

Orange Pi Lite 2

User Manual

History

Ver	Data	Author	Brief	Publish	Memo
1.0	2018-05-07	Leeboby	Create Files		
1.1	2019-7-10	Leeboby	Update audio instructions		
1.2	2019-12-16	Leeboby			

Content

I. Basic features of Orange Pi Lite 2

1. What is Orange Pi Lite 2?

Orange Pi is an open source single board computer, a new generation of arm64 development board, which can run Android 7.0, Ubuntu and Debian operating systems. Orange Pi single board computer uses Allwinner H6 system-on-chip and has 1GB LPDDR3 memory.

2. Use of Orange Pi Lite 2

We can use it to build:

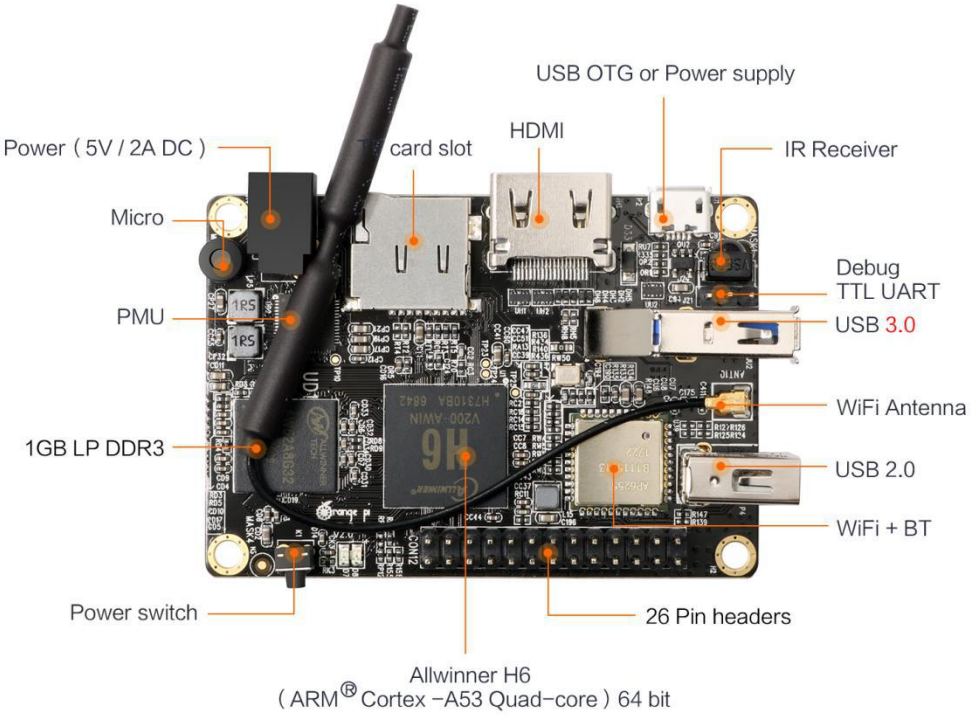
- a computer
- A web server
- Game console
- HD video player
- Speakers
- Speakers
- ...

There are many more features because Orange Pi Lite 2 is open source.

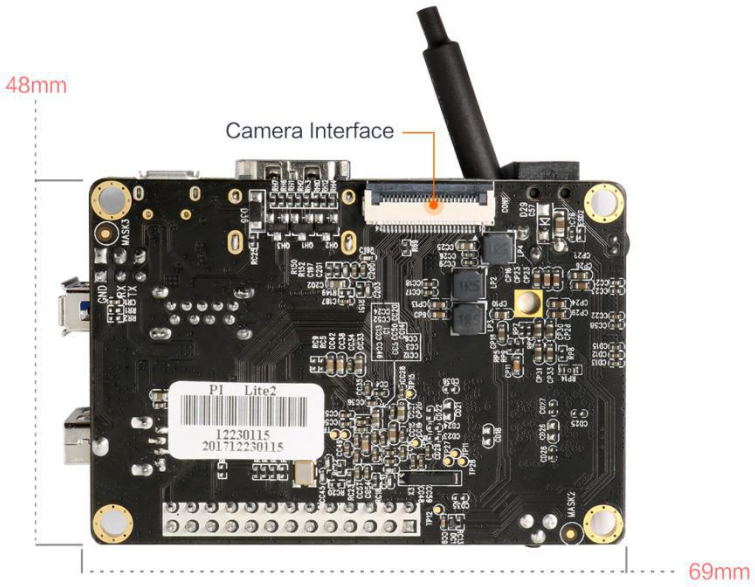
3. Who is the Orange Pi Lite 2 for?

Orange Pi Lite 2 is not just a consumer product, it is also designed for anyone who wants to use technology to create and innovate. It is a very simple, interesting and practical tool that you can use to build the world around you.

Top view



Bottom view



4. Hardware Features of Orange Pi Lite 2

Hardware specification	
CPU	H6 Quad-core 64-bit 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 (SDRAM)	1GB LPDDR3 (shared with GPU)
On-board Storage	TF card (Max. 32GB) /MMC card slot
WIFI+BT	AP6255, IEEE 802.11 AC/bb/g/n, BT4.1
Audio Input	MIC
Video Input	A CSI input connector Camera: Support 8 bit digital camera interface Maximum still capture resolution for parallel interface to 5M Maximum video capture resolution for parallel interface to 1080p@30fps
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
Video Outputs	HDMI 2.0a TX with HDCP 2.2 output
Audio Output	HDMI 2.0a
Power Source	DC input, USB OTG input can supply power
PMU	AXP805
USB 2.0 Ports	One USB 2.0 Host, one Micro USB 2.0
USB 3.0 Ports	One USB 3.0 Host
Low-level peripherals	26 Pins Header

5. GPIO specifications

The following figure is the GPIO pin function diagram of Orange Pie Lite 2:

Orange Pi Lite 2 pin correspondence table		
CON12-P01	VCC-3.3V	VCC-IO
CON12-P02	VCC-5V	DCIN
CON12-P03	TWI1-SDA	PH06
CON12-P04	VCC-5V	DCIN
CON12-P05	TWI1-SCK	PH05
CON12-P06	GND	GND
CON12-P07	PWM1	PH04
CON12-P08	PD21	PD21
CON12-P09	GND	GND
CON12-P10	PD22	PD22
CON12-P11	UART3_RX	PD24
CON12-P12	PC09	PC09
CON12-P13	UART3_TX	PD23
CON12-P14	GND	GND
CON12-P15	UART3_CTS	PD26
CON12-P16	PC08	PC08
CON12-P17	VCC-3V3	VCC-IO
CON12-P18	PC07	PC07
CON12-P19	SPI0_MOSI	PC02
CON12-P20	GND	GND
CON12-P21	SPI0_MISO	PC03
CON12-P22	UART3_RTS	PD25
CON12-P23	SPI0_CLK	PC00
CON12-P24	SPI0_CS0	PC05
CON12-P25	GND	GND
CON12-P26	PH03	PH03

II. Development board instructions

1. Prepare hardware and software tools

(1) Hardware requirements:

- Orange Pi Lite 2 development board
- TF card, minimum 8GB capacity, class 10, recommended brand TF card, such as SanDisk 16G TF card
- For a compiling host, the configuration should preferably meet the following conditions:
  64bit CPU
  8 GB memory and above
  100GB of free disk space
  The operating system is best: **Ubuntu14.04 (for compiling Android)**
  Ubuntu18.04 (for compiling Linux source code)

(2) Software Requirements:

- Orange Pi Lite 2 SDK
- Orange Pi Lite 2 firmware
- Android and Linux flashing tools

The above software can be obtained through Github, Mega network disk and Baidu cloud disk.

http://www.orangepi.org/downloadresources/
http://www.orangepi.cn/downloadresourcescn/

2. Development board power supply

There are two ways to power the development board:

- DC (5V 2A) power supply:

Power on after inserting the DC adapter

- Micro USB (5V 2A) OTG power supply:

Plug in the Micro USB adapter to power on.

III. Android compilation environment

The following operations are performed on a PC with Ubuntu 14.04 installed. Other versions of Ubuntu systems or Linux distributions may have some differences.

1. Get SDK source zip

After downloading the Android source package, first you need to combine multiple compressed files into one, and then decompress.

```
$ mkdir OrangePi_Lite2
$ cat H6-2018-1-2.tar.gza* > OrangePi_Lite2.tar
$ tar xf OrangePi_Lite2.tar -C OrangePi_Lite2
```

2. Build a compilation environment

(1) Install JDK

Android 7.0 development can only use the version of openjdk8, higher or lower than this version and Oracle's JDK will cause the compilation to fail. Openjdk-8 installation command is as follows:

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

(2) Configure JAVA environment variables

For example, the installation path is / usr / lib / jvm / java-8-openjdk-amd64. You can run the following command in the terminal to configure the environment variables:

```
$ 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
```

(3) Install platform support software

For Ubuntu 14.04:

```
$ 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. Compile SDK source code

After the SDK is decompressed, there will be two subdirectories andorid and lichee under the decompressed directory. The main contents of the lichee directory are as follows:

```
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    #Solution hardware configuration, packaging tools, etc.
```

(1) Kernel compilation process

Enter the following command in the lichee directory:

```
$ cd OrangePi_Lite2/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
```

The output after compilation is as follows:

```
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
----------------------------------------
```

The kernel code is in the lichee / linux-3.10 directory. Executing the above commands will automatically copy the configuration file from lichee / linux-3.10 / arch / arm64 / configs / sun50iw6p1smp_android_7.x_defconfig to lichee / linux-3.10 / .config as the default configuration before compiling The next time you compile, you can run ./build.sh directly in the lichee directory, and will continue to use the last .config configuration.

(2) uboot / boot0 compilation process (optional)

Under normal circumstances, you do not need to recompile uboot, but if you have customized modifications to uboot, you can compile. The compilation method is as follows:

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

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

If uboot / boot0 is not compiled, the default is to use the pre-compiled results of lichee / tools / pack / chips / sun50iw6p1 / bin. After recompiling with the above command, the above files will be replaced automatically.

(3) Android code compilation process

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

The pack command is to generate the firmware. If the compilation and packaging process passes successfully, the following message will be prompted:

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

OrangePi_Lite2/lichee/tools/pack/sun50iw6p1_android_petrel-p1_uart0.img

pack finish
```

According to the prompt, you can see the generated Android firmware sun50iw6p1_android_petrel-p1_uart0.img in the OrangePi_Lite2 / lichee / tools / pack / directory. Below, please refer to the "Android Firmware Burning" section to complete the Android firmware burning.

IV. Linux compilation environment setup

1. Get the Linux SDK source code from Github

(1) Orange Pi Linux Source Downloader

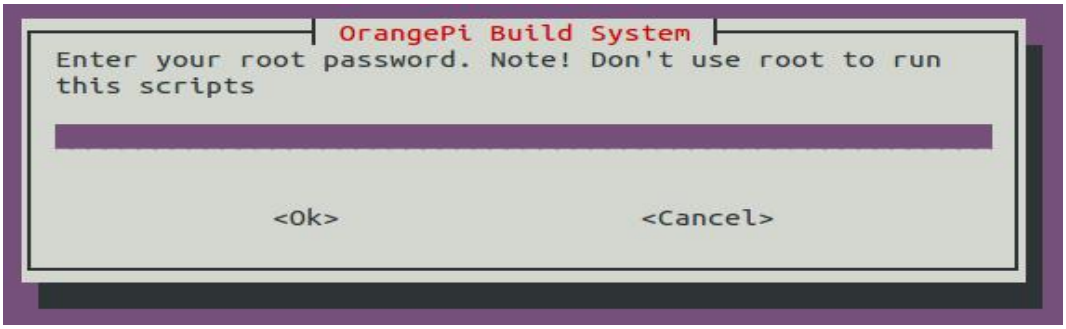
Orange Pi lite2's Linux source code has been uploaded to GitHub. The currently supported kernel version is Linux 4.9 and the mainline kernel (some drivers are still under development). We can use the Orange Pi Linux source downloader to download and obtain the downloader source code. Here's how:

```
$ 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
```

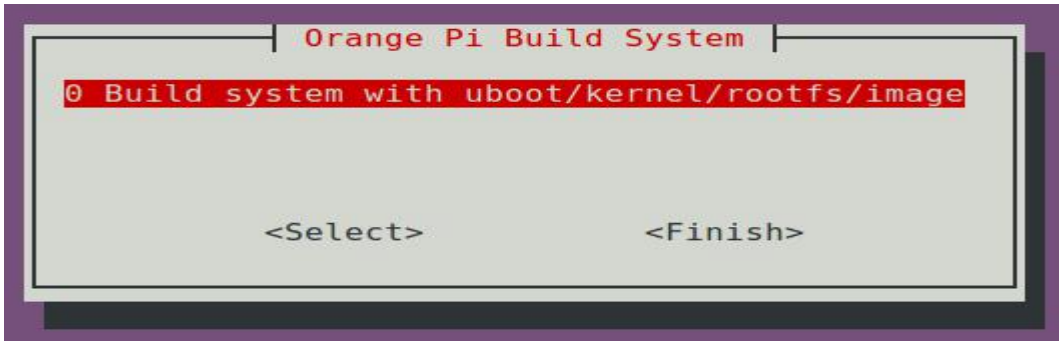
(2) Run the downloader

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

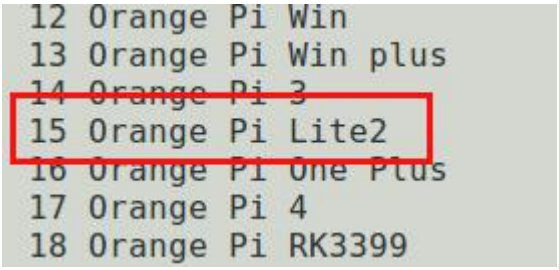
Enter the root password and press enter



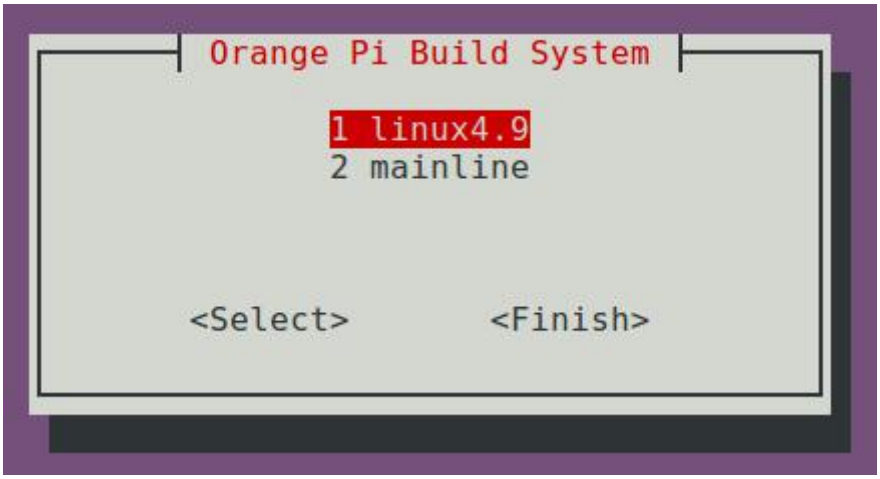
Select 0 Build system with uboot / kernel / rootfs / image to enter the interface of the development board model selection



Select orange pi lite2, enter the kernel version selection interface after entering



At present, the orangepi lite2 development board supports the kernel code of Linux 4.9 and mainline. Select one of them and press Enter to download the corresponding SDK source.



The downloaded source code will be stored in the same directory of OrangePi_Build

```
$ ls ../OrangePi_Build -l
  OrangePi_Build
  OrangePiH6_Linux4.9   (The kernel version is Linux4.9)
  OrangePiH6_mainline   (The current kernel version is Linux5.3.5)
```

2. Get the source code of Linux SDK from Baidu Cloud Disk

If GitHub fails to download the code, you can download the source code compression package of Linux SDK directly from Baidu Cloud Disk.
The download link is:

https://pan.baidu.com/s/15NOF_eAwbN9ah3dfWx0ArQ

3. Build a compilation environment

The Orange Pi H6 Linux SDK has only been tested on PCs with Ubuntu 18.04. Please prepare the host environment of ubuntu 18.04 before use.
The Linux source directory structure for Orange Pi H6 is shown below:

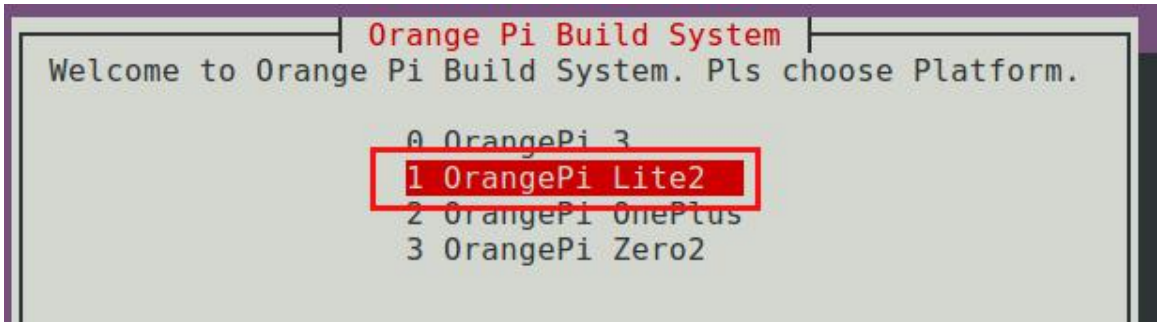
```
$ cd OrangePiH6_Linux4.9
$ tree -L 1
.
├── build.sh -> scripts/build.s    Compile startup script
├── external                    Store additional configuration files
├── kernel                        Linux kernel source
├── output                         Store the output file, which will
be generated after compiling the source code
├── scripts                   Script files used during compilation
├── toolchain    Cross-compilation toolchain used by the kernel and u-boot
└── uboot                       Store boot0 and u-boot source code
6 directories, 1 file
```

4. Compile Linux and U-boot source code

(1) Execute compile startup script

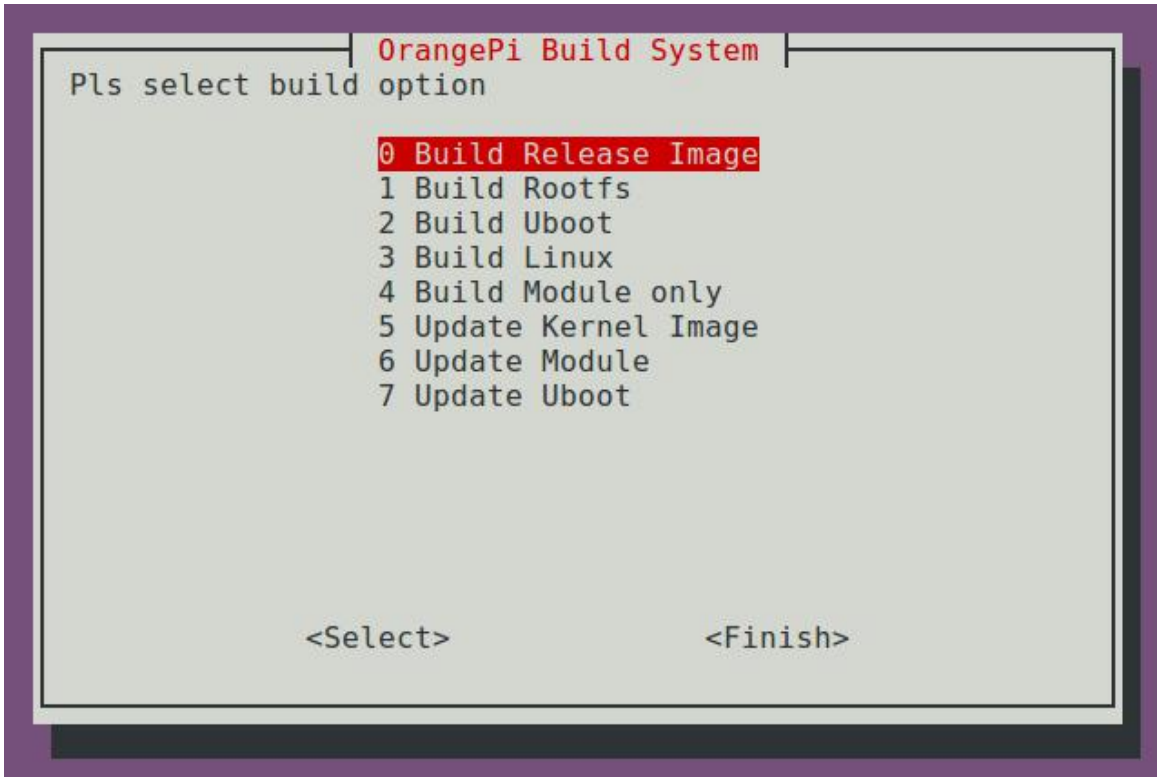
```
$ cd OrangePiH6_Linux4.9
$ sudo ./build.sh
```

Select orangepi lite2 and press enter



The functions of each option are as follows:

- **0 Build Release Image —— Compile Ubuntu or Debian Distribution Images**
- **1 Build Rootfs —— Compile rootfs for ubuntu or debian**
- **2 Build Uboot —— Compile boot0 and u-boot source code**
- **3 Build Linux —— Compile Linux kernel source code**
- **4 Build Module only —— Compile Linux kernel modules**
- **5 Update kernel Image —— Update Kernel in SD Card Linux System**
- **6 Update Module —— Update the kernel module in SD card Linux system**
- **7 Update Uboot —— Update boot0 and u-boot and dtb configuration of SD card Linux system**



The final file generated by the compilation will be saved in the output directory

```
$ cd output
$ tree -L 2
.
├── images                  //Generated image file
│   └──
OrangePi_lite2_ubuntu_xenial_server_linux4.9.118_v2.0.2.img
├── kernel                  //Compile the generated kernel
└── uImage_lite2
├── rootfs                  //Compile the generated rootfs
├── uboot                   //Compile the generated uboot image
│   ├── boot0_sdcard_sun50iw6p1.bin
│   ├── H6.dtb
│   └── u-boot-sun50iw6p1.bin
└── xenial_arm64_server_rootfs.tar.gz
```

5. Linux SDK usage example

The following will fully demonstrate the use of the Linux SDK by adding a rtl8812AU USB WIFI kernel module to the kernel source code.

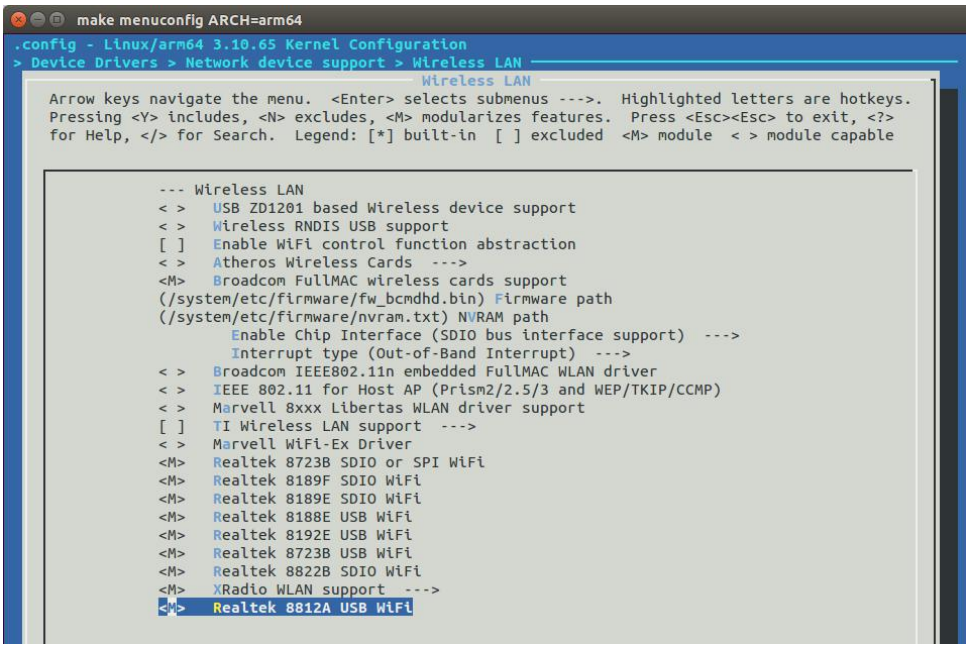
(1) Get the source code of rtl8812AU from github

```
$ cd OrangePiH6_Linux4.9/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.
```

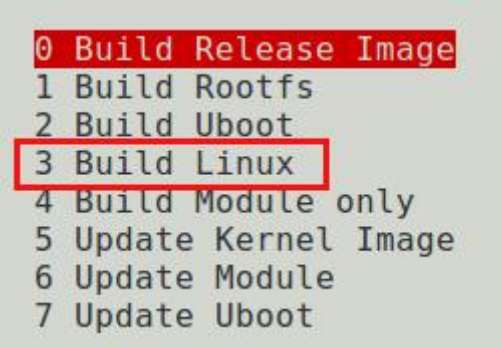
(2) Add rtl8812AU configuration

```
$ cd OrangePiH6_Linux4.9/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/
```

(3) Select Realtek 8812A USB WiFi in the kernel configuration and compile it into a kernel module



(4) Recompile the kernel according to the method in `` Compiling Linux and U-boot Source ''



Partially compiled log looks like this:

```
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
```

After compiling, you can find the compiled kernel module in output / lib / modules / 4.9.118 + / kernel / drivers / net / wireless / rtl8812AU

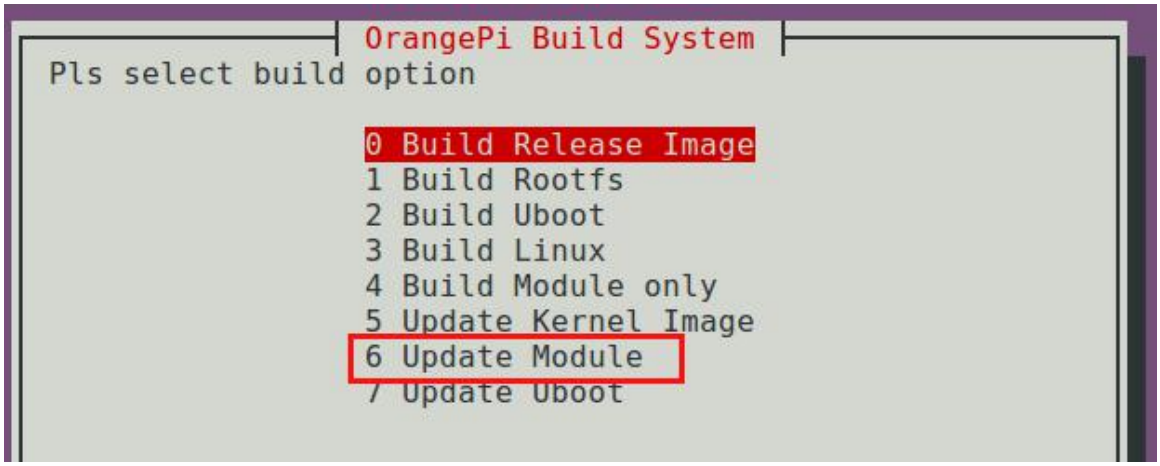
```
$ cd output/lib/modules/4.9.118+/kernel/drivers/net/wireless/rtl8812AU
$ ls
8812au.ko
```

(5) Update the kernel module

First, insert the SD card that has been burned with Linux firmware into the PC used to compile Linux source code (physical or virtual machine with Ubuntu 18.04 installed). After the system recognizes and successfully mounts the inserted SD card, we can / media / $ LOGNAME see the name of the corresponding partition.

```
$ cd /media/$LOGNAME
$ ls
BOOT      Store the kernel
rootfs    Root file system
```

Then choose 6 Update Module to update the kernel module according to the instructions in the section `` Compiling Linux and U-boot Source ''



Next, you can start the system through the SD card and use the new 8812au.ko kernel module to drive the USB WIFI network card.

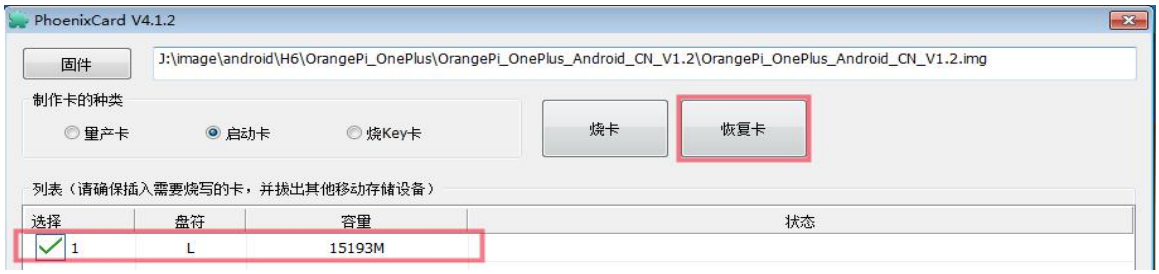
V. Android firmware burning

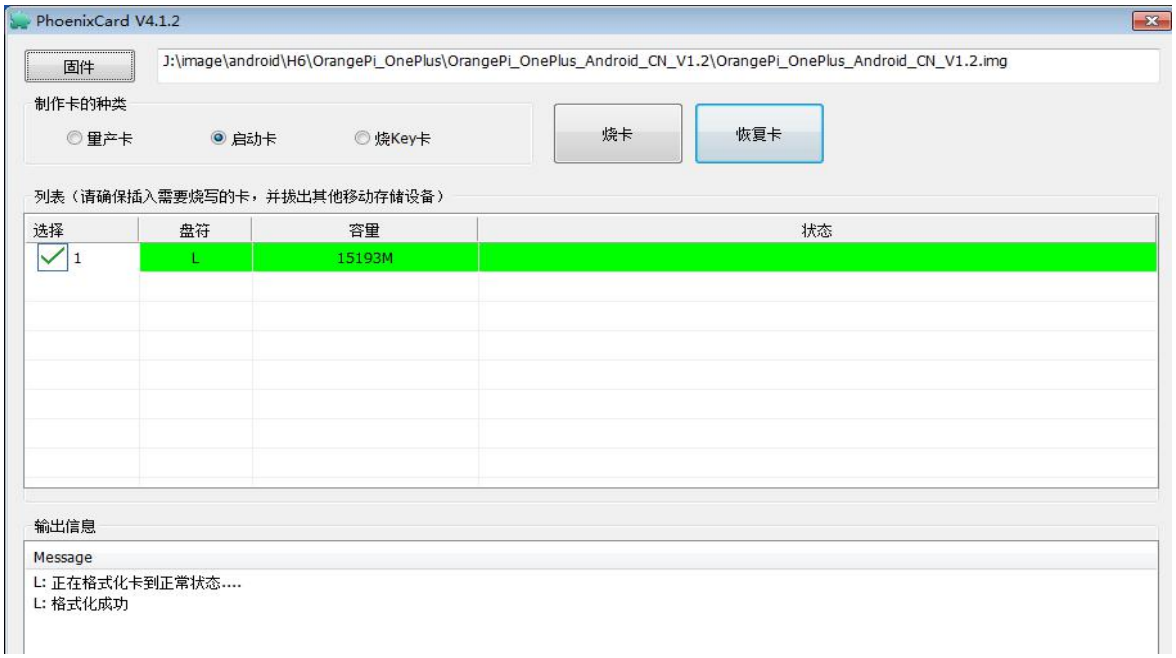
Android firmware cannot be written to the TF card via the dd command in Linux or the Win32 Diskimager tool in Windows. Need to use the tool PhoenixCard to write, PhoenixCard current latest version is PhoenixCard V4.1.2, can be downloaded from the official tool on the official website download page.

1. Android firmware burning steps

(1) Format TF card

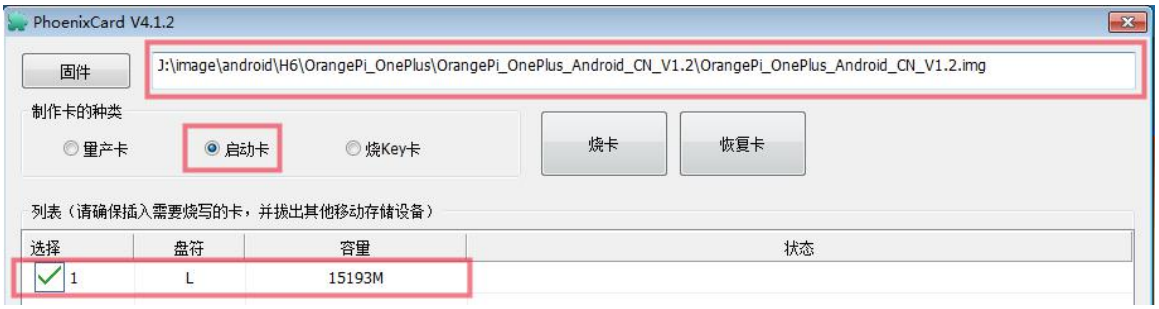
Check whether the inserted TF card is consistent with the selected drive letter, click the "Restore Card" button to start formatting the TF



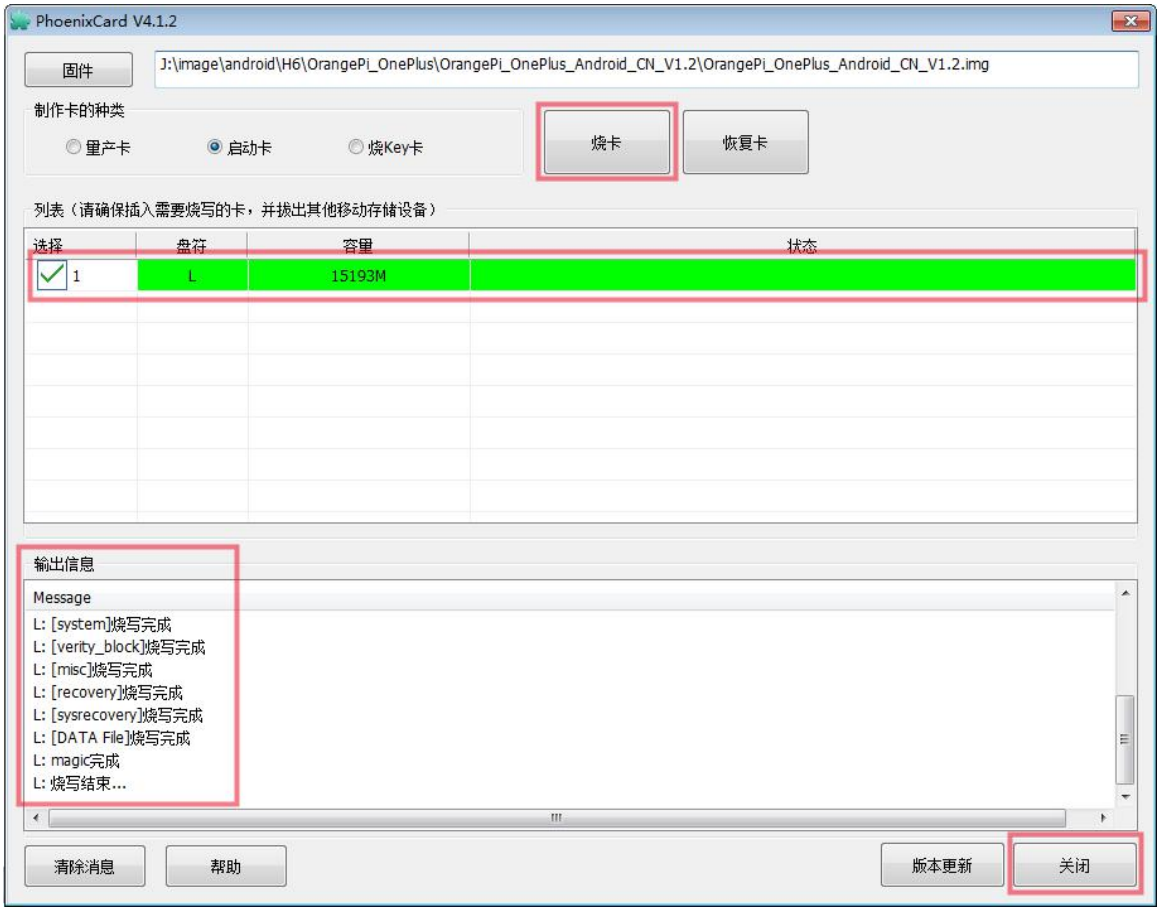


(2) Then select firmware, select boot card.

Please note the red mark in the picture below



(3) Click "Burn Card" to start writing to the TF card and wait for the burning to complete.



After the Android firmware is successfully programmed, click the "Close" button, and then you can insert the TF card into the development board to start the system

VI. Linux firmware burning

We can burn the Linux firmware of Orange Pi Lite 2 into the TF card through Etcher. Since Orange Pi Lite 2 has no on-board eMMC, we can only start the system through TF. Etcher supports the following operating systems:

- Linux (most distros, such as Ubuntu)
- MacOS 10.9 and later
- Windows 7 and later

The Etcher software installation package can be downloaded from its official website https://etcher.io/, or it can be downloaded from the official tools on the official download page of Orange Pi Lite 2.

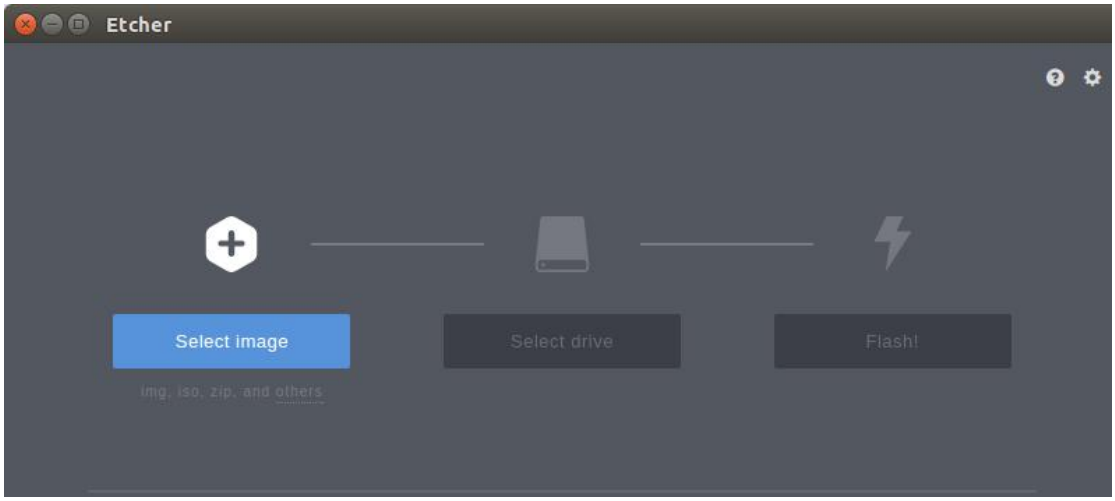
1. How to install Etcher

- The installation method of Etcher in Windows is the same as that of ordinary software.
- Etcher is installed on Ubuntu and Debian systems as follows

```
1. Add Etcher Debian repository:

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

2. How to flash Linux firmware through Etcher

(1) First open Etcher, its interface is shown below



(2) Then use "Select image" to select the Linux firmware to be burned

(3) Then insert the TF card, Etcher will automatically identify the corresponding drive

(4) Finally, click “Flash!” To start burning. After burning, you can insert the development board to start the system.

VII. Instructions for Linux systems

1. Linux boot light description

After booting, the onboard LED light will first turn on the red light, then the red light will be off and the yellow light will be on.

2. Login account and password

Username root, password: orangepi
Username orangepi, password: orangepi

3. Extend rootfs partition

After the system operating card is prepared, the rootfs partition of the file system should be expanded immediately. This will greatly improve the performance of the system and avoid various complicated problems caused by insufficient space.

We can use the system's built-in script resize_rootfs.sh to expand the capacity after entering the system:

```
The amount of free space in the system before expansion
root@OrangePi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk0p2  1.1G  520M  488M  52% /
devtmpfs        481M     0  481M   0% /dev
tmpfs           489M     0  489M   0% /dev/shm
tmpfs           489M  6.6M  483M   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           489M     0  489M   0% /sys/fs/cgroup
/dev/mmcblk0p1   50M   29M   22M  58% /boot

Run the built-in expansion script
root@OrangePi:~# resize_rootfs.sh
The amount of space available in the system after the expansion
root@OrangePi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk0p2  7.2G  539M  6.4G   8% /
devtmpfs        481M     0  481M   0% /dev
tmpfs           489M     0  489M   0% /dev/shm
tmpfs           489M   13M  477M   3% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
```

```
tmpfs           489M     0  489M   0% /sys/fs/cgroup
/dev/mmcblk0p1   50M   29M   22M  58% /boot
```

4. Recording playback test method

Note: The Audio Codec audio driver for the mainline kernel is not yet available

(1) Test recording function

```
root@orangepilite2:~# arecord -d 10 -f cd -D hw:1,0 -t wav Test.wav
Recording WAVE 'Test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz,
  Stereo
root@orangepilite2:~#
```

(2) Test the HDMI playback function, just use the aplay command to play

```
root@orangepilite2:~# aplay Test.wav
```

5. WIFI configuration method

Add the following configuration to / etc / network / interface and restart

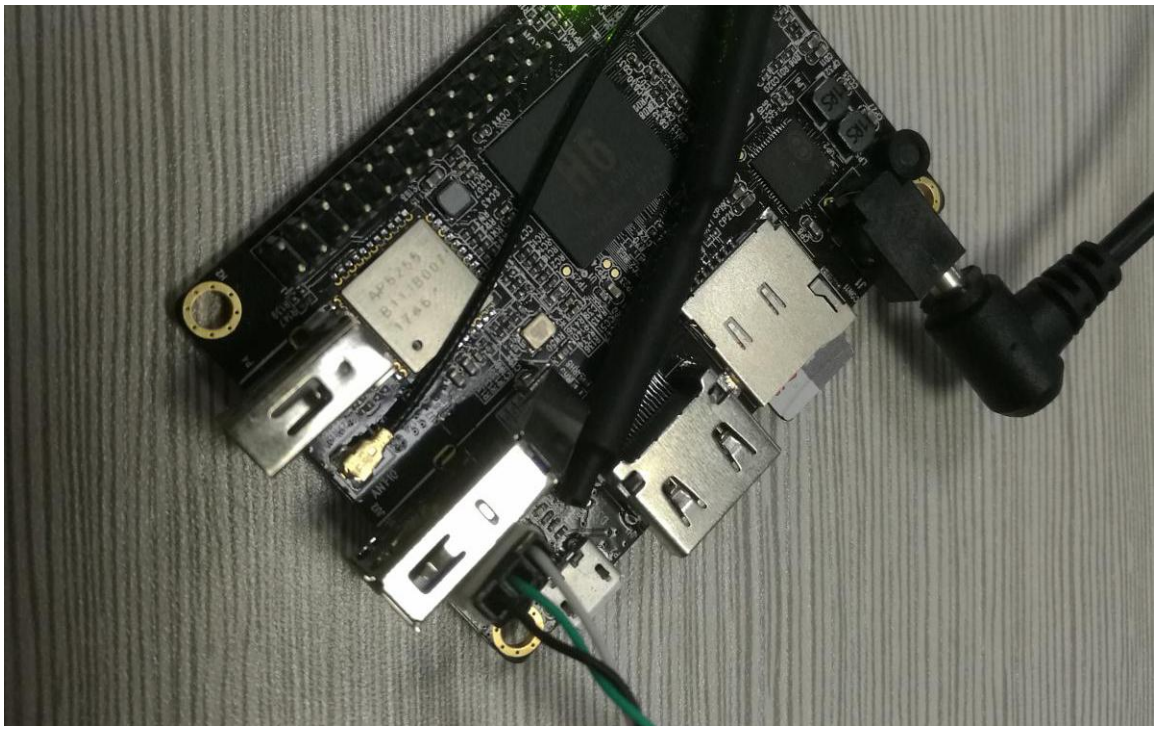
```
auto wlan0
iface wlan0 inet dhcp
wpa-ssid orangepi     //Fill in the WIFI account here (orangepi)
wpa-psk orangepi     //Fill in the WIFI password here (orangepi)
```

VIII. Introduction to serial debugging tools

First you need to prepare a USB to TTL serial cable similar to the picture below

Connect the serial cable as shown in the figure below. From the silk screen on the back of the board, you can see that the corresponding functions of the different colors are as follows

- Black-GND
- Green-RX
- White-TX



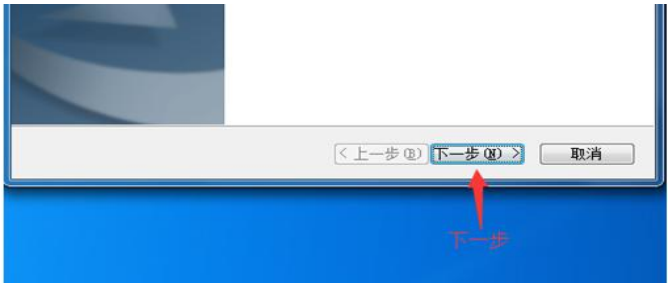
1. Use on Windows platform

In the process of using OrangePi for project development, in order to obtain more debugging information, OrangePi supports serial port information debugging by default. For developers, they only need to prepare the materials mentioned above to get serial debugging information. The serial debugging tools used by different host computers are similar. Basically, you can refer to the methods below to deploy. There are many tools for serial debugging on the Windows platform. The commonly used tool is putty. This section uses putty as an example to explain deployment.

(1) Install USB driver

Download the latest version of the driver PL2303_Prolific_DriverInstaller_v130.zip, downloading codecs.

As an administrator select the application installation



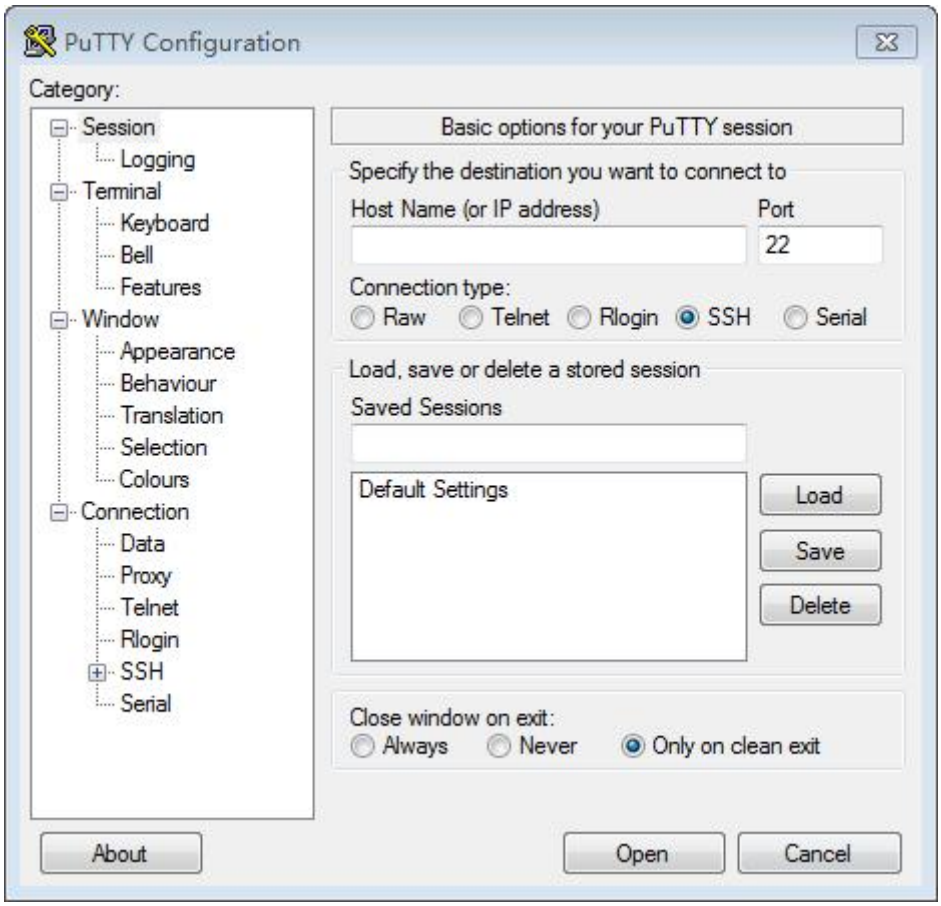
Wait for the installation to complete

(2) Download and install Putty

Putty can be downloaded from the following address, please choose the version suitable for your development environment.

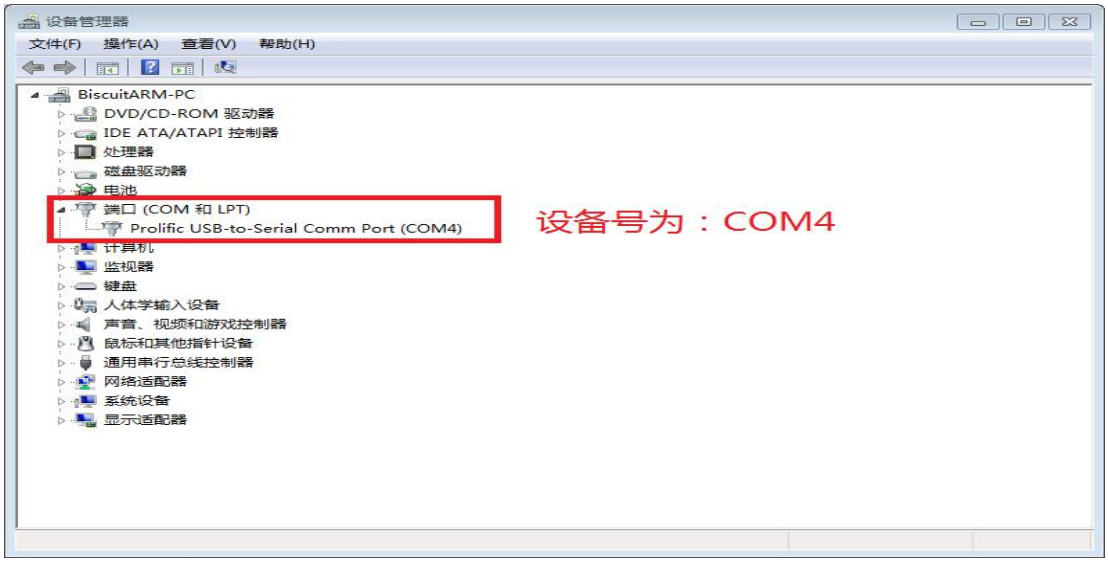
```
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
```

Double-click the downloaded putty.exe directly to open putty. The software interface is shown in the figure below.



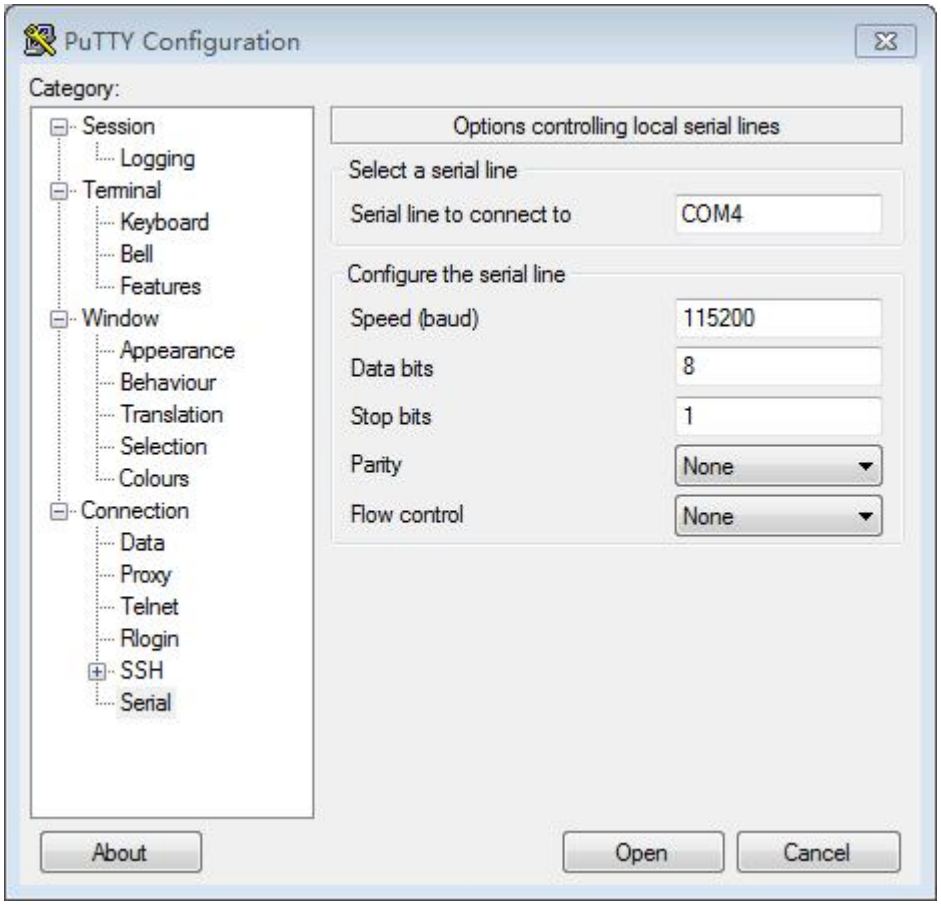
(3) Acquisition of equipment information

In Windows 7, we can check whether the serial port connection is normal and the device number of the serial port through the device manager. If the device is not recognized normally, please check whether the driver is successfully installed. If there is a problem with the driver installation, you can try to use 360 Driver Master to scan and install the driver.



(4) Putty configuration

Set the serial port to the corresponding port number (COM4), disable flow control, and set the speed to 115200



(5) Start debugging serial output

OrangePi is powered on and putty will automatically print serial port log information.

2. Use on Linux platforms

There is not much difference between using putty on the Linux platform and the Windows platform. The following mainly describes the steps where there are differences. All operations are based on Ubuntu 14.04 system.

(1) Install and launch Putty

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

(2) Configure Putty

The serial number can be viewed through ls / dev / ttyUSB *
Baud rate needs to be set to 115200
And turn off flow control