx00codec      ]: sndacx00-codec - sndacx00-codec
                          sndacx00-codec
```

Test record function

```
1.  Check device node, codec is 5
2.  Use tinymix program to connect i2s3 TX（codec）and APBIF0 RX, open i2s3 in    with
following command:
      $ cd out/tinyalsa-arm64
      $ ./tinymix 13 7
      $ ./tinymix 20 1
3. Execute the following command and open codec in control
      $ ./tinymix -D 3 13 1
      $ ./tinymix -D 3 15 1
      $ ./tinymix -D 3 20 1
      $ ./tinymix -D 3 27 1
4. Record command
      $ ./tincap test.wav -D 0 -d 0 -t 10
```

Test HDMI play sound function

```
1.  Check device node, HDMI is 1
2.  Use tinymix program to connect i2s1 and APBIF_TXDIF0,open i2s1 out
       $ ./tinymix 9 1
       $ ./tinymix 17 1
3.  HDMI play command
       $ ./tinyplay test.wav -D 0 -d 0
```

5. WIFI configuration

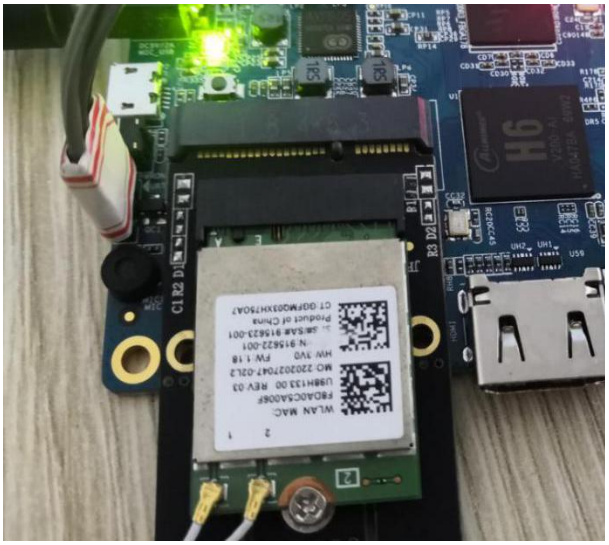
Add the following configuration under /etc/network/interface, then restart the orange pi 3:

```
auto wlan0
iface wlan0 inet dhcp
wpa-ssid orangepi          //enter the wifi account（it is orangepi right now）
wpa-psk orangepi           //enter the wifi WIFI password（it is orangepi right now）
```

6. Test Method of PCIE Interface

Matters need attention: USB2.0 cant be used simultaneously with PCLE, otherwise error will occur.

At present, the driver of rtl8822be has been integrated in the Linux kernel by default. After inserting RTL8822BE wireless network card module into the system according to the method shown below, the system will automatically identify and load the 88x2be.ko kernel module



The lsmod command can be used to check whether the driver is loaded successfully, and the ifconfig command can be used to check the corresponding network nodes of the PCIE wireless network card

```
root@OrangePi:~# lsmod
Module                         Size    Used by
```

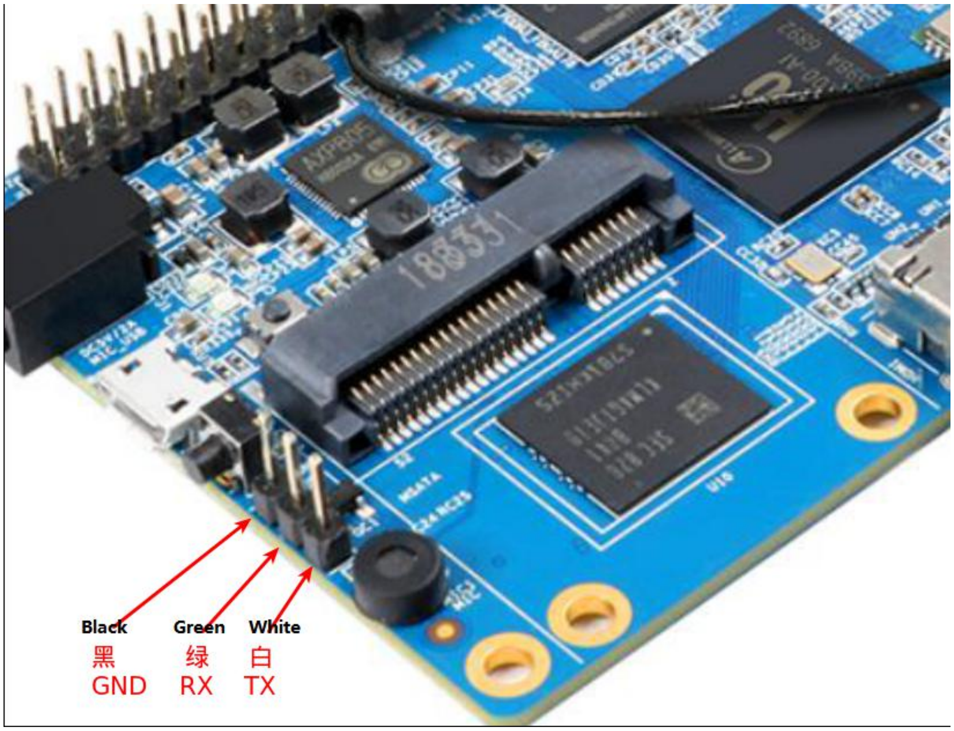
```
88x2be                  2116402   0
ss                        44426   0
rtk_btusb                37346   0
sunxi_ir_rx              10075   0
bcmdhd                   721838   0

root@OrangePi:~# ifconfig
eth0        Link encap:Ethernet   HWaddr 72:d6:05:4f:b9:3b
             inet addr:192.168.1.131   Bcast:192.168.1.255
             inet6 addr: fe80::70d6:5ff:fe4f:b93b/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST   MTU:1500   Metric:1
             RX packets:112 errors:0 dropped:0 overruns:0 frame:0
             TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:9214 (9.2 KB)   TX bytes:2056 (2.0 KB)
             Interrupt:44

wlan0       Link encap:Ethernet   HWaddr 6c:21:a2:14:dc:3a
             UP BROADCAST MULTICAST   MTU:1500   Metric:1
             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:0 (0.0 B)   TX bytes:0 (0.0 B)

wlp1s0      Link encap:Ethernet   HWaddr f8:da:0c:5a:00:6f
             UP BROADCAST MULTICAST   MTU:1500   Metric:1
             RX packets:0 errors:0 dropped:6 overruns:0 frame:0
             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:0 (0.0 B)   TX bytes:0 (0.0 B)
```

VIII. Using Debug tools

Prepare a USB to TTL cable like the following:

Connect the debug port with cable like the following, or you could check the function onthe board silk

- Black——GND
- Green——RX
- White——TX

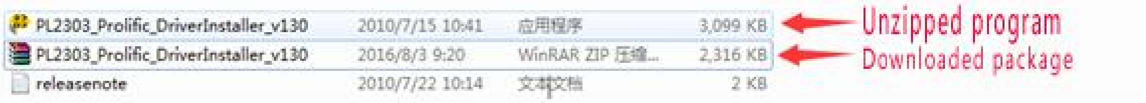


1. Operation Steps on Windows

In order to get more debugging information in the project development process of using OrangePi, OrangePi default support for serial information debugging. For developers, you can simply get the serial port debugging information with the materials mentioned above. The host computer using different serial debugging tools are similar, basically can reference with the following manual for deployment. There are a lot of debugging tools for Windows platform, the most commonly used tool is putty. This section takes putty as an example to explain the deployment.

- **Install USB Driver**

Download and unzip the latest version of driver PL2303_Prolific_DriverInstaller_v130.zip

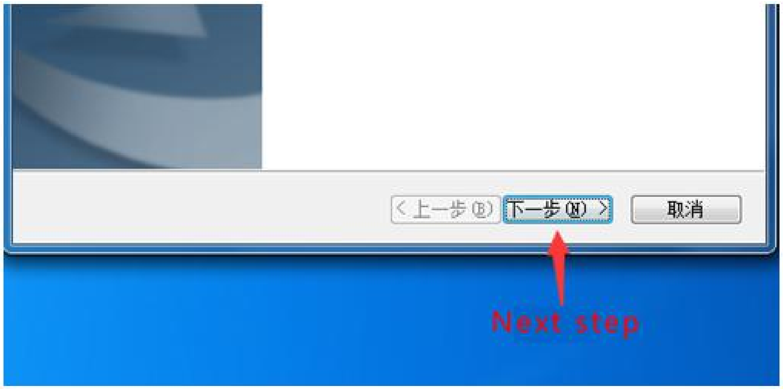


PL2303_Prolific_DriverInstaller_v130	2010/7/15 10:41	应用程序	3,099 KB
PL2303_Prolific_DriverInstaller_v130	2016/8/3 9:20	WinRAR ZIP 压缩...	2,316 KB
releasenote	2010/7/22 10:14	文本文档	2 KB

Unzipped program

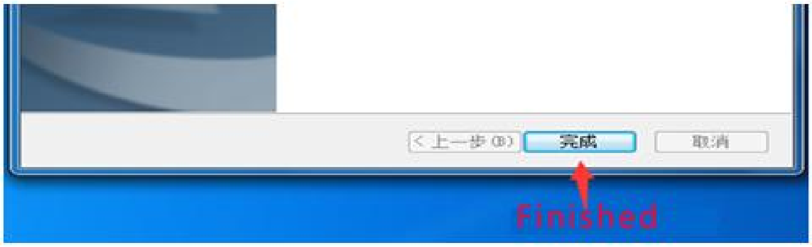
Downloaded package

Choose application installation as Administrator



Next step

Wait for completing installation

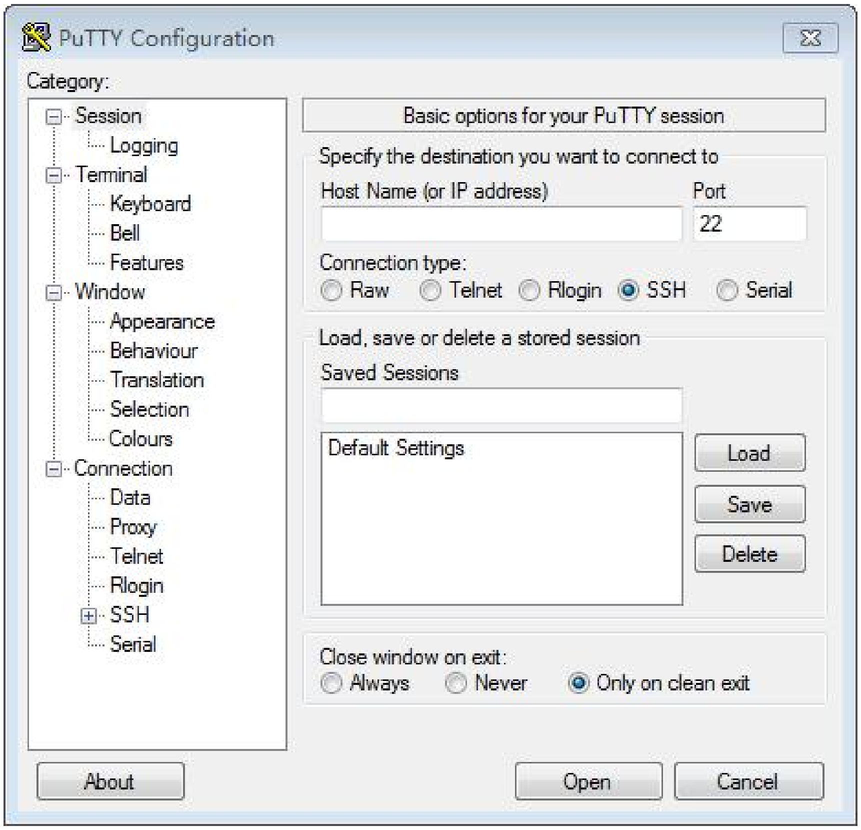


- **Install Putty**

You could download Putty from the following address and select a suitable version to your are development environment

```
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
```

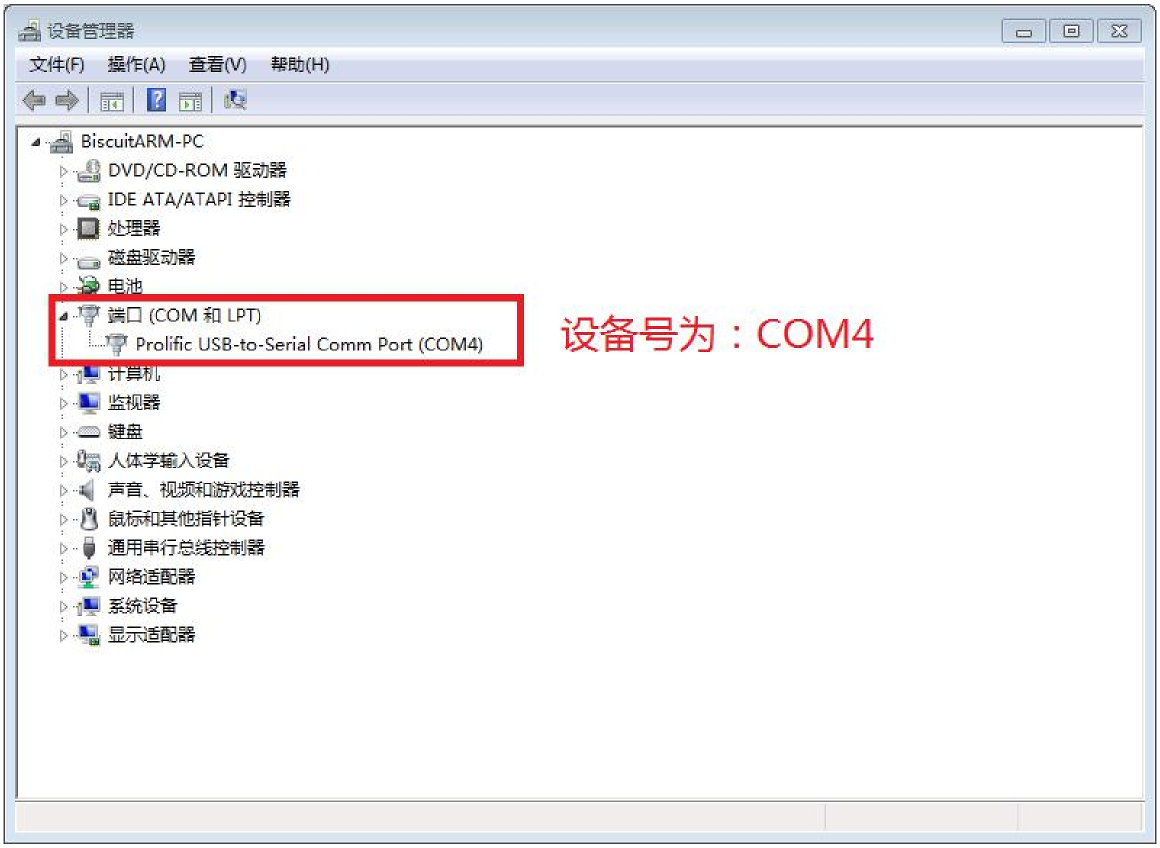
- Double click putty.exe to run putty, interface shown as below



Host Name (or IP address):
Port: 22
Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial
Close window on exit: ○ Always ○ Never ● Only on clean exit

- **设备信息的获取 Equipment information acquisition**

在 Windows7 中，我们可以通过设备管理器查看串口连接是否正常以及串口的设备号。如果设备没有正常识别，请检查驱动是否安装成功。如果驱动安装有问题，可以尝试使用 360 驱动大师扫描安装驱动。

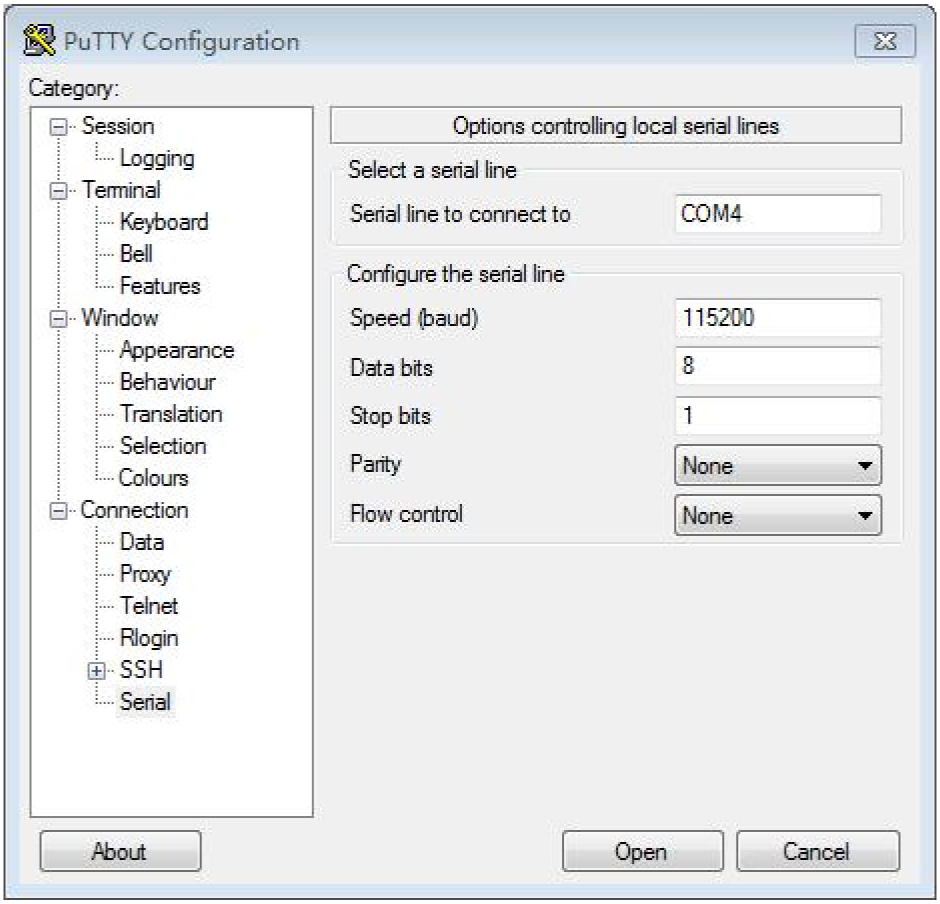


BiscuitARM-PC
- DVD/CD-ROM 驱动器
- IDE ATA/ATAPI 控制器
- 处理器
- 磁盘驱动器
- 电池
- 端口 (COM 和 LPT)
  - Prolific USB-to-Serial Comm Port (COM4)
- 计算机
- 监视器
- 键盘
- 人体学输入设备
- 声音、视频和游戏控制器
- 鼠标和其他指针设备
- 通用串行总线控制器
- 网络适配器
- 系统设备
- 显示适配器

设备号为：COM4

- **Putty Configure**

Serial port should set to the corresponding port number (COM4), close flow control and set the speed as 115200



Options controlling local serial lines

Select a serial line

Serial line to connect to: COM4

Configure the serial line

Speed (baud): 115200

Data bits: 8

Stop bits: 1

Parity: None

Flow control: None

- **Start debug serial port with output**

Power OrangePi on, putty will print out serial log information auto.

2. Operation Steps on Linux

When running putty, , there is not too much difference on Linux platform or Windows platform. Here the instruction is based on Ubuntu 14.04 OS

- **Install and start Putty**

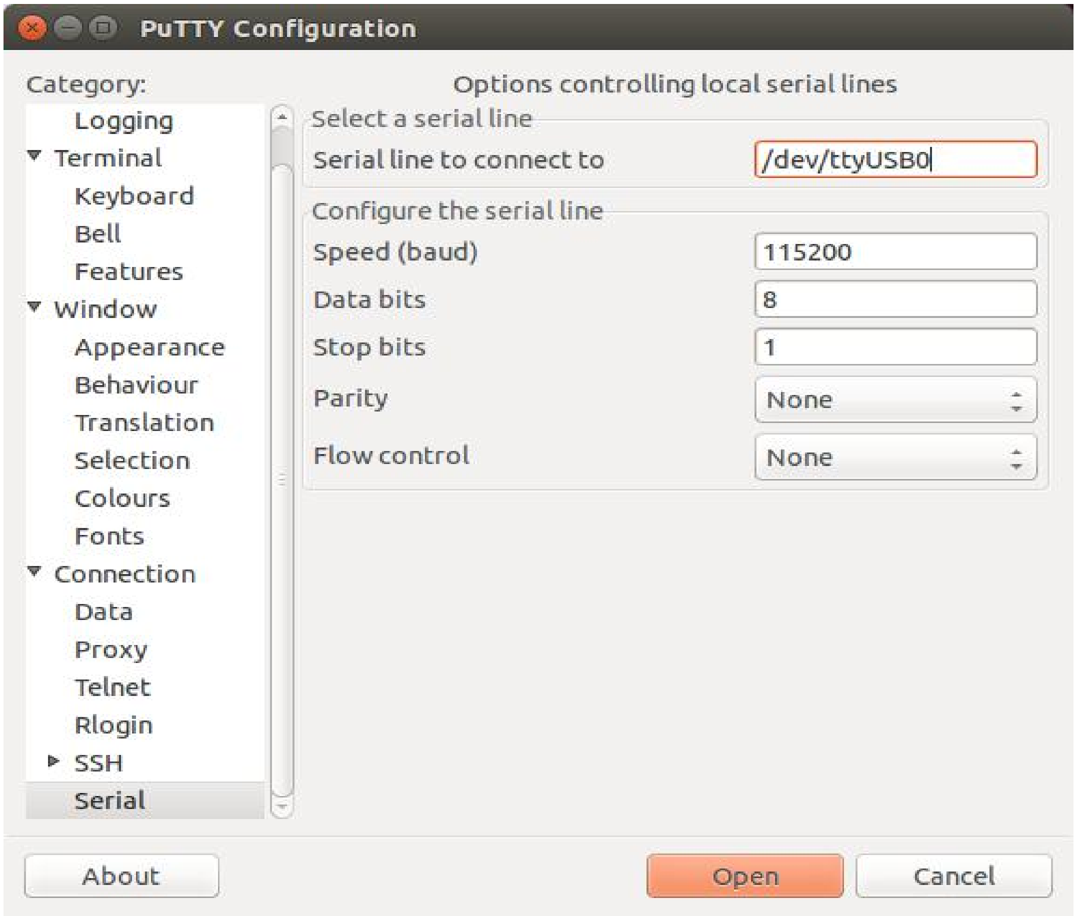
```
$ sudo apt-get install putty
$ sudo putty
```

- **Configure Putty**

Check serial number via ls /dev/ttyUSB*

Set baud rate into 115200

Close flow control



Serial line to connect to /dev/ttyUSB0

Speed (baud) 115200

Data bits 8

Stop bits 1

Parity None

Flow control None