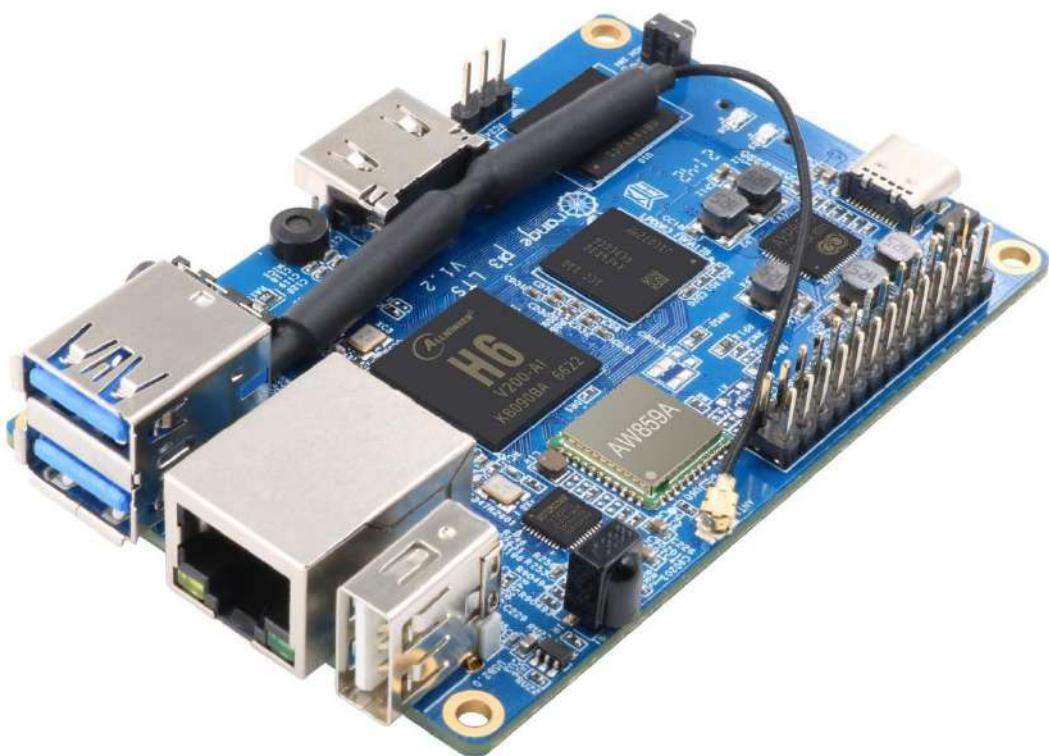




# Orange Pi 3 LTS

## User Manual





# Contents

1. Basic Features Of Orange Pi 3 LTS .....	1
1. 1. What is Orange Pi 3 LTS .....	1
1. 2. Purpose of Orange Pi 3 LTS .....	1
1. 3. Who is Orange Pi 3 LTS designed for?.....	1
1. 4. Hardware features of Orange Pi 3 LTS .....	2
1. 5. Top view and bottom view of Orange Pi 3 LTS .....	3
1. 6. Interface details of Orange Pi 3 LTS .....	4
2. Introduction to the use of the development board .....	6
2. 1. Prepare the necessary accessories .....	6
2. 2. Download the image of the development board and related materials .....	10
2. 3. Use the Android TV image pre-installed in EMMC to test the function of the development board .....	11
2. 4. Method of burning Linux image to TF card based on Windows PC .....	11
2. 4. 1. How to use balenaEtcher to burn a Linux image .....	11
2. 4. 2. How to use Win32Diskimager to burn Linux image .....	15
2. 5. The method of burning Linux image to TF card based on Ubuntu PC .....	17
2. 6. Method of programming Linux image to EMMC .....	21
2. 7. How to burn Android firmware to TF card .....	22
2. 8. How to program Android firmware to EMMC .....	26
2. 9. Start The Orange Pi Development Board .....	31
2. 10. How to use the debugging serial port .....	32
2. 10. 1. Connection instructions for debugging serial port.....	32
2. 10. 2. How to use the debugging serial port on Ubuntu platform .....	33



2. 10. 3. How to use the debugging serial port on Windows platform .....	36
2. 11. Instructions for using the 5v pin in the 26pin interface of the development board for power supply .....	39
3. Instructions for Debian and Ubuntu systems .....	40
3. 1. Supported linux distribution types and kernel versions .....	41
3. 2. Linux kernel driver adaptation .....	44
3. 3. Description of the linux command format in this manual .....	44
3. 4. Linux system login instructions .....	46
3. 4. 1. Linux system default login account and password .....	46
3. 4. 2. How to set the automatic login of the Linux system terminal .....	46
3. 4. 3. Instructions for automatic login of Linux desktop system .....	48
3. 4. 4. Setting method for automatic login of root user in Linux desktop system	50
3. 4. 5. How to disable the desktop in the Linux desktop system .....	51
3. 5. On-board LED light test description .....	53
3. 6. Operation instructions for the capacity of the rootfs partition of the Linux system in the TF card .....	54
3. 6. 1. The first boot will automatically expand the capacity of the rootfs partition in the TF card .....	54
3. 6. 2. The method of prohibiting the automatic expansion of the rootfs partition capacity in the TF card .....	56
3. 6. 3. How to manually expand the capacity of the rootfs partition in the TF card .....	58
3. 6. 4. Method to reduce the capacity of rootfs partition in TF card .....	63
3. 7. How to modify the linux log level (loglevel) .....	67
3. 8. Network connection test .....	68
3. 8. 1. Ethernet port test .....	68
3. 8. 2. WIFI connection test .....	70
3. 8. 3. How to use Hostapd to establish a WIFI hotspot .....	78
3. 8. 4. How to set static IP address .....	83
3. 8. 5. Set up the method of automatically connecting to the network when the Linux system starts up for the first time .....	92



3. 9. SSH remote login development board .....	96
3. 9. 1. SSH remote login development board under Ubuntu .....	96
3. 9. 2. SSH remote login development board under Windows .....	98
3. 10. HDMI related test items .....	99
3. 10. 1. HDMI Display Test .....	99
3. 10. 2. HDMI to VGA display test .....	100
3. 10. 3. Linux4.9 system HDMI resolution setting .....	101
3. 10. 4. How to Modify the Width and Height of Framebuffer in Linux 4.9 System .....	103
3. 10. 5. Framebuffer cursor related settings .....	106
3. 10. 6. How to hide the mouse cursor on the USB touch screen .....	106
3. 11. How to use Bluetooth .....	107
3. 11. 1. Test method for desktop image .....	107
3. 11. 2. How to use the server version image .....	111
3. 12. USB interface test .....	114
3. 12. 1. USB2.0 and USB3.0 interface description .....	114
3. 12. 2. Connect the mouse or keyboard to test .....	115
3. 12. 3. Connect USB storage device test .....	115
3. 12. 4. USB wireless network card test .....	116
3. 12. 5. USB network card test .....	119
3. 12. 6. USB camera test .....	120
3. 13. Audio Test .....	126
3. 13. 1. How to use the command line to play audio .....	126
3. 13. 2. Testing the audio method in the desktop system .....	127
3. 13. 3. MIC recording test .....	129
3. 14. Infrared receiving test .....	130
3. 15. Temperature sensor .....	131
3. 16. How to install Docker .....	132
3. 17. Pin Interface Pin Description .....	134
3. 18. How to install wiringOP .....	135
3. 19. 26pin interface GPIO, I2C, UART, SPI and PWM test .....	136



---

3. 19. 1. 26pin GPIO port test .....	136
3. 19. 2. 26pin SPI test .....	138
3. 19. 3. 26pin I2C test .....	139
3. 19. 4. 26pin UART test .....	141
3. 19. 5. PWM test method .....	143
3. 19. 6. How to use 0.96-inch OLED module with I2C interface .....	145
3. 19. 7. The method of outputting the kernel print information to the 26pin serial port.....	148
3. 20. How to use SPI LCD display .....	149
3. 20. 1. 2.4 inch SPI LCD display .....	149
3. 20. 2. 3.2 inch RPi SPI LCD display .....	154
3. 20. 3. 3.5 inch SPI LCD display .....	158
3. 21. Hardware watchdog test .....	162
3. 22. Set up Chinese environment and install Chinese input method .....	163
3. 22. 1. Installation method of Ubuntu system .....	163
3. 22. 2. Installation method of Debian system .....	170
3. 23. View the chipid of the H6 chip .....	178
3. 24. How to program linux image to eMMC .....	178
3. 25. The usage of eMMC storage space and memory after the Linux system is started .....	182
3. 26. The method of modifying the picture displayed in the startup phase of the Linux 4.9 system .....	184
3. 27. The method of remotely logging in to the Linux system desktop .....	185
3. 27. 1. Remote login using NoMachine .....	185
3. 27. 2. Remote login using VNC .....	196
3. 28. How to install Klipper firmware host computer using Kiauh .....	203
3. 29. Installation method of pagoda Linux panel .....	235
3. 30. Python related instructions .....	240
3. 30. 1. Method of compiling and installing Python source code .....	240
3. 30. 2. How to replace pip source in Python .....	241



---

3. 31. Installation method of OpenCV .....	242
3. 31. 1. Using apt to install OpenCV .....	242
3. 32. How to install Home Assistant .....	243
3. 32. 1. Installation via docker .....	243
3. 32. 2. Install via python .....	247
3. 33. Debian11 Kodi image usage instructions .....	249
3. 34. Tencent ncnn high-performance neural network forward computing framework test .....	257
3. 35. Installation and testing method of face_recognition face recognition library ..	265
3. 35. 1. Automatic installation of face_recognition using script .....	266
3. 35. 2. Manual installation of face_recognition .....	267
3. 35. 3. Test method of face_recognition .....	269
3. 36. Installation method of Tensorflow .....	280
3. 36. 1. The method of using script to automatically install Tensorflow .....	280
3. 36. 2. Steps to manually install Tensorflow .....	280
3. 37. ROS installation method .....	282
3. 37. 1. How to install ROS 1 Noetic .....	282
3. 37. 2. How to install ROS 2 Galactic .....	288
3. 38. Installation method of OpenMediaVault .....	292
3. 38. 1. Install OpenMediaVault 5.x on Debian 10 .....	292
3. 38. 2. Install OpenMediaVault 6.x on Debian11 .....	295
3. 39. Installation method of Pi-hole .....	307
3. 40. Introduction to the use of GotoHTTP .....	314
3. 41. Installation method of QT .....	318
3. 41. 1. How to install QT5 .....	318
3. 41. 2. How to install QT Creator .....	322
3. 42. GPU test description .....	331
3. 42. 1. Ubuntu 22.04 Linux5.16 system GPU test instructions .....	331
3. 43. Ubuntu22.04 method of installing browser .....	334
3. 44. Partial programming language test supported by Linux system .....	334



---

3. 44. 1. Debian Bullseye System .....	334
3. 44. 2. Debian Buster System .....	336
3. 44. 3. Ubuntu Jammy system .....	338
3. 44. 4. Ubuntu Focal system .....	340
3. 44. 5. Ubuntu Bionic system .....	342
3. 45. How to install kernel header files .....	343
3. 46. Shutdown and restart methods .....	346
<b>4. Instructions for use of Android TV system .....</b>	<b>348</b>
4. 1. Supported Android TV Versions .....	348
4. 2. Android 9.0 TV function adaptation .....	348
4. 3. On-board LED light display description .....	349
4. 4. How to return to the previous interface on Android .....	349
4. 5. How to use ADB .....	349
4. 5. 1. Enable USB debugging option .....	349
4. 5. 2. Using network connection adb debugging .....	351
4. 5. 3. Use the data cable to connect adb debugging .....	352
4. 6. HDMI 4K Display Instructions .....	353
4. 7. HDMI to VGA display test .....	354
4. 8. Wi-Fi connection method .....	355
4. 9. How to use WI-FI hotspot .....	358
4. 10. Bluetooth connection method .....	361
4. 11. How to use the USB camera .....	364
4. 12. eMMC storage space description .....	366
4. 13. Test method of serial port in 26pin interface .....	366
4. 14. Android system ROOT description .....	370
4. 15. Some Android APP installation instructions .....	372
4. 15. 1. Browser Installation Instructions .....	372
4. 15. 2. Tencent Video Installation Instructions .....	373



4. 15. 3. Youku Video Installation Instructions .....	373
4. 15. 4. iQIYI Video Installation Instructions .....	373
4. 15. 5. Installation Instructions .....	374
5. Linux SDK - Instructions for using the old version of orangepi-build .....	374
5. 1. Compilation system requirements .....	374
5. 2. Get the source code of linux sdk .....	377
5. 2. 1. Download orangepi-build from github .....	377
5. 2. 2. Download the cross-compilation toolchain .....	378
5. 2. 3. Orangepi-build complete directory structure description .....	380
5. 3. Compile u-boot.....	382
5. 4. Compile the linux kernel .....	386
5. 5. Compile rootfs.....	392
5. 6. Compile the linux image .....	395
6. Linux SDK - Instructions for using the new version of orangepi-build .....	399
6. 1. Compilation system requirements .....	400
6. 2. Get the source code of linux sdk .....	402
6. 2. 1. Download orangepi-build from github .....	402
6. 2. 2. Download the cross-compilation toolchain .....	404
6. 2. 3. Orangepi-build complete directory structure description .....	406
6. 3. Compile u-boot.....	407
6. 4. Compile the linux kernel .....	411
6. 5. Compile rootfs .....	415
6. 6. Compile the linux image .....	420
7. Android 9.0 SDK Instructions .....	424
7. 1. Download the source code of Android SDK .....	424
7. 2. Build Android Compilation Environment .....	425
7. 3. Compile Android image .....	426
7. 3. 1. Compile the kernel .....	426
7. 3. 2. Compile Android source code .....	428



# Version Update History

Version	Date	Update Instructions
v1.9	2022-04-19	<ol style="list-style-type: none"><li>1. How to test audio in desktop Linux system</li><li>2. Linux5.1x pwm test method</li><li>3. Test method for Linux5.1x headset audio playback</li><li>4. Linux: Tencent ncnn neural network forward computing framework test method</li><li>5. Ubuntu 20.04 ROS 2 Galactic installation method</li><li>6. Add a method for setting the automatic login of the linux system terminal</li><li>7. Linux: How to create a WIFI hotspot using Hostapd</li><li>8. Linux: How to use nmcli command to set static IP address</li><li>9. Linux: How to compile Python source code and how to replace pip source</li><li>10. Linux: How to install OpenCV using apt</li><li>11. Linux: Add Home Assistant installation method</li><li>12. Debian11 Kodi image usage instructions</li><li>13. Add a description of the linux command format in this manual</li><li>14. Add new version orangepi-build instructions</li><li>15. Android: Add a test method for the serial port in the 26pin interface</li><li>16. Add some notes</li></ol>
v2.0	2022-05-19	<ol style="list-style-type: none"><li>1. Support Ubuntu22.04 server and desktop image</li><li>2. Setting method for automatic login of root user in Linux desktop system</li><li>3. The method of disabling the desktop in the Linux desktop version system</li><li>4. Linux: The method of manually expanding the rootfs partition capacity in the TF card</li><li>5. Linux: Method to reduce the capacity of rootfs partition in TF card</li><li>6. How to hide the mouse cursor on the touch screen of the</li></ol>



		<p>Linux desktop version system</p> <p>7. Linux:face_recognition face recognition library installation and testing methods</p> <p>8. Debian: How to install OpenMediaVault 5.x and 6.x</p> <p>9. Installation method of Ubuntu 20.04 ROS 1 Noetic</p> <p>10. Linux: How to install Pi-hole</p> <p>11. Debian10: Tensorflow installation method</p> <p>12. Linux: How to install QT</p> <p>13. Linux: Introduction to the use of GotoHTTP</p> <p>14. Linux: How to use Cheese to test a USB camera</p> <p>15. Ubuntu22.04 Linux5.16 GPU test instructions</p>
v2.1	2022-06-01	<p>1. Linux: How to install kernel header files</p> <p>2. How to set up the method of automatically connecting to the network when the Linux system starts up for the first time</p>
v2.2	2022-06-1x	<p>1. KlipperScreen open mouse cursor method</p>



# 1. Basic Features Of Orange Pi 3 LTS

## 1. 1. What is Orange Pi 3 LTS

Orange Pi is an open source single-board computer, a new generation of arm64 development boards, which can run operating systems such as Android TV 9.0, Ubuntu and Debian. The Orange Pi 3 LTS uses the Allwinner H6 chip and has 2GB LPDDR3 memory.

## 1. 2. Purpose of Orange Pi 3 LTS

We can use it to build:

- A small Linux desktop computer
- A small Linux web server
- Install Klipper host computer to control 3D printer
- Android TV box

**Of course, there are other more functions, because the Orange Pi development board can install Linux systems such as Debian and Ubuntu, as well as systems such as Android TV, which means that we can implement it within the scope of the hardware and software support of the development board. Various functions.**

## 1. 3. Who is Orange Pi 3 LTS designed for?

The Orange Pi development board is not just a consumer product, it is designed for anyone who wants to use technology to create and innovate. It's a simple, fun, and useful tool that you can use to shape the world around you.



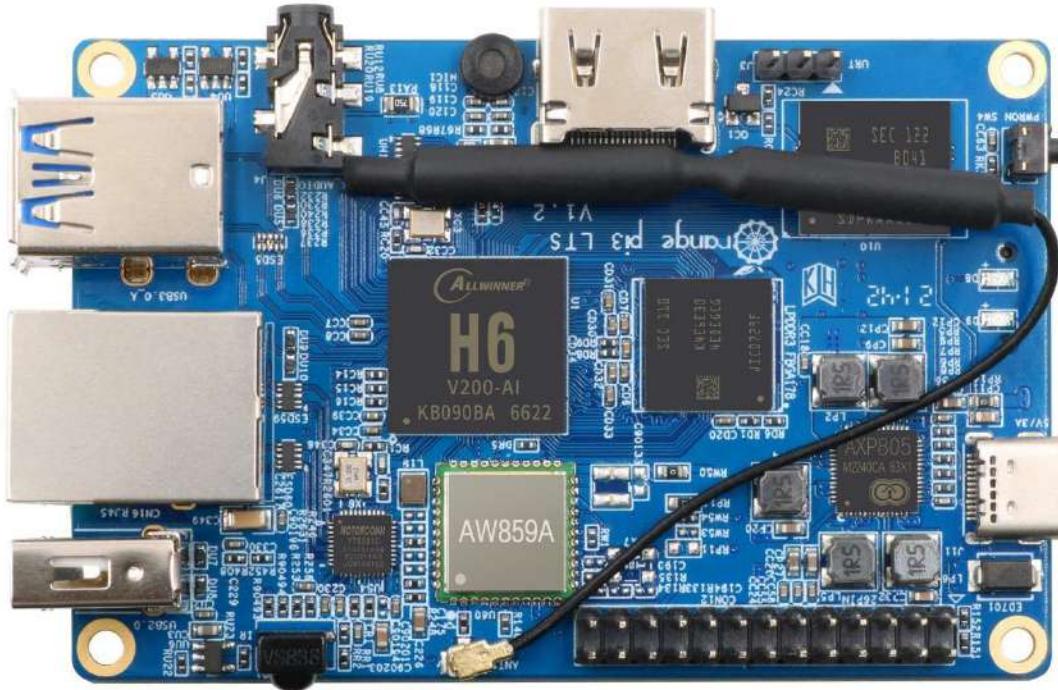
## 1. 4. Hardware features of Orange Pi 3 LTS

<b>Hardware Features Introduction</b>	
CPU	Allwinner H6 quad-core 64-bit 1.8GHz high-performance Cortex-A53 processor
GPU	<ul style="list-style-type: none"><li>• High-performance multi-core GPU Mali T720</li><li>• OpenGL ES3.1/3.0/2.0/1.1</li></ul>
Memory	2GB LPDDR3(shared with GPU)
Onboard storage	TF card slot、8GB EMMC
Ethernet	YT8531C chip, Support 10/100M/1000M Ethernet
WIFI+Bluetooth	AW859Achip、Support IEEE 802.11 a/b/g/n/ac, BT5.0
Video output	HDMI 2.0a 、TV CVBS output
Audio output	HDMI output, 3.5mm audio port
Power supply	USB Type-Cpower supply
PMU	AXP805
USB port	1xUSB 3.0 HOST、1xUSB 2.0 HOST、1xUSB2.0 OTG
26pin header	1xI2C、1xSPI、1xUART and multiple GPIO
Debug serial port	UART-TX、UART-RX and GND
LED lights	Power indicator and status indicator
IR receiver	Infrared remote control for orange pi
Button	power button (SW4)
Supported OS	Android9.0 TV、Ubuntu、Debian and Manjaro OS
<b>Appearance specification introduction</b>	
Product Size	85mm×56mm
Weight	45g
Orange Pi™ is the registered trademark of Shenzhen Xunlong Software Co., Ltd.	

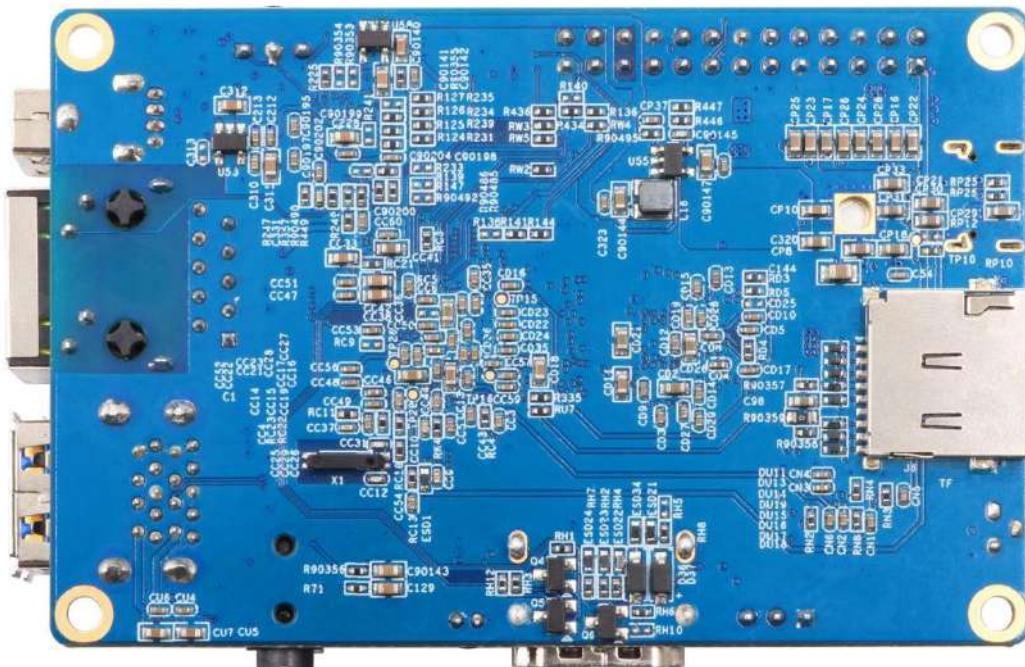


## 1. 5. Top view and bottom view of Orange Pi 3 LTS

top view:

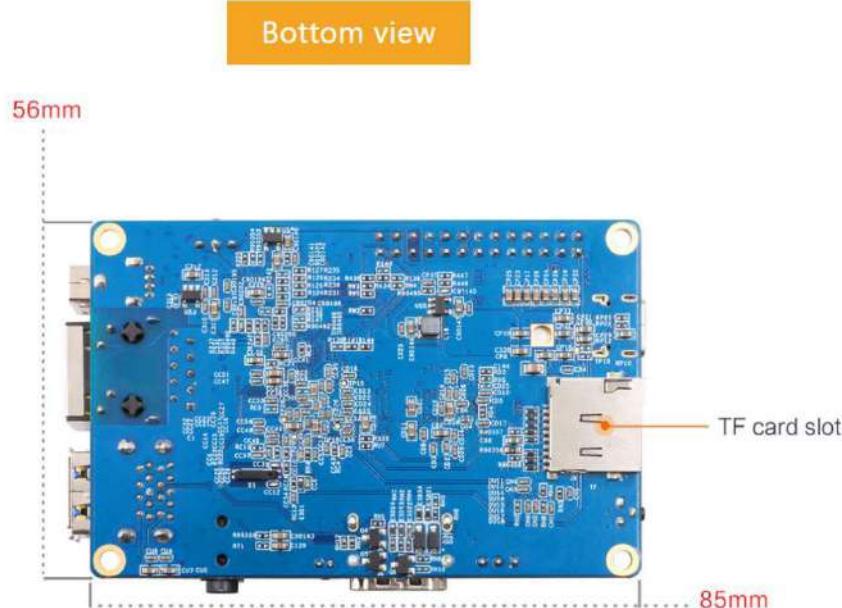
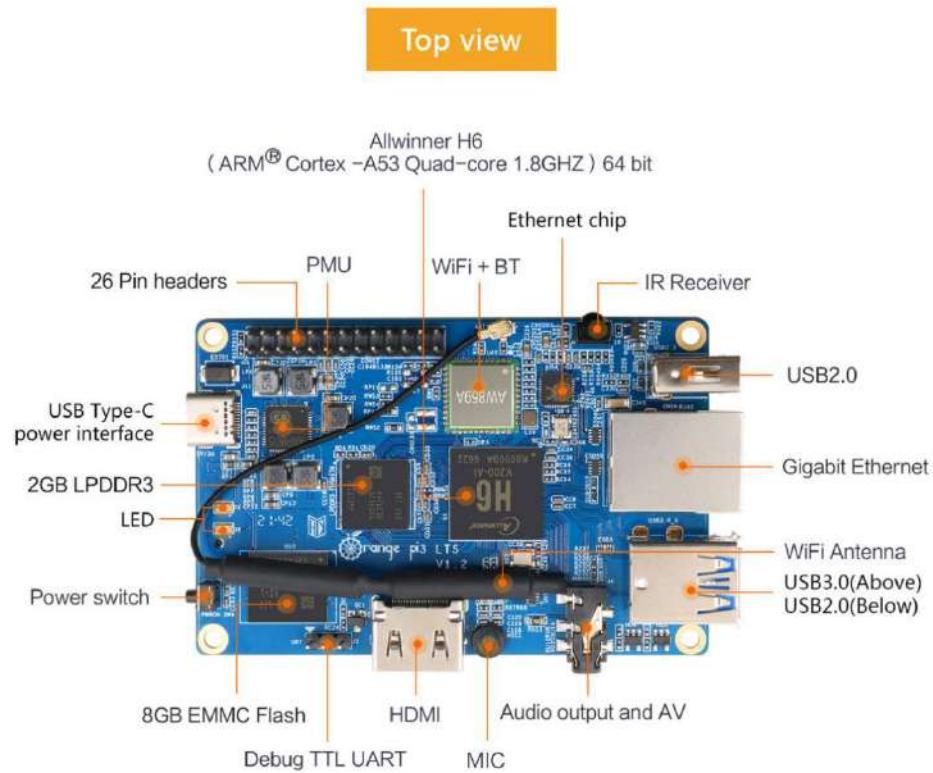


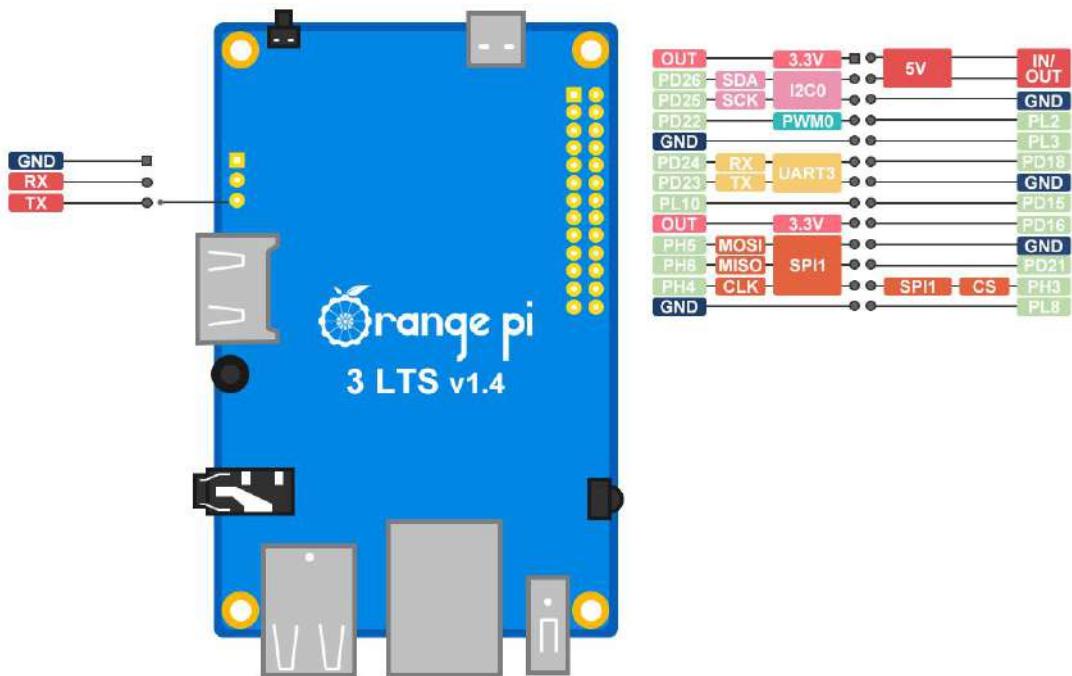
bottom view:





## 1. 6. Interface details of Orange Pi 3 LTS





The diameters of the four positioning holes are all 3.0mm.



## 2. Introduction to the use of the development board

### 2. 1. Prepare the necessary accessories

- 1) TF card, class10 or above high-speed card with a minimum capacity of 8GB, it is recommended to use SanDisk TF card, Orange Pi test is to use SanDisk TF card, other brands of TF card may cause the problem that the system cannot be started



- 2) TF card reader, used to read and write TF card



- 3) HDMI to HDMI cable, used to connect the development board to an HDMI monitor or TV for display



- 4) Power supply, if there is a 5V/2A or 5V/3A power supply head, you only need to prepare a data cable with the USB to Type C interface shown in the picture on the left below, and you can also use a cable similar to the picture shown on the right below. 5V/2A or 5V/3A high-quality USB Type C interface power adapter integrated with the power head



- 5) USB interface mouse and keyboard, as long as it is a standard USB interface mouse and keyboard, the mouse and keyboard can be used to control the Orange Pi development board
- 6) Infrared remote control, mainly used to control the Android system

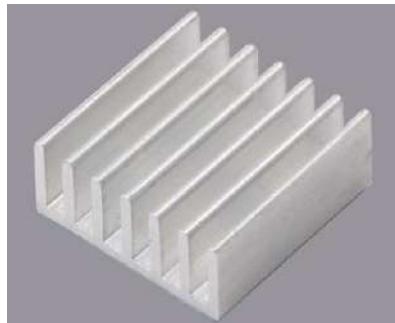


**Note that the remote control of the air conditioner or the remote control of the TV cannot control the Orange Pi development board, only the remote control provided by the Orange Pi can.**

- 7) Fast or Gigabit Ethernet cable to connect the development board to the Internet
- 8) AV video cable, if you want to display the video through the CVBS interface instead of the HDMI interface, then you need to connect the development board to the TV through the AV video cable



- 9) Heat sink, if you are worried that the temperature of the development board is too high, you can add a heat sink, and the heat sink can be attached to the H6 chip. If you think the temperature is still too high, you can also stick a heat sink on the memory chip on the right side of the H6.



- 10) The **5V** cooling fan is **optional**. As shown in the figure below, there are 5V and GND pins on the 26pin interface of the development board that can be connected to the cooling fan. The spacing between the pins of the 26pin interface is **2.54mm**. Refer to this specification for the power interface of the cooling fan. Go to Taobao to buy

**After the development board is plugged into the Type C power supply, the 5V pin can be used directly without other settings. Also, please note that the 5V pin cannot be controlled by software.**



- 11) Matching case, which can be used to protect the development board



- 12) USB to 3.3v TTL module and DuPont cable, when using the debug serial port, USB to TTL module and DuPont cable are required to connect the development board and computer



**Note that the TTL level used by the development board is 3.3v. Except for the USB to TTL module shown in the figure above, other similar 3.3v USB to TTL modules are generally available.**

- 13) A PC with Ubuntu and Windows operating systems installed



1	Ubuntu18.04 PC	Optional, used to compile Linux source code and Android source code
2	Windows PC	For burning Android and Linux images

## 2. 2. Download the image of the development board and related materials

- 1) The download URL of the English version is:

<http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-pi-3-LTS.html>



- 2) The information mainly includes
  - Android source code:** Save on Google network disk
  - Linux source code:** Save on github
  - User Manual and Schematics:** The data sheet related to the chip will also be



- placed here
- d. **official tool:** It mainly includes the software that needs to be used during the use of the development board
  - e. **Android image:** Save on Google network disk
  - f. **Ubuntu image:** Save on Google network disk
  - g. **Debian image:** Save on Google network disk

## 2. 3. Use the Android TV image pre-installed in EMMC to test the function of the development board

The development board has 8GB EMMC. After getting the development board, you can use the Android9.0 TV image pre-installed in the EMMC to test the functions of the development board. After confirming that all the hardware functions of the development board are correct, you can burn the one you want to use. system.

## 2. 4. Method of burning Linux image to TF card based on Windows PC

Note that the Linux image mentioned here specifically refers to a Linux distribution image such as Debian, Ubuntu or Manjaro downloaded from the Orange Pi data download page.

### 2. 4. 1. How to use balenaEtcher to burn a Linux image

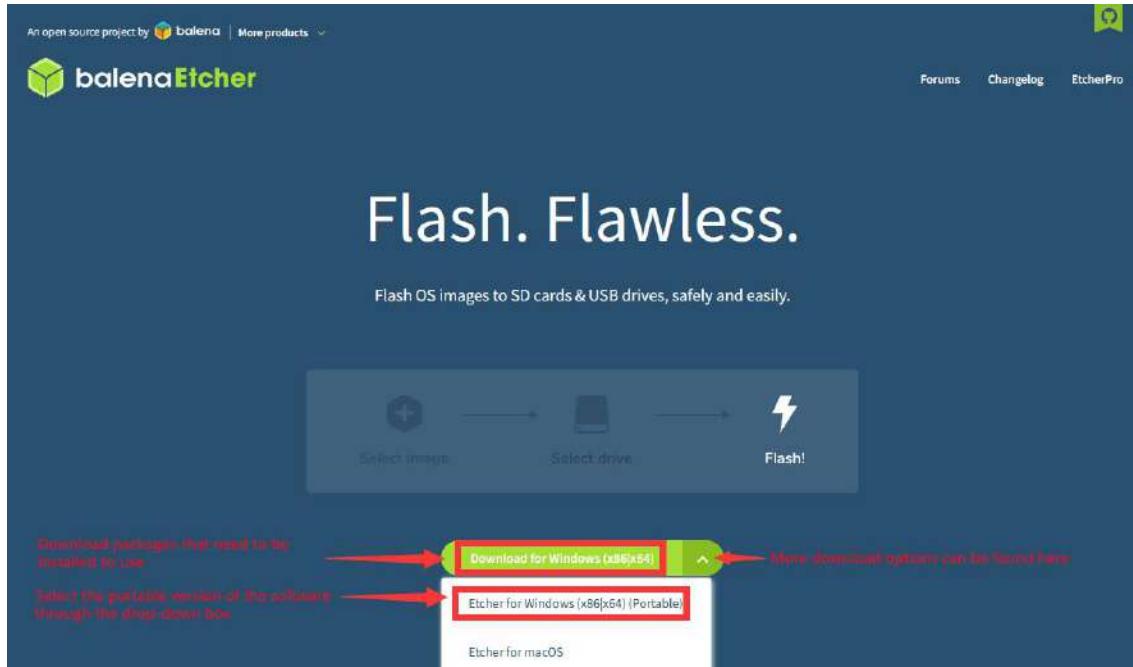
- 1) First prepare a TF card with a capacity of 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk
- 2) Then use the card reader to insert the TF card into the computer
- 3) Download the compressed package of the Linux operating system image file you want to burn from [the data download page of Orange Pi](#), and then use the decompression software to decompress it. In the decompressed file, the file ending with ".img" is the image file of the operating system. The size is generally more than 1GB
- 4) Then download the burning software of the Linux image - **balenaEtcher**, the



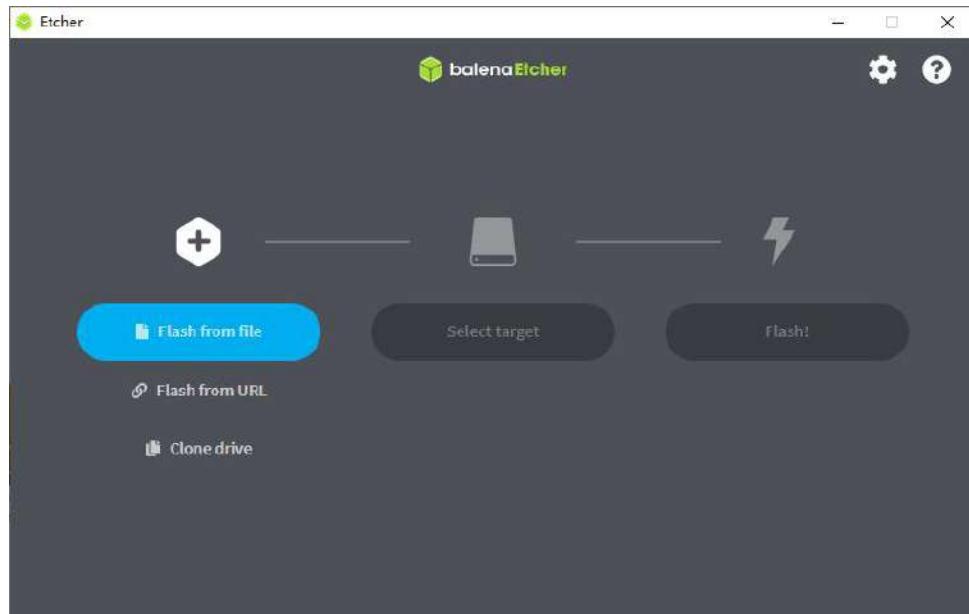
download address is

<https://www.balena.io/etcher/>

5) After entering the balenaEtcher download page, click the green download button to download the installation package of balenaEtcher. You can also select the Portable version of balenaEtcher through the drop-down box. The Portable version does not need to be installed. Double-click to open it and use it



6) If you download a version of balenaEtcher that needs to be installed, please install it before using it. If you download the Portable version of balenaEtcher, just double-click to open it. The opened balenaEtcher interface is shown in the figure below.

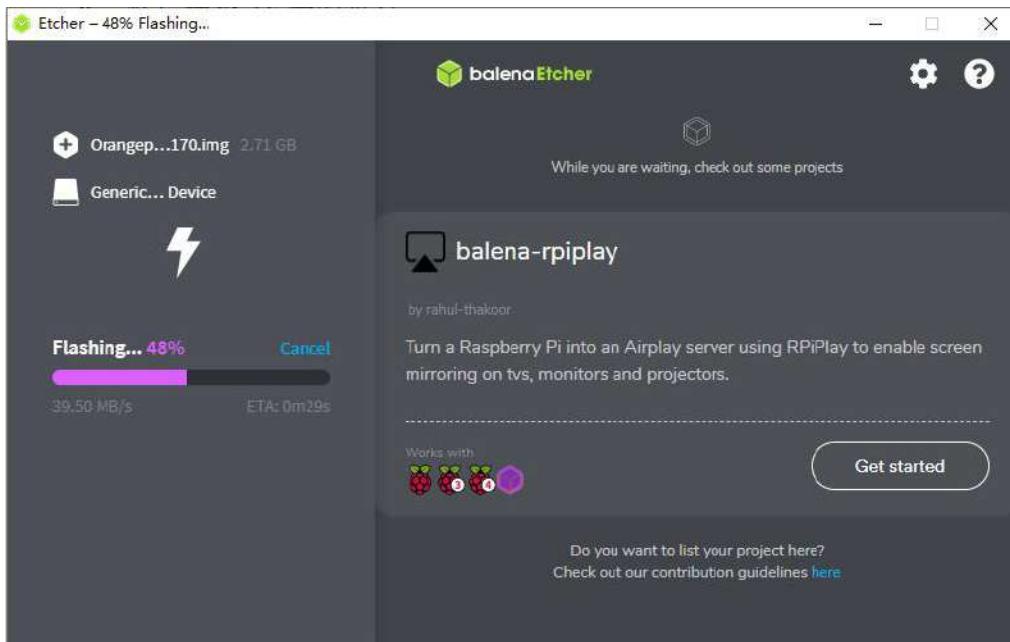


7) The specific steps to use balenaEtcher to burn a Linux image are as follows

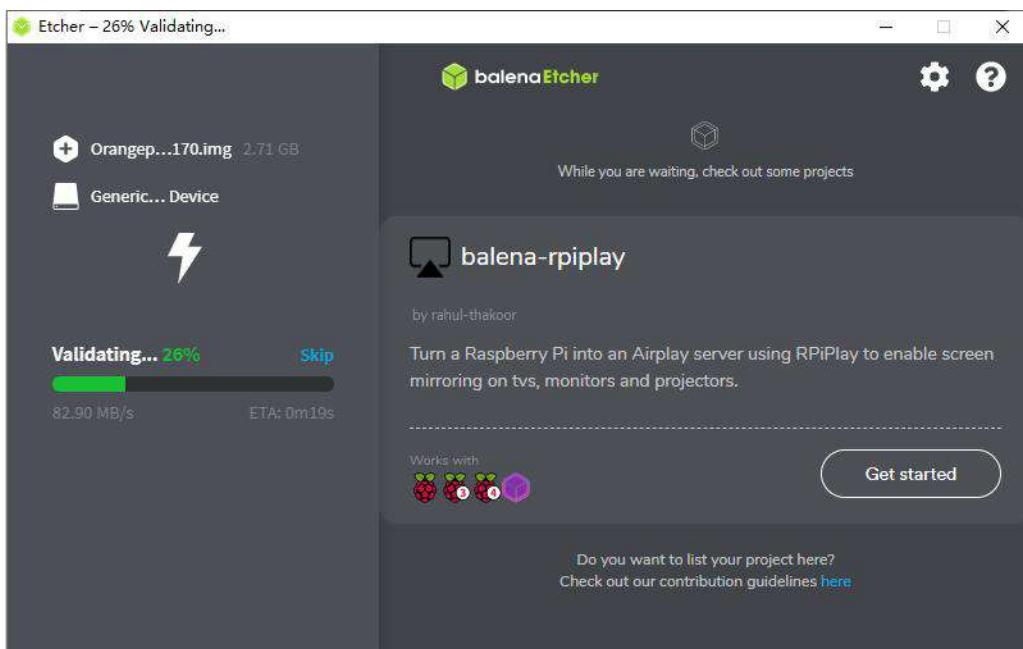
- First select the path of the Linux image file to be burned
- Then select the drive letter of the TF card
- Finally, click Flash to start burning the Linux image to the TF card



8) The interface displayed in the process of balenaEtcher burning the Linux image is shown in the figure below. In addition, the progress bar shows purple to indicate that the Linux image is being burned to the TF card.



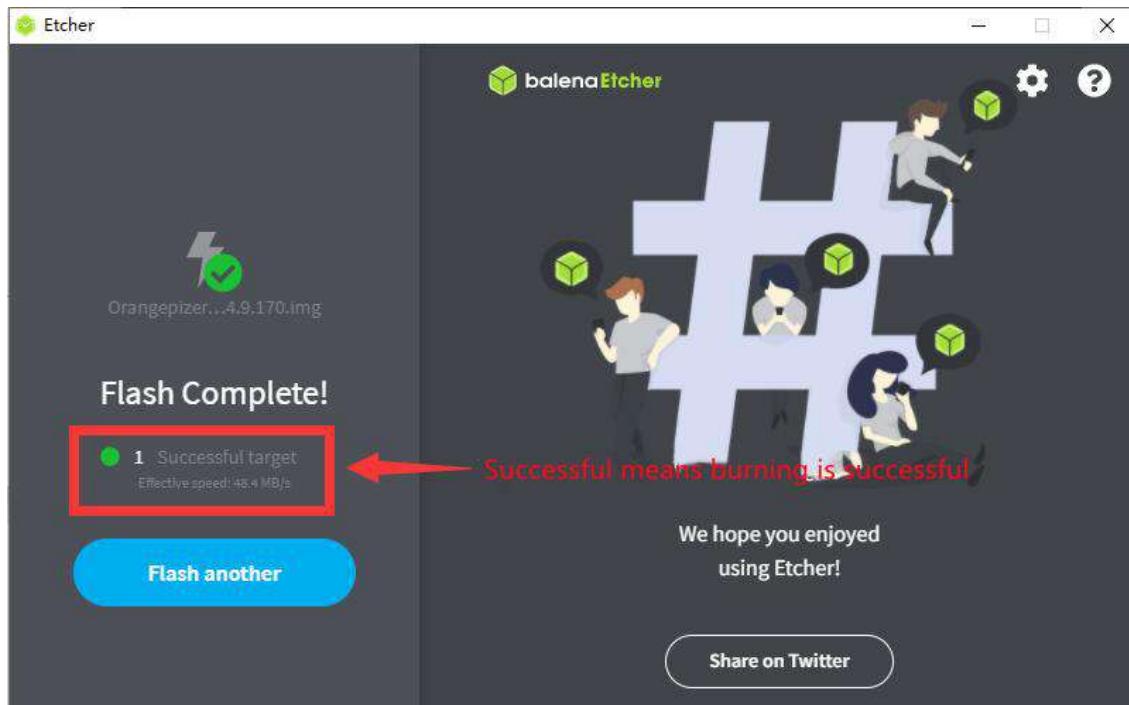
- 9) After the Linux image is burned, balenaEtcher will also verify the image burned to the TF card by default to ensure that there is no problem in the burning process. As shown in the figure below, a green progress bar indicates that the image has been burned, and balenaEtcher is verifying the burned image.



- 10) After the successful burning is completed, the display interface of balenaEtcher is shown in the figure below. If a green indicator icon is displayed, it means that the image



burning is successful. At this time, you can exit balenaEtcher, and then pull out the TF card and insert it into the TF card slot of the development board.



## 2. 4. 2. How to use Win32Diskimager to burn Linux image

1) First prepare a TF card with a capacity of 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk

2) Then use the card reader to insert the TF card into the computer

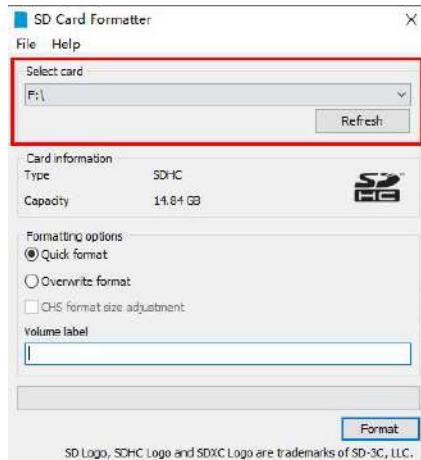
3) Then format the TF card

- a. The **SD Card Formatter** software can be used to format the TF card, and its download address is

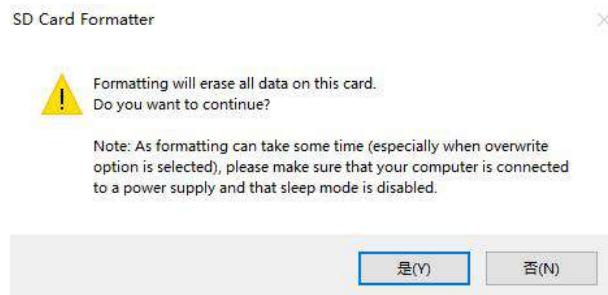
[https://www.scdcard.org/downloads/formatter/eula\\_windows/SDCardFormatterv5\\_WinEN.zip](https://www.scdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip)

- b. After downloading, unzip and install directly, and then open the software

- c. If only the TF card is inserted into the computer, the “**Select card**” column will display the drive letter of the TF card. If multiple USB storage devices are inserted into the computer, you can select the drive letter corresponding to the TF card through the drop-down box.



- d. Then click "**Format**", a warning box will pop up before formatting, select "**Yes (Y)**" to start formatting



- e. After formatting the TF card, the information shown in the figure below will pop up, click OK.



- 4) Download the compressed package of the Linux operating system image file you want to burn from [the data download page of Orange Pi](#), and then use the decompression software to decompress it. In the decompressed file, the file ending with ".img" is the image file of the operating system. The size is generally more than 1GB



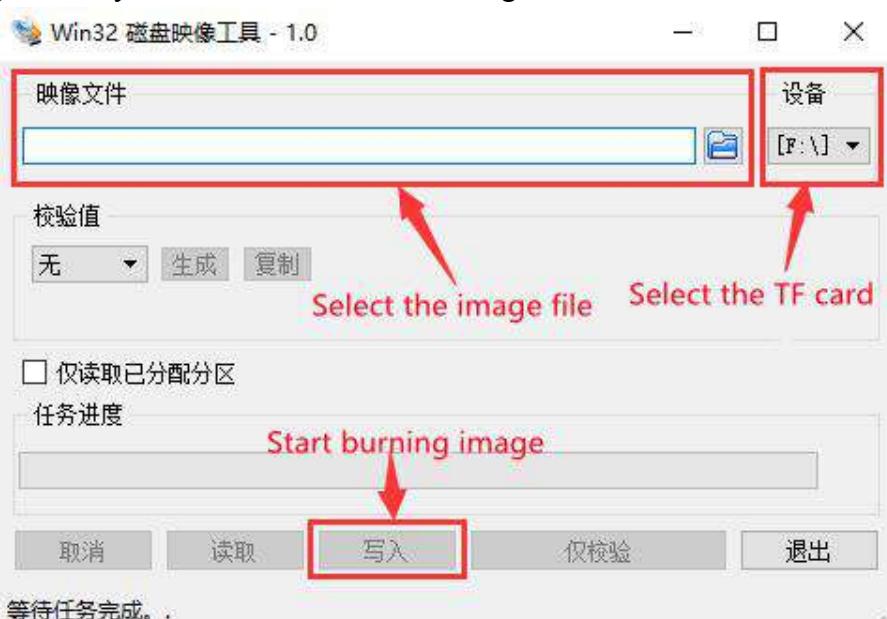
### 5) Use Win32Diskimager to burn the Linux image to the TF card

- The download page of Win32Diskimager is

<http://sourceforge.net/projects/win32diskimager/files/Archive/>

- After downloading, install it directly. The Win32Diskimager interface is as follows

- First select the path of the image file
- Then confirm that the drive letter of the TF card is consistent with the one displayed in the "Device" column
- Finally click "Write" to start burning



- After the image writing is completed, click the "Exit" button to exit, and then you can pull out the TF card and insert it into the development board to start

## 2. 5. The method of burning Linux image to TF card based on Ubuntu PC

Note that the Linux image mentioned here specifically refers to a Linux distribution image such as Debian, Ubuntu or Manjaro downloaded from the Orange Pi data download page, and Ubuntu PC refers to a personal computer with Ubuntu installed.



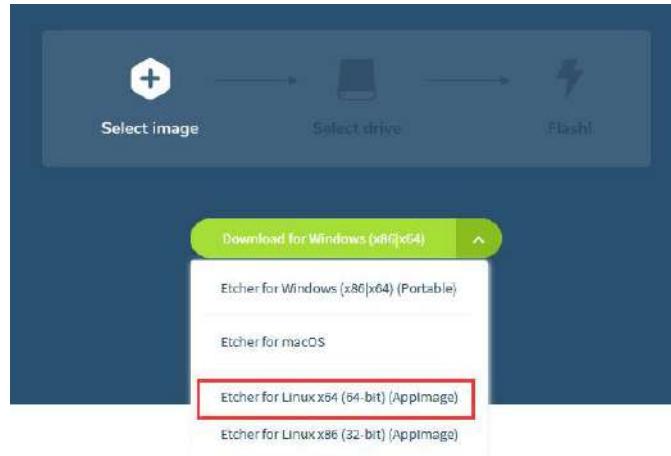
1) TF First prepare a TF card with a capacity of 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk

2) Then use the card reader to insert the TF card into the computer

3) Download balenaEtcher software, the download address is

<https://www.balena.io/etcher/>

4) After entering the balenaEtcher download page, please select the Linux version of the software through the drop-down box to download



5) After downloading, please use the **unzip** command to decompress the downloaded compressed package. The decompressed **balenaEtcher-1.5.109-x64.AppImage** is the software needed to burn the Linux image

```
test@test:~$ unzip balena-etcher-electron-1.5.109-linux-x64.zip
Archive: balena-etcher-electron-1.5.109-linux-x64.zip
      inflating: balenaEtcher-1.5.109-x64.AppImage
test@test:~$ ls
balenaEtcher-1.5.109-x64.AppImage  balena-etcher-electron-1.5.109-linux-x64.zip
```

6) Download the compressed package of the Linux operating system image file you want to burn from **the data download page of Orange Pi**, and then use the decompression software to decompress it. In the decompressed file, the file ending with ".img" is the image file of the operating system. The size is generally more than 1GB

The decompression command for the compressed package ending in 7z is as follows

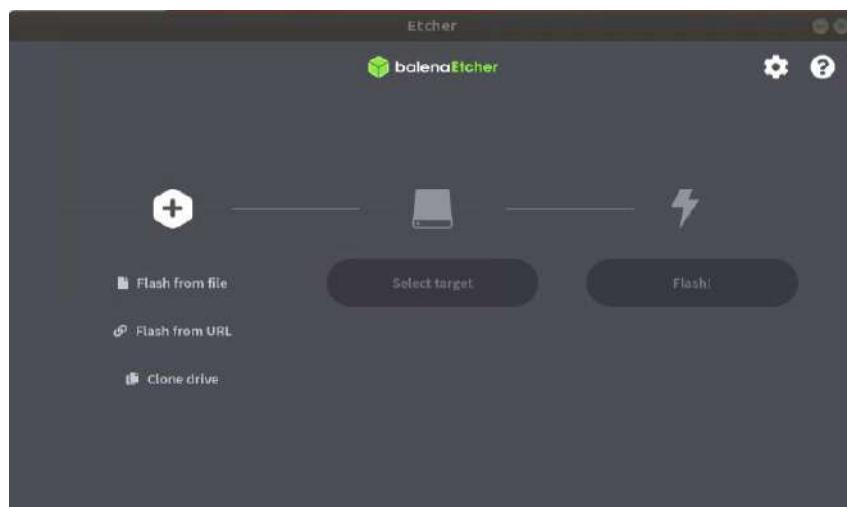


```
test@test:~$ 7z x OrangePi3-Lts_2.1.8_ubuntu_bionic_server_linux4.9.118.7z
test@test:~$ ls OrangePi3-Lts_2.1.8_ubuntu_bionic_server_linux4.9.118.* 
OrangePi3-lts_2.1.8_ubuntu_bionic_server_linux4.9.118.7z
OrangePi3-lts_2.1.8_ubuntu_bionic_server_linux4.9.118.img.sha #checksum file
OrangePi3-lts_2.1.8_ubuntu_bionic_server_linux4.9.118.img #image file
```

7) After decompressing the image, you can use the **sha256sum -c \*.sha** command to calculate whether the checksum is correct. If the message is **successful**, it means that the downloaded image is correct. You can safely burn it to the TF card. If the **checksum does not match**, it means that There is a problem with the downloaded image, please try to download again

```
test@test:~$ sha256sum -c *.sha
OrangePi3-lts_2.1.8_ubuntu_bionic_server_linux4.9.118.img: success
```

8) Then double-click **balenaEtcher-1.5.109-x64.AppImage** on the graphical interface of Ubuntu PC to open balenaEtcher (**no installation required**), and the interface after balenaEtcher is opened is shown in the following figure



9) The specific steps to use balenaEtcher to burn a Linux image are as follows

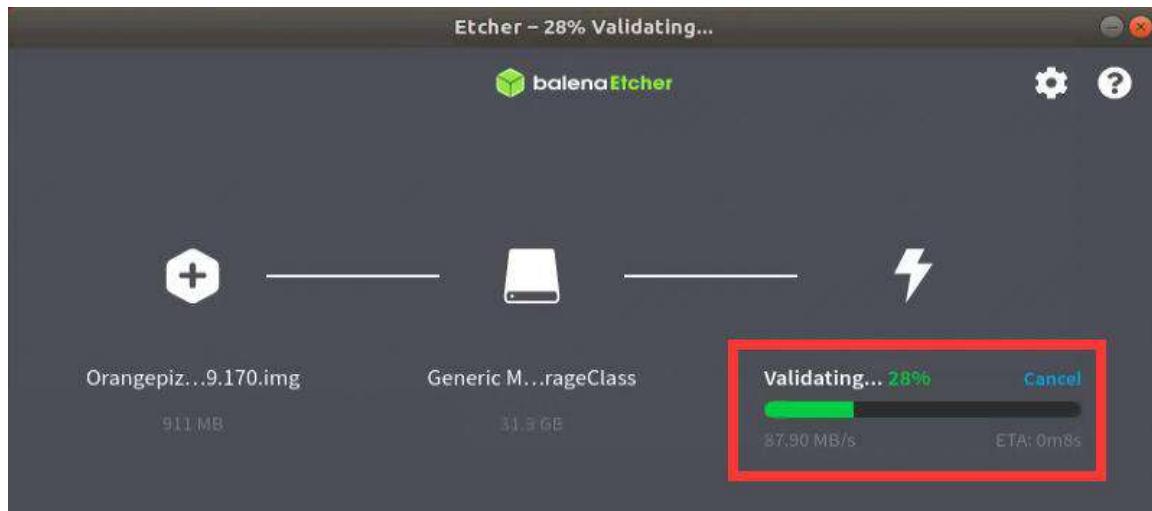
- First select the path of the Linux image file to be burned
- Then select the drive letter of the TF card
- Finally, click Flash to start burning the Linux image to the TF card



- 10) The interface displayed in the process of balenaEtcher burning the Linux image is shown in the figure below. In addition, the progress bar shows purple to indicate that the Linux image is being burned to the TF card.



- 11) After the Linux image is burned, balenaEtcher will also verify the image burned to the TF card by default to ensure that there is no problem in the burning process. As shown in the figure below, a green progress bar indicates that the image has been burned, and balenaEtcher is verifying the burned image.



12) After the successful burning, the display interface of balenaEtcher is shown in the figure below. If the green indicator icon is displayed, it means that the image burning is successful. At this time, you can exit balenaEtcher, and then pull out the TF card and insert it into the TF card slot of the development board.



## 2. 6. Method of programming Linux image to EMMC

See [the method of burning linux image to EMMC](#)



The blue word part is a hyperlink, click to jump to the corresponding position.

## 2. 7. How to burn Android firmware to TF card

The Android image of the development board can only be burned into the TF card using the **PhoenixCard** software under the Windows platform, and the version of the PhoenixCard software cannot be lower than **PhoenixCard v4.1.2**.

Please do not use software that burns Linux images, such as Win32Diskimager or balenaEtcher, to burn Android images.

In addition, the PhoenixCard software does not have versions for Linux and Mac platforms, so it is impossible to burn Android images to TF cards under Linux and Mac platforms.

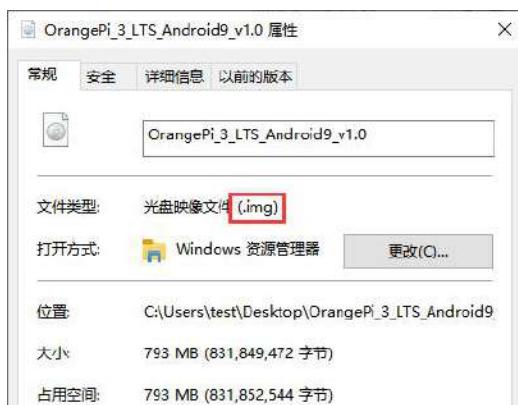
- 1) First prepare a TF card with a capacity of 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk
- 2) Then use the card reader to insert the TF card into the computer
- 3) Download Android 9.0 firmware and PhoenixCard programming tool from **Orange Pi's data download page**, please make sure that the version of PhonenixCrad tool is **PhoenixCard v4.1.2 or PhoenixCard v4.1.2 or later**
- 4) Then use the decompression software to decompress the downloaded Android image compression package. In the decompressed file, the file ending with ".img" is the Android image file
  - a. If you don't know how to decompress the compressed package of the Android image, you can install a **360 compression software**



- b. After decompressing with 360 compression software, you can see the following files

名称	修改日期	类型	大小
OrangePi_3_LTS_Android9_v1.0	2021/12/21 13:13	光盘映像文件	812,353 KB
OrangePi_3_LTS_Android9_v1.0.img.md5sum	2021/12/21 13:29	MD5SUM 文件	1 KB

- c. The first 800M file is the Android image file to be burned. Check its properties as shown in the figure below



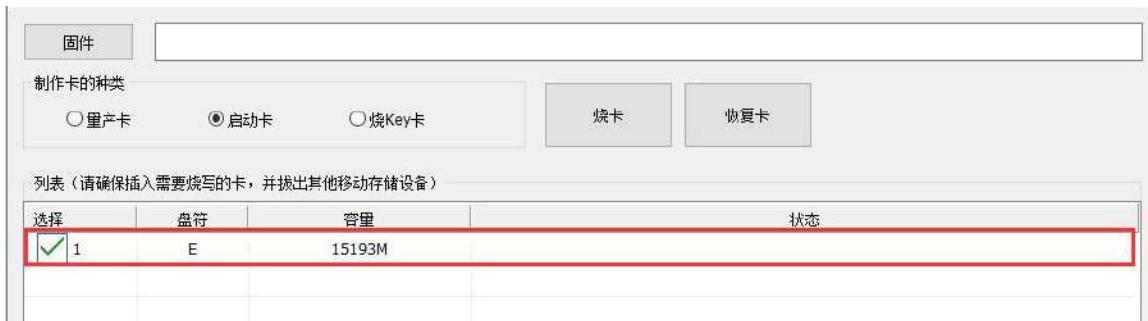
- 5) Use the decompression software to decompress **PhoenixCard v4.1.2.rar**, this software does not need to be installed, just find **PhoenixCard** in the decompressed folder and open it

option.cfg	2017/6/27 16:53	CFG 文件	1 KB
ParserManager.dll	2017/6/28 17:51	应用程序扩展	81 KB
PhoenixCard ManualV4.1.1	2017/7/5 15:05	DOC 文档	382 KB
PhoenixCard	2017/10/25 15:03	应用程序	1,742 KB
PhoenixCard.lan	2017/10/25 15:16	LAN 文件	3 KB
PhoenixCard.pdb	2017/10/25 15:03	PDB 文件	22,971 KB

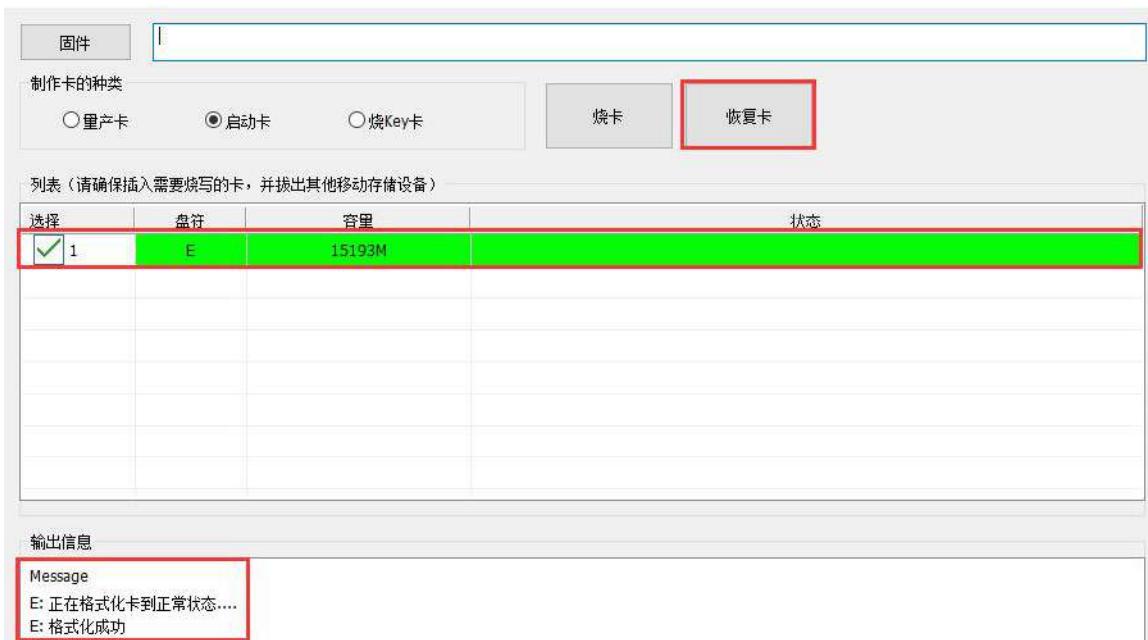
- 6) After opening PhoenixCard, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that the displayed drive letter is the same as the drive letter of the TF card you want to burn.**



If There is no display, you can try to unplug the TF card

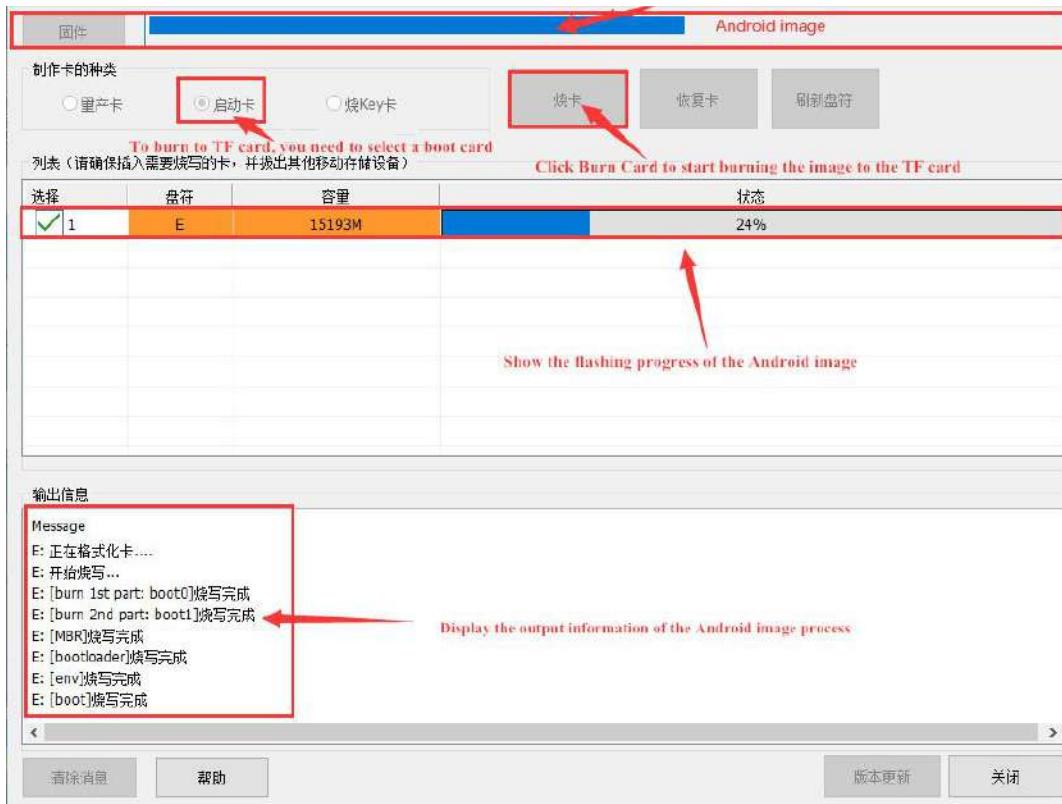


7) After confirming the drive letter, first format the TF card, click the recovery card button in PhoenixCard, or use the **SD Card Formatter** mentioned above to format the TF card

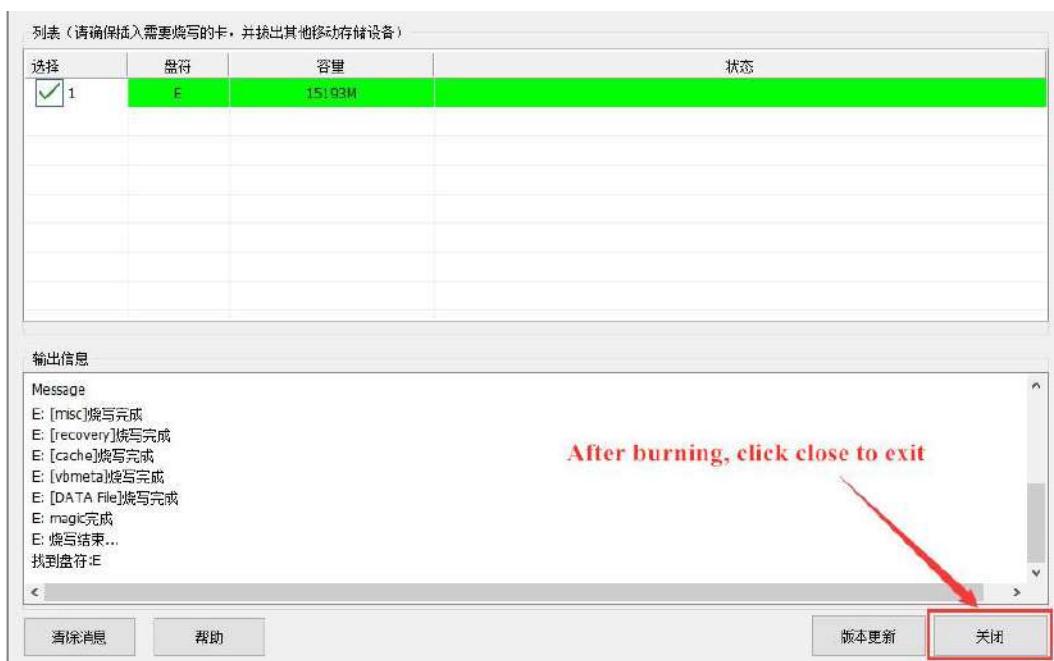


8) Then start writing Android firmware to TF card

- d. First select the path of the Android firmware in the "**Firmware**" column
- e. Select "**Startup Card**" in "**Type of Creation Card**"
- f. Then click the "**burn card**" button to start burning



9) After burning, the display of PhoenixCard is as shown in the figure below. At this time, click the **close** button to exit PhoenixCard, and then you can pull out the TF card from the computer and insert it into the development board to start.





## 2. 8. How to program Android firmware to EMMC

The Android image of the development board can only be burned to the TF card using the **PhoenixCard** software under the Windows platform, and then burned to the EMMC through the TF card. The version of the PhoenixCard software cannot be lower than **PhoenixCard v4.1.2**.

Please do not use software that burns Linux images, such as Win32Diskimager or balenaEtcher, to burn Android images.

In addition, the PhoenixCard software does not have versions for Linux and Mac platforms, so it is impossible to burn Android images to TF cards under Linux and Mac platforms.

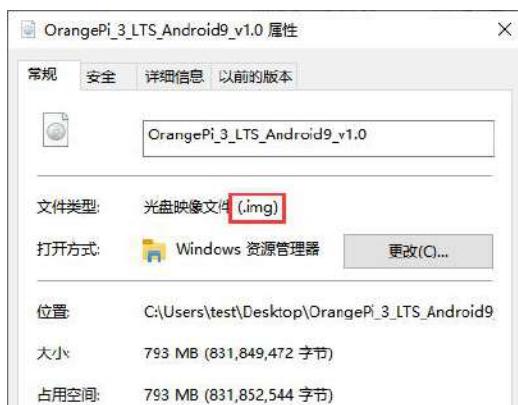
- 1) First of all, please note that burning the Android firmware into the EMMC of the development board needs to be done with the help of a TF card, which is mainly divided into the following two steps
  - a. First use PhoenixCard to burn the Android firmware to the TF card in the form of a mass production card
  - b. Then use TF card to burn Android firmware into EMMC
- 2) First prepare a TF card with a capacity of 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk
- 3) Then use the card reader to insert the TF card into the computer
- 4) Download the Android 9.0 firmware and PhoenixCard programming tool from [Orange Pi's data download page](#), please make sure that the version of the PhoenixCard tool is **PhoenixCard v4.1.2 or a version above PhoenixCard v4.1.2**
- 5) Use the decompression software to decompress the downloaded Android firmware compressed package. In the decompressed file, the file ending with ".img" is the Android firmware
  - a. If you don't know how to decompress the compressed package of the Android image, you can install a [360 compression software](#)



- b. After decompressing with 360 compression software, you can see the following files

名称	修改日期	类型	大小
OrangePi_3_LTS_Android9_v1.0	2021/12/21 13:13	光盘映像文件	812,353 KB
OrangePi_3_LTS_Android9_v1.0.img.md5sum	2021/12/21 13:29	MD5SUM 文件	1 KB

- c. The first 800M file is the Android image file to be burned. Check its properties as shown in the figure below



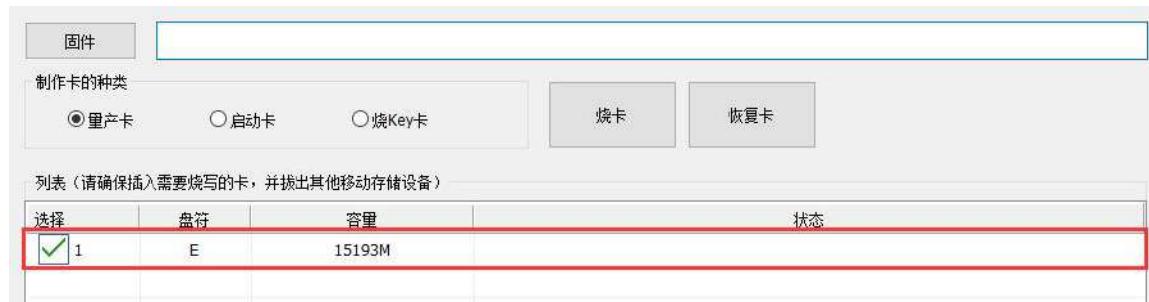
- 6) Use the decompression software to decompress **PhoenixCard v4.1.2.rar**, this software does not need to be installed, just find **PhoenixCard** in the decompressed folder and open it

option.cfg	2017/6/27 16:53	CFG 文件	1 KB
ParserManager.dll	2017/6/28 17:51	应用程序扩展	81 KB
PhoenixCard ManualV4.1.1	2017/7/5 15:05	DOC 文档	382 KB
PhoenixCard	2017/10/25 15:03	应用程序	1,742 KB
PhoenixCard.lan	2017/10/25 15:16	LAN 文件	3 KB
PhoenixCard.pdb	2017/10/25 15:03	PDB 文件	22,971 KB

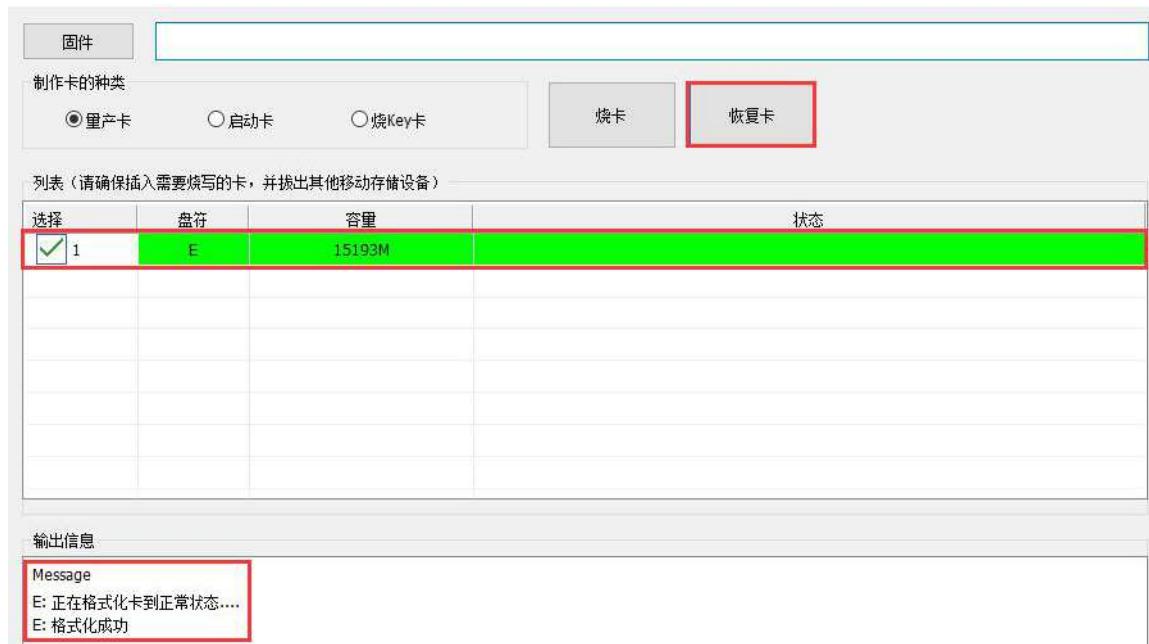
- 7) After opening **PhoenixCard**, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that**



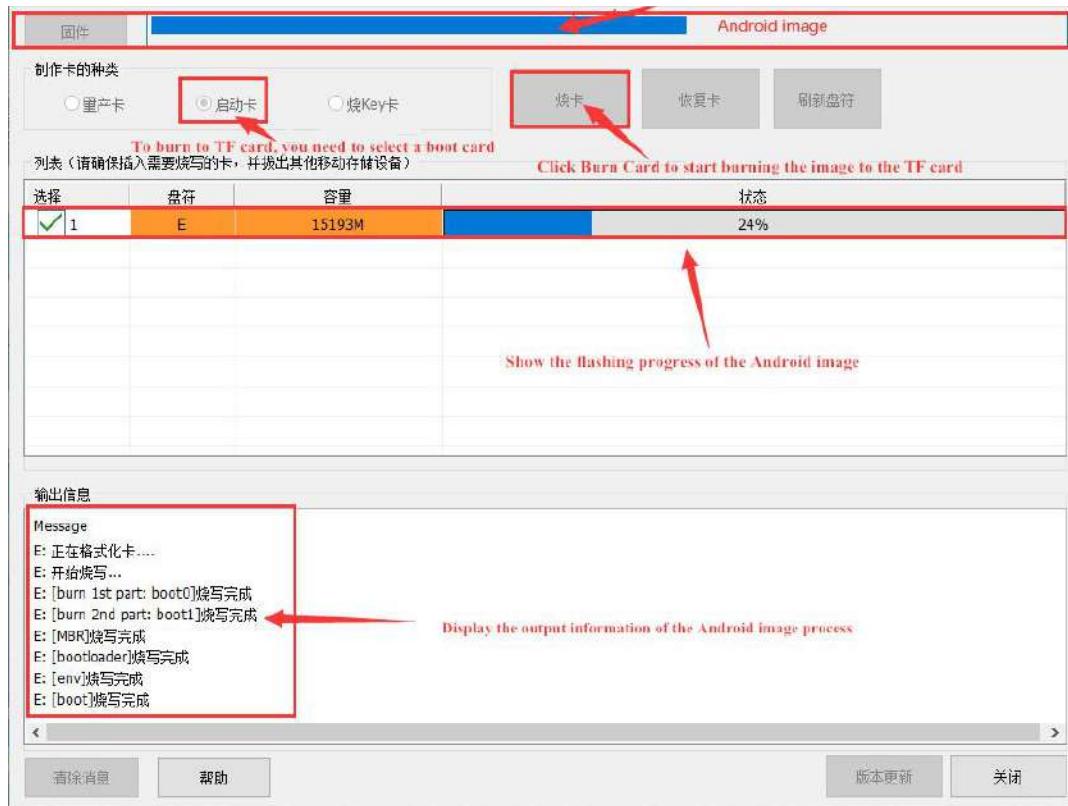
**the displayed drive letter is the same as the drive letter of the TF card you want to burn.** If There is no display, you can try to unplug the TF card



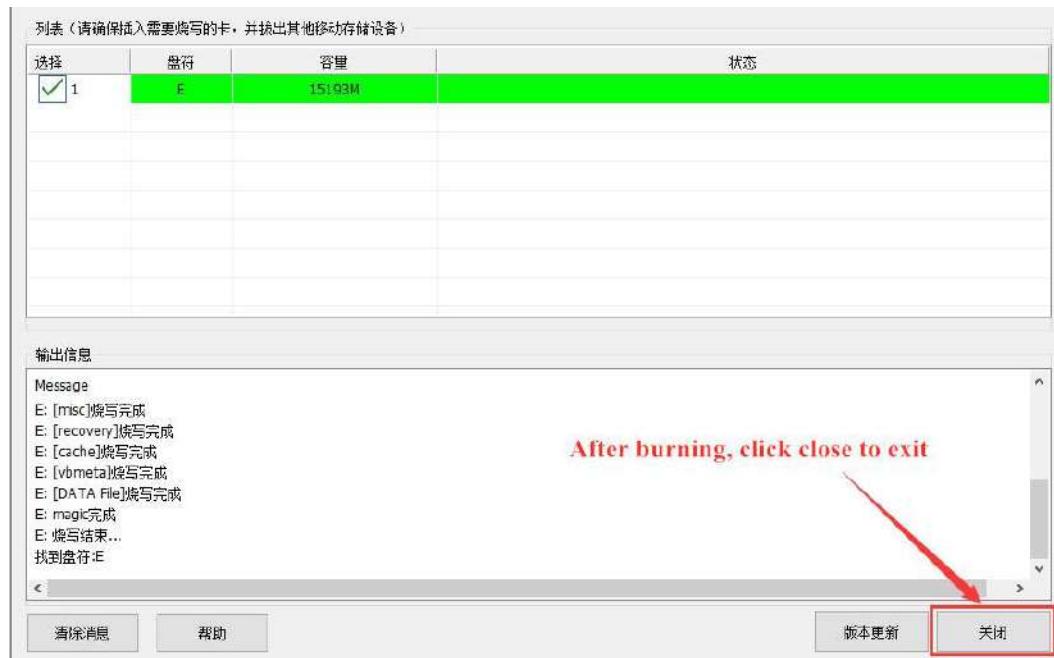
8) After confirming the drive letter, first format the TF card, click the **recovery card** button in PhoenixCard, or use the **SD Card Formatter** mentioned above to format the TF card



- 2) Then start writing Android firmware to TF card
  - a. First select the path of the Android firmware in the "Firmware" column
  - b. Select "**Mass Production Card**" in the "**Production**" card type
  - c. Then click the "**burn card**" button to start burning



3) After burning, the display of PhoenixCard is as shown in the figure below. At this time, click the **close button** to exit PhoenixCard



4) Then insert the TF card into the development board. After powering on the



development board, it will automatically burn the Android firmware in the TF card to the EMMC of the development board. During the burning process, the red light on the development board will keep flashing. If the development board is connected to an HDMI display, a progress bar for burning Android firmware to EMMC will be displayed on the HDMI display



5) After the burning is completed, the HDMI display is as shown in the figure below, and then the development board will automatically shut down



6) At this point, you can shut down, pull out the TF card, and then power on again, and the Android system in EMMC will be started.



## 2. 9. Start The Orange Pi Development Board

- 1) The development board has on-board eMMC, and the Android 9.0 image is burned in it by default. You can directly use the image in the eMMC for startup and full-function testing when you get the development board.
- 2) If you need to use a linux image, you can insert the TF card with the linux image burned into the TF card slot of the Orange Pi development board
- 3) The development board has an HDMI interface, and the development board can be connected to a TV or HDMI display through an HDMI to HDMI cable
- 4) Connect the USB mouse and keyboard to control the orange pi development board
- 5) The development board has an Ethernet port, which can be plugged into a network cable for Internet access
- 6) Connect a **5V/2A** or **5V/3A** high-quality Type C interface power adapter

**Remember not to insert a power adapter with a voltage output greater than 5V, it will burn out the development board.**

**Many unstable phenomena during system power-on and startup are basically caused by power supply problems, so a reliable power adapter is very important. If you find that it keeps restarting during the startup process, please replace the power supply or the Type C data cable and try again.**

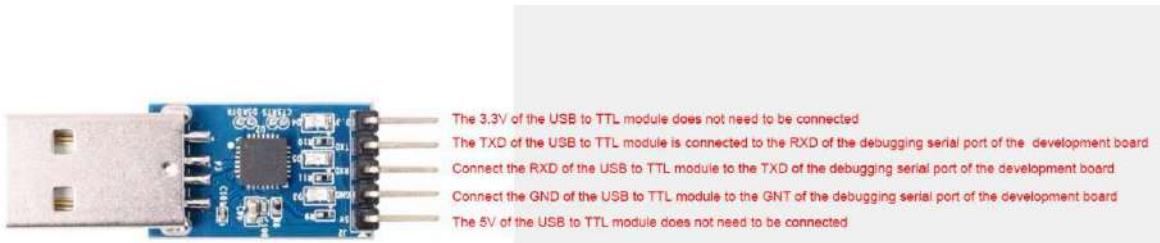
- 7) Then turn on the switch of the power adapter. If everything is normal, the HDMI display can see the startup screen of the system.
- 8) If you want to view the output information of the system through the debugging serial port, please use the USB to TTL module and DuPont cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section on how to **use the debugging serial port**.



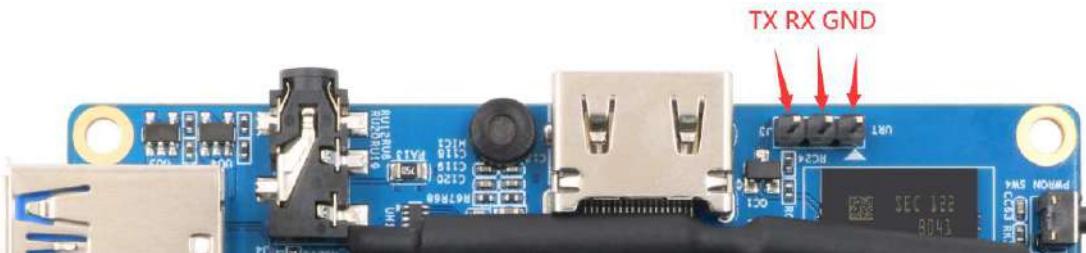
## 2. 10. How to use the debugging serial port

### 2. 10. 1. Connection instructions for debugging serial port

- 1) First, you need to prepare a **3.3v** USB to TTL module, and then insert one end of the USB interface of the USB to TTL module into the USB interface of the computer



- 2) The corresponding relationship between the debug serial port GND, TX and RX pins of the development board is shown in the figure below



- 3) The GND, TX and RX pins of the USB to TTL module need to be connected to the debug serial port of the development board through a DuPont cable

- a. Connect the GND of the USB to TTL module to the GND of the development board
- b. The RX of the USB to TTL module is connected to **the TX of the development board**
- c. The TX of the USB to TTL module is connected to **the RX of the development board**

- 4) The schematic diagram of connecting the USB to TTL module to the computer and the Orange Pi development board is shown below



Schematic diagram of connecting USB to TTL module to computer and Orange Pi development board

**The TX and RX of the serial port need to be cross-connected. If you don't want to distinguish the order of TX and RX carefully, you can connect the TX and RX of the serial port casually first. If the test serial port has no output, then exchange the order of TX and RX. There is an order**

## 2. 10. 2. How to use the debugging serial port on Ubuntu platform

There are many serial debugging software that can be used under Linux, such as **putty**, **minicom**, etc. The following demonstrates how to use **putty**

- 1) First, insert the USB to TTL module into the USB interface of the Ubuntu computer. If the USB to TTL module is connected normally, you can see the corresponding device node name under **/dev** of the Ubuntu PC. Remember this node name. It will be used when set the serial port software

```
test@test:~$ ls /dev/ttYS*
```

```
/dev/ttYS0
```

- 2) Then use the following command to install putty on Ubuntu PC

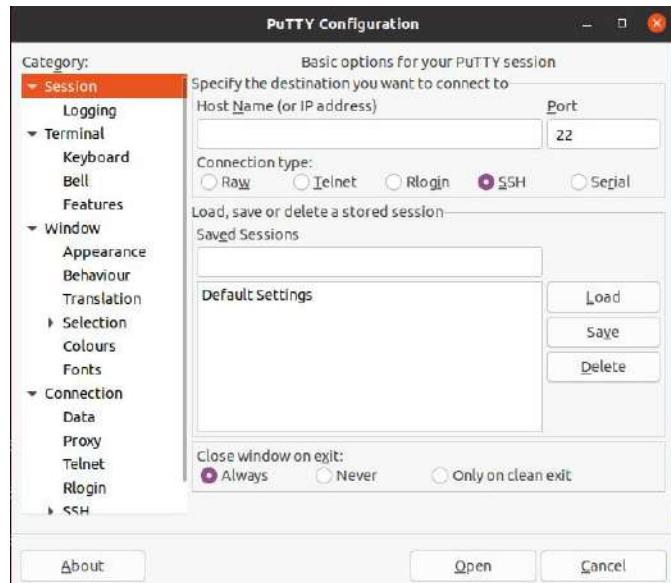
```
test@test:~$ sudo apt update
```

```
test@test:~$ sudo apt -y install putty
```

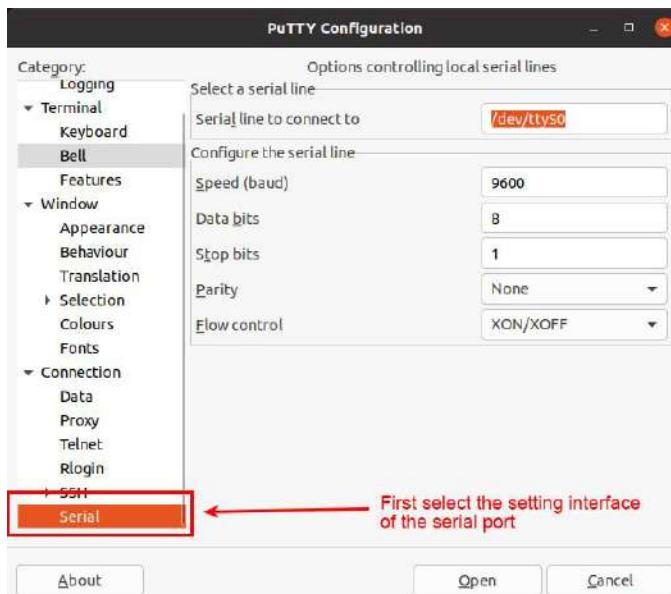
- 3) Then run putty, **remember to add sudo permissions**

```
test@test:~$ sudo putty
```

- 4) After executing the putty command, the following interface will pop up

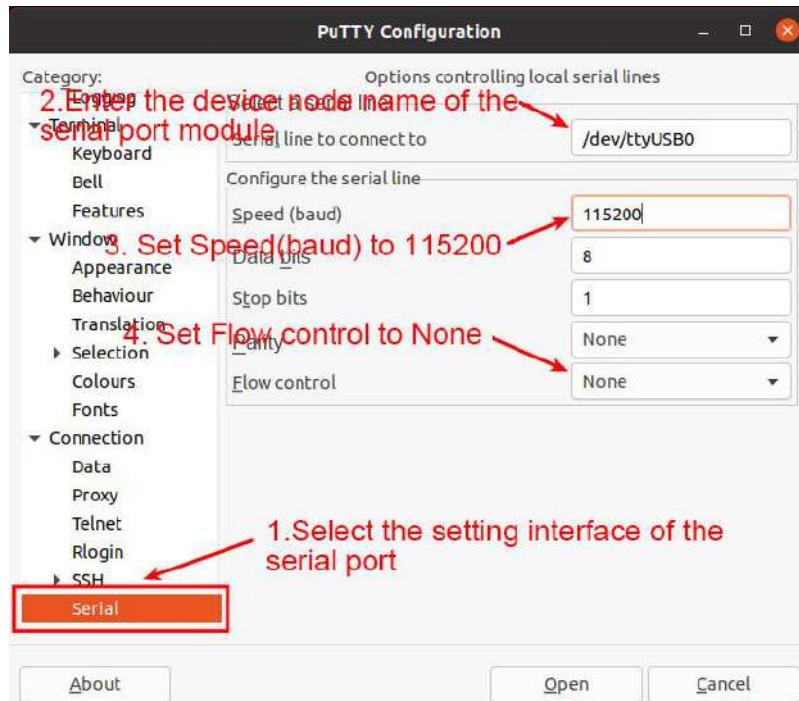


### 5) First select the setting interface of the serial port

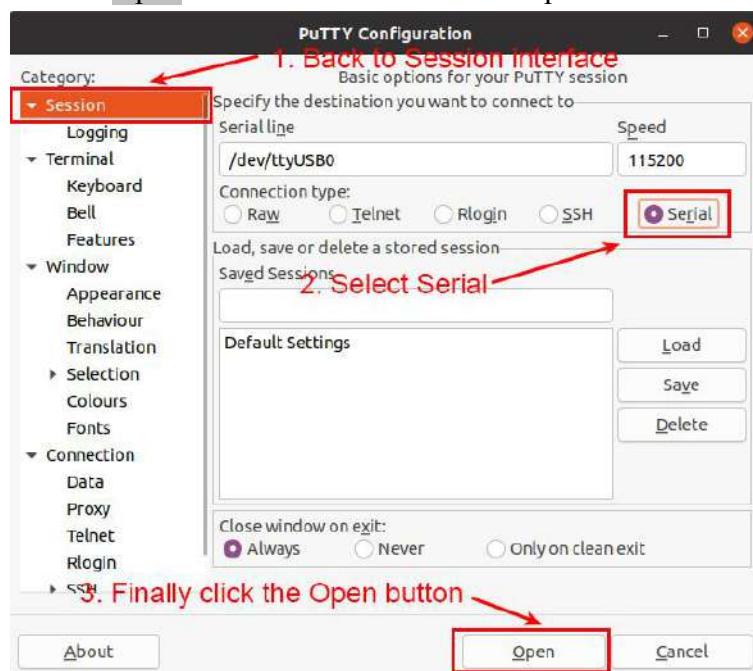


### 6) Then set the parameters of the serial port

- a. Set **Serial line to connect to** to **/dev/ttYS0** (modify to the corresponding node name, usually **/dev/ttYS0**)
- b. Set **Speed(baud)** to 115200 (the baud rate of the serial port)
- c. Set **Flow control** to **None**



- 7) After setting the serial port setting interface, go back to the Session interface
- Serial First select the Connection type as Serial
  - Then click the Open button to connect the serial port



- 8) After starting the development board, you can see the Log information output by the system from the open serial terminal

### 2. 10. 3. How to use the debugging serial port on Windows platform

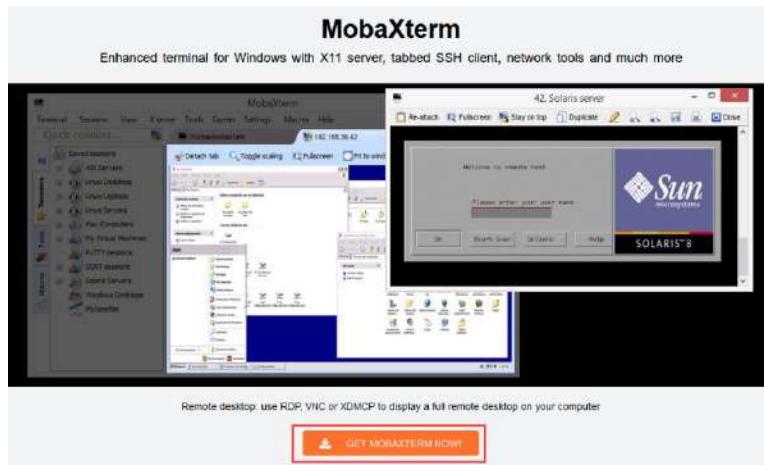
There are many serial port debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following shows how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

## 1) Download MobaXterm

a. Download MobaXterm URL as follows

<https://mobaxterm.mobatek.net/>

b. After entering the MobaXterm download page, click **GET XOBATERM NOW!**



c. Then choose to download the Home version



Home Edition	Professional Edition
<b>Free</b>	<b>\$69 / 49€ per user*</b>
Full X server and SSH support	* Excluding tax. Volume discounts <a href="#">available</a>
Remote desktop (RDP, VNC, Xdmcp)	
Remote terminal (SSH, telnet, rlogin, Mosh)	
X11-Forwarding	
Automatic SFTP browser	<b>Every feature from Home Edition +</b>
Master password protection	Customize your startup message and logo
Plugins support	Modify your profile script
Portable and installer versions	Remove unwanted games, screensaver or tools
Full documentation	Unlimited number of sessions
Max. 12 sessions	Unlimited number of tunnels and macros
Max. 2 SSH tunnels	Unlimited run time for network daemons
Max. 4 macros	Enhanced security settings
Max. 360 seconds for Tftp, Nfs and Cron	12-months updates included
	Deployment inside company
	Lifetime right to use

[Download now](#)    [Subscribe online / Get a quote](#)

- d. Then select the Portable portable version. After downloading, there is no need to install it, just open it and use it

MobaXterm Home Edition

Download MobaXterm Home Edition (current version):

[MobaXterm Home Edition v20.3 \(Portable edition\)](#) [MobaXterm Home Edition v20.3 \(installer edition\)](#)

Download previous stable version: [MobaXterm Portable v20.2](#) [MobaXterm installer v20.2](#)

You can also get early access to the latest features and improvements by downloading MobaXterm Preview version:

[MobaXterm Preview Version](#)

By downloading MobaXterm software, you accept [MobaXterm terms and conditions](#).

You can download MobaXterm and plugins sources [here](#).

If you use MobaXterm inside your company, you should consider subscribing to [MobaXterm Professional Edition](#); your subscription will give you access to professional support and to the "Customizer" software. This customizer will allow you to generate personalized versions of MobaXterm including your own logo, your default settings and your welcome message. Please [contact us](#) for more information.

- 2) After downloading, use the decompression software to decompress the downloaded compressed package, you can get the executable software of MobaXterm, and then double-click to open it

名称	修改日期	类型	大小
CygUtils.plugin	2020/5/21 4:06	PLUGIN 文件	15,570 KB
<b>MobaXterm_Personal_20.3</b>	2020/6/5 4:30	应用程序	14,104 KB

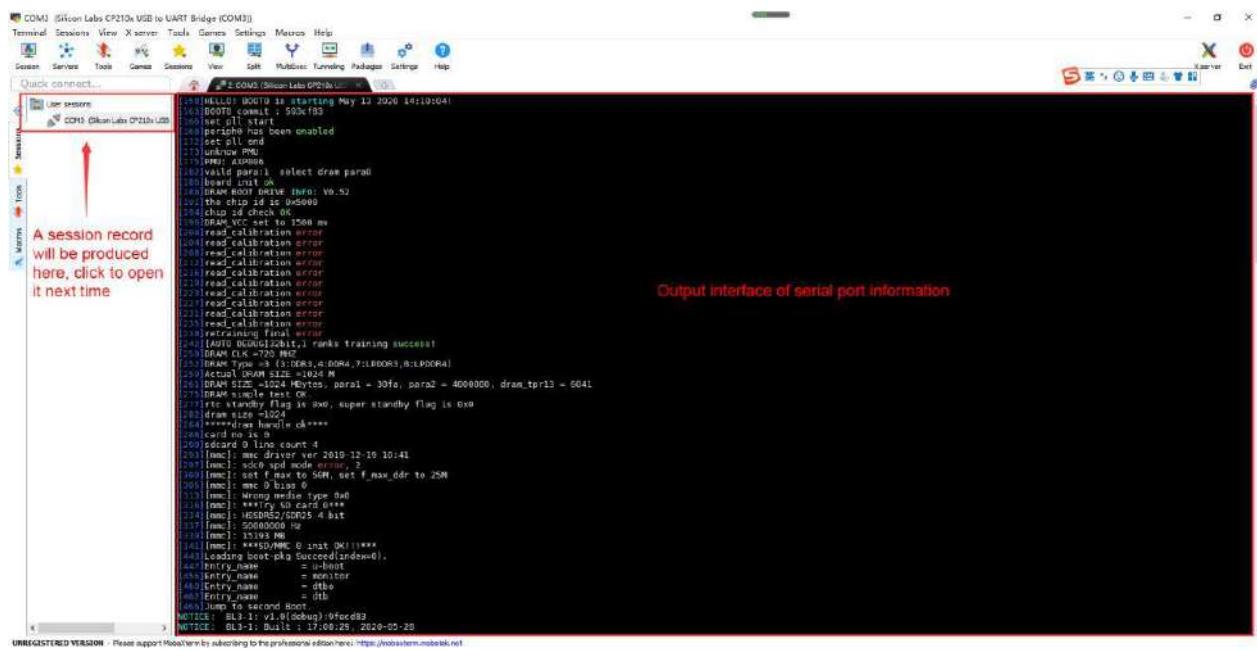
- 3) After opening the software, the steps to set the serial port connection are as follows
- Open the session settings interface
  - Select the serial port type
  - Select the port number of the serial port (select the corresponding port number according to the actual situation). If you cannot see the port number, please use the **360 driver master** to scan and install the driver for the USB to TTL serial port chip.



- d. Select the baud rate of the serial port to be 115200
- e. Finally click the "OK" button to complete the setting



- 4) After clicking the "OK" button, it will enter the following interface. At this time, you can see the output information of the serial port when you start the development board.





## 2. 11. Instructions for using the 5v pin in the 26pin interface of the development board for power supply

The power supply method we recommend for the development board is to use the power cord of the 5V/2A or 5V/3A Type C interface to be plugged into the Type C power interface of the development board to supply power. If you need to use the 5V pin in the 26pin interface to power the development board, please make sure that the power cable used can meet the power supply requirements of the development board. If there is unstable use, please switch back to the Type C power supply.

- 1) First, you need to prepare a power cord as shown in the figure below



The power cord shown in the picture above can be bought on Taobao, please search and buy by yourself

- 2) Use the 5V pin in the 26pin interface to supply power to the development board. The connection method of the power cable is as follows

- a. The USB A port of the power cord shown in the figure above needs to be plugged into the 5V/2A or 5V/3A power adapter connector (**it is not recommended to plug it into the USB port of the computer to supply power, it will be unstable or unable to start normally**)
- b. The red Dupont line needs to be plugged into the 5V pin of the 26pin interface of the development board
- c. The black Dupont wire needs to be plugged into the GND pin of the 26pin interface



- d. The positions of the 5V pin and GND pin of the 26pin interface in the development board are as shown in the figure below. **Remember not to connect them in reverse.**



### 3. Instructions for Debian and Ubuntu systems

Ubuntu images and Debian images are generally collectively referred to as Linux images (they use the Linux kernel), so when you see a Linux image or a Linux system in a manual, it refers to an image or system like Ubuntu or Debian.

Many people will have doubts about whether they can use pure Ubuntu or pure Debian systems (pure here can be understood as systems downloaded from Ubuntu or Debian official websites). The answer is no, because Ubuntu and Debian do not provide a system for the development board of the Orange Pi.

We can see from the official websites of Ubuntu and Debian that they both support the arm64 architecture (the SOC of the development board is the arm64 architecture), but please note that the support mentioned here only refers to the arm64 version of the software repository provided by Ubuntu or Debian (including tens of thousands of packages) or rootfs (which is what Orange Pi uses to make Ubuntu or Debian systems). To make an Ubuntu or Debian system that can be used for a certain development board, you need to transplant U-boot and Linux kernel and other things, and also need to fix the bugs encountered and optimize some functions. These are all done by Orange Pi.

Since Orange Pi only maintains Ubuntu, Debian and Manjaro systems, if these Linux distributions such as CentOS, Kali or OpenWRT are not ported by other developers or adapted by themselves, they cannot be used on the development board of Orange Pi (hardware running These systems are fine).

In addition, people often ask if systems from other development boards can be



**used on the Orange Pi development board. The answer is no, because different development boards use chips and circuit connections are generally different. A system developed for a certain development board cannot be used on other development boards.**

### 3. 1. Supported linux distribution types and kernel versions

release type	kernel version	Server Edition	desktop version
Ubuntu 18.04 - Bionic	linux4.9	support	support
Ubuntu 20.04 - Focal	linux4.9	support	support
Debian 10 - Buster	linux4.9	support	support
Ubuntu 20.04 - Focal	linux5.10	support	support
Debian 10 - Buster	linux5.10	support	support
Debian 11 - Bullseye	linux5.10	support	support
Ubuntu 20.04 - Focal	linux5.16	support	support
Ubuntu 22.04 - Jammy	linux5.16	support	support
Debian 11 - Bullseye	linux5.16	support	support

All the above images may not be compiled and put on Google network disk for everyone to download and use. Because if they are all released, the number will be too large. Many students who are new to Linux may have difficulty in choosing. In addition, Ubuntu and Debian generally use one of the versions. If the needs are met, there is no need for each different version. version to download and use.

If there are some special requirements (such requirements are basically very few), you must use a certain version of Ubuntu or Debian, but if there is no ready-made image to download, you can use the source code provided by Orange Pi to compile the desired image yourself. If you write If it is supported, then it is adapted in the code. Please refer to Chapter 5 and Chapter 6 for the compilation method of Linux image.

- 1) You can see the following download options on [the data download page of Orange Pi](#). Ubuntu image and Debian image are generally referred to as Linux image.



## Downloads



- Click the **Ubuntu image** download link to download Ubuntu related image. For example, after opening the Google network disk, you can see the following Ubuntu image (**the image version number may be updated**)

<input type="checkbox"/>		OrangePi3-lts_2.2.2_ubuntu_focal_server_linux5.10.75.7z	278.9M
<input type="checkbox"/>		OrangePi3-lts_2.2.2_ubuntu_focal_desktop_linux5.10.75.7z	715.8M
<input type="checkbox"/>		OrangePi3-lts_2.1.8_ubuntu_bionic_desktop_linux4.9.118.7z	653.2M

- Click the **Debian image** download link to download Debian-related image. For example, after opening the Google network disk, you can see the following Debian image (**the image version number may be updated**)

<input type="checkbox"/>		OrangePi3-lts_3.0.0_debian_bullseye_server_linux5.16.17.7z	383.6M
<input type="checkbox"/>		OrangePi3-lts_3.0.0_debian_bullseye_server_linux5.10.76_kodi_test.7z	546.7M
<input type="checkbox"/>		OrangePi3-lts_3.0.0_debian_bullseye_desktop_xfce_linux5.16.17.7z	973M
<input type="checkbox"/>		OrangePi3-lts_2.2.2_debian_buster_server_linux5.10.75.7z	357.6M
<input type="checkbox"/>		OrangePi3-lts_2.2.2_debian_buster_desktop_linux5.10.75.7z	890.9M
<input type="checkbox"/>		OrangePi3-lts_2.1.8_debian_buster_server_linux4.9.118.7z	335.1M
<input type="checkbox"/>		OrangePi3-lts_2.1.8_debian_buster_desktop_linux4.9.118.7z	832.9M

## 2) Naming rules for Linux images

**Development board model\_version number\_Linux distribution type\_distribution code\_server or desktop\_kernel version**

- The model of the development board:** all are **OrangePi3-lts**. The model names of different development boards are generally different. Before burning the



- image, please make sure that the model name of the selected image matches the development board.
- b. **Version number:** such as **2.x.x or 3.x.x**, this version number will increase with the update of the image function. In addition, the last number of the version number of the Linux image of the development board is an even number.
  - c. **Types of Linux distributions:** **Ubuntu** and **Debian** are currently supported. Since Ubuntu is derived from Debian, there is not much difference in use between the two systems. However, the default configuration of some software and the use of commands are slightly different. In addition, Ubuntu and Debian both have software repositories that are maintained and supported, and there are also slight differences in the supported and installable software packages. These require personal experience in order to have a deeper understanding. For more details, you can refer to the official documentation provided by Ubuntu and Debian.
  - d. **Distribution code:** used to distinguish different versions of a specific Linux distribution such as Ubuntu or Debian. Among them, **bionic** and **focal** are both Ubuntu distributions, bionic means Ubuntu18.04, focal means Ubuntu20.04, and jammy means Ubuntu22.04. The biggest difference between different versions is that many software in the software warehouse maintained by the new version of Ubuntu system are Newer than those in older Ubuntu systems, such as Python and GCC compilation toolchains. **buster** is the specific version code of Debian, buster means Debian10, **bullseye** means Debian11, and Debian11 is the latest stable version released by Debian.
  - e. **Server or desktop:** It is used to indicate whether the system has a desktop environment. If it is **server**, it means that the system does not have a desktop environment installed. The storage space and resources occupied by the image are relatively small, and the command line is mainly used to operate the control system. If it is **desktop**, it means that the XFCE4 desktop environment is installed by default in the system, and the storage space and resources occupied by the image are relatively large. You can connect the monitor, mouse and keyboard to operate the operating system through the interface. Of course, the desktop version of the system can also be operated through the command line like the server version of the system.
  - f. **Kernel version:** used to indicate the version number of the linux kernel, currently supports **linux4.9.118**, **linux5.10.75** and **linux5.16.17**.



### 3. 2. Linux kernel driver adaptation

Function	Linux4.9	Linux5.10	Linux5.16
HDMI video	OK	OK	OK
HDMI audio	OK	OK	OK
USB3.0	OK	OK	OK
USB2.0x2	OK	OK	OK
TF Card start	OK	OK	OK
Network card	OK	OK	OK
IR Receiver	OK	OK	OK
WIFI	OK	OK	OK
Bluetooth	OK	OK	OK
Headphone Audio	OK	OK	OK
MIC Recording	OK	OK	OK
USB Camera	OK	OK	OK
LED Lamp	OK	OK	OK
26pin GPIO	OK	OK	OK
I2C0	OK	OK	OK
SPI1	OK	OK	OK
UART3	OK	OK	OK
PWM	OK	OK	OK
Temperature Sensor	OK	OK	OK
Hardware watchdog	OK	OK	OK
Switch button	OK	OK	OK
Emmc start up	OK	OK	OK
Mali GPU	NO	NO	OK
TV-OUT	NO	NO	NO
Video codec	NO	NO	NO

### 3. 3. Description of the linux command format in this manual

- 1) All commands in this manual that need to be entered in the Linux system will be framed by the boxes below



As shown below, the content in the yellow box indicates the content that needs special attention, except for the commands in this.

## 2) Description of the prompt type in front of the command

- a. The prompt in front of the command refers to the content of the red part in the box below. This part of the content is not part of the linux command, so when entering a command in the linux system, please do not enter the content of the red font part.

```
orangeipi@orangeipi:~$ sudo apt update  
root@orangeipi:~# vim /boot/boot.cmd  
test@test:~$ ssh root@192.168.1.xxx  
root@test:~# ls
```

- b. **root@orangeipi:~\$** The prompt indicates that this command is entered in the **linux system of the development board**. The **\$** at the end of the prompt indicates that the current user of the system is an ordinary user. When executing a privileged command, **sudo** needs to be added.
- c. **root@orangeipi:~#** The prompt indicates that this command is entered in the **linux system of the development board**. The **#** at the end of the prompt indicates that the current user of the system is the root user, and can execute any command you want to execute.
- d. **test@test:~\$** The prompt indicates that this command was entered in the Ubuntu PC or Ubuntu virtual machine, not the linux system of the development board. The **\$** at the end of the prompt indicates that the current user of the system is an ordinary user. When executing a privileged command, you need to add **sudo**.
- e. **root@test:~#** The prompt indicates that the command was entered in the Ubuntu PC or Ubuntu virtual machine, not the linux system of the development board. The **#** at the end of the prompt indicates that the current user of the system is the root user and can execute any command you want to execute.

## 3) What are the commands that need to be entered?

- a. As shown below, **the black and bold part** is the command that needs to be input, and the content below the command is the output content (some commands have output, some may not), this part of the content does not need to be input



```
root@orangepi:~# cat /boot/orangepiEnv.txt
verbosity=7
bootlogo=false
console=serial
```

- b. As shown below, some commands that cannot be written in one line will be placed on the next line, as long as the black and bold parts are commands that need to be entered. When these commands are entered on a line, the "\\" at the end of each line needs to be removed, which is not part of the command. In addition, there are spaces in different parts of the command, please don't miss them

```
orangepi@orangepi:~$ echo \
"deb [arch=$(dpkg --print-architecture) \
signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
https://download.docker.com/linux/debian \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

### 3. 4. Linux system login instructions

#### 3. 4. 1. Linux system default login account and password

Account	Password
root	orangepi
orangepi	orangepi

Note that when entering the password, **the specific content of the entered password will not be displayed on the screen**, please do not think that there is any fault, just press Enter after entering it.

**When the input password is wrong, or there is a problem with the ssh connection, please note that as long as you use the Linux image provided by Orange Pi, please do not suspect that the above password is wrong, but find other reasons.**

#### 3. 4. 2. How to set the automatic login of the Linux system terminal

- 1) Method for root user to automatically log in to the terminal
  - a. First enter the following command to create a configuration file for terminal automatic login



```
root@orangepi:~# mkdir -p /etc/systemd/system/getty@.service.d/
root@orangepi:~# mkdir -p /etc/systemd/system/serial-getty@.service.d/
root@orangepi:~# cat <<-EOF >  \
/etc/systemd/system/serial-getty@.service.d/override.conf
[Service]
ExecStartPre=/bin/sh -c 'exec /bin/sleep 10'
ExecStart=
ExecStart=-/sbin/agetty --noissue --autologin root %I \$TERM
Type=idle
EOF
root@orangepi:~# cp /etc/systemd/system/serial-getty@.service.d/override.conf  \
/etc/systemd/system/getty@.service.d/override.conf
```

- b. Then restart the system to see that the terminal will automatically log in (no need to enter an account and password), the user used is **root**

- 2) Method for orangepi users to automatically log in to the terminal
    - a. First enter the following command to create a configuration file for terminal automatic login

```
root@orangepi:~# mkdir -p /etc/systemd/system/getty@.service.d/
root@orangepi:~# mkdir -p /etc/systemd/system/serial-getty@.service.d/
root@orangepi:~# cat <<-EOF >  \
/etc/systemd/system/serial-getty@.service.d/override.conf
[Service]
ExecStartPre=/bin/sh -c 'exec /bin/sleep 10'
```

**ExecStart=****ExecStart=-/sbin/agetty --noissue --autologin orangepi %I \\$TERM****Type=idle****EOF**

```
root@orangepi:~# cp /etc/systemd/system/serial-getty@.service.d/override.conf \
/etc/systemd/system/getty@.service.d/override.conf
```

- b. Then restart the system to see that the terminal will automatically log in (no need to enter the account and password), the user used is **orangepi**

Starting kernel ...

orangepi3-lts login: **orangepi (automatic login)**

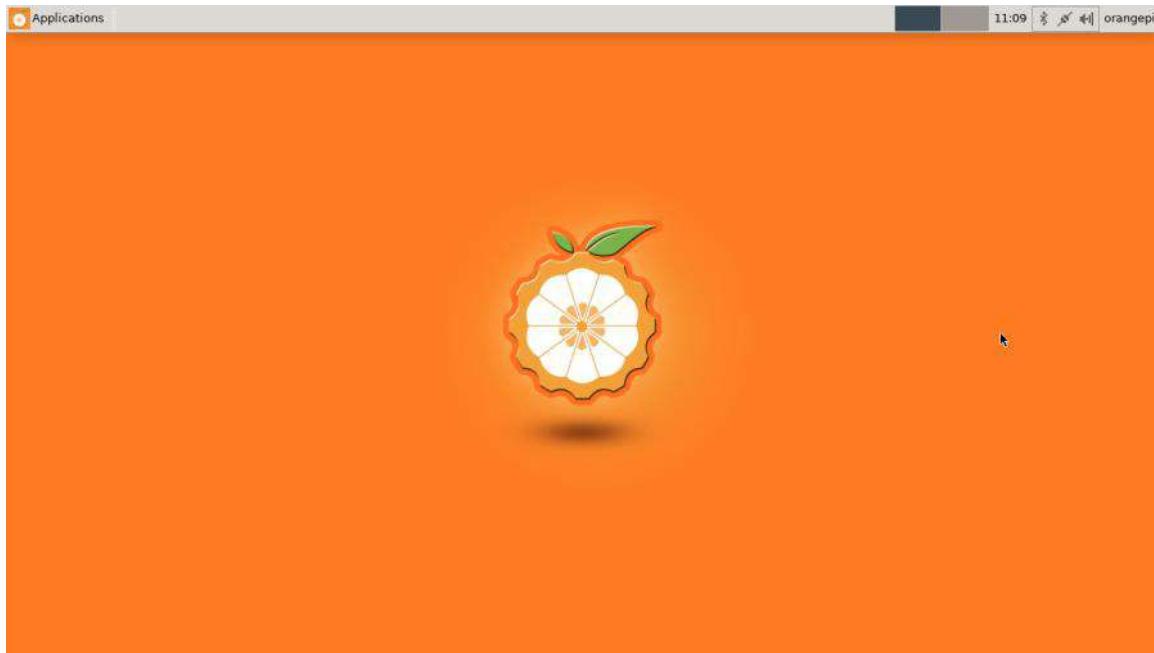


Welcome to Orange Pi 3.0.0 Bullseye with Linux 5.10.75-sun50iw6

System load:	31%	Up time:	0 min
Memory usage:	7% of 1.94G	IP:	192.168.1.21
CPU temp:	80°C	Usage of /:	11% of 15G

### 3.4.3. Instructions for automatic login of Linux desktop system

- 1) After the desktop version system is started by default, it will automatically use the orangepi user to log in to the desktop without entering a password



2) Modify the configuration in

**/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf** to prohibit the desktop version system from automatically logging in to the desktop. The modification command is as follows, or you can open the configuration file and modify it directly

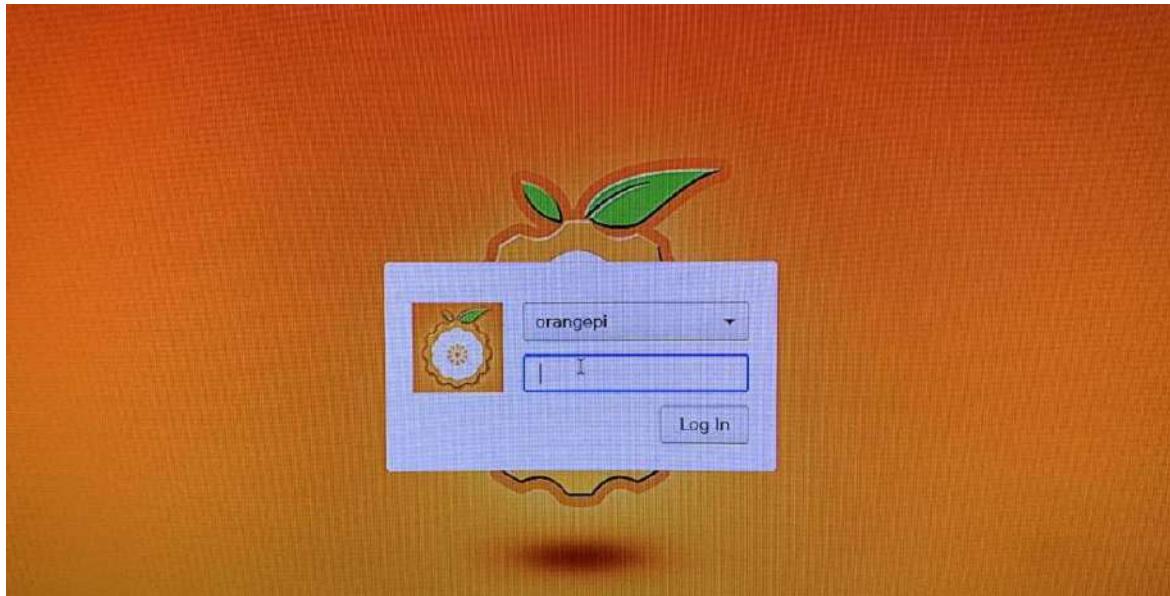
```
orangepi@orangepi:~$ sudo sed -i '\
"s/autologin-user=orangepi/#autologin-user=orangepi/"' \
/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
```

3) After modification, the configuration of

**/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf** is as follows

```
orangepi@orangepi:~$ cat /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
[Seat:*]
#autologin-user=orangepi
autologin-user-timeout=0
user-session=xfce
```

4) Then restart the system, the login dialog box will appear, at this time, you need to enter the **password** to enter the system



### 3. 4. 4. Setting method for automatic login of root user in Linux desktop system

- 1) First set the automatic login user as **root** in **22-orangepi-autologin.conf**

```
orangepi@orangepi:~$ sudo vim /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
[Seat:*)
autologin-user=root
autologin-user-timeout=0
user-session=xfce
```

- 2) Then modify the **lightdm-autologin** configuration file, change root to anything, and remove the root user restriction

**Note , Ubuntu22.04 does not need to set this step.**

```
orangepi@orangepi:~$ sudo vim /etc/pam.d/lightdm-autologin
# Allow access without authentication
#auth      required pam_succeed_if.so user != root quiet_success
auth      required pam_succeed_if.so user != anything quiet_success
```

- 3) Then restart the system, it will automatically log in to the Linux system desktop with the root user

**Note that if you use the root user to log in to the desktop system, you cannot use the pulseaudio in the upper right corner to manage audio devices.**



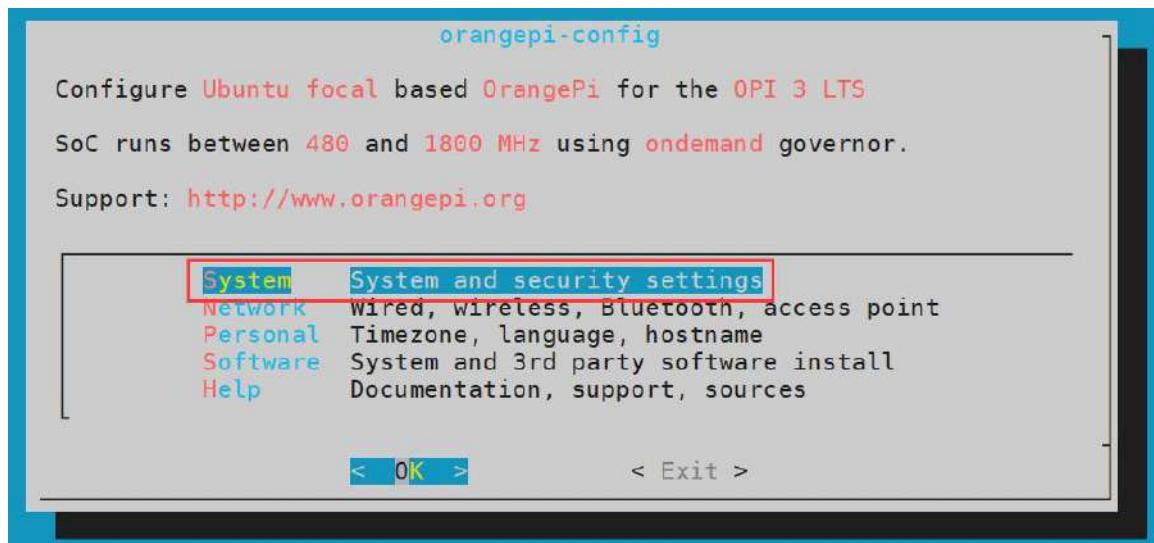
Also note that this is not a bug, since pulseaudio is not allowed to run under root.

### 3. 4. 5. How to disable the desktop in the Linux desktop system

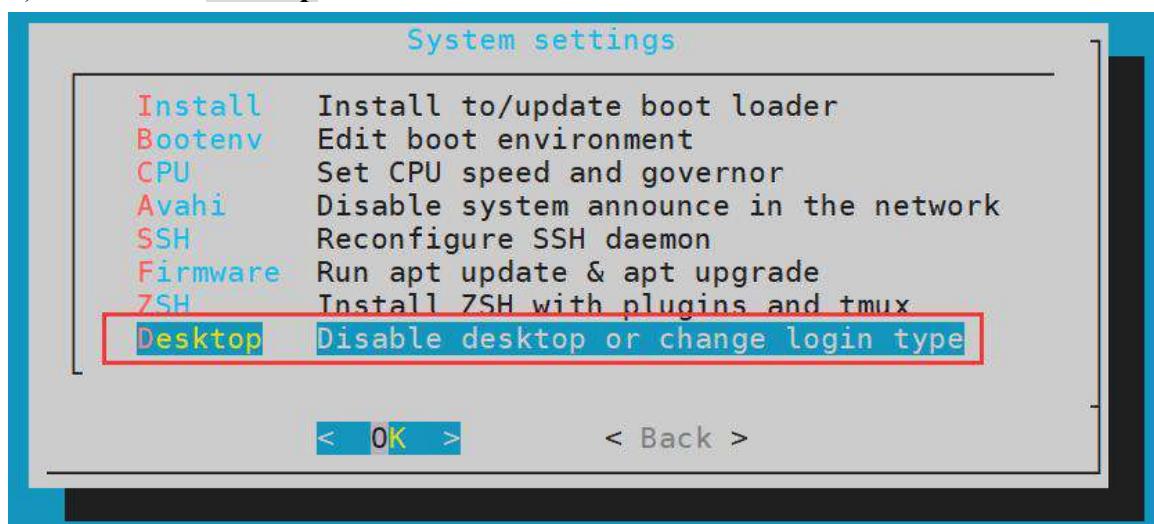
1) First enter the following command in the command line, **please remember to add sudo permissions**

```
orangeipi@orangeipi:~$ sudo orangepi-config
```

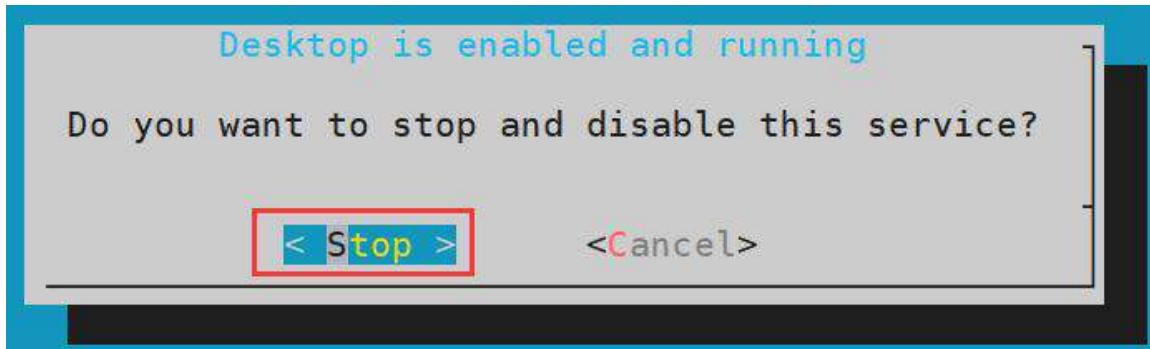
2) Then select **System**



3) Then select **Desktop**



4) Then select <Stop>



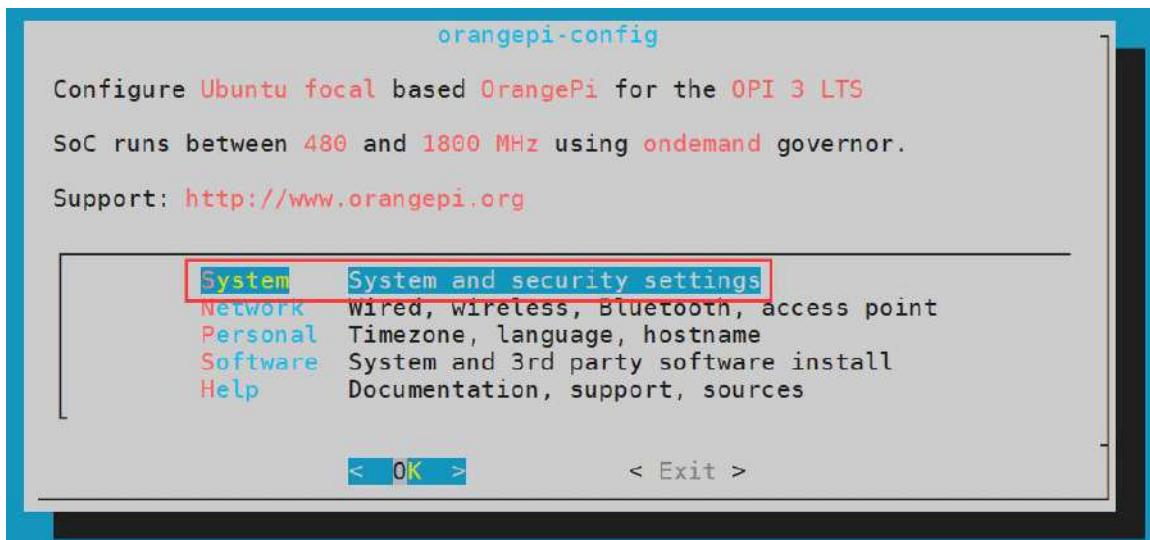
5) Then restart the Linux system and you will find that the desktop will not be displayed

6) The steps to reopen the desktop are as follows:

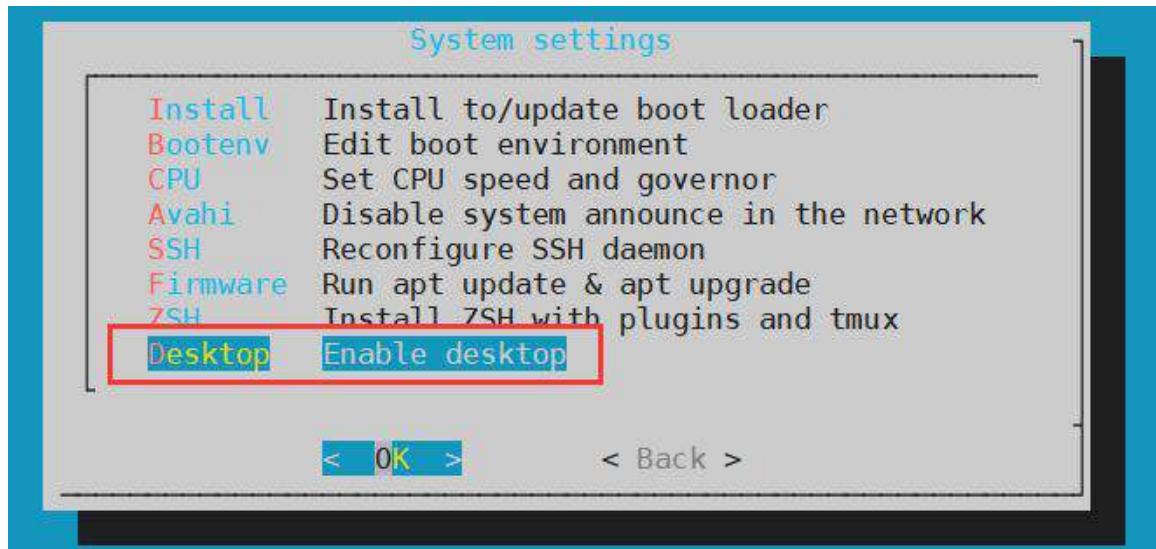
- First, enter the following command in the command line, **please remember to add sudo permissions**

```
orangeipi@orangeipi:~$ sudo orangeipi-config
```

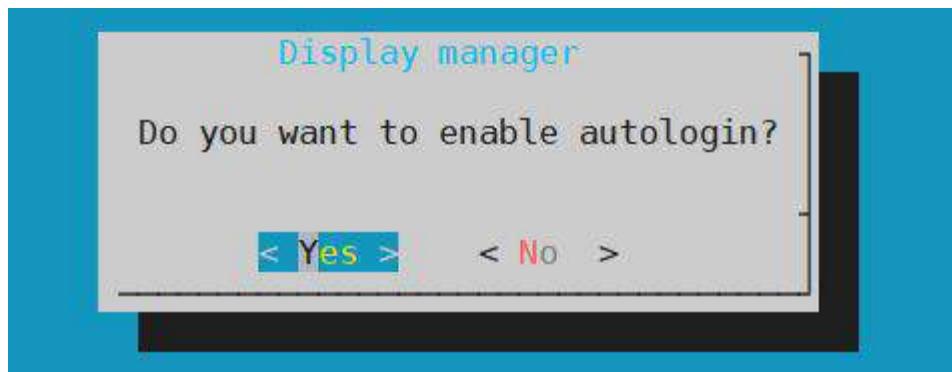
- Then select **System**



- Then select **Desktop    Enable desktop**



- d. Then choose whether you need to automatically log in to the desktop. If you choose <Yes>, you will automatically log in to the desktop. If you choose <No>, the user and password input interface will be displayed. You need to enter the password to enter the desktop.



- e. After selecting, the HDMI monitor will display the desktop

### 3. 5. On-board LED light test description

- 1) There are two LED lights on the development board, one yellow light and one red light. The default display of the LED lights when the system starts is as follows

	yellow light	red light
u-boot startup phase	OFF	ON
The kernel boots into the system	ON	OFF
GPIO port	PL7	PL4



2) The method of setting the yellow light on and off and flashing is as follows

a. First enter the setting directory of the yellow light

```
root@orangeipi:~# cd /sys/class/leds/orangeipi\:yellow\:status
```

b. The command to set the yellow light off is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:yellow:status# echo 0 > brightness
```

c. The command to set the yellow light to be always on is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:yellow:status# echo 1 > brightness
```

d. The command to set the yellow light to flash is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:yellow:status# echo heartbeat > trigger
```

e. The command to set the yellow light to stop flashing is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:yellow:status# echo none > trigger
```

3) The method of setting the red light on and off and flashing is as follows

a. First enter the setting directory of the red light

```
root@orangeipi:~# cd /sys/class/leds/orangeipi\:red\:status
```

b. The command to set the red light off is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:red:status# echo 0 > brightness
```

c. The command to set the red light to be always on is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:red:status# echo 1 > brightness
```

d. The command to set the red light to flash is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:red:status# echo heartbeat > trigger
```

e. The command to set the red light to stop flashing is as follows

```
root@orangeipi:/sys/class/leds/orangeipi:red:status# echo none > trigger
```

### 3. 6. Operation instructions for the capacity of the rootfs partition of the Linux system in the TF card

#### 3. 6. 1. The first boot will automatically expand the capacity of the rootfs partition in the TF card

1) After burning the Linux image of the development board to the TF card, you can view the usage of the TF card capacity on the **Ubuntu computer**. The steps are as follows:

**Note that if this step is not performed, it will not affect the automatic expansion of the Linux system of the development board. Here I just want to explain how to**

**check the capacity of the TF card after burning the Linux image on the TF card.**

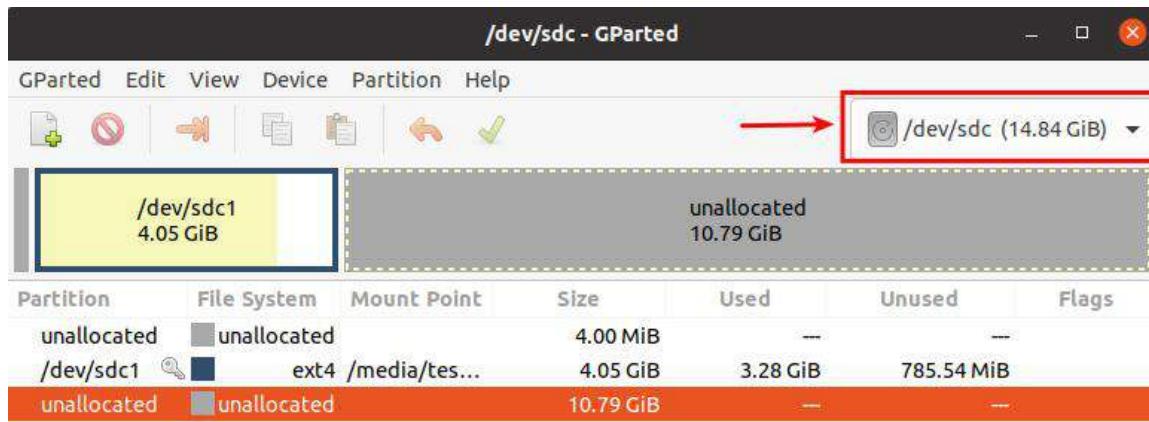
- First install the software gparted in the Ubuntu computer

```
test@test:~$ sudo apt install -y gparted
```

- Then open gparted

```
test@test:~$ sudo gparted
```

- After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity



- The above picture shows the situation of the TF card after burning the Linux desktop version system. It can be seen that although the total capacity of the TF card is 16GB (displayed as 14.84GiB in GParted), the rootfs partition (/dev/ sdc1) actually only allocated 4.05GiB, leaving 10.79GiB unallocated

- Then you can insert the TF card with the programmed Linux system into the development board to start. When the TF card starts the Linux system for the first time, it will call the **orangeipi-resize-filesystem** script through the systemd service

**orangeipi-resize-filesystem**.service to automatically perform The expansion of the rootfs partition, so **there is no need to manually expand**

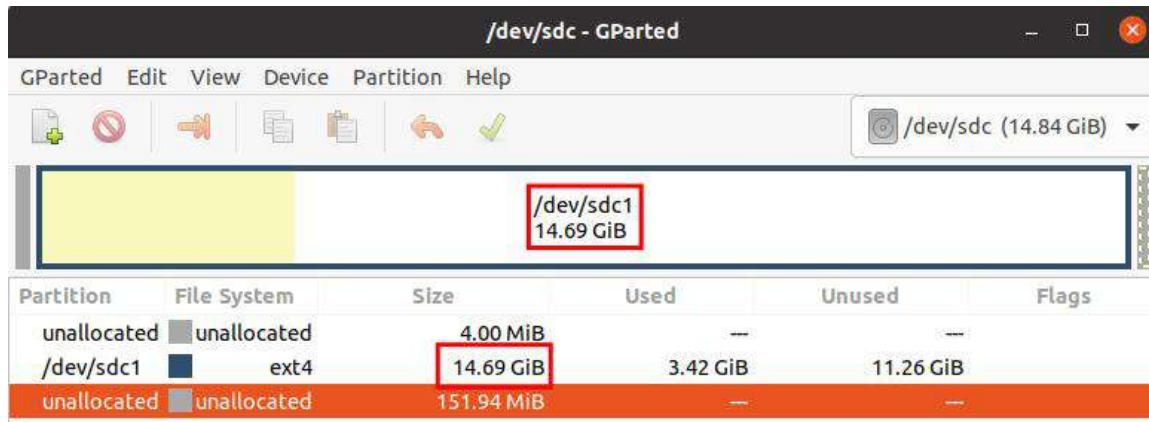
- After logging in to the system, you can use the **df -h** command to check the size of the rootfs. If it is consistent with the actual capacity of the TF card, it means that the automatic expansion is running correctly.

```
orangeipi@orangeipi:~$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
udev	430M	0	430M	0%	/dev
tmpfs	100M	5.6M	95M	6%	/run
<b>/dev/mmcblk0p1</b>	<b>15G</b>	<b>915M</b>	<b>14G</b>	<b>7%</b>	<b>/</b>
tmpfs	500M	0	500M	0%	/dev/shm



- 4) After booting the Linux system for the first time, we can also remove the TF card from the development board and reinsert it into the **Ubuntu computer**, and then use gparted again to check the status of the TF card, as shown in the figure below, the rootfs partition (/dev/ sdc1) has been expanded to 14.69GiB



**It should be noted that the Linux system has only one partition in ext4 format, and does not use a separate BOOT partition to store files such as kernel images, so there is no problem of expanding the BOOT partition.**

### 3. 6. 2. The method of prohibiting the automatic expansion of the rootfs partition capacity in the TF card

- 1) First burn the linux image of the development board into the TF card on the **Ubuntu computer** (not Windows), and **then re-plug the TF card**

- 2) Then the Ubuntu computer will generally automatically mount the partition of the TF card. If the automatic mounting is normal, you can use the ls command to see the following output. The partition name of the TF card is not necessarily the same as the name shown in the following command.

```
test@test:~$ ls /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/
bin  boot  dev  etc  home  lib  lost+found  media  mnt  opt  proc  root  run
sbin  selinux  srv  sys  tmp  usr  var
```

- 3) Then switch the current user to the root user in the Ubuntu computer

```
test@test:~$ sudo -i
[sudo] test password:
root@test:~#
```



- 4) Then enter the root directory of the Linux system in the TF card and create a new file named **.no\_rootfs\_resize**

```
root@test:~# cd /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db# cd root
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# touch .no_rootfs_resize
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# ls .no_rootfs*
.no_rootfs_resize
```

- 5) Then you can uninstall the TF card, then pull out the TF card and insert it into the development board to start. When the linux system starts, when the file **.no\_rootfs\_resize** is detected in the **/root** directory, the rootfs will no longer be automatically expanded.

- 6) After entering the Linux system after prohibiting rootfs automatic expansion, you can see that the total capacity of the rootfs partition is only 4GB (the image of the desktop version is tested here), which is much smaller than the actual capacity of the TF card, indicating that the automatic expansion of rootfs is prohibited successfully.

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            925M    0   925M   0% /dev
tmpfs           199M   3.2M  196M   2% /run
/dev/mmcblk0p1  4.0G  3.2G  686M  83% /
```

- 7) If you need to re-expand the capacity of the rootfs partition in the TF card, you only need to execute the following command, and then restart the Linux system of the development board.

**Note, please execute the following commands under the **root** user.**

```
root@orangepi:~# rm /root/.no_rootfs_resize
root@orangepi:~# systemctl enable orangepi-resize-filesystem.service
root@orangepi:~# reboot
```

After restarting, enter the Linux system of the development board again, and you can see that the rootfs partition has been expanded to the actual capacity of the TF card.

```
root@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            925M    0   925M   0% /dev
```



```
tmpfs          199M  3.2M  196M   2% /run
/dev/mmcblk0p1  15G  3.2G  12G  23% /
```

### 3. 6. 3. How to manually expand the capacity of the rootfs partition in the TF card

If the total capacity of the TF card is very large, such as 128GB, you do not want the rootfs partition of the Linux system to use all the capacity of the TF card, but only want to allocate a part of the capacity, such as 16GB, for the Linux system, and then the remaining capacity of the TF card can be used for other use. Then you can use the content described in this section to manually expand the capacity of the rootfs partition in TF.

- 1) First burn the linux image of the development board into the TF card on the **Ubuntu computer** (not Windows), and **then re-plug the TF card**
- 2) Then the Ubuntu computer will generally automatically mount the partition of the TF card. If the automatic mounting is normal, you can use the **ls** command to see the following output. The partition name of the TF card is not necessarily the same as the name shown in the following command.

```
test@test:~$ ls /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/
bin  boot  dev  etc  home  lib  lost+found  media  mnt  opt  proc  root  run
sbin  selinux  srv  sys  tmp  usr  var
```

- 3) Then switch the current user to the root user in the Ubuntu computer

```
test@test:~$ sudo -i
[sudo] test password:
root@test:~#
```

- 4) Then enter the root directory of the Linux system in the TF card and create a new file named **.no\_rootfs\_resize**

```
root@test:~# cd /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db# cd root
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# touch .no_rootfs_resize
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# ls .no_rootfs*
.no_rootfs_resize
```



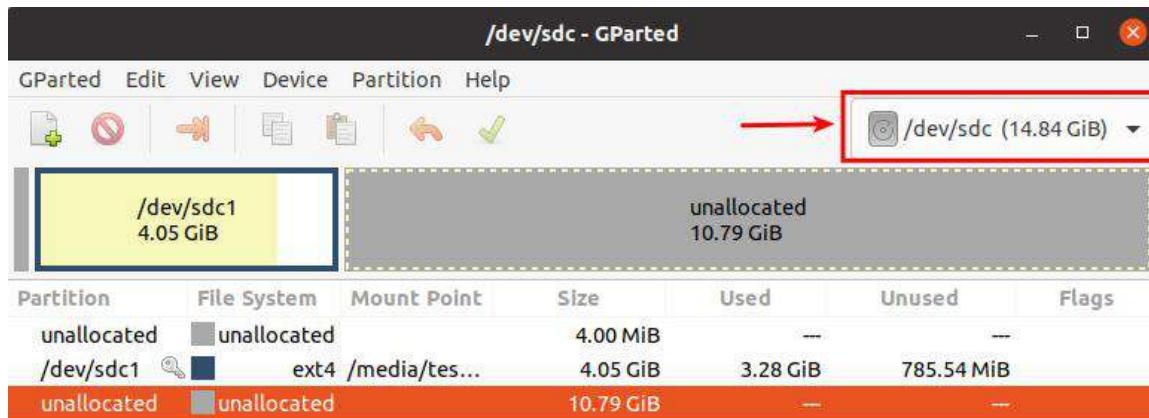
5) Then install the software gparted on the Ubuntu computer

```
test@test:~$ sudo apt install -y gparted
```

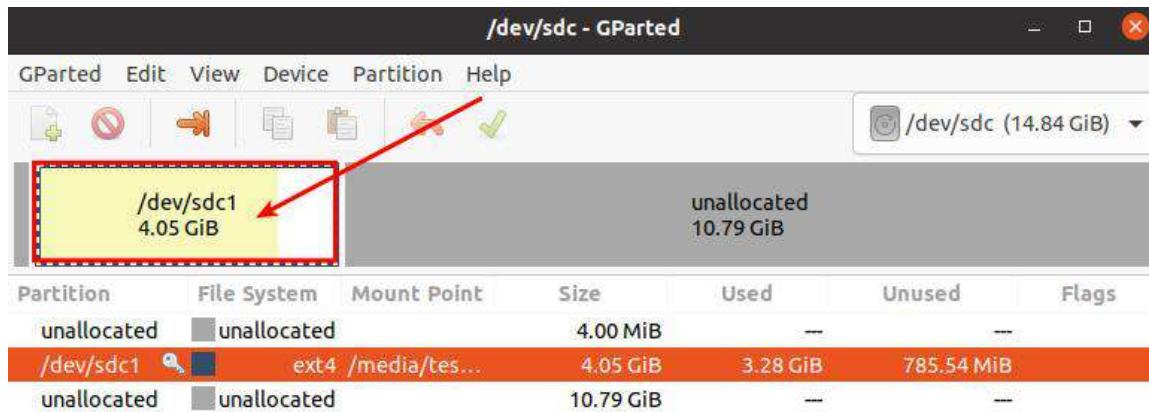
6) Then open gparted

```
test@test:~$ sudo gparted
```

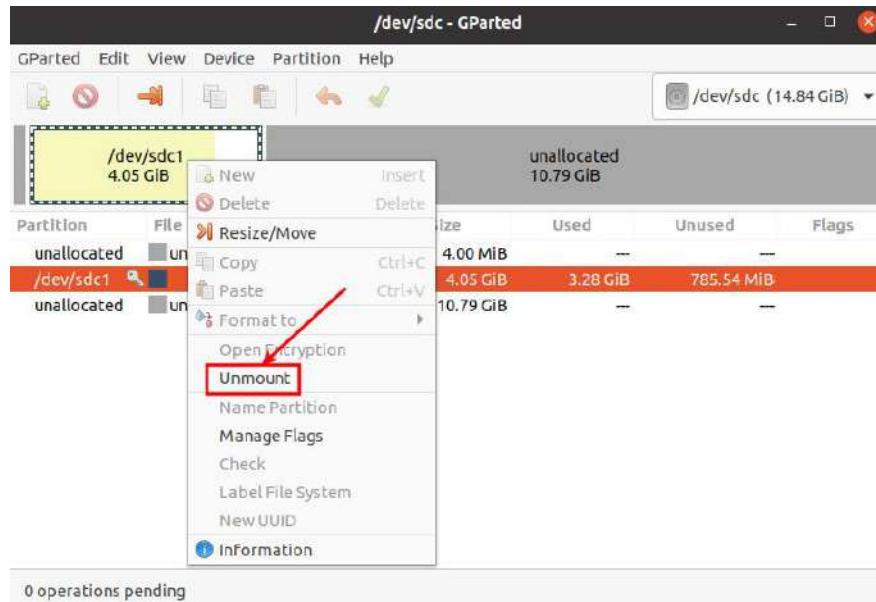
7) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity. The figure below shows the TF card after burning the Linux desktop version system. It can be seen that although the total capacity of the TF card is 16GB (displayed as 14.84GiB in GParted), the rootfs partition (/dev/sdc1) Only 4.05GiB are actually allocated, leaving 10.79GiB unallocated



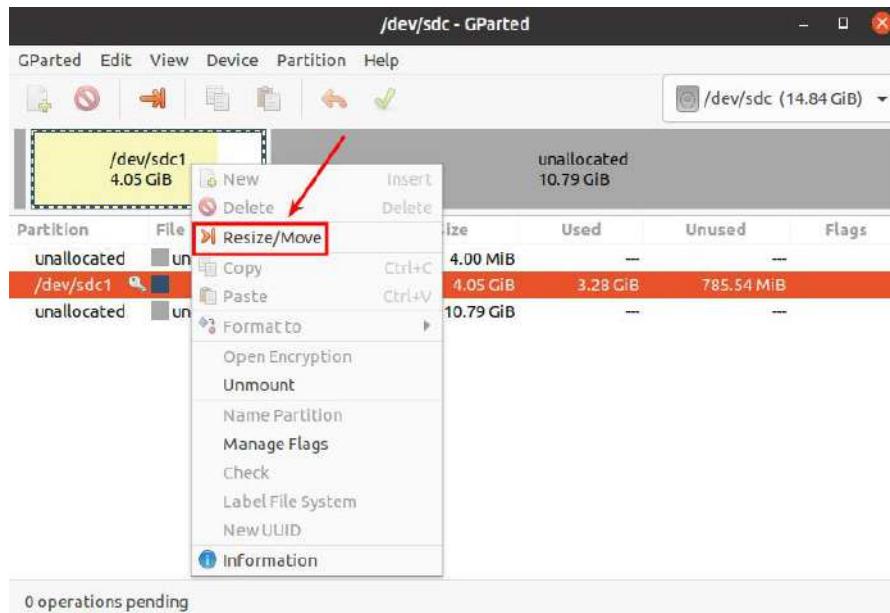
8) Then select the rootfs partition (/dev/sdc1)



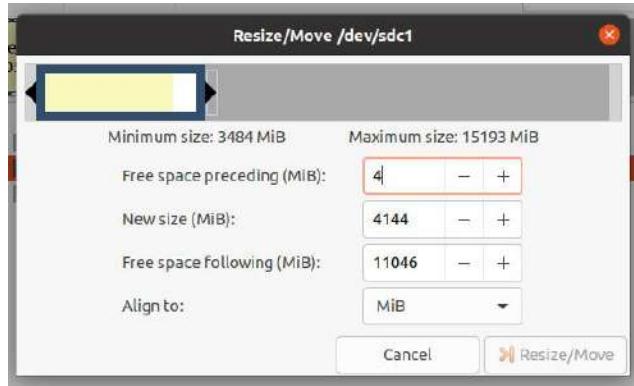
9) Click the right mouse button again to see the operation options shown in the figure below. If the TF card has been mounted, you need to Umount the rootfs partition of the TF card first.



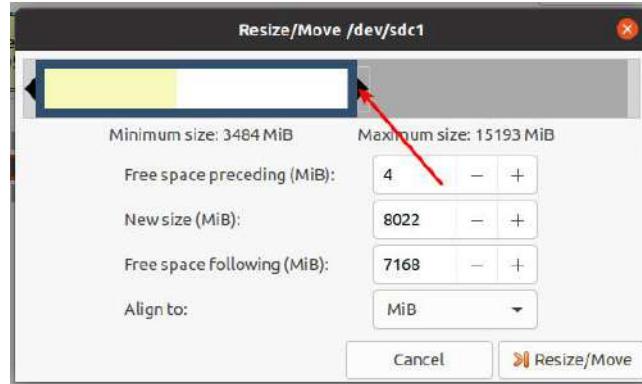
- 10) Then select the rootfs partition again, right-click, and select **Resize/Move** to start expanding the size of the rootfs partition



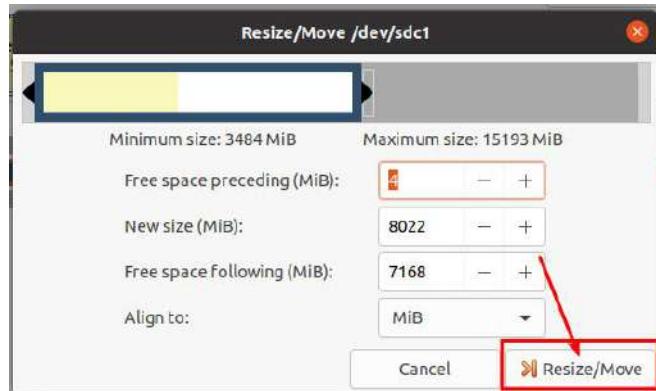
- 11) After the **Resize/Move** option is turned on, the following setting interface will pop up



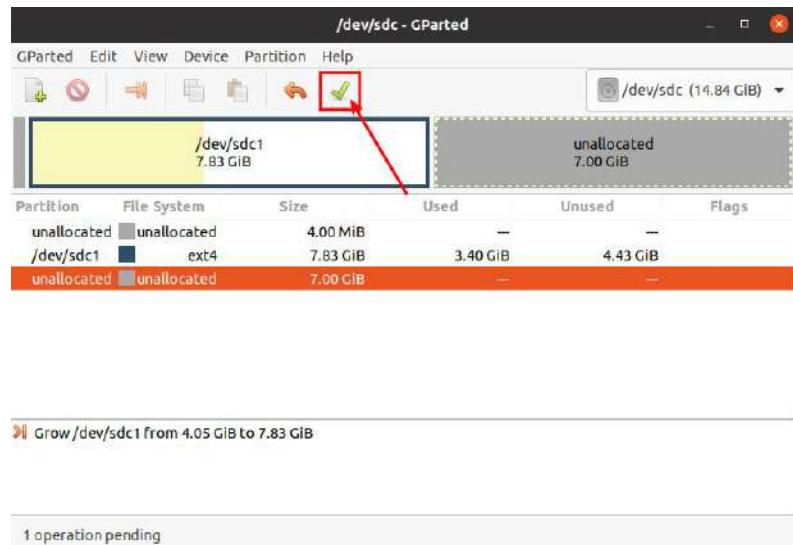
- 12) Then you can directly drag the position shown in the figure below to set the size of the capacity, or you can set the size of the rootfs partition by setting the number in **New size (MiB)**



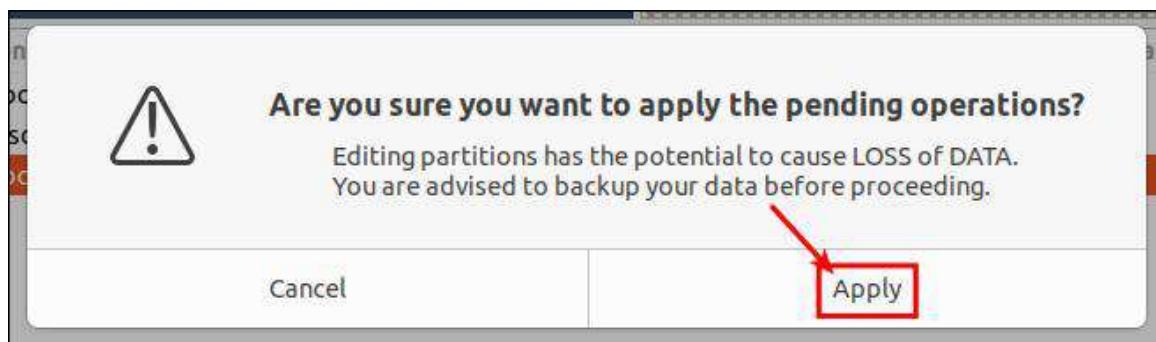
- 13) After setting the capacity, click **Resize/Move** in the lower right corner



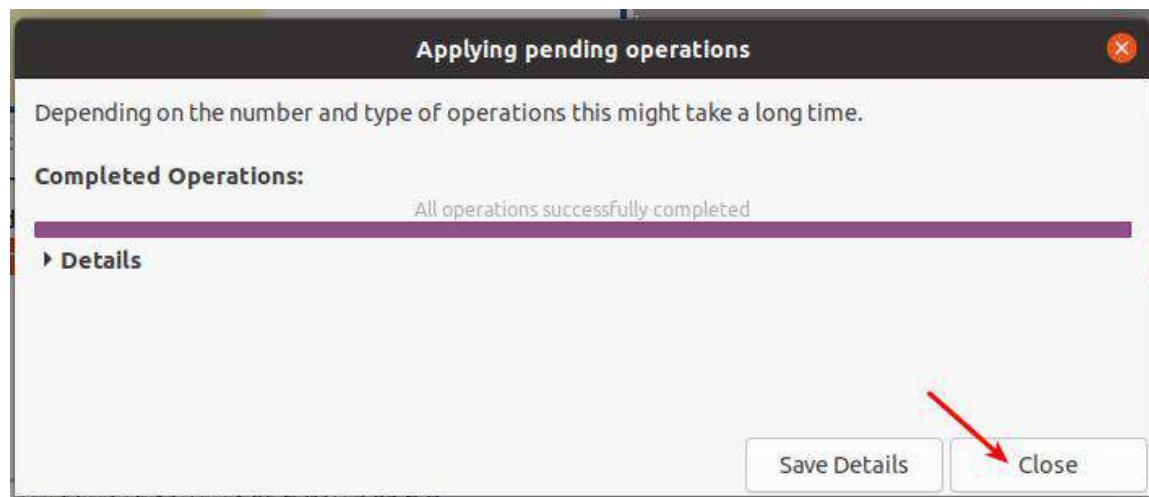
- 14) After the final confirmation, click the green ✓ as shown in the figure below



- 15) Then select **Apply**, it will officially start to expand the capacity of the rootfs partition



- 16) After the expansion is complete, click **Close** to close it





17) Then you can unplug the TF card, and then insert it into the development board to start. After entering the Linux system of the development board, if you use the **df -h** command to see that the size of the rootfs partition is the same as the size set earlier, it means manual operation. Expansion succeeded

```
root@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            925M    0   925M   0% /dev
tmpfs           199M  3.2M  196M   2% /run
/dev/mmcblk0p1  7.7G  3.2G  4.4G  42% /
```

### 3. 6. 4. Method to reduce the capacity of rootfs partition in TF card

After configuring the application program or other development environment in the Linux system of the TF card, if you want to backup the Linux system in the TF card, you can use the method in this section to reduce the size of the rootfs partition first, and then start the backup.

1) First insert the TF card you want to operate in the **Ubuntu computer** (not Windows)

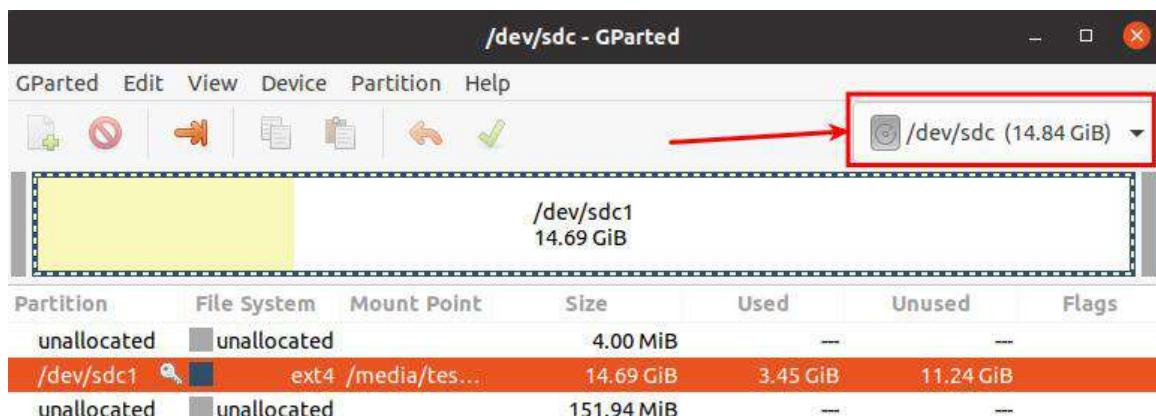
2) Then install the software gparted on the Ubuntu computer

```
test@test:~$ sudo apt install -y gparted
```

3) Then open gparted

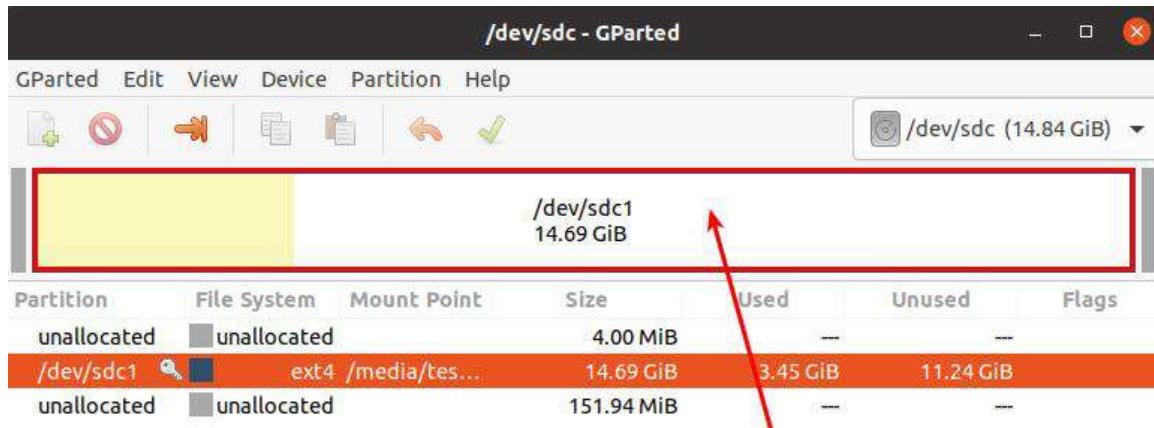
```
test@test:~$ sudo gparted
```

4) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity

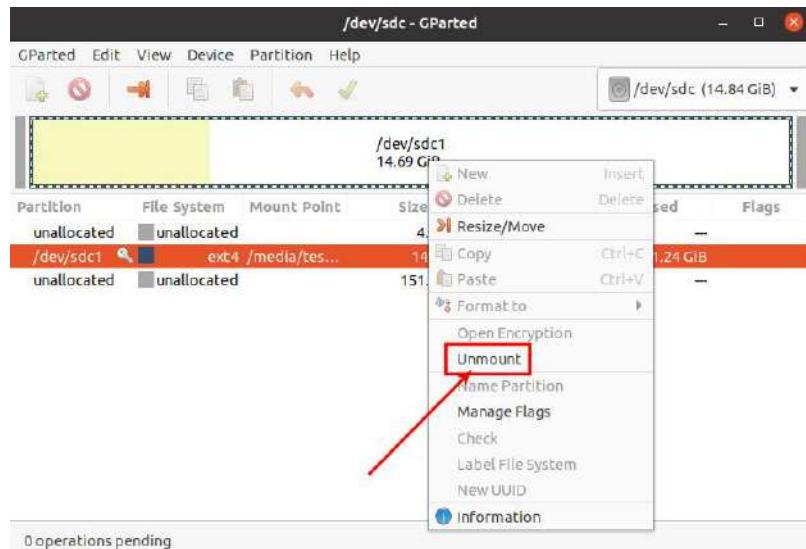




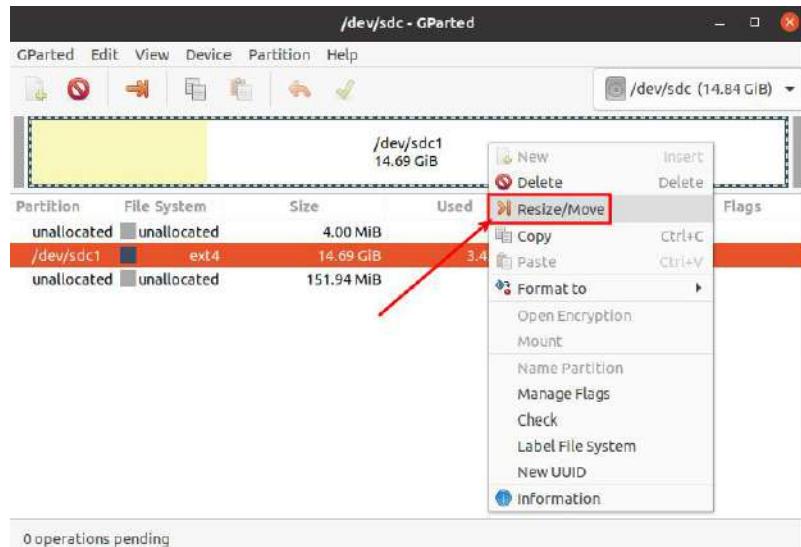
5) Then select the rootfs partition (/dev/sdc1)



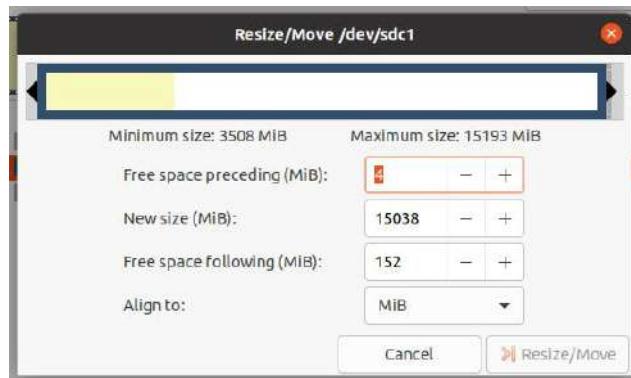
6) Then click the right mouse button to see the operation options shown in the figure below. If the TF card has been mounted, you need to Unmount the rootfs partition of the TF card first.



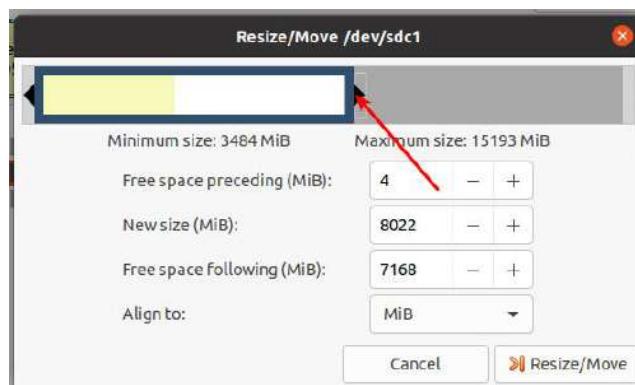
7) Then select the rootfs partition again, right-click, and select **Resize/Move** to start setting the size of the rootfs partition



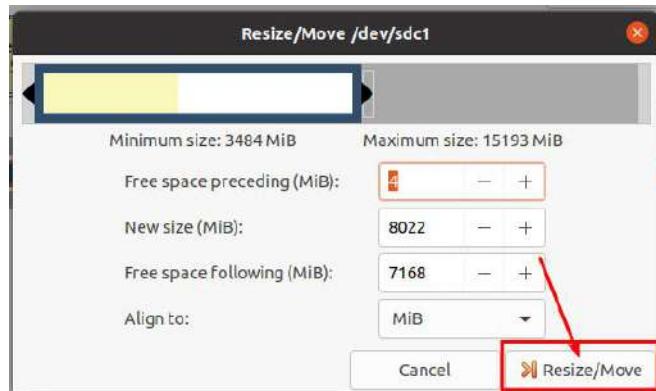
8) After the **Resize/Move** option is turned on, the following setting interface will pop up



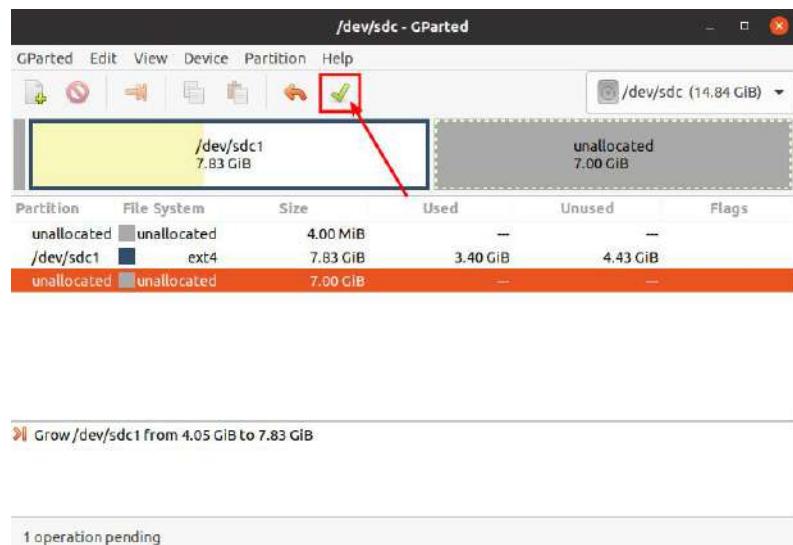
9) Then you can directly drag the position shown in the figure below to set the size of the capacity, or you can set the size of the rootfs partition by setting the number in **New size (MiB)**



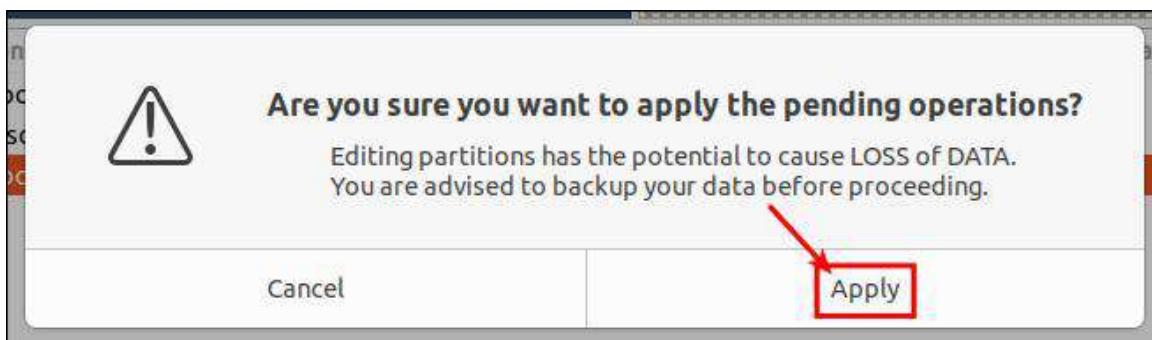
18) After setting the capacity, click **Resize/Move** in the lower right corner



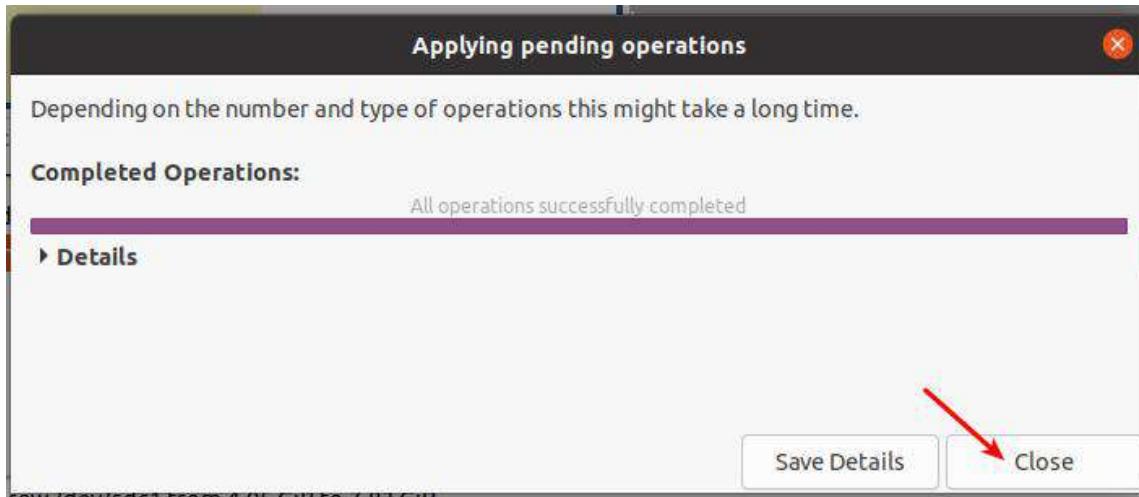
- 19) After the final confirmation, click the green ✓ as shown in the figure below



- 20) Then select **Apply**, it will officially start to expand the capacity of the rootfs partition



- 21) After the expansion is completed, click **Close** to close it



22) Then you can unplug the TF card and insert it into the development board to start it. After entering the Linux system of the development board, if you use the **df -h** command, you can see that the size of the rootfs partition is the same as the size set earlier, it means shrinking capacity success

```
root@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            925M    0  925M   0% /dev
tmpfs           199M  3.2M  196M   2% /run
/dev/mmcblk0p1  7.7G  3.2G  4.4G  42% /
```

### 3. 7. How to modify the linux log level (loglevel)

1) The loglevel of the linux system is set to 1 by default. When using the serial port to view the startup information, the kernel output log is as follows, basically all shielded

```
Starting kernel ...
```

```
Uncompressing Linux... done, booting the kernel.
```

```
Orange Pi 2.1.8 Focal ttyS0
```

```
orangepi login:
```

2) When there is a problem with the startup of the Linux system, you can use the



following method to modify the value of loglevel, so as to print more log information to the serial port display, which is convenient for debugging. If the Linux system fails to start and cannot enter the system, you can insert the TF card into the Ubuntu PC through the card reader, and then directly modify the configuration of the Linux system in the TF card after mounting the TF card in the Ubuntu PC. Insert the TF card into the development board to start

```
orangepi@orangepi:~$ sudo sed -i "s/verbosity=1/verbosity=7/" /boot/orangepiEnv.txt  
orangepi@orangepi:~$ sudo sed -i "s/console=both/console=serial/" /boot/orangepiEnv.txt
```

- 3) The above commands actually set the variables in **/boot/orangepiEnv.txt**. After setting, you can open **/boot/orangepiEnv.txt** to check.

```
orangepi@orangepi:~$ cat /boot/orangepiEnv.txt  
verbosity=7  
bootlogo=false  
console=serial
```

- 4) Then restart the development board, the output information of the kernel will be printed to the serial port output

## 3. 8. Network connection test

### 3. 8. 1. Ethernet port test

- 1) First, insert the network cable into the Ethernet interface of the development board, and ensure that the network is unblocked
- 2) After the system starts, it will automatically assign an IP address to the Ethernet card through DHCP, **no other configuration is required**
- 3) The command to view the IP address is as follows

```
orangepi@orangepi:~$ ip addr show eth0  
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000  
    link/ether 5e:ac:14:a5:93:b3 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.1.16/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0  
        valid_lft 259174sec preferred_lft 259174sec
```



```
inet6 240e:3b7:3240:c3a0:e269:8305:dc08:135e/64  scope  global  dynamic
      noprefixroute
          valid_lft 259176sec preferred_lft 172776sec
inet6 fe80::957d:bbbd:4928:3604/64 scope link  noprefixroute
          valid_lft forever preferred_lft forever
```

**There are three ways to check the IP address after the development board is started:**

- 1. Connect the HDMI display, then log in to the system and use the `ip addr show eth0` command to view the IP address**
- 2. Enter the `ip addr show eth0` command in the debug serial terminal to view the IP address**
- 3. If there is no debugging serial port and no HDMI display, you can also view the IP address of the network port of the development board through the management interface of the router. However, this method often fails to see the IP address of the development board. If you can't see it, the debug method looks like this:**
  - A) First check whether the Linux system has started normally. If the green light of the development board is on, it is generally started normally. If only the red light is on, or the red and green lights are not on, the system has not started normally;**
  - B) Check whether the network cable is plugged in tightly, or try another network cable;**
  - C) Try another router (the router has encountered many problems, such as the router cannot assign an IP address normally, or the IP address has been assigned normally but cannot be seen in the router);**
  - D) If there is no router to replace, you can only connect the HDMI display or use the debug serial port to view the IP address.**

**In addition, it should be noted that the development board DHCP automatically assigns an IP address without any settings.**

- 4) The command to test the network connectivity is as follows, the ping command can be interrupted by the **Ctrl+C** shortcut key

```
orangepi@orangepi:~$ ping www.baidu.com -I eth0
```

```
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
```



```
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=56 time=6.74 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=56 time=6.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=56 time=6.26 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=56 time=7.27 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 6.260/6.770/7.275/0.373 ms
```

### 3.8.2. WIFI connection test

Please do not connect to WIFI by modifying the /etc/network/interfaces configuration file. There will be problems with connecting to the WIFI network in this way.

#### 3.8.2.1. The server version image is connected to WIFI through the command

When the development board is not connected to the Ethernet, not connected to the HDMI display, and only connected to the serial port, it is recommended to use the commands demonstrated in this section to connect to the WIFI network. Because nmtui can only display characters in some serial port software (such as minicom), it cannot display the graphical interface normally. Of course, if the development board is connected to an Ethernet or HDMI display, you can also use the commands demonstrated in this section to connect to the WIFI network

- 1) First log in to the linux system, there are the following three ways
  - a. If the development board is connected to the network cable, you can [log in to the Linux system remotely through ssh](#)
    - a. If the development board is connected to the debugging serial port, you can use the serial port terminal to log in to the linux system
    - b. If the development board is connected to the HDMI display, you can log in to the linux system through the HDMI display terminal

- 2) First use the `nmcli dev wifi` command to scan the surrounding WIFI hotspots

```
orangepi@orangepi:~$ nmcli dev wifi
```



IN-USE	BSSID	SSID	MODE	CHAN	RATE	SIGNAL	BARS	SECURITY
	28:6C:07:6E:87:2E	orangePi	Infra	9	260 Mbit/s	97		WPA1 WPA2
	D8:D8:66:A5:BD:D1	██████████	Infra	10	270 Mbit/s	90		WPA1 WPA2
	A0:40:A0:A1:72:20	██████████	Infra	4	405 Mbit/s	82		WPA2
	28:6C:07:6E:87:2F	orangePi_5G	Infra	149	540 Mbit/s	80		WPA1 WPA2
	CA:50:E9:89:E2:44	ChinaNet_TC15	Infra	1	130 Mbit/s	79		WPA1 WPA2
	A0:40:A0:A1:72:31	NETSEARCH	Infra	100	405 Mbit/s	67		WPA2
	D4:EE:07:08:A9:E0	██████████	Infra	4	130 Mbit/s	55		WPA1 WPA2
	88:C3:97:49:25:13	██████████	Infra	6	130 Mbit/s	52		WPA1 WPA2
	00:BD:82:51:53:C2	██████████	Infra	12	130 Mbit/s	49		WPA1 WPA2
	C0:61:18:FA:49:37	██████████	Infra	149	270 Mbit/s	47		WPA1 WPA2
	04:79:70:8D:0C:B8	██████████	Infra	153	270 Mbit/s	47		WPA2
	04:79:70:FD:0C:B8	██████████	Infra	153	270 Mbit/s	47		WPA2
	9C:A6:15:DD:E6:0C	██████████	Infra	10	270 Mbit/s	45		WPA1 WPA2
	B4:0F:3B:45:D1:F5	██████████	Infra	48	270 Mbit/s	45		WPA1 WPA2
	E8:CC:18:4F:7B:44	██████████	Infra	157	135 Mbit/s	45		WPA1 WPA2
	B0:95:8E:D8:2F:ED	██████████	Infra	11	405 Mbit/s	39		WPA1 WPA2
	C0:61:18:FA:49:36	██████████	Infra	11	270 Mbit/s	24		WPA1 WPA2

3) Then use the **nmcli** command to connect to the scanned WIFI hotspot, where:

- wifi\_name** Need to be changed to the name of the WIFI hotspot you want to connect to
- wifi\_passwd** need to change to the password of the WIFI hotspot you want to connect to

```
orangepi@orangepi:~$ nmcli dev wifi connect wifi_name password wifi_passwd
```

```
Device 'wlan0' successfully activated with 'cf937f88-ca1e-4411-bb50-61f402eef293'.
```

4) You can view the IP address of the wifi through the **ip addr show wlan0** command

```
orangepi@orangepi:~$ ip addr show wlan0
11: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
    link/ether 23:8c:d6:ae:76:bb brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.11/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 259192sec preferred_lft 259192sec
        inet6 240e:3b7:3240:c3a0:c401:a445:5002:ccdd/64 scope global dynamic
noprefixroute
            valid_lft 259192sec preferred_lft 172792sec
        inet6 fe80::42f1:6019:a80e:4c31/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

5) Use the **ping** command to test the connectivity of the wifi network. The **ping** command can be interrupted by the **Ctrl+C** shortcut key



```
orangeipi@orangeipi:~$ ping www.orangeipi.org -I wlan0
PING www.orangeipi.org (182.92.236.130) from 192.168.1.49 wlan0: 56(84) bytes of
data.
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=1 ttl=52 time=43.5 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=2 ttl=52 time=41.3 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=3 ttl=52 time=44.9 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=4 ttl=52 time=45.6 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=5 ttl=52 time=48.8 ms
^C
--- www.orangeipi.org ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 41.321/44.864/48.834/2.484 ms
```

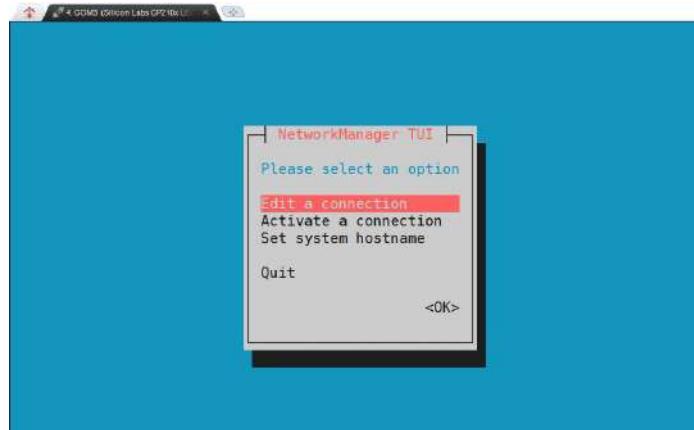
### 3.8.2.2. The server version image connects to WIFI through a graphical method

- 1) First log in to the linux system, there are the following three ways
  - a. If the development board is connected to the network cable, you can [log in to the Linux system remotely through ssh](#)
  - b. If the development board is connected to the debugging serial port, you can use the serial port terminal to log in to the linux system (please use MobaXterm for serial port software, and the graphical interface cannot be displayed using minicom)
  - c. If the development board is connected to the HDMI display, you can log in to the linux system through the HDMI display terminal

- 2) Then enter the nmtui command in the command line to open the wifi connection interface

```
orangeipi@orangeipi:~$ nmtui
```

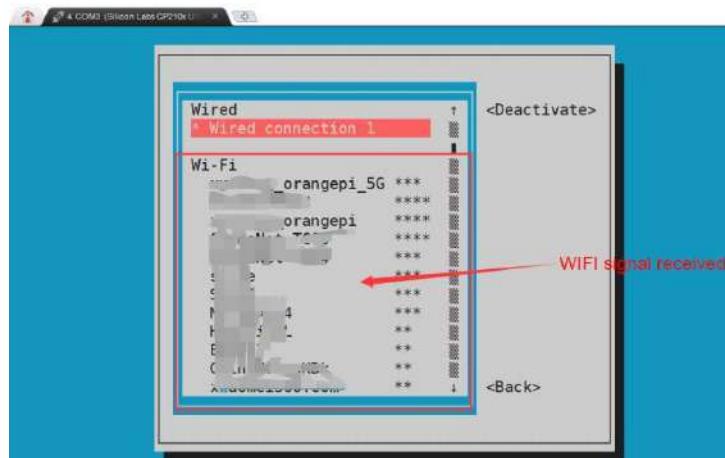
- 3) Enter the nmtui command to open the interface as shown below



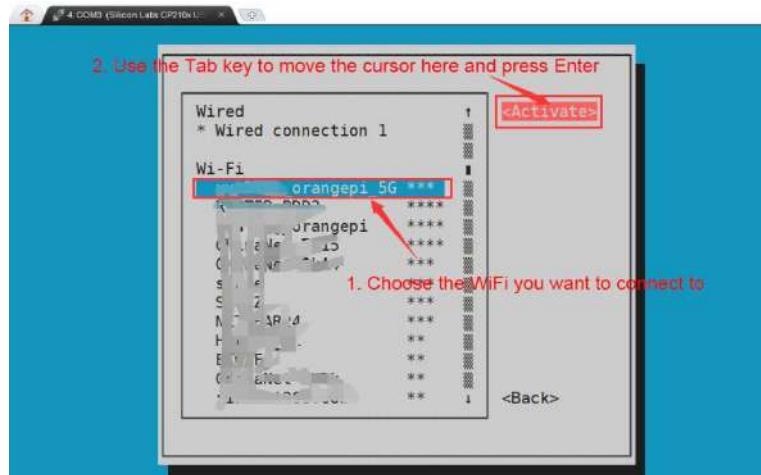
- 4) Select **Activate a connect** and press Enter



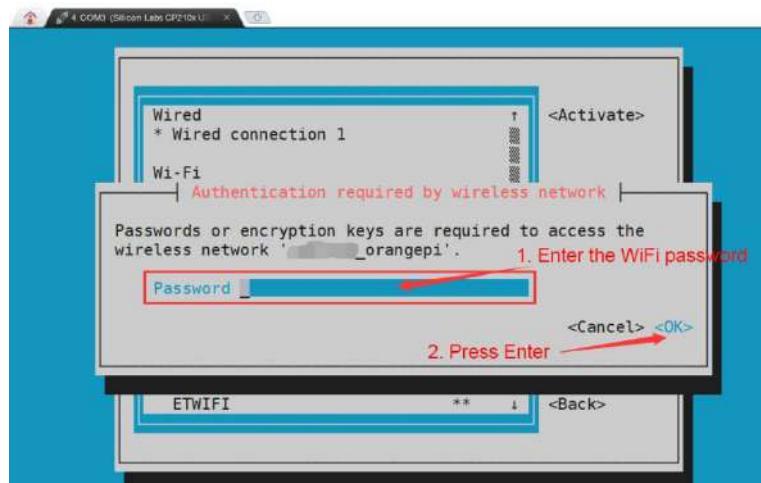
- 5) Then you can see all the searched WIFI hotspots



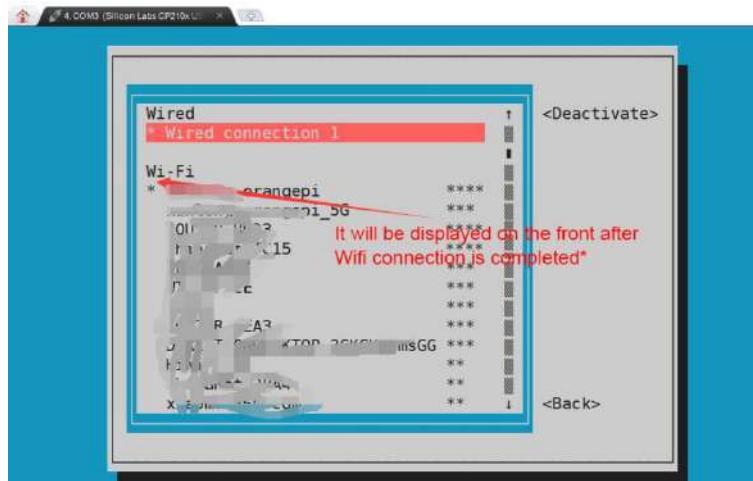
- 6) Select the WIFI hotspot you want to connect, then use the Tab key to position the cursor to **Activate** and press Enter



- 7) Then a dialog box for entering a password will pop up, enter the corresponding password in **Pssword** and press Enter to start connecting to WIFI



- 8) After the WIFI connection is successful, a "\*" will be displayed in front of the connected WIFI name



- 9) You can view the IP address of the wifi through the **ip addr show wlan0** command

```
orangepi@orangepi:~$ ip addr show wlan0
11: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
    link/ether 24:8c:d3:aa:76:bb brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.11/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 259069sec preferred_lft 259069sec
        inet6 240e:3b7:3240:c4a0:c401:a445:5002:ccdd/64 scope global dynamic
noprefixroute
            valid_lft 259071sec preferred_lft 172671sec
        inet6 fe80::42f1:6019:a80e:4c31/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

- 10) Use the **ping** command to test the connectivity of the wifi network. The **ping** command can be interrupted by the **Ctrl+C** shortcut key

```
orangepi@orangepi:~$ ping www.orangepi.org -I wlan0
PING www.orangepi.org (182.92.236.130) from 192.168.1.49 wlan0: 56(84) bytes of
data.
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=1 ttl=52 time=43.5 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=2 ttl=52 time=41.3 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=3 ttl=52 time=44.9 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=4 ttl=52 time=45.6 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=5 ttl=52 time=48.8 ms
^C
```



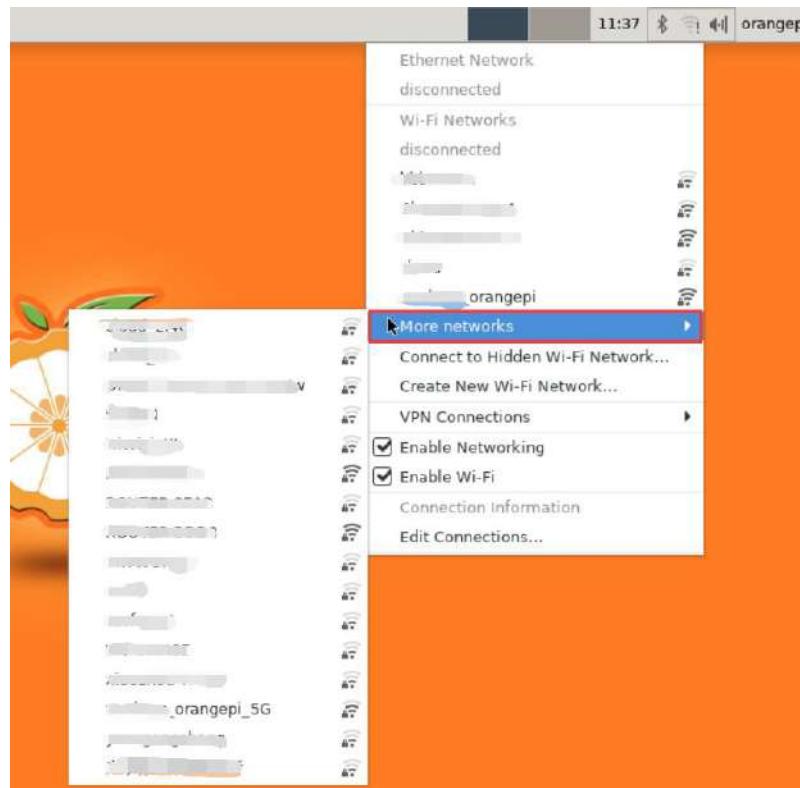
```
--- www.orangepi.org ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4006ms  
rtt min/avg/max/mdev = 41.321/44.864/48.834/2.484 ms
```

### 3. 8. 2. 3. Test method for desktop image

- 1) Click the network configuration icon in the upper right corner of the desktop (please do not connect the network cable when testing WIFI)



- 2) Click **More networks** in the pop-up drop-down box to see all scanned WIFI hotspots, and then select the WIFI hotspot you want to connect to

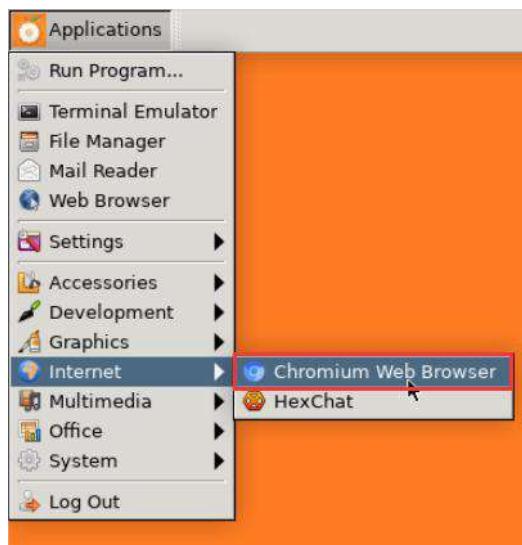




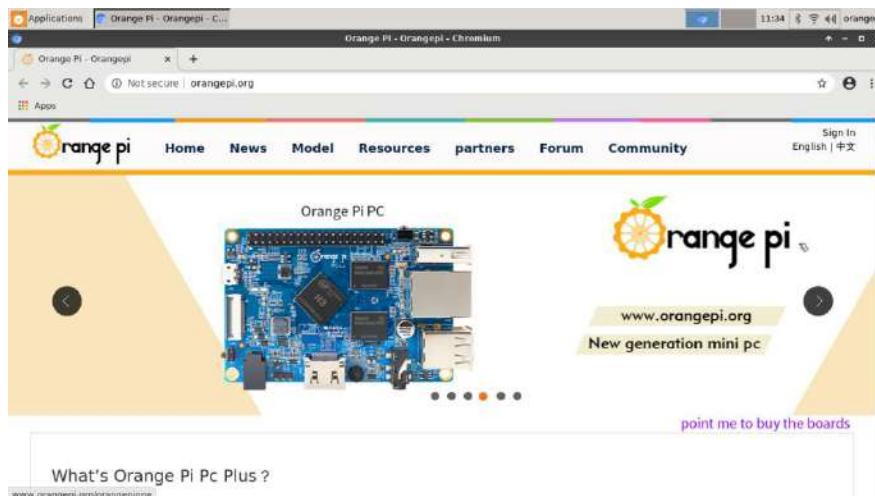
- 3) Then enter the password of the WIFI hotspot, and then click **Connect** to start connecting to the WIFI



- 4) After connecting to WIFI, you can open the browser to check whether you can access the Internet. The entrance of the browser is shown in the figure below.



- 5) After opening the browser, if you can see the page of the Orange Pi website, or you can open other pages, it means the WIFI connection is normal



### 3. 8. 3. How to use Hostapd to establish a WIFI hotspot

First, please make sure that the development board is connected to the network cable and can access the Internet normally. Because the process of setting up Hostapd to establish a WIFI hotspot needs to download some software packages from the Internet. If the development board cannot access the Internet through **the wired network port**, the installation will fail.

If there is no network cable connected, when other network devices (such as mobile phones or computers) are connected to the WIFI hotspot launched by the development board, they cannot access the external network normally (such as the mobile phone browser cannot open the webpage).

After using Hostapd to establish a WIFI hotspot (note that after setting it up), if you do not need to access the external network, but only need to connect to the WIFI hotspot launched by the development board, then it is also possible to not connect the network cable.

In addition, before setting up Hostapd, please make sure that WIFI is not connected to the network, otherwise, it will prompt that WIFI is in use and cannot set up Hostapd normally.

- 1) First update the software source index of the system

```
root@orangepi:~$ sudo apt-get update
```

- 2) Then enter the **orangepi-config** command in the terminal, **remember to add sudo permissions**

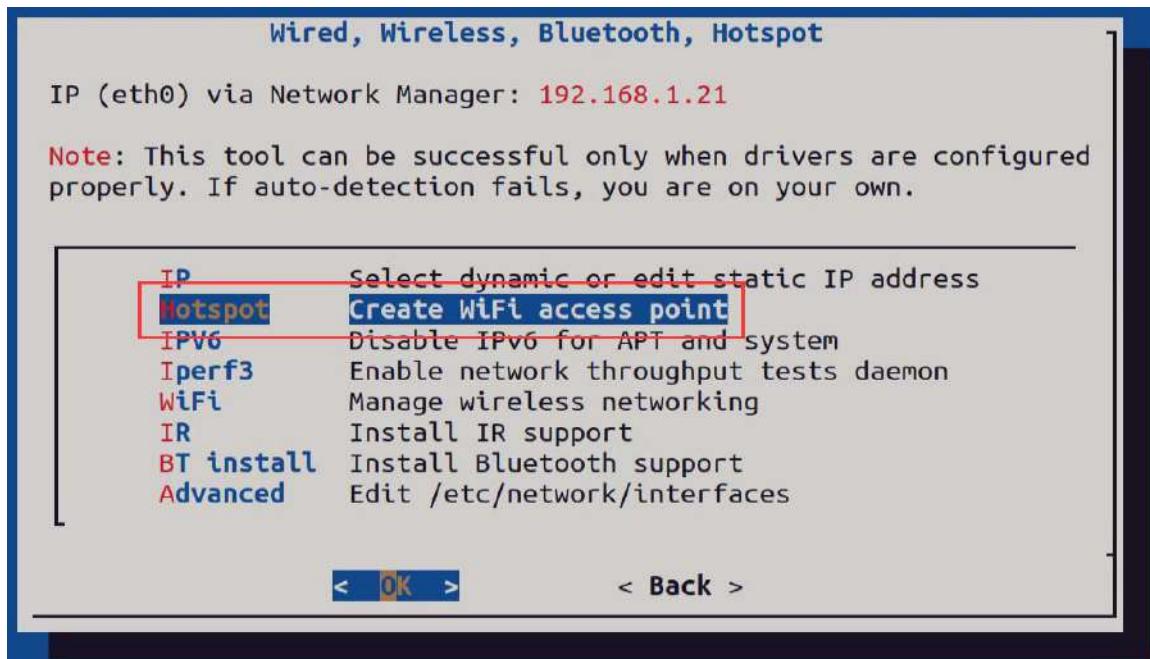
```
root@orangepi:~$ sudo orangepi-config
```



- 3) The interface after orangepi-config is opened is shown in the figure below, select the **Network** option to enter the network-related settings interface



- 4) Hotspot Then select **Hotspot Create WiFi access point** option to start setting up Hotspot

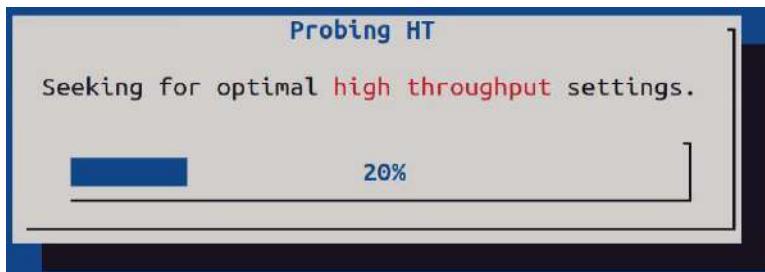


- 5) When the following selection box pops up, select **wlan0**

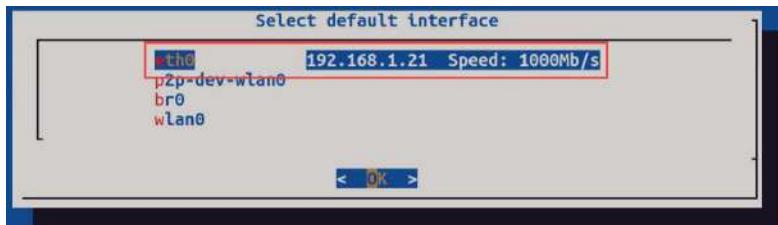
Note that this checkbox does not appear in Ubuntu Bionic and Debian Buster.



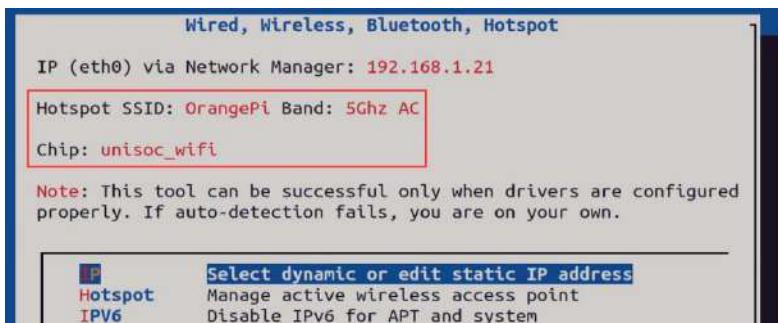
- 6) Then orangepi-config will start a series of settings, just wait patiently at this time



- 7) After waiting for a while, the following selection box will pop up, please select the first **eth0**



- 8) After all settings are completed, if orangepi-config displays the following interface, it means that Hostapd is set correctly



- 9) The name of the WIFI hotspot set by Hotspot is: **OrangePi** by default, and the password is: **12345678**. If everything is normal, the mobile phone or computer can search

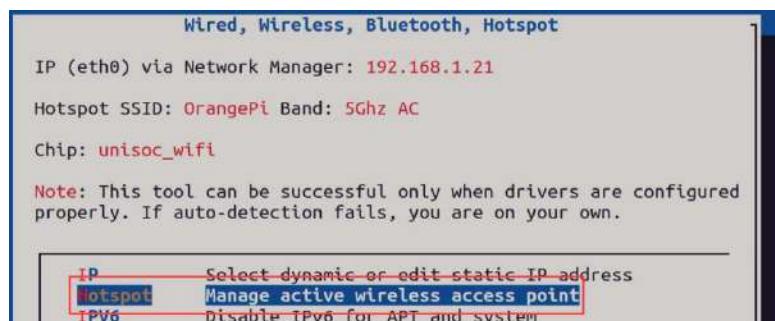


for the WIFI hotspot named OrangePi. Hostapd is set up correctly. The following figure is a schematic diagram of the WIFI hotspot emitted by the mobile phone connected to the development board:



10) After Hotspot is set up, open Hostapd in orangepi-config to configure it

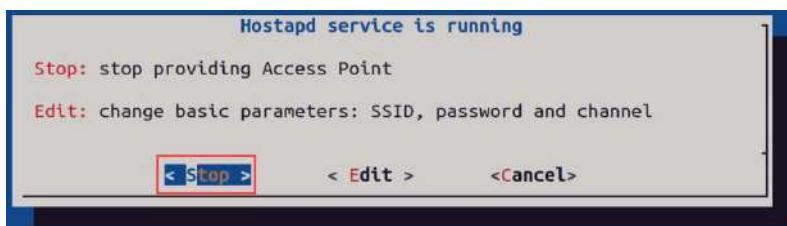
- First select **Hotspot** in config-config



- Then you can see the following selection interface



- Select **Stop** to stop the Hostapd service

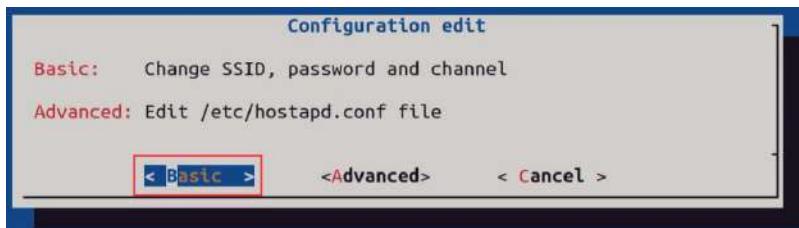


- Hostapd 的配置 Select **Edit** to edit the configuration of Hostapd

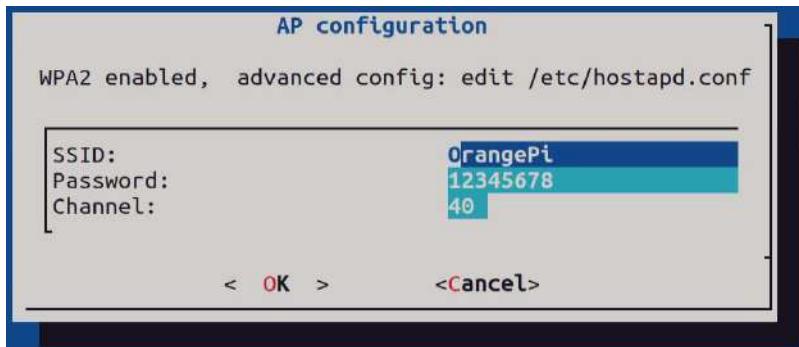
- The **Edit** option location of Hostapd is shown in the figure below



- b) The **Edit** option is opened as shown below



- i. Select **Basic** to modify the name and password of the WIFI hotspot.  
After modification, select <OK> to save



**Note: If you want to change the password, the changed password must not be less than 8 characters, otherwise the Hostapd service will not work properly.**

- ii. Select **Advanced** to directly modify the name and password of the WIFI hotspot and other configurations in the hostapd configuration file **/etc/hostapd.conf**. After modification, select <Save> to save it.



```
# orangepi hostapd configuration example
#
# nl80211 mode
#
ssid=OrangePi
interface=wlan0
hw_mode=g
channel=40
#bridge=br0
driver=nl80211

logger_syslog=0
logger_syslog_level=0
wmm_enabled=1
wpa=2
preamble=1

wpa_psk=66eb31d2b48d19ba216f2e50c6831ee11be98e2fa3a8075e30b866f4a5ccda27
wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
↓(+)
```

50%

&lt; Save &gt;

&lt;Cancel&gt;

**Note: If you want to change the password, the changed password must not be less than 8 characters, otherwise the Hostapd service will not work properly.**

### 3.8.4. How to set static IP address

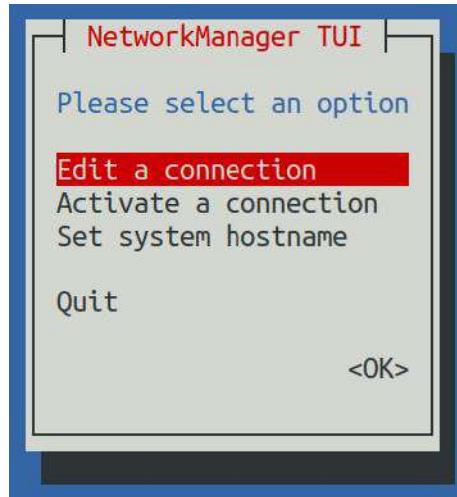
Please do not set a static IP address by modifying the `/etc/network/interfaces` configuration file.

#### 3.8.4.1. Using the nmtui command to set a static IP address

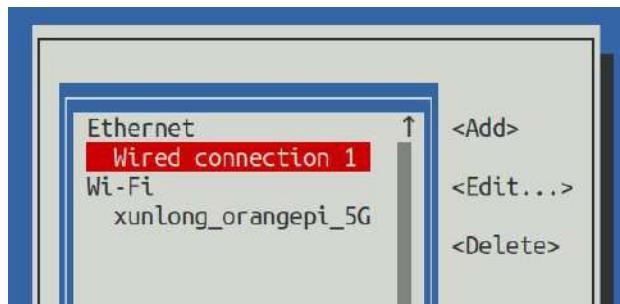
- 1) First run the `nmtui` command

```
orangepi@orangepi:~$ nmtui
```

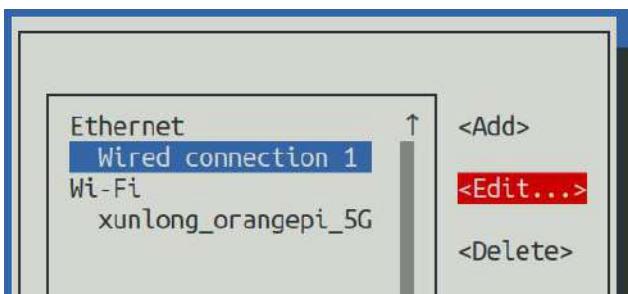
- 2) Then select **Edit a connection** and press enter



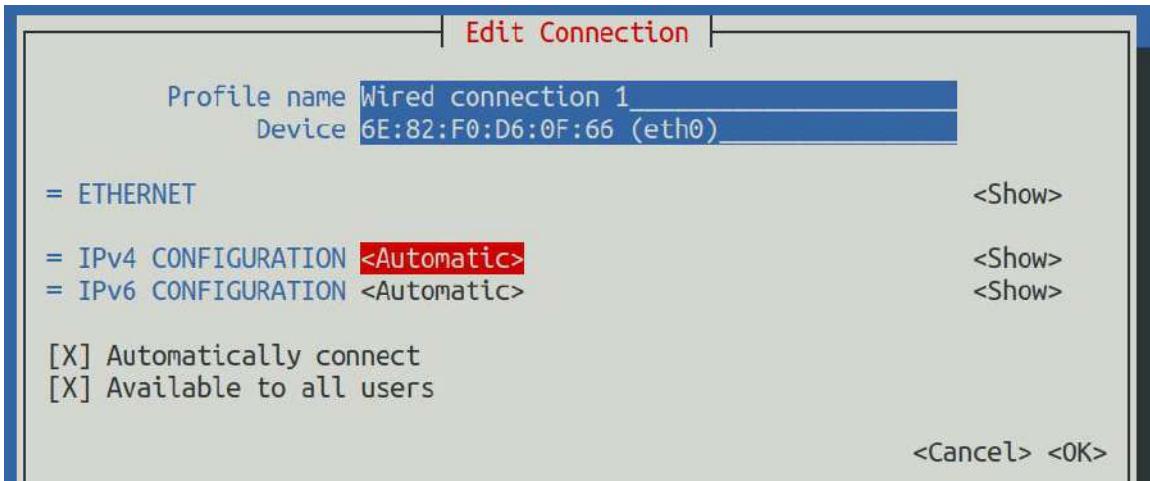
- 3) Then select the network interface that needs to set a static IP address, such as setting the static IP address of the **Ethernet** interface and select **Wired connection 1**.



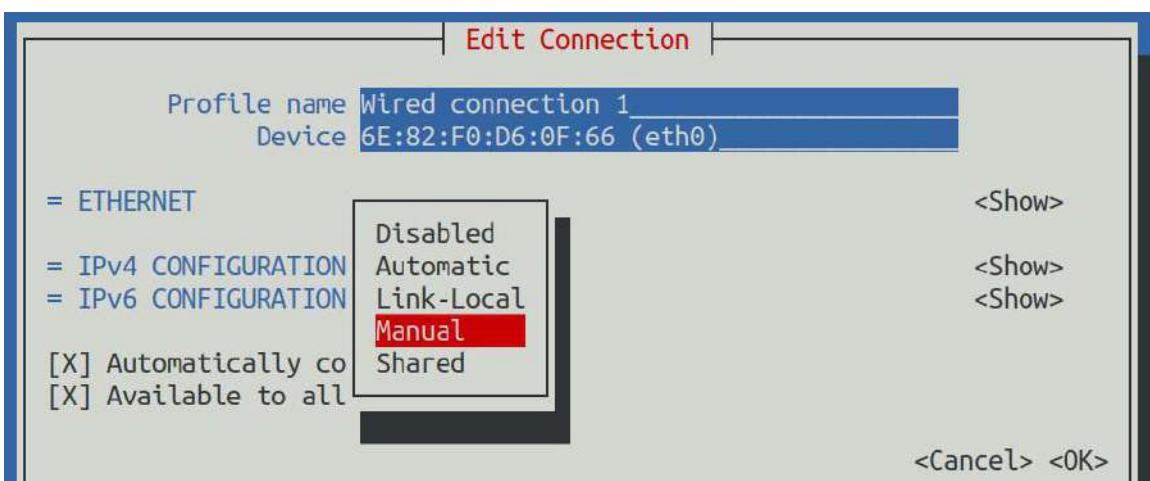
- 4) Then select **Edit** by **Tab** key and press Enter



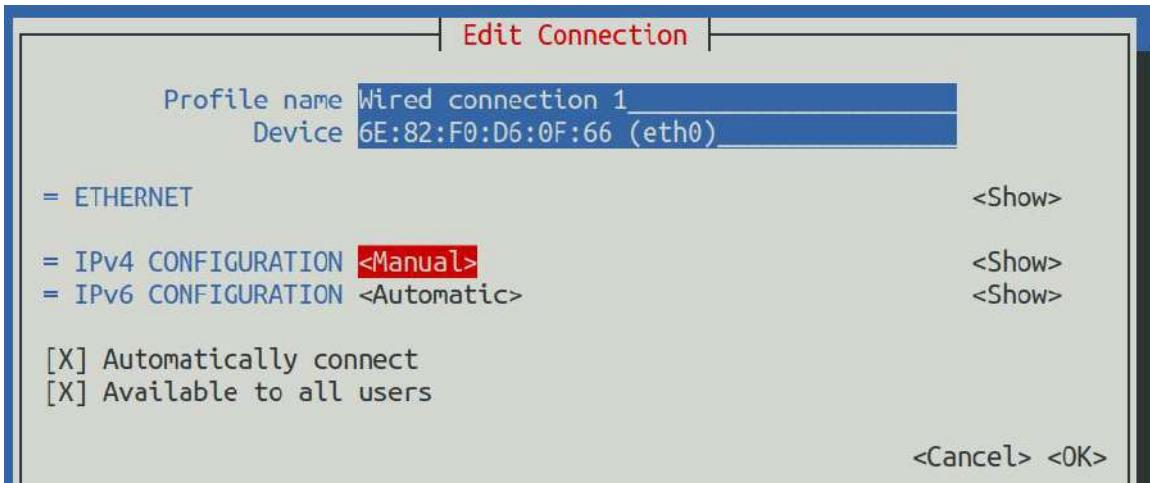
- 5) Then use the Tab key to move the cursor to the **<Automatic>** position shown in the figure below to configure IPv4



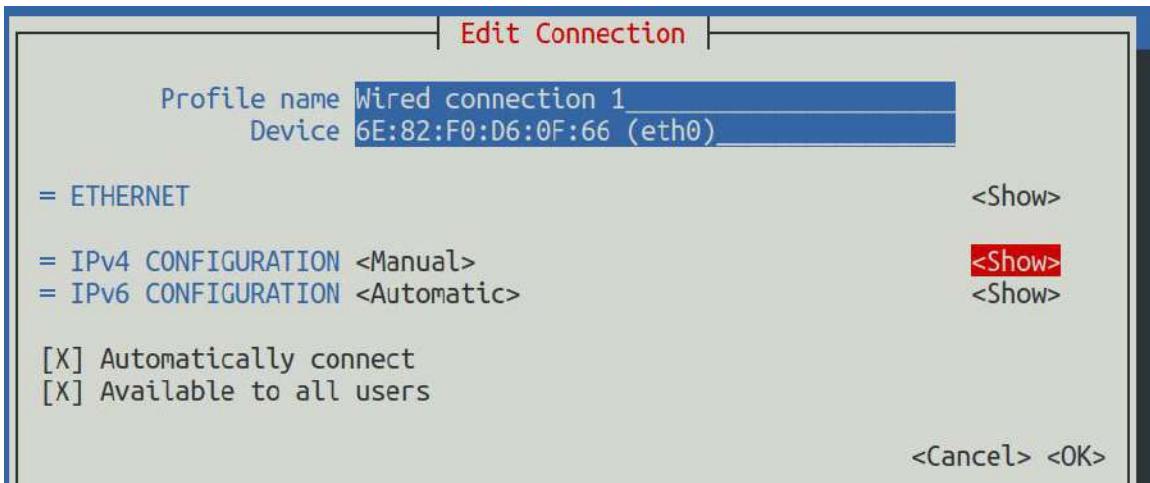
- 6) Then press Enter, use the up and down arrow keys to select **Manual**, then press Enter to confirm



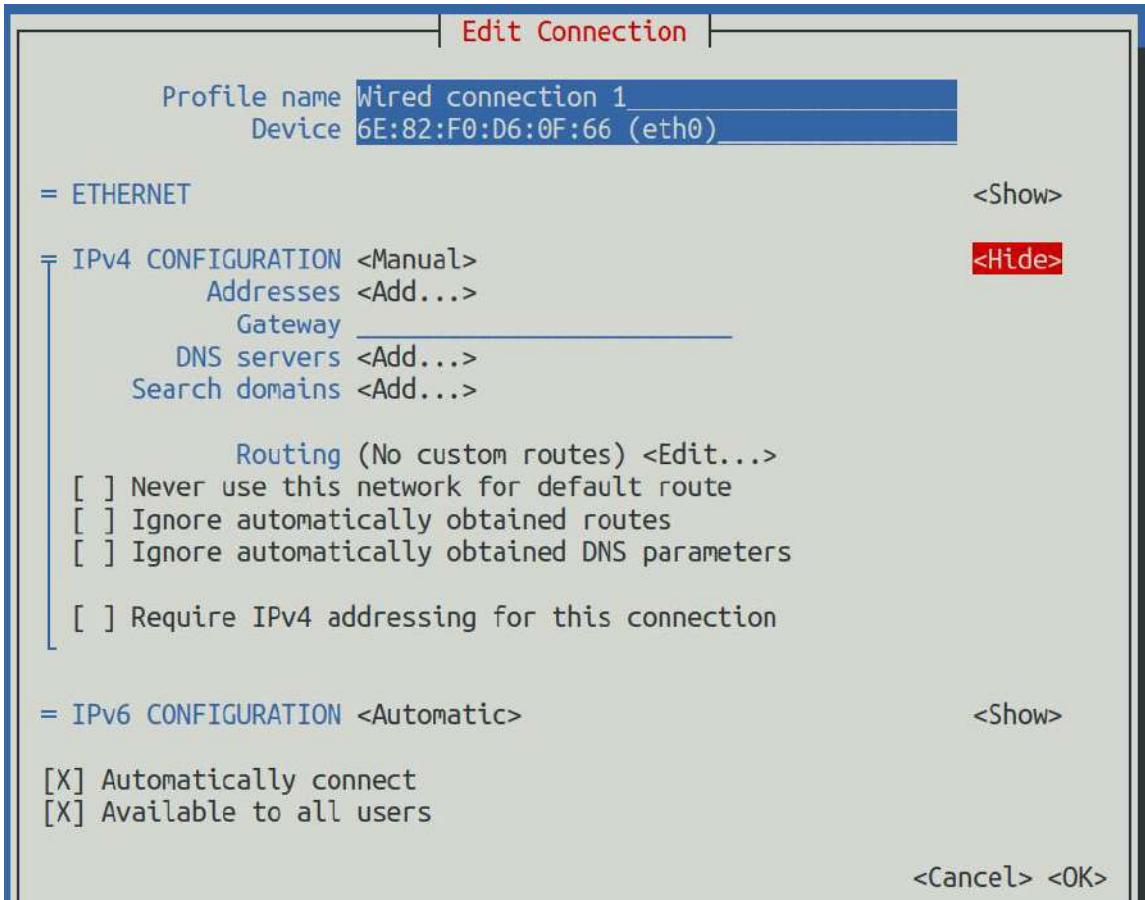
- 7) The display after selection is as shown below



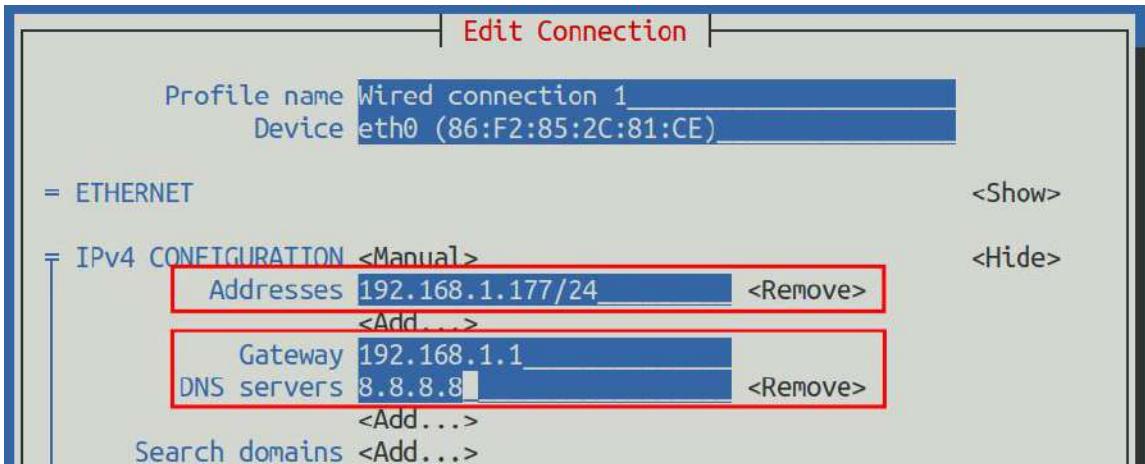
- 8) Then move the cursor to <Show> by Tab key



- 9) Then press Enter, the following setting interface will pop up after pressing Enter



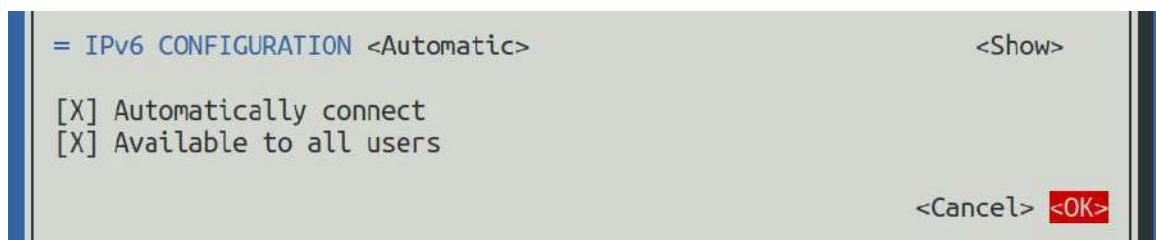
- 10) Then you can set the IP address (Addresses), gateway (Gateway) and DNS server address in the location shown in the figure below (there are many other setting options, please explore by yourself), **please set according to your specific needs**, The value set in the image below is just an example



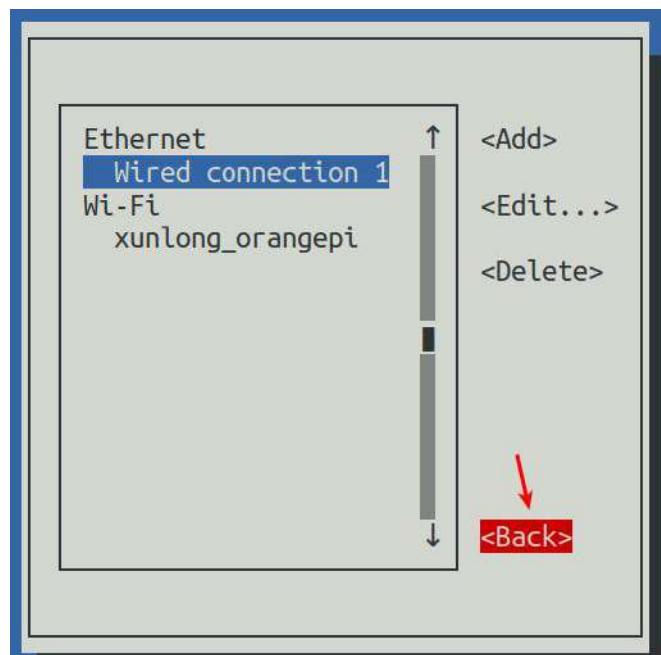
- 11) After setting, move the cursor to <OK> in the lower right corner, then press Enter to



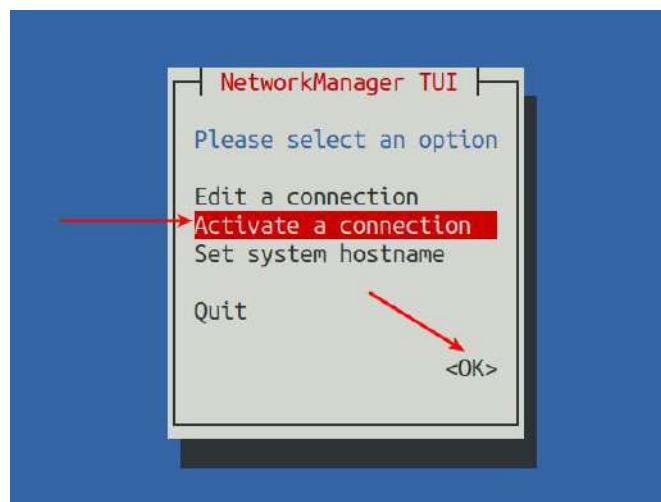
confirm



12) Then click <Back> to return to the previous selection interface

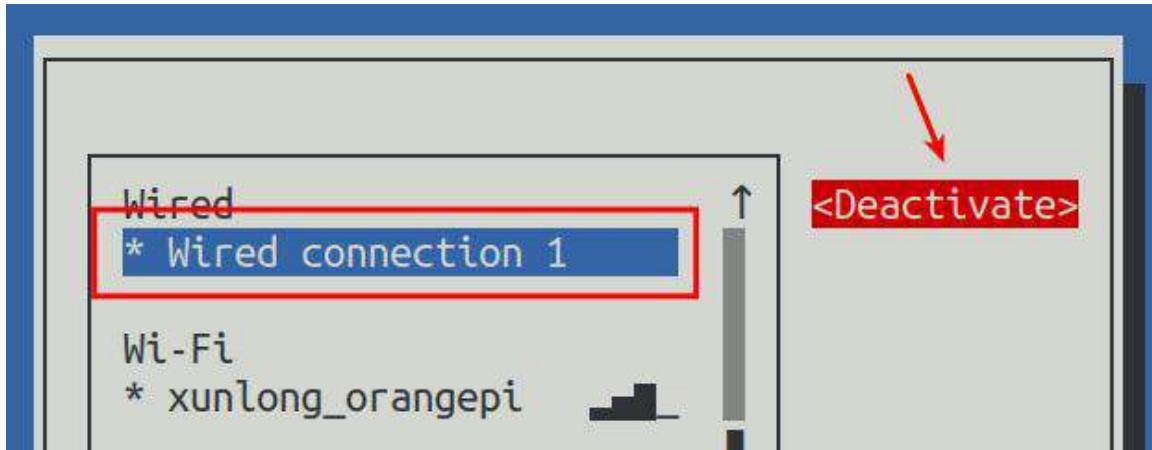


13) Then select **Activate a connection**, move the cursor to <OK>, and finally click Enter

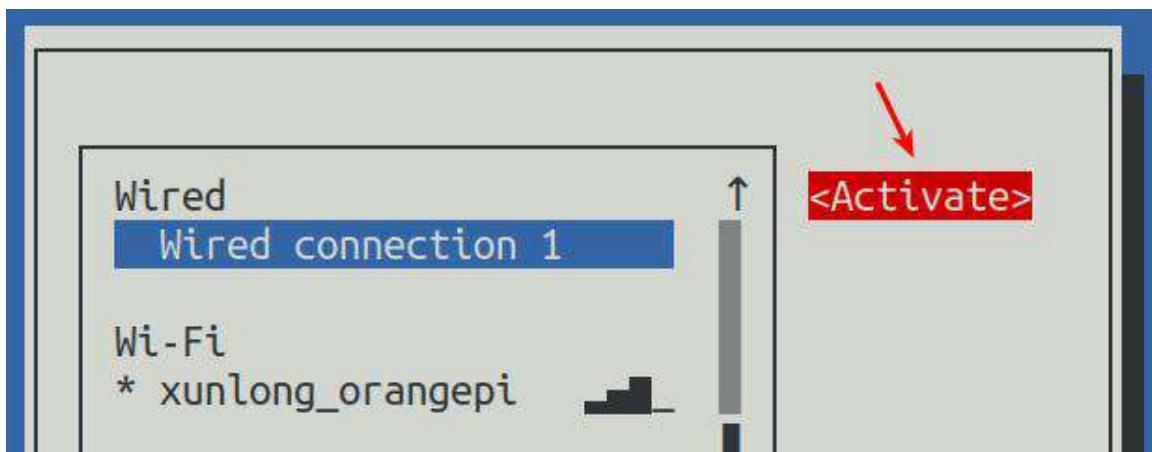




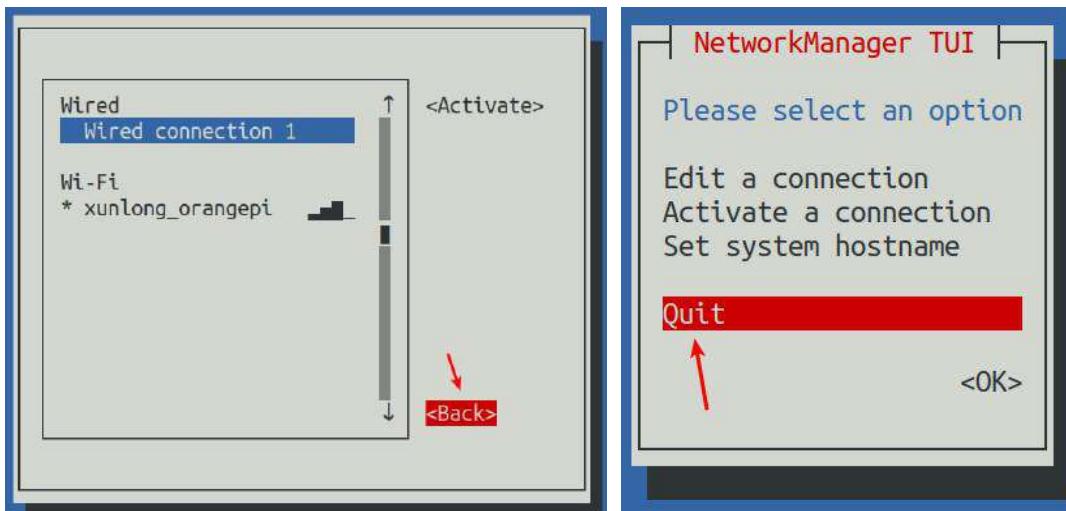
- 14) Then select the network interface to be set, such as **Wired connection 1**, then move the cursor to **<Deactivate>**, and press Enter to disable **Wired connection 1**



- 15) Then please do not move the cursor, and then press the Enter key to re-enable **Wired connection 1**, so that the static IP address set earlier will take effect



- 16) Then exit nmtui through the **<Back>** and **Quit** buttons



- 17) Then through **ip addr show eth0**, you can see that the IP address of the network port has become the static IP address set earlier

```
orangepi@orangepi:~$ ip addr show eth0
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 5e:ac:14:a5:92:b3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.177/24 brd 192.168.1.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 241e:3b8:3240:c3a0:e269:8305:dc08:135e/64 scope global dynamic
noprefixroute
        valid_lft 259149sec preferred_lft 172749sec
    inet6 fe80::957d:bbbe:4928:3604/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

- 18) Then you can test the connectivity of the network to check whether the IP address is configured OK. The **ping** command can be interrupted by the **Ctrl+C** shortcut key

```
orangepi@orangepi:~$ ping 192.168.1.47 -I eth0
PING 192.168.1.47 (192.168.1.47) from 192.168.1.188 eth0: 56(84) bytes of data.
64 bytes from 192.168.1.47: icmp_seq=1 ttl=64 time=0.233 ms
64 bytes from 192.168.1.47: icmp_seq=2 ttl=64 time=0.263 ms
64 bytes from 192.168.1.47: icmp_seq=3 ttl=64 time=0.273 ms
64 bytes from 192.168.1.47: icmp_seq=4 ttl=64 time=0.269 ms
64 bytes from 192.168.1.47: icmp_seq=5 ttl=64 time=0.275 ms
^C
```



```
--- 192.168.1.47 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4042ms
rtt min/avg/max/mdev = 0.233/0.262/0.275/0.015 ms
```

### 3. 8. 4. 2. Using nmcli command to set static IP address

- 1) If you want to set the static IP address of the network port, please insert the network cable into the development board first. **If you need to set the static IP address of the WIFI, please connect the WIFI first**, and then start to set the static IP address
- 2) Then you can view the name of the network device through the **nmcli con show** command, as shown below
  - a. **orangeipi** is the name of the WIFI network interface (the names are not necessarily the same)
  - b. **Wired connection 1** is the name of the Ethernet interface

```
orangeipi@orangeipi:~$ nmcli con show
NAME           UUID                                  TYPE      DEVICE
orangeipi      cfc4f922-ae48-46f1-84e1-2f19e9ec5e2a    wifi      wlan0
Wired connection 1  9db058b7-7701-37b8-9411-efc2ae8bfa30  ethernet  eth0
```

- 3) Then enter the following command, where
  - a. "**Wired connection 1**" means to set the static IP address of the Ethernet port. If you need to set the static IP address of the WIFI, please modify it to the name corresponding to the WIFI network interface (you can get it through the **nmcli con show** command)
  - b. After **b.ipv4.addresses** is the static IP address to be set, which can be modified to the value you want to set
  - c. **ipv4.gateway** represents the address of the gateway

```
orangeipi@orangeipi:~$ nmcli con mod "Wired connection 1" \
ipv4.addresses "192.168.1.110" \
ipv4.gateway "192.168.1.1" \
ipv4.dns "8.8.8.8" \
ipv4.method "manual"
```

- 4) Then restart the linux system



```
orangeipi@orangeipi:~$ sudo reboot
```

- 5) Then re-enter the linux system and use the **ip addr show eth0** command to see that the IP address has been set to the desired value

```
orangeipi@orangeipi:~$ ip addr show eth0
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 5e:ae:14:a5:91:b3 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.110/32 brd 192.168.1.110 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
            inet6 240e:3b7:3240:c3a0:97de:1d01:b290:fe3a/64 scope global dynamic noprefixroute
                valid_lft 259183sec preferred_lft 172783sec
            inet6 fe80::3312:861a:a589:d3c/64 scope link noprefixroute
                valid_lft forever preferred_lft forever
```

### 3. 8. 5. Set up the method of automatically connecting to the network when the Linux system starts up for the first time

The development board has an Ethernet port. If you want to remotely log in to the Linux system of the development board through the Ethernet port, you only need to plug a network cable that can access the Internet normally to the Ethernet port. After the Linux system is started, it will automatically pass DHCP to the Ethernet port. Assign an IP address, and then we can obtain the IP address of the Ethernet port through the HDMI screen, serial port or by viewing the router background, and then we can log in to the Linux system remotely.

The development board also has wireless WIFI. If you want to remotely log in to the Linux system of the development board through WIFI, you need to remotely log in to the Linux system through the IP address of the Ethernet port through ssh and then connect to the WIFI through commands, or use commands on the HDMI screen or serial port. Connect to WIFI.

However, if there is no HDMI screen and serial port module, although there is a network cable, the IP address of the development board cannot be viewed through the router background. Or if there is no HDMI screen, serial port module and network cable, and only WIFI can be connected, you can use the method described in this section to automatically connect to WIFI and also set the static IP address of the WIFI or automatically set the static IP address of the Ethernet port.



To use the method in this section, you first need to prepare a Linux system machine. For example, a computer or virtual machine with Ubuntu system installed.

Why do you need a Linux system machine? Because the root file system of the Linux system of the development board burned in the TF card is in ext4 format, the Linux system machine can mount it normally, and then modify the configuration file.

If you want to modify it in Windows system, you can use Paragon ExtFS for Windows software. Since this software needs to be paid, and there is no similar free software that is easy to use, we will not demonstrate it here.

In addition, if you have any problems trying to use Paragon ExtFS for Windows, please solve it by yourself, we will not answer any questions.

1) First burn the Linux image of the development board you want to use into the TF card, and then use the card reader to insert the TF card with the Linux image of the development board into the machine installed with the Linux system (such as the one with the Ubuntu system installed). Computer, the following is an example of Ubuntu computer to demonstrate)

2) When the TF card is inserted into the Ubuntu computer, the Ubuntu computer will generally automatically mount the partition of the Linux root file system in the TF card, which can be known from the following commands,

/media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f, It is the path to mount the Linux root file system in the TF card (the path name is based on what you actually see, not necessarily the same)

```
test@test:~$ df -h | grep "media"
/dev/sdd1  1.4G  1.2G  167M  88%
/media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f
test@test:~$ ls /media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f
bin  boot  dev  etc  home  lib  lost+found  media  mnt  opt  proc  root  run
sbin  selinux  srv  sys  tmp  usr  var
```

3) Then enter the /boot directory of the Linux system burned in the TF card

```
test@test:~$ cd /media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f/boot
```



- 4) Then copy the **orangeipi\_first\_run.txt.template** as **orangeipi\_first\_run.txt**. Through the orangeipi\_first\_run.txt configuration file, you can set the development board to automatically connect to a WIFI hotspot when the Linux system starts for the first time, or you can set the WIFI or Ethernet port. static IP address

```
test@test:/media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f/boot$ sudo cp \
orangeipi_first_run.txt.template orangeipi_first_run.txt
```

- 5) You can open the orangeipi\_first\_run.txt file through the following command, and then you can view and modify the content

```
test@test:/media/test/d17d5e4f-ae41-4554-a727-bb5c9c94134f/boot$ sudo vim \
orangeipi_first_run.txt
```

- 6) Instructions for using variables in the orangeipi\_first\_run.txt file

- a. **FR\_general\_delete\_this\_file\_after\_completion** The variable is used to set whether to delete the orangeipi\_first\_run.txt file after the first startup. The default value is 1, that is, delete. If it is set to 0, orangeipi\_first\_run.txt will be renamed to orangeipi\_first\_run.old after the first startup. Generally keep the default value
- b. **FR\_net\_change\_defaults** The variable is used to set whether to change the default network settings, this must be set to 1, otherwise all network settings will not take effect
- c. **FR\_net\_ethernet\_enabled** The variable is used to control whether the configuration of the Ethernet port is enabled. If you need to set the static IP address of the Ethernet port, please set it to 1
- d. **FR\_net\_wifi\_enabled** The variable is used to control whether it can be configured with WIFI. If you need to set the development board to automatically connect to the WIFI hotspot, you must set it to 1. Also, please note that if this variable is set to 1, the setting of the Ethernet port will be invalid. That is to say, WIFI and Ethernet ports cannot be set at the same time (why, because there is no need...)
- e. **FR\_net\_wifi\_ssid** The variable is used to set the name of the WIFI hotspot you want to connect to
- f. **FR\_net\_wifi\_key** The variable is used to set the password of the WIFI hotspot you want to connect to
- g. **FR\_net\_use\_static** The variable is used to set whether to set the static IP



- address of the WIFI or Ethernet port
- h. **FR\_net\_static\_ip** The variable is used to set the address of the static IP, please set it according to your actual situation
  - i. **FR\_net\_static\_gateway** The variable is used to set the gateway, please set it according to your actual situation
- 7) The following demonstrates several specific setting examples:
- a. For example, if you want the Linux system of the development board to automatically connect to the WIFI hotspot after the first boot, you can set it like this
    - a) Set **FR\_net\_change\_defaults** to 1
    - b) Set **FR\_net\_wifi\_enabled** to 1
    - c) Set **FR\_net\_wifi\_ssid** to the name of the WIFI hotspot you want to connect to
    - d) Set **FR\_net\_wifi\_key** as the password of the WIFI hotspot you want to connect to
  - b. For example, after the Linux system that wants to develop the board is started for the first time, it will automatically connect to the WIFI hotspot, and set the IP address of the WIFI to a specific static IP address (so that when the Linux system starts, you can directly use the set static IP address to ssh remote Log in to the development board, you don't need to check the IP address of the development board through the router background), you can set it like this:
    - a) Set **FR\_net\_change\_defaults** to 1
    - b) Set **FR\_net\_wifi\_enabled** to 1
    - c) Set **FR\_net\_wifi\_ssid** to the name of the WIFI hotspot you want to connect to
    - d) Set **FR\_net\_wifi\_key** as the password of the WIFI hotspot you want to connect to
    - e) Set **FR\_net\_use\_static** to 1
    - f) Set **FR\_net\_static\_ip** to the desired IP address
    - g) Set **FR\_net\_static\_gateway** to the corresponding gateway address
  - c. For example, after the Linux system that wants to develop the board is started for the first time, the IP address of the Ethernet port is automatically set to the desired static IP address, which can be set as follows:



- a) Set **FR\_net\_change\_defaults** to 1
- b) Set **FR\_net\_etherent\_enabled** to 1
- c) Set **FR\_net\_use\_static** to 1
- d) Set **FR\_net\_static\_ip** to the desired IP address
- e) Set **FR\_net\_static\_gateway** to the corresponding gateway address

8) After modifying the orangepi\_first\_run.txt file, you can exit the /boot directory of the Linux system of the development board in the TF card, uninstall the TF card, and then insert the TF card into the development board to start

9) If a static IP address is not set, you still need to check the IP address through the router background. If a static IP address is set, you can ping the static IP address set on the computer. If you can ping, the system has been started normally, and The network has also been set correctly, and then you can use the set IP address to ssh to log in to the Linux system of the development board remotely.

**After the Linux system of the development board is started for the first time, orangepi\_first\_run.txt will be deleted or renamed to orangepi\_first\_run.txt.old. At this time, even if the orangepi\_first\_run.txt configuration file is reset, and then restart the Linux system of the development board, orangepi\_first\_run. The configuration in txt will not take effect again, because this configuration will only work after the first boot after the Linux system is burned, please pay special attention to this.**

### 3. 9. SSH remote login development board

**By default, Linux systems enable ssh remote login and allow root users to log in to the system. Before ssh login, you need to make sure that the Ethernet or wifi network is connected, and then use the ip addr show command or obtain the IP address of the development board by viewing the router.**

#### 3. 9. 1. SSH remote login development board under Ubuntu

1) Obtain the IP address of the development board

2) Then you can log in to the Linux system remotely through the ssh command

test@test:~\$ **ssh root@192.168.1.36** (It needs to be replaced with the IP address of



the development board)

root@192.168.1.36's password: (Enter the password here, the default password is orangepi)

Note that when entering the password, **the specific content of the entered password will not be displayed on the screen**, please do not think that there is any fault, just press Enter after entering it.

If you are prompted to refuse the connection, as long as the image provided by Orange Pi is used, **please do not doubt whether the password of orangepi is wrong, but find other reasons.**

3) After successfully logging in to the system, the display is as shown below

```
test@test:~$ ssh root@192.168.1.165
root@192.168.1.165's password:
```



```
Welcome to Orange Pi Focal with Linux 5.10.46-sunxi64
```

```
System load: 0.21 0.07 0.02 Up time: 1 min
Memory usage: 6 % of 1989MB IP: 192.168.1.165
CPU temp: 51°C
Usage of /: 9% of 15G
```

```
[ General system configuration (beta): orangepi-config ]
```

```
Last login: Thu Jul 1 01:23:45 2021 from 192.168.1.87
```

```
root@orangepi3-lts:~# 
```

If ssh cannot log in to the linux system normally, please first check whether the IP address of the development board can be pinged. Can connect:

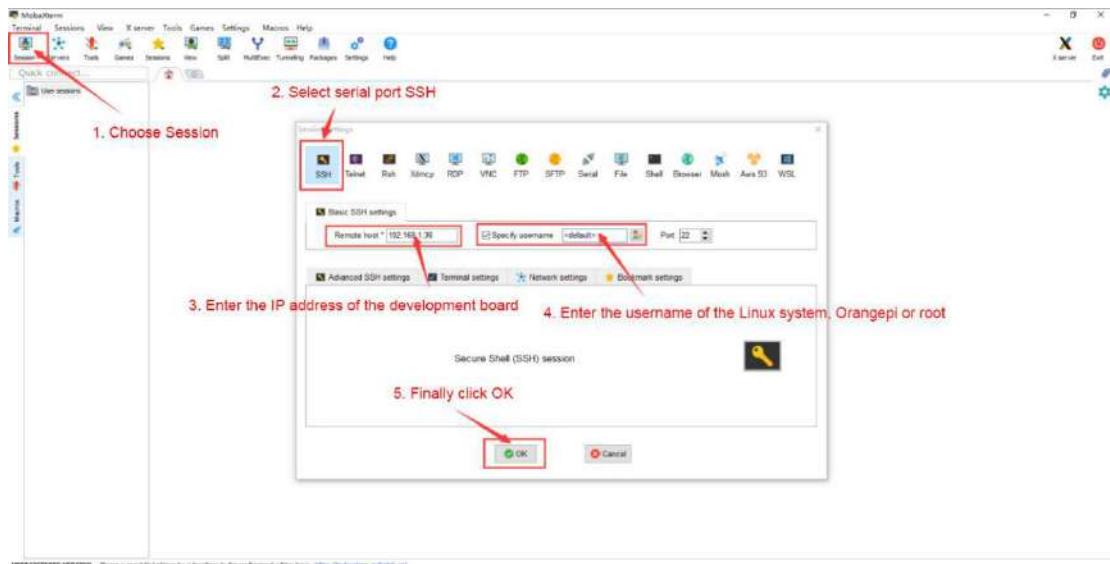
```
root@orangepi:~# rm /etc/ssh/ssh_host_*
root@orangepi:~# dpkg-reconfigure openssh-server
```



If it still doesn't work, please restart the system and try it.

### 3.9.2. SSH remote login development board under Windows

- 1) First get the IP address of the development board
- 2) Under Windows, you can use MobaXterm to remotely log in to the development board, first create a new ssh session
  - a. Open Session
  - b. Then select SSH in Session Setting
  - c. Then enter the IP address of the development board in Remote host
  - d. Then enter the username **root** or **orangeipi** of the linux system in Specify username
  - e. Finally click **OK**

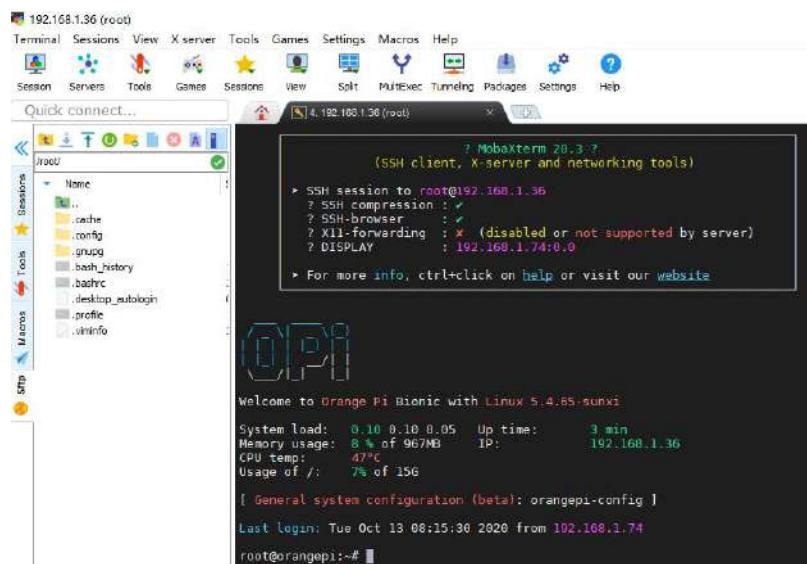


- 3) Then you will be prompted to enter a password. The default passwords for both root and orangeipi users are orangeipi

Note that when entering the password, the specific content of the entered password will not be displayed on the screen, please do not think that there is any fault, just press Enter after entering it.



- 4) After successfully logging in to the system, the display is as shown below



### 3. 10. HDMI related test items

#### 3. 10. 1. HDMI Display Test

- 1) Use HDMI to HDMI cable to connect Orange Pi development board and HDMI display



- 2) After starting the linux system, if the HDMI display has image output, it means that



the HDMI interface is working normally

**Note that although many laptops have an HDMI interface, the HDMI interface of the notebook generally only has the output function, and does not have the function of HDMI in, which means that the HDMI output of other devices cannot be displayed on the screen of the notebook.**

**When you want to connect the HDMI of the development board to the HDMI port of the laptop, please make sure that your laptop supports the HDMI in function.**

**When the HDMI does not display, please check whether the HDMI cable is plugged in tightly. After confirming that the connection is correct, you can try a different screen to see if there is any display.**

### 3. 10. 2. HDMI to VGA display test

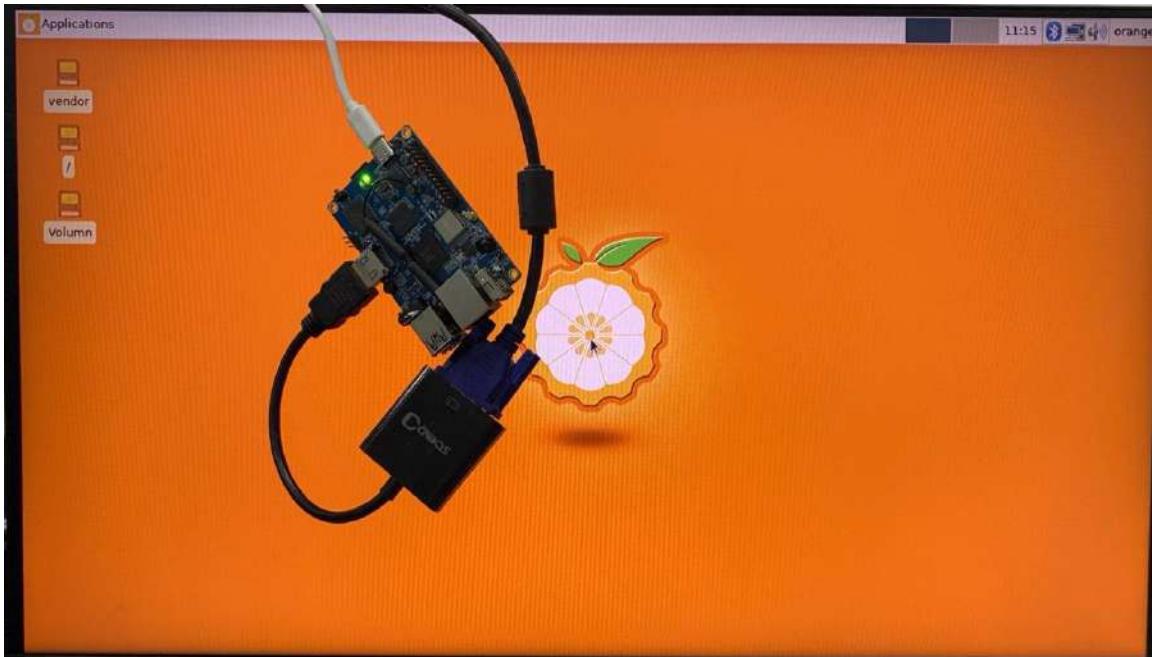
- 1) First you need to prepare the following accessories
  - a. HDMI to VGA converter



- b. A VGA cable



- c. A monitor or TV that supports VGA interface
- 2) HDMI to VGA display test as shown below



When using HDMI to VGA display, the development board and the Linux system of the development board do not need to do any settings, as long as the Micro HDMI interface of the development board can display normally. So if there is a problem with the test, please check the HDMI to VGA converter, VGA cable and monitor if there is any problem

### 3. 10. 3. Linux4.9 system HDMI resolution setting

Note: This method is only applicable to systems with linux4.9 kernel

- 1) There is a disp\_mode variable in **/boot/orangepiEnv.txt** of the linux4.9 system, which can be used to set the resolution of the HDMI output. The default resolution of the linux system is **1080p60**

```
root@orangepi:/boot# cat orangepiEnv.txt
verbosity=1
disp_mode=1080p60
```

- 2) The values supported by the disp\_mode variable are shown in the following table

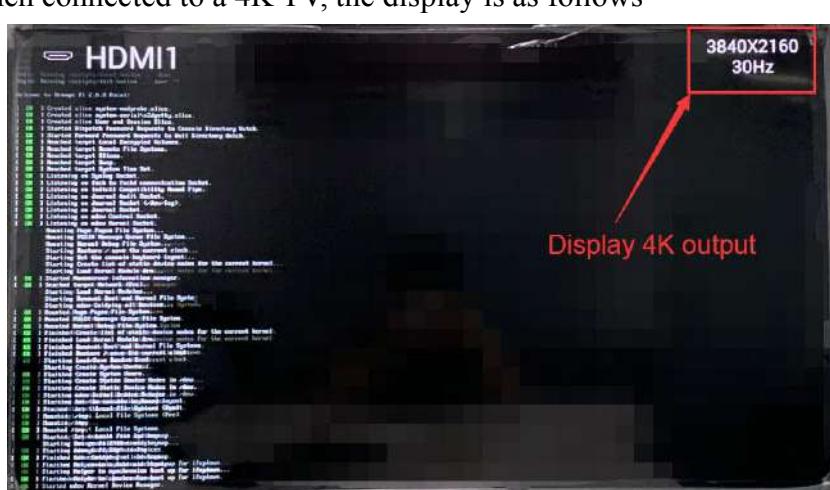
Models supported by disp_mode	HDMI resolution	HDMI refresh rate
<b>480i</b>	<b>720x480</b>	<b>60</b>
<b>576i</b>	<b>720x480</b>	<b>50</b>



<b>480p</b>	<b>720x480</b>	<b>60</b>
<b>576p</b>	<b>720x576</b>	<b>60</b>
<b>720p50</b>	<b>1280x720</b>	<b>50</b>
<b>720p60</b>	<b>1280x720</b>	<b>60</b>
<b>1080i50</b>	<b>1920x1080</b>	<b>50</b>
<b>1080i60</b>	<b>1920x1080</b>	<b>60</b>
<b>1080p24</b>	<b>1920x1080</b>	<b>24</b>
<b>1080p50</b>	<b>1920x1080</b>	<b>50</b>
<b>1080p60</b>	<b>1920x1080</b>	<b>60</b>
<b>2160p24</b>	<b>3840x2160</b>	<b>24</b>
<b>2160p25</b>	<b>3840x2160</b>	<b>25</b>
<b>2160p30</b>	<b>3840x2160</b>	<b>30</b>

Note that when the HDMI resolution is set to **2160p24**, **2160p25** and **2160p30**, the Framebuffer can only be set to a maximum of 1920x1080.

- 3) Modify the value of the disp\_mode variable to the resolution you want to output, then restart the system, and HDMI will output the set resolution
- 4) If the output resolution of HDMI is set to **2160p24**, **2160p25** or **2160p30**, HDMI needs to be connected to a TV or monitor that supports 4K display to display normally
  - a. When connected to a 4K TV, the display is as follows



- b. If it is connected to a TV or monitor that does not support 4K display, it will not be able to display. As shown in the figure below, the TV connected to 1080p directly displays the format that **does not support**



5) The method of checking the output resolution of the HDMI driver is as follows (the resolution of the HMDI output shown in the figure below is 2160p25). If the displayed resolution is the same as the set resolution, it means that the setting on this side of the development board is correct.

```
root@orangepi:~# cat /sys/class/disp/disp/attr/sys
```

```
root@orangepi:/sys/class/disp/disp/attr# cat sys
screen 0:
de_rate 696000000 hz, ref_fps:25
mgr0: 3840x2160 fmt[rgb] cs[0x0] range[limit] eotf[0x0] bits[8bits] err[0] force_sync[0] unblank direct_show[false]
dmabuf: cache[0] cache_max[0] umap skip[0] overflow[0]
        hdmi output mode[29]    fps:25.2    3840x2160
        err:0 skip1 lrg:69215 vsync:0 vsync_skip:0
        BUF enable ch[1] lyr[0] z[0] prem[N] alglobl 255] fmt[ 0] fb[1920,1080;1920,1080;1920,1080] crop[ 0, 0,1920,1080] frame[ 0, 0,3840,2160] addr[fe000000,
0, 0] flags[0x 0] trd[0,0]
depthf[ 0] transf[0]
```

### 3. 10. 4. How to Modify the Width and Height of Framebuffer in Linux 4.9 System

**Note: This method is only applicable to systems with linux4.9 kernel**

1) There are two variables `fb0_width` and `fb0_height` in `/boot/orangepiEnv.txt` of the linux system, which can be used to set the width and height of the Framebuffer. The default settings of the linux system are `fb0_width=1280`, `fb0_height=720`

```
root@orangepi:~# cat /boot/orangepiEnv.txt
verbosity=1
console=both
disp_mode=1080p60
fb0_width=1280
fb0_height=720
```

2) The **reference values** corresponding to different resolutions of `fb0_width` and



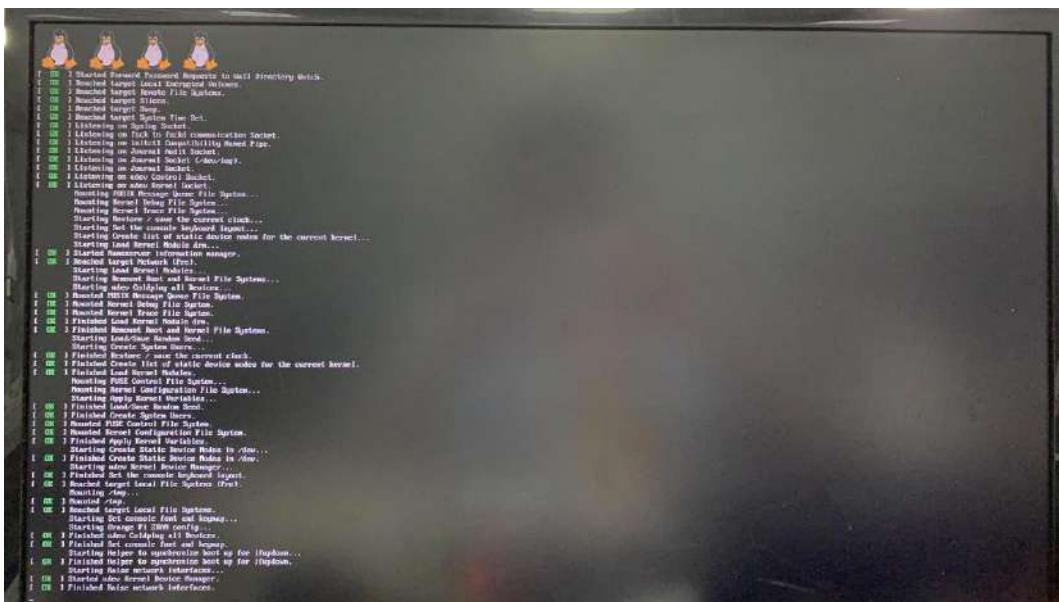
`fb0_height` are as follows. It should be noted that when the resolution of HDMI is set to **2160p24**, **2160p25** and **2160p30**

**Note that when the HDMI resolution is set to 2160p24, 2160p25 and 2160p30, the Framebuffer can only be set to a maximum of 1920x1080**

HDMI Resolution	fb0_width	fb0_height
480p	720	480
576p	720	576
720p	1280	720
1080p	1920	1080
2160p	1920	1080

3) Under the same HDMI resolution, the display conditions of different fb0\_width and fb0\_height are as follows. When the values of fb0\_width and fb0\_height are larger, the text displayed on the screen is smaller. When the values of fb0\_width and fb0\_height are smaller, The larger the text displayed on the screen

- a. HDMI resolution is 1080p60, fb0 width and fb0 height are 1920x1080 display



- b. HDMI resolution is 1080p60, fb0 width and fb0 height are 1280x720 display

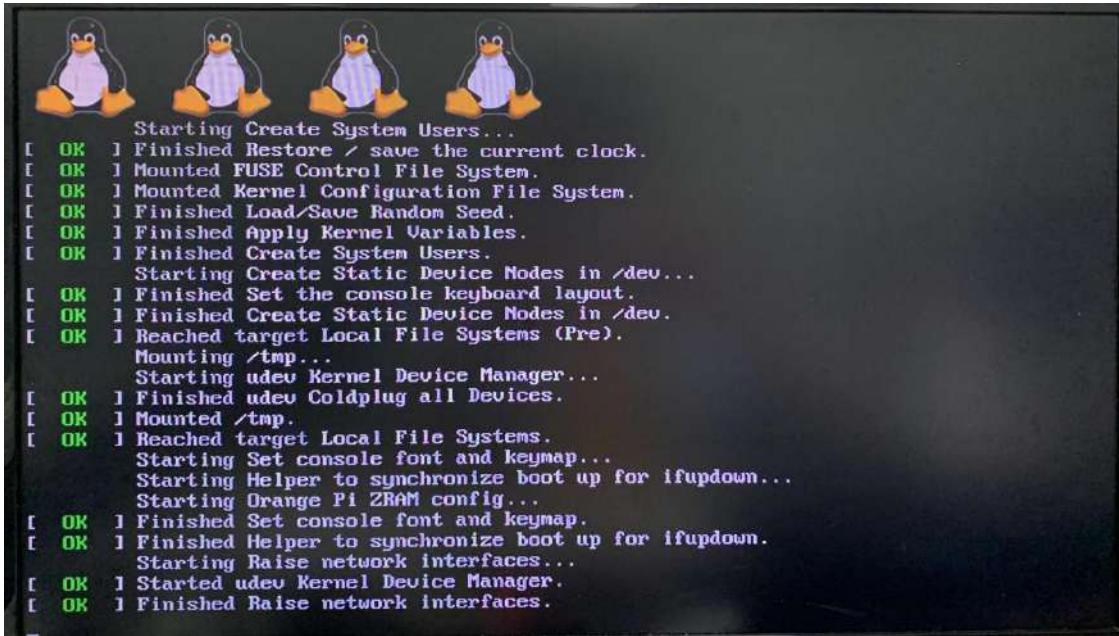


```
[ OK ] Reached target Network (Pre).
[ OK ] Starting Load Kernel Modules...
[ OK ] Starting Remount Root and Kernel File Systems...
[ OK ] Starting udev Coldplug all Devices...
[ OK ] Mounted POSIX Message Queue File System.
[ OK ] Mounted Kernel Debug File System.
[ OK ] Mounted Kernel Trace File System.
[ OK ] Finished Create list of static device nodes for the current kernel.
[ OK ] Finished Load Kernel Module drm.
[ OK ] Finished Remount Root and Kernel File Systems.
[ OK ] Finished Restore / save the current clock.
[ OK ] Finished Load Kernel Modules...
[ OK ] Mounting FUSE Control File System...
[ OK ] Mounting Kernel Configuration File System...
[ OK ] Starting Load/Save Random Seed...
[ OK ] Starting Apply Kernel Variables...
[ OK ] Starting Create System Users...
[ OK ] Mounted FUSE Control File System.
[ OK ] Mounted Kernel Configuration File System.
[ OK ] Finished Load/Save Random Seed.
[ OK ] Finished Create System Users.
[ OK ] Starting Create Static Device Nodes in /dev...
[ OK ] Finished Set the console keyboard layout.
[ OK ] Finished Apply Kernel Variables.
[ OK ] Finished Create Static Device Nodes in /dev.
[ OK ] Reached target Local File Systems (Pre).
[ OK ] Mounting /tmp...
[ OK ] Starting udev Kernel Device Manager...
[ OK ] Finished udev Coldplug all Devices.
[ OK ] Mounted /tmp.
[ OK ] Reached target Local File Systems.
[ OK ] Starting Set console font and keymap...
[ OK ] Starting Orange Pi ZRAM config...
[ OK ] Finished Set console font and keymap.
[ OK ] Finished Helper to synchronize boot up for ifupdown...
[ OK ] Finished Helper to synchronize boot up for ifupdown.
[ OK ] Starting Raise network interfaces...
[ OK ] Started udev Kernel Device Manager...
[ OK ] Finished Raise network interfaces.
```

- c. The display case with HDMI resolution of 1080p60, fb0\_width and fb0\_height of 720x576

```
[ OK ] Finished Create list of static device nodes for the current kernel.
[ OK ] Finished Load Kernel Module drm.
[ OK ] Finished Load Kernel Modules...
[ OK ] Mounting FUSE Control File System...
[ OK ] Mounting Kernel Configuration File System...
[ OK ] Starting Apply Kernel Variables...
[ OK ] Finished Restore / save the current clock.
[ OK ] Finished Remount Root and Kernel File Systems.
[ OK ] Mounted FUSE Control File System.
[ OK ] Mounted Kernel Configuration File System.
[ OK ] Finished Apply Kernel Variables.
[ OK ] Starting Load/Save Random Seed...
[ OK ] Starting Create System Users...
[ OK ] Finished Load/Save Random Seed.
[ OK ] Finished Create System Users.
[ OK ] Starting Create Static Device Nodes in /dev...
[ OK ] Finished Set the console keyboard layout.
[ OK ] Finished Create Static Device Nodes in /dev.
[ OK ] Reached target Local File Systems (Pre).
[ OK ] Mounting /tmp...
[ OK ] Starting udev Kernel Device Manager...
[ OK ] Finished udev Coldplug all Devices.
[ OK ] Mounted /tmp.
[ OK ] Reached target Local File Systems.
[ OK ] Starting Set console font and keymap...
[ OK ] Starting Orange Pi ZRAM config...
[ OK ] Finished Set console font and keymap.
[ OK ] Finished Helper to synchronize boot up for ifupdown...
[ OK ] Finished Helper to synchronize boot up for ifupdown.
[ OK ] Starting Raise network interfaces...
```

- d. The display case with HDMI resolution of 1080p60, fb0\_width and fb0\_height of 720x480



### 3.10.5. Framebuffer cursor related settings

- 1) The softcursor used by Framebuffer, the method to set the cursor to flicker or not to flicker is as follows

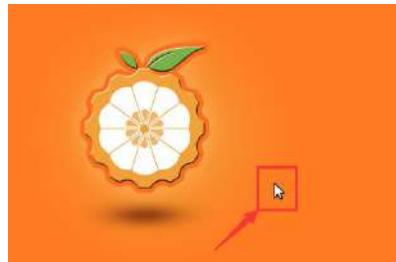
```
root@orangepi:~# echo 1 > /sys/class/graphics/fbcon/cursor_blink      #Cursor blinks  
root@orangepi:~# echo 0 > /sys/class/graphics/fbcon/cursor_blink      #Cursor does not  
                                blink
```

- 2) If you need to hide the cursor, you can add **vt.global\_cursor\_default=0** to the extraargs variable of **/boot/orangepiEnv.txt** (the value of **extraargs** will be assigned to the **bootargs** environment variable and finally passed to the kernel) (if **vt.global\_cursor\_default=1**, it is displayed cursor), then restart the system to see that the cursor has disappeared

```
root@orangepi:~# cat /boot/orangepiEnv.txt  
verbosity=1  
console=both  
disp_mode=1080p60  
fb0_width=1280  
fb0_height=720  
extraargs=cma=25M vt.global_cursor_default=0
```

### 3.10.6. How to hide the mouse cursor on the USB touch screen

- 1) When using the touch screen, if you want to hide the mouse cursor shown in the figure below, you can use the method described in this section



- 2) First open the following configuration file, and then add the configuration of the red font part to it

```
orangeipi@orangeipi:~$ sudo vim /etc/lightdm/lightdm.conf.d/11-orangeipi.conf
[Seat:*)
user-session=xfce
greeter-show-manual-login=false
greeter-hide-users=false
allow-guest=false
xserver-command=X -bs -core -nocursor
```

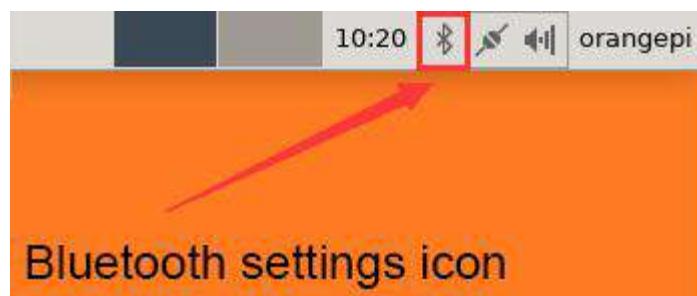
- 3) **Then restart the Linux system**

- 4) Then use the touch screen, you will find that the mouse cursor is gone

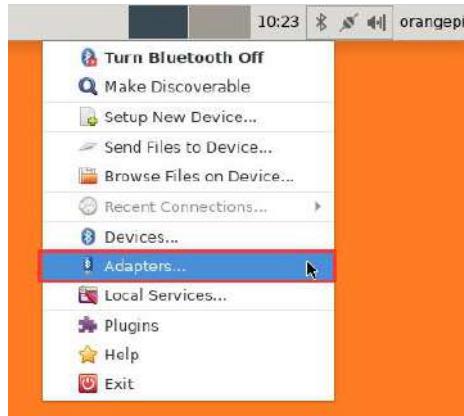
### 3. 11. How to use Bluetooth

#### 3. 11. 1. Test method for desktop image

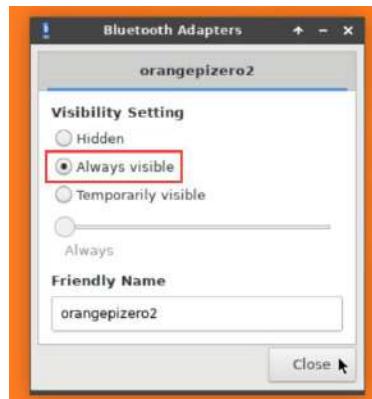
- 1) Click the Bluetooth icon in the upper right corner of the desktop



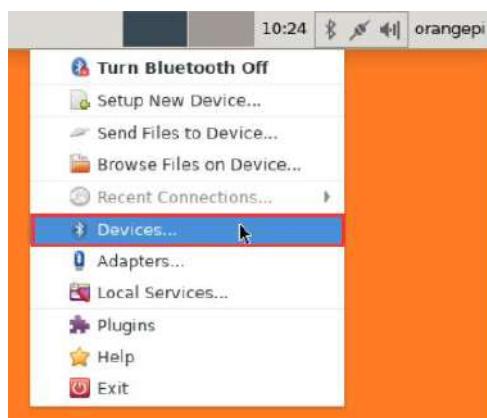
- 2) Then select adapter



- 3) Set the **Visibility Setting** to **Always visible** in the Bluetooth adapter setting interface, and then click close to **close**



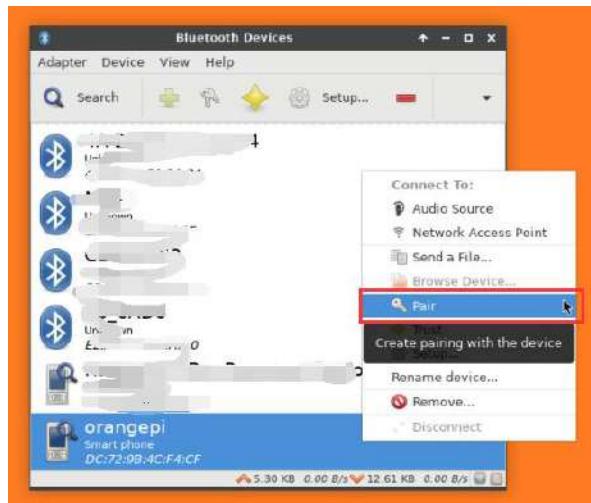
- 4) Then open the configuration interface of the Bluetooth device



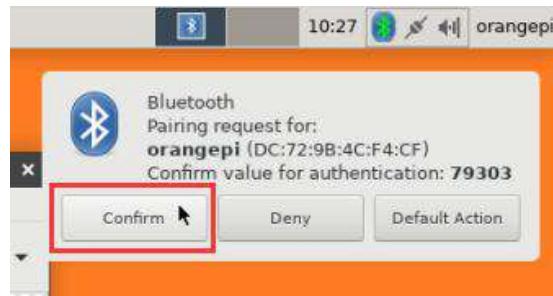
- 5) Click **Search** to start scanning for surrounding Bluetooth devices



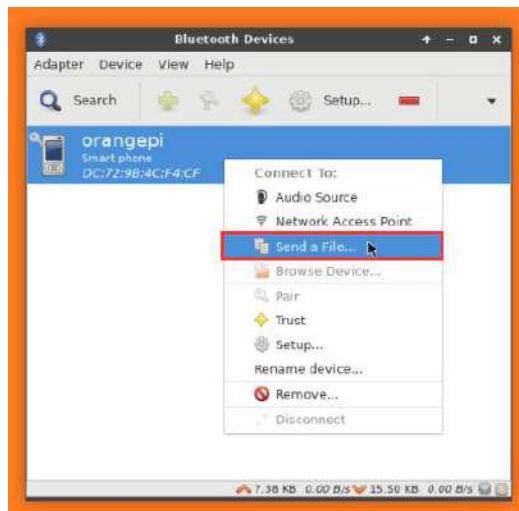
- 6) Then select the Bluetooth device you want to connect, and then click the right mouse button to pop up the operation interface of the Bluetooth device. Select **Pair** to start pairing. The demonstration here is pairing with an Android phone.



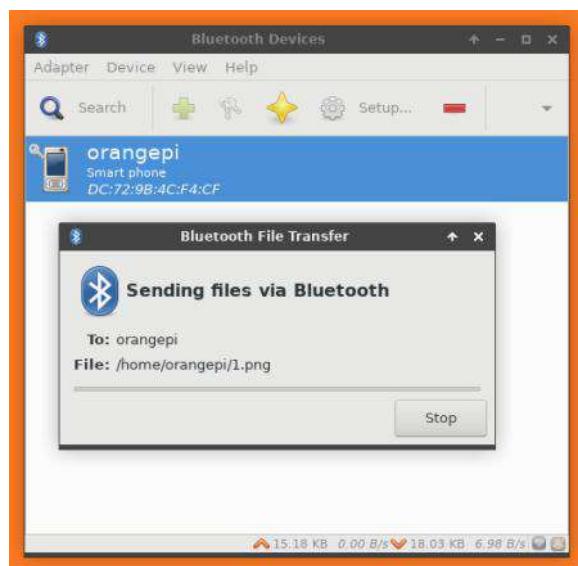
- 7) When pairing, a pairing confirmation box will pop up in the upper right corner of the desktop. Select **Confirm** to confirm. At this time, the mobile phone also needs to be confirmed.



- 8) After pairing with the mobile phone, you can select the paired Bluetooth device, then right-click and select **Send a File** to start sending a picture to the mobile phone



- 9) The interface for sending pictures is as follows





### 3.11.2. How to use the server version image

- 1) After entering the system, you can use the hciconfig command to check whether there is a Bluetooth device node. If there is, it means that the Bluetooth initialization is normal.

```
orangepi@orangepi:~$ sudo apt update && sudo apt install bluez
orangepi@orangepi:~$ hciconfig -a
hci0:  Type: Primary  Bus: UART
          BD Address: 10:11:12:13:14:15  ACL MTU: 1021:8  SCO MTU: 240:3
          UP RUNNING
          RX bytes:646 acl:0 sco:0 events:37 errors:0
          TX bytes:2650 acl:0 sco:0 commands:37 errors:0
          Features: 0xbff 0xff 0x8d 0xfe 0xdb 0x3d 0x7b 0xc7
          Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
          Link policy:
          Link mode: SLAVE ACCEPT
          Name: 'orangepi3-lts'
          Class: 0x000000
          Service Classes: Unspecified
          Device Class: Miscellaneous,
          HCI Version: 5.0 (0x9)  Revision: 0x400
          LMP Version: 5.0 (0x9)  Subversion: 0x400
          Manufacturer: Spreadtrum Communications Shanghai Ltd (492)
```

- 2) Scan for bluetooth devices using bluetoothctl

```
orangepi@orangepi:~$ sudo bluetoothctl
[NEW] Controller 10:11:12:13:14:15 orangepi3-lts [default]
Agent registered
[bluetooth]# power on      #enable controller
Changing power on succeeded
[bluetooth]# discoverable on    #Make the controller discoverable
Changing discoverable on succeeded
[CHG] Controller 10:11:12:13:14:15 Discoverable: yes
[bluetooth]# pairable on     #Set the controller to be pairable
Changing pairable on succeeded
[bluetooth]# scan on       #Start scanning for nearby bluetooth devices
Discovery started
[CHG] Controller 10:11:12:13:14:15 Discovering: yes
```



```
[NEW] Device 76:60:79:29:B9:31 76-60-79-29-B9-31
[NEW] Device 9C:2E:A1:42:71:11 Xiaomi phone
[NEW] Device DC:72:9B:4C:F4:CF orangepi
[bluetooth]# scan off          #After scanning the Bluetooth device you want to connect,
you can close the scan, and then write down the MAC address of the Bluetooth device.
The Bluetooth device tested here is an Android phone, the Bluetooth name is orangepi,
and the corresponding MAC address is DC:72:9B:4C :F4:CF
Discovery stopped
[CHG] Controller 10:11:12:13:14:15 Discovering: no
[CHG] Device DC:72:9B:4C:F4:CF RSSI is nil
```

- 3) After scanning the device you want to pair, you can pair it. Pairing needs to use the MAC address of the device

```
[bluetooth]# pair DC:72:9B:4C:F4:CF      #Use the scanned MAC address of the
Bluetooth device for pairing
Attempting to pair with DC:72:9B:4C:F4:CF
[CHG] Device DC:72:9B:4C:F4:CF Connected: yes
Request confirmation
[leeb1m[agent] Confirm passkey 764475 (yes/no): yes  #Enter yes here, you also need
to confirm on the phone
[CHG] Device DC:72:9B:4C:F4:CF Modalias: bluetooth:v010Fp107Ed1436
[CHG] Device DC:72:9B:4C:F4:CF UUIDs: 0000046a-0000-1000-8000-00805f9b34fb
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: yes
[CHG] Device DC:72:9B:4C:F4:CF Paired: yes
Pairing successful    #Prompt for successful pairing
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: no
[CHG] Device DC:72:9B:4C:F4:CF Connected: no
```

- 4) After the pairing is successful, the display of the Bluetooth interface of the mobile phone is as follows



- 5) To connect a bluetooth device, you need to install the **pulseaudio-module-bluetooth** package, and then start the **pulseaudio** service

```
orangeipi@orangeipi:~$ sudo apt update  
orangeipi@orangeipi:~$ sudo apt -y install pulseaudio-module-bluetooth  
orangeipi@orangeipi:~$ pulseaudio --start
```

- 6) How to connect a Bluetooth device

```
orangeipi@orangeipi:~$ sudo bluetoothctl  
Agent registered  
[bluetooth]# paired-devices      #View the MAC address of a paired Bluetooth device  
Device DC:72:9B:4C:F4:CF orangeipi  
[bluetooth]# connect DC:72:9B:4C:F4:CF      #Connect a Bluetooth device using the  
                                         #MAC address  
Attempting to connect to DC:72:9B:4C:F4:CF  
[CHG] Device DC:72:9B:4C:F4:CF Connected: yes  
Connection successful  
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: yes  
[CHG] Controller 10:11:12:13:14:15 Discoverable: no  
[orangeipi]#      #This prompt appears to indicate that the connection is successful
```

- 7) After connecting the bluetooth device, the bluetooth configuration interface of the Android phone can see the prompt of the **connected audio for calls and media**

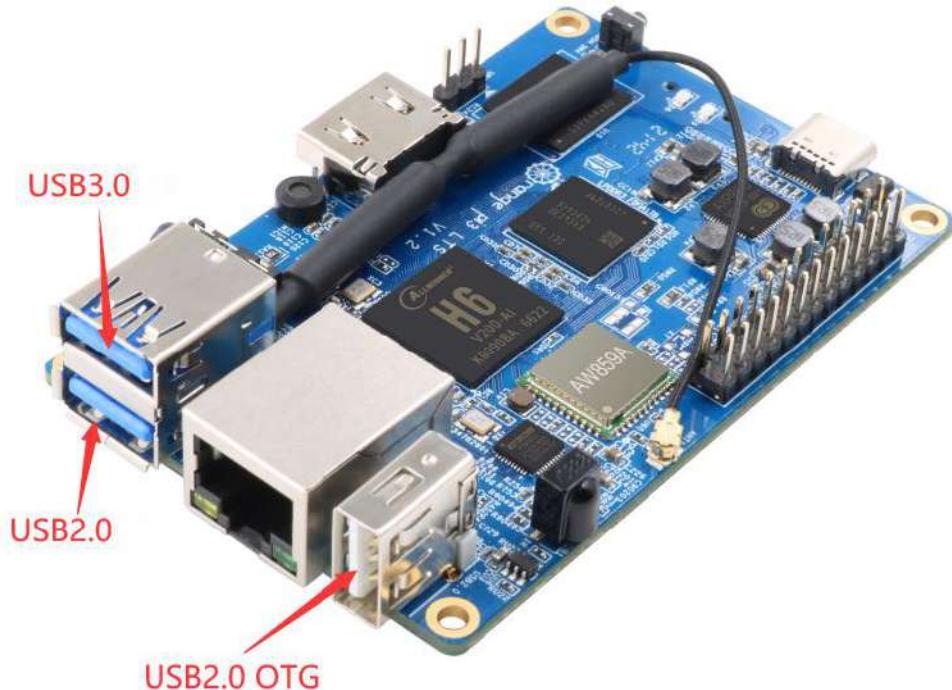


### 3. 12. USB interface test

The three USB ports of the development board can be connected to a USB hub to expand the number of USB ports.

#### 3. 12. 1. USB2.0 and USB3.0 interface description

1) Orange Pi 3 LTS has a total of 3 USB interfaces, including a USB3.0, a USB2.0 and a USB2.0 OTG interface, of which the USB2.0 OTG interface defaults to Host mode, which can be connected to a mouse, keyboard or U disk.



Although the USB interface in the lower left corner is blue, it only has the function of USB2.0, please pay special attention to this.



### 3. 12. 2. Connect the mouse or keyboard to test

- 1) Insert the keyboard of the USB interface into the USB interface of the Orange Pi development board
- 2) When using the mouse, you need to connect the Orange Pi development board to the HDMI display

3) If the mouse or keyboard can operate normally, the USB interface is in normal use (the mouse can only be used in the desktop version of the system)

### 3. 12. 3. Connect USB storage device test

- 1) First, insert the U disk or USB mobile hard disk into the USB interface of the Orange Pi development board
- 2) Execute the following command, if you can see the output of sdX, it means that the U disk is successfully recognized

```
orangepi@orangepi:~$ cat /proc/partitions | grep "sd*"
major minor #blocks name
 8        0   30044160 sda
 8        1   30043119 sda1
```

3) Use the mount command to mount the U disk to **/mnt**, and then you can view the files in the U disk

```
orangepi@orangepi:~$ sudo mount /dev/sda1 /mnt/
orangepi@orangepi:~$ ls /mnt/
test.txt
```

To mount a U disk in **exfat** format in Linux system, you can use the following command

```
orangepi@orangepi:~$ sudo apt-get install exfat-utils exfat-fuse
orangepi@orangepi:~$ sudo mount -t exfat /dev/sda1 /mnt/
```

4) After mounting, you can view the capacity usage and mount point of the U disk through the **df -h** command

```
orangepi@orangepi:~$ df -h | grep "sd"
/dev/sda1      29G  208K  29G  1% /mnt
```



### 3. 12. 4. USB wireless network card test

**Note: This method is only applicable to the system with the linux5.1x kernel, and the system with the linux4.9 kernel is not debugged.**

The available USB wireless network cards **tested** by the Linux 5.10 system are as follows. Please test other types of USB wireless network cards yourself. If they cannot be used, you need to transplant the corresponding USB wireless network card driver.

serial number	model
1	RTL8723BU
2	RTL8821CU

#### 3. 12. 4. 1. RTL8723BU test

- 1) First insert the RTL8723BU wireless network card module into the USB interface of the development board
- 2) Then the linux system will automatically load the RTL8723BU related kernel modules, you can see the following output through the lsmod command

```
root@orangepi:~# lsmod | grep "rtl8"
rtl8xxxu           118784  0
mac80211          503808  1 rtl8xxxu
```

- 3) You can see the loading information of the RTL8723BU module through the dmesg command

```
root@orangepi:~# dmesg | tail
[ 1583.908354] usb 3-1.1: 1e0: ff ff ff ff ff ff ff ff
[ 1583.908357] usb 3-1.1: 1e8: ff ff ff ff ff ff ff
[ 1583.908360] usb 3-1.1: 1f0: ff ff ff ff ff ff ff
[ 1583.908363] usb 3-1.1: 1f8: ff ff ff ff ff ff ff
[ 1583.908369] usb 3-1.1: RTL8723BU rev E (SMIC) 1T1R, TX queues 3, WiFi=1,
BT=1, GPS=0, HI PA=0
[ 1583.908373] usb 3-1.1: RTL8723BU MAC: 00:13:ef:f4:58:ae
[ 1583.908377] usb 3-1.1: rtl8xxxu: Loading firmware rtlwifi/rtl8723bu_nic.bin
[ 1583.908536] usb 3-1.1: Firmware revision 35.0 (signature 0x5301)
[ 1584.013655] Bluetooth: hci1: RTL: fw version 0x1e4cc3ff
```



```
[ 1584.813524] rtl8xxxu 3-1.1:1.2 wlx0013eff458ae: renamed from wlan1
```

- 4) Then you can see the device node of RTL8723BU WIFI through the ifconfig command. For the **connection and test method of WIFI**, please refer to the section on WIFI connection test, which will not be repeated here.

```
root@orangeipi:~# ifconfig wlx0013eff458ae
wlx0013eff458ae: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
          ether 00:13:ef:f4:58:ae  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
```

- 5) Then you can see two Bluetooth devices through the hciconfig command. The node whose Bus type is USB is the Bluetooth node of RTL8723BU. **For the Bluetooth test method**, please refer to the Bluetooth usage section, which will not be repeated here.

```
root@orangeipi:~# hciconfig
hci1:  Type: Primary  Bus: USB
        BD Address: 00:13:EF:F4:58:AE  ACL MTU: 1021:8  SCO MTU: 255:16
        UP RUNNING PSCAN ISCAN
        RX bytes:3994 acl:0 sco:0 events:206 errors:0
        TX bytes:29459 acl:0 sco:0 commands:173 errors:0

hci0:  Type: Primary  Bus: UART
        BD Address: 10:11:12:13:14:15  ACL MTU: 1021:8  SCO MTU: 240:3
        UP RUNNING
        RX bytes:2981 acl:0 sco:0 events:127 errors:0
        TX bytes:5423 acl:0 sco:0 commands:61 errors:0
```

### 3. 12. 4. 2. RTL8821CU test

- 1) First insert the RTL8821CU wireless network card module into the USB interface of the development board
- 2) Then the linux system will automatically load the RTL8821CU-related kernel



modules, and you can see the following output through the lsmod command

```
root@orangepi:~# lsmod | grep "8821"
8821cu           1941504  0
cfg80211        356352   3 8821cu,brcmfmac,mac80211
```

3) You can see the loading information of the RTL8821CU module through the dmesg command

```
root@orangepi:~# dmesg | tail
[ 1835.290228] usb 3-1.1: Product: 802.11ac NIC
[ 1835.290239] usb 3-1.1: Manufacturer: Realtek
[ 1835.290249] usb 3-1.1: SerialNumber: 123456
[ 1835.371795] Bluetooth: hci1: RTL: examining hci_ver=08 hci_rev=000c lmp_ver=08
lmp_subver=8821
[ 1835.372753] Bluetooth: hci1: RTL: rom_version status=0 version=1
[ 1835.372771] Bluetooth: hci1: RTL: loading rtl_bt/rtl8821c_fw.bin
[ 1835.373136] Bluetooth: hci1: RTL: loading rtl_bt/rtl8821c_config.bin
[ 1835.373349] Bluetooth: hci1: RTL: cfg_sz 10, total sz 21678
[ 1835.784650] Bluetooth: hci1: RTL: fw version 0x826ca99e
[ 1835.816409] rtl8821cu 3-1.1:1.2 wlxd0c0bf8742cd: renamed from wlan1
```

4) Then you can see the device node of RTL8821CU WIFI through the ifconfig command. For the [connection and test method of WIFI](#), please refer to the section on WIFI connection test, which will not be repeated here.

```
root@orangepi:~# ifconfig wlxd0c0bf8742cd
wlxd0c0bf8742cd: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
          ether d0:c0:bf:87:42: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
```

5) Then you can see two Bluetooth devices through the hciconfig command. The node whose Bus type is USB is the Bluetooth node of RTL8821CU. For the Bluetooth test method, please refer to the [Bluetooth usage section](#), which will not be repeated here.

```
root@orangepi:~# hciconfig
```



```
hci1: Type: Primary Bus: USB
      BD Address: D0:C0:BF:87:42:CE  ACL MTU: 1021:8  SCO MTU: 255:12
      UP RUNNING
      RX bytes:1343 acl:0 sco:0 events:137 errors:0
      TX bytes:24227 acl:0 sco:0 commands:137 errors:0

hci0: Type: Primary Bus: UART
      BD Address: 43:45:C5:00:1F:AC  ACL MTU: 1021:8  SCO MTU: 64:1
      UP RUNNING
      RX bytes:1941 acl:0 sco:0 events:66 errors:0
      TX bytes:2816 acl:0 sco:0 commands:66 errors:0
```

### 3. 12. 5. USB network card test

- 1) The currently **tested and usable** USB network cards are as follows

serial number	model
1	RTL8152B USB 100M Ethernet
2	RTL8153 USB 1000M Ethernet

- 2) First insert the USB network card into the USB port of the development board, and then insert the network cable into the USB network card to ensure that the network cable can access the Internet normally. If you can see the following log information through the dmesg command, it means that the USB network card is recognized normally.

```
root@orangepi:~# dmesg | tail
[ 121.985016] usb 3-1: USB disconnect, device number 2
[ 126.873772] sunxi-ehci 5311000.ehci3-controller: ehci_irq: highspeed device connect
[ 127.094054] usb 3-1: new high-speed USB device number 3 using sunxi-ehci
[ 127.357472] usb 3-1: reset high-speed USB device number 3 using sunxi-ehci
[ 127.557960] r8152 3-1:1.0 eth1: v1.08.9
[ 127.602642] r8152 3-1:1.0 enx00e04c362017: renamed from eth1
[ 127.731874] IPv6: ADDRCONF(NETDEV_UP): enx00e04c362017: link is not ready
[ 127.763031] IPv6: ADDRCONF(NETDEV_UP): enx00e04c362017: link is not ready
[ 129.892465] r8152 3-1:1.0 enx00e04c362017: carrier on
[ 129.892583] IPv6: ADDRCONF(NETDEV_CHANGE): enx00e04c362017: link becomes ready
```

- 3) Then you can see the device node of the USB network card and the automatically



assigned IP address through the ifconfig command

```
root@orangepi:~# ifconfig
enx00e04c362017: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>      mtu
1500
          inet 192.168.1.177  netmask 255.255.255.0  broadcast 192.168.1.255
              inet6 fe80::681f:d293:4bc5:e9fd  prefixlen 64  scopeid 0x20<link>
                ether 00:e0:4c:36:20:17  txqueuelen 1000  (Ethernet)
                  RX packets 1849  bytes 134590 (134.5 KB)
                  RX errors 0  dropped 125  overruns 0  frame 0
                  TX packets 33  bytes 2834 (2.8 KB)
                  TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4) The command to test network connectivity is as follows

```
root@orangepi:~# ping www.baidu.com -I enx00e04c362017
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=56 time=6.74 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=56 time=6.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=56 time=6.26 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=56 time=7.27 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 6.260/6.770/7.275/0.373 ms
```

### 3. 12. 6. USB camera test

- 1) First insert the USB camera into the USB port of the Orange Pi development board
- 2) Then through the lsmod command, you can see that the kernel automatically loads the following modules
  - a. linux4.9 The system displays as follows

```
orangepi@orangepi:~$ lsmod | grep "uvc"
Module           Size  Used by
uvcvideo        106496  0
```

- b. linux5.1x The system displays as follows
- ```
orangepi@orangepi:~$ lsmod | grep "uvc"
uvcvideo        102400  0
```



|                                                          |               |          |                                                 |
|----------------------------------------------------------|---------------|----------|-------------------------------------------------|
| <b>videobuf2_vmalloc</b>                                 | <b>20480</b>  | <b>1</b> | <b>uvcvideo</b>                                 |
| <b>videobuf2_v4l2</b>                                    | <b>28672</b>  | <b>1</b> | <b>uvcvideo</b>                                 |
| <b>videobuf2_common</b>                                  | <b>53248</b>  | <b>2</b> | <b>videobuf2_v4l2,uvcvideo</b>                  |
| <b>videodev</b>                                          | <b>233472</b> | <b>3</b> | <b>videobuf2_v4l2,uvcvideo,videobuf2_common</b> |
| <b>mc</b>                                                | <b>49152</b>  | <b>4</b> |                                                 |
| <b>videodev,videobuf2_v4l2,uvcvideo,videobuf2_common</b> |               |          |                                                 |

- 3) Through the v4l2-ctl command, you can see that the device node information of the USB camera is /dev/video0

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y v4l-utils
orangepi@orangepi:~$ v4l2-ctl --list-devices
USB 2.0 Camera (usb-xhci-hcd.0.auto-1.2):
    /dev/video0
```

**Note that the l in v4l2 is a lowercase l, not the number 1.**

**In addition, the serial numbers of videos are not necessarily all video0, please refer to the actual ones.**

- 4) Use fswebcam to test the USB camera

- a. Install fswebcam

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt-get install -y fswebcam
```

- b. After installing fswebcam, you can use the following command to take pictures

- a) The -d option is used to specify the device node of the USB camera
- b) --no-banner is used to remove the watermark of the photo
- c) -r Option to specify the resolution of the photo
- d) -S Option to set to skip previous frames
- e) ./image.jpg Used to set the name and path of the generated photo

```
orangepi@orangepi:~$ fswebcam -d /dev/video0 --no-banner -r 1280x720 -S \
5 ./image.jpg
```

- c. In the server version of the Linux system, after taking the photo, you can use the scp command to transfer the captured image to the Ubuntu PC for image viewing

```
orangepi@orangepi:~$ scp image.jpg test@192.168.1.xx:/home/test (Modify the IP)
```

**address and path according to the actual situation)**

- d. In the desktop version of the Linux system, you can directly view the captured pictures through the HDMI display

## 5) Use motion to test the USB camera

- a. Install the camera test software motion

```
orangeipi@orangeipi:~$ sudo apt update  
orangeipi@orangeipi:~$ sudo apt install -y motion
```

- b. Modify the configuration of **/etc/default/motion** and modify **start\_motion\_daemon=no** to **start\_motion\_daemon=yes**

**Note that Ubuntu22.04 does not need to set this step.**

```
orangeipi@orangeipi:~$ sudo sed -i '\  
"s/start_motion_daemon=no/start_motion_daemon=yes/"' \/  
/etc/default/motion
```

- c. Modify the configuration of **/etc/motion/motion.conf**

```
orangeipi@orangeipi:~$ sudo sed -i '\  
"s/stream_localhost on/stream_localhost off/"' \/  
/etc/motion/motion.conf
```

- d. In addition, make sure that the videodevice of **/etc/motion/motion.conf** is set to the device node corresponding to the USB camera

**Note that the serial numbers of videos are not necessarily all video0, please refer to what you actually see.**

```
orangeipi@orangeipi:~$ sudo vim /etc/motion/motion.conf  
# Video device (e.g. /dev/video0) to be used for capturing.  
videodevice /dev/video0
```

- e. Then run motion

```
orangeipi@orangeipi:~$ sudo motion -b
```

- f. Before using motion, please make sure that the Orange Pi development board can connect to the network normally, and then obtain the IP address of the development board through the ifconfig command
- g. Then enter **[development board IP address: 8081]** in the Ubuntu PC or Windows PC or Firefox browser on the same local area network as the development board to see the video output by the camera



6) Use mjpg-streamer to test the USB camera

a. Download mjpg-streamer

a) Github download address:

```
orangeipi@orangeipi:~$ git clone https://github.com/jacksonliam/mjpg-streamer
```

b) The image download address of Gitee is:

```
orangeipi@orangeipi:~$ git clone https://gitee.com/leeboby/mjpg-streamer
```

b. Install dependent packages

a) Ubuntu system

```
orangeipi@orangeipi:~$ sudo apt-get install -y cmake libjpeg8-dev
```

b) Debian system

```
orangeipi@orangeipi:~$ sudo apt-get install -y cmake libjpeg62-turbo-dev
```

c. Compile and install mjpg-streamer

```
orangeipi@orangeipi:~$ cd mjpg-streamer/mjpg-streamer-experimental
```

```
orangeipi@orangeipi:~/mjpg-streamer/mjpg-streamer-experimental$ make -j4
```

```
orangeipi@orangeipi:~/mjpg-streamer/mjpg-streamer-experimental$ sudo make install
```

d. Then enter the following command to start mjpg\_streamer

**Note that the serial numbers of videos are not necessarily all video0, please refer to what you actually see.**

```
orangeipi@orangeipi:~/mjpg-streamer/mjpg-streamer-experimental$ export \
```

```
LD_LIBRARY_PATH=.
```

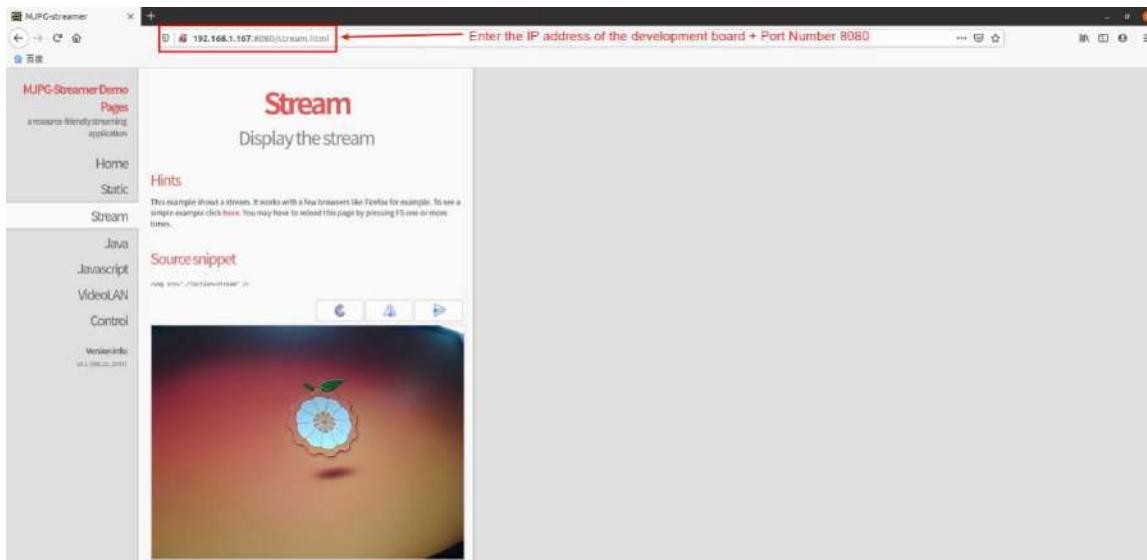
```
orangeipi@orangeipi:~/mjpg-streamer/mjpg-streamer-experimental$ sudo \
```

```
./mjpg_streamer -i "./input_uvc.so -d /dev/video0 -u -f 30" \
```

```
-o "./output_http.so -w ./www"
```



- e. Then enter [**IP address of the development board: 8080**] in the browser of the Ubuntu PC or Windows PC or mobile phone on the same LAN as the development board to see the video output by the camera.



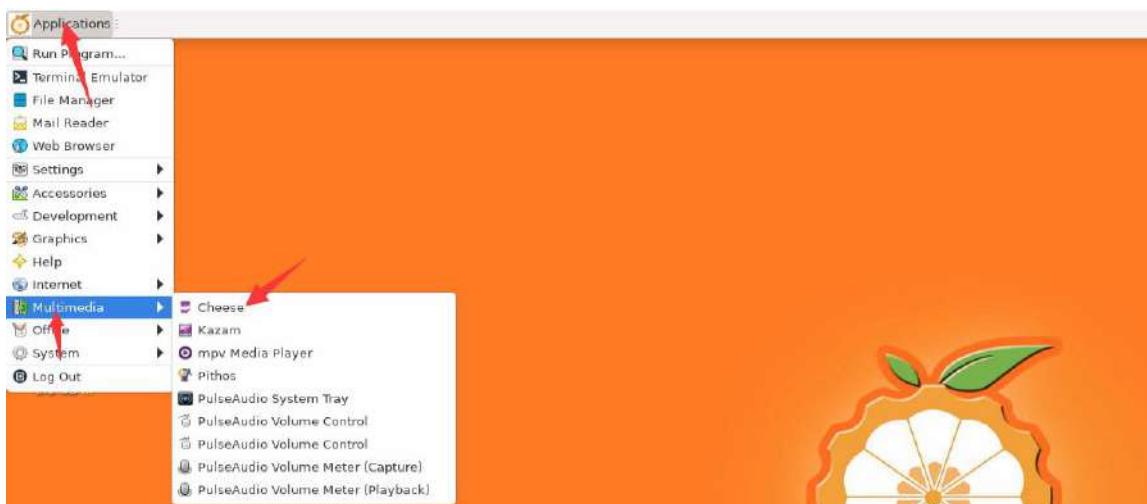
- f. It is recommended to use mjpg-streamer to test the USB camera, which is much smoother than motion, and you will not feel any lag when using mjpg-streamer

## 7) How to use Cheese to test USB camera

- First, you need to ensure that the Ubuntu or Debian system used by the development board is the **desktop version** of the system
- Then use the following command to install Cheese

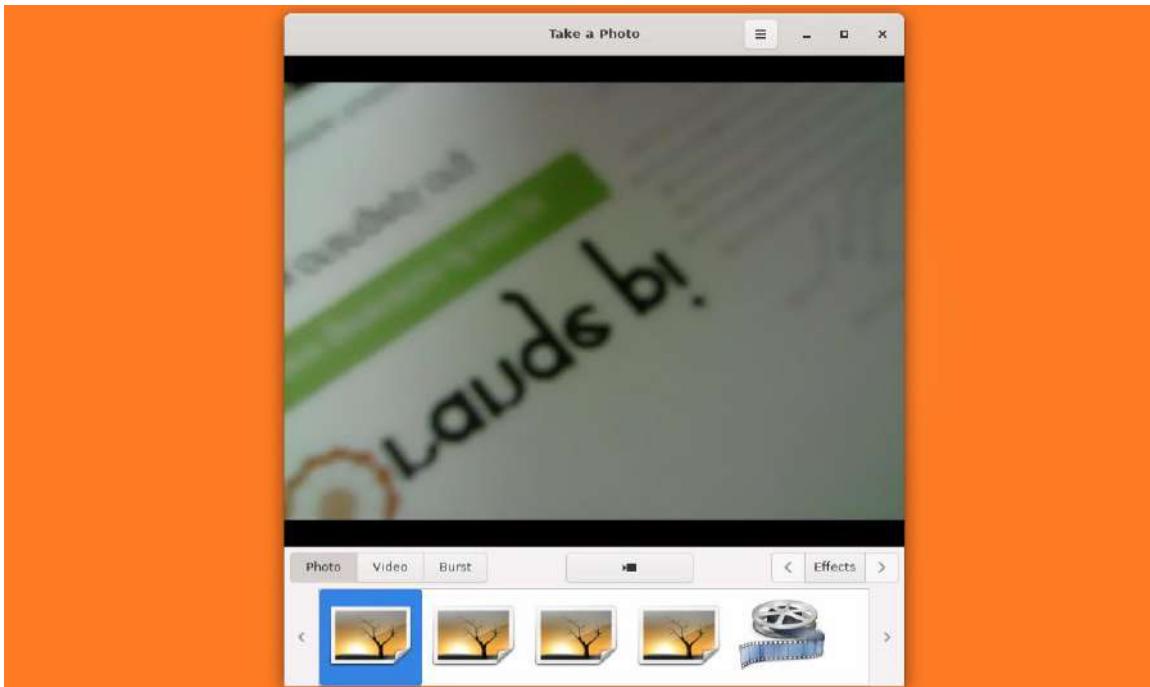
```
orangepi@orangepi:~$ sudo apt-get update  
orangepi@orangepi:~$ sudo apt-get -y install cheese
```

- Then open Cheese on the desktop



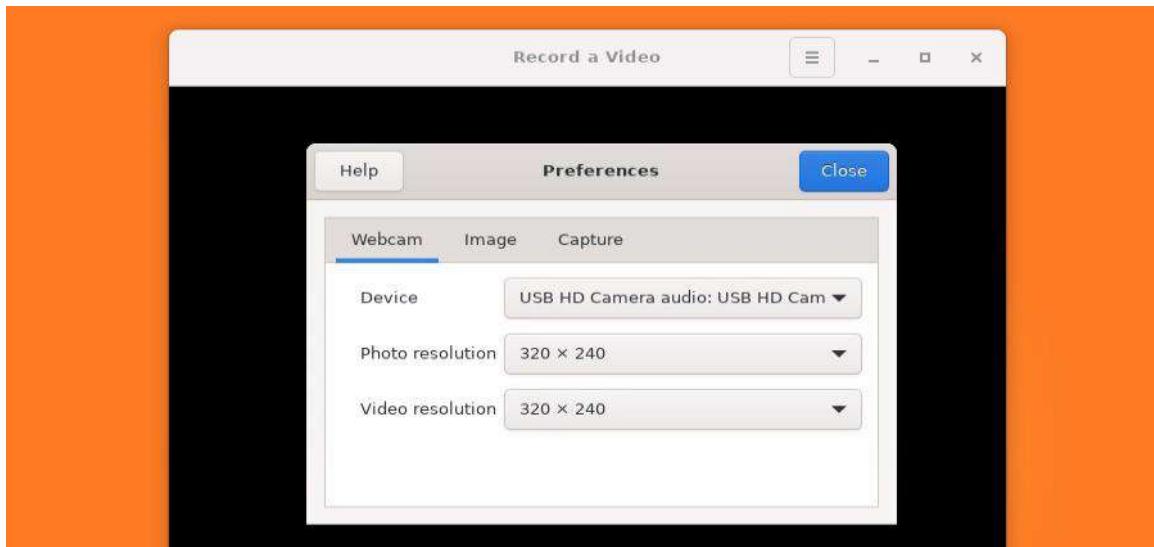


- d. After d.Cheese is opened, you can see the preview screen of the USB camera without other settings



- e. You can set the camera resolution and other parameters in **Preferences**. The lower the resolution, the smoother the display will be.





### 3.13. Audio Test

#### 3.13.1. How to use the command line to play audio

##### 3.13.1.1. Headphone jack audio playback test

- 1) First insert headphones into the audio interface
- 2) You can view the sound card devices supported by the Linux system through the **aplay -l** command

a. The output of a.linux4.9 system is as follows

```
root@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: sndhdmi [sndhdmi], device 0: SUNXI-HDMIAUDIO audiohdmi-dai-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: sndacx00codec [sndacx00-codec], device 0: SUNXI-AUDIO acx00-dai-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

b. The output of the Linux5.1x system is as follows

```
root@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: allwinnerac200c [allwinner,ac200-codec], device 0: 508f000.i2s-acx00-dai
```



**acx00-codec-0 [508f000.i2s-acx00-dai acx00-codec-0]**

**Subdevices: 1/1**

**Subdevice #0: subdevice #0**

card 1: sun50ih6hdmi [sun50i-h6-hdmi], device 0: 5091000.i2s-i2s-hifi i2s-hifi-0 [5091000.i2s-i2s-hifi i2s-hifi-0]

Subdevices: 1/1

Subdevice #0: subdevice #0

- 3) Then you can use the aplay command to play the audio test file that comes with the system. If the headset can hear the sound, it means the audio playback is normal

a. The playback command of Linux4.9 is as follows

```
root@orangeipi:~# aplay -D hw:1,0 /usr/share/sounds/alsa/audio.wav
```

```
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

b. The playback command of Linux5.1x is as follows

```
root@orangeipi:~# aplay -D hw:0,0 /usr/share/sounds/alsa/audio.wav
```

```
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

### 3. 13. 1. 2. HDMI audio playback test

- 1) First connect the Orange Pi development board to the TV with an HDMI cable (other HDMI monitors need to ensure that they can play audio)

- 2) HDMI audio playback does not require other settings, just use the aplay command to play audio to hear the sound

a. Linux4.9

```
root@orangeipi:~# aplay -D hw:0,0 /usr/share/sounds/alsa/audio.wav
```

b. Linux5.1x

```
root@orangeipi:~# aplay -D hw:1,0 /usr/share/sounds/alsa/audio.wav
```

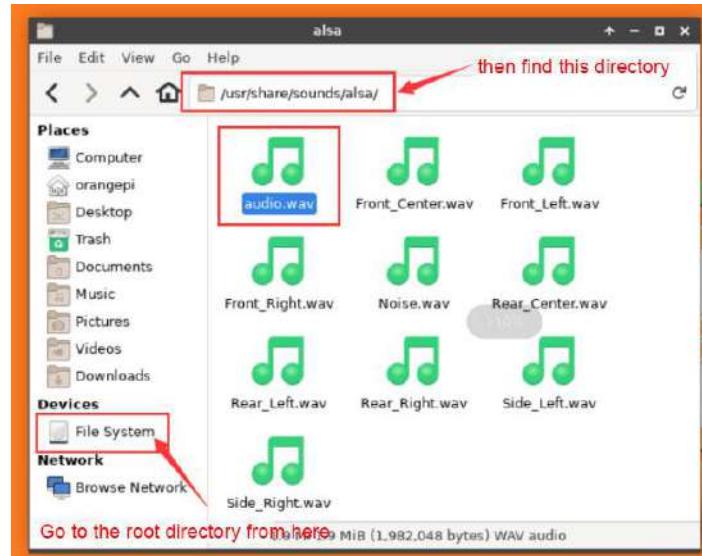
### 3. 13. 2. Testing the audio method in the desktop system

- 1) First open the file manager

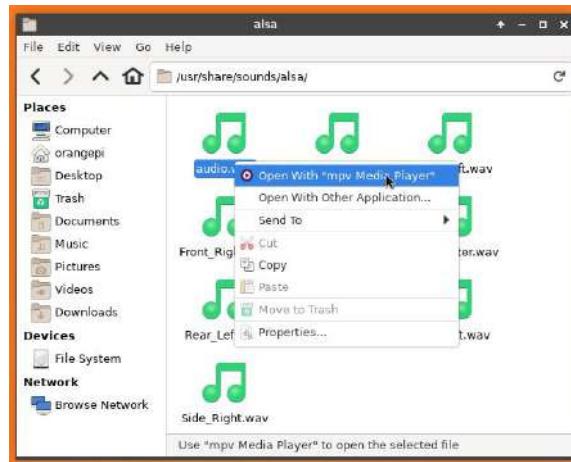




- 2) Then find the following file (if the audio file does not exist in the system, you can upload an audio file to the system by yourself)

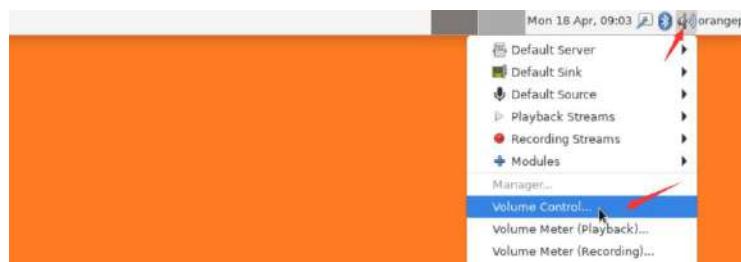


- 3) Then select the audio.wav file, right-click and select Open with mpv to start playing

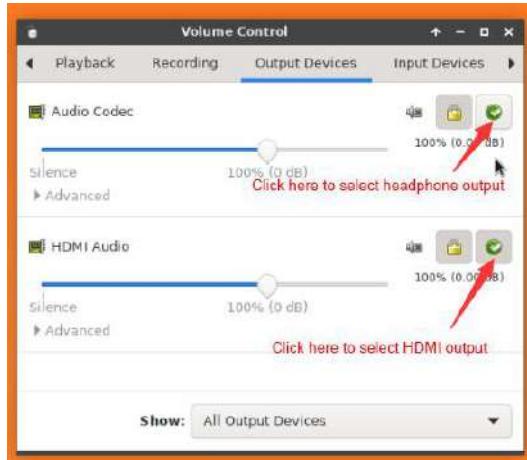


- 4) How to switch between HDMI playback and headphone playback

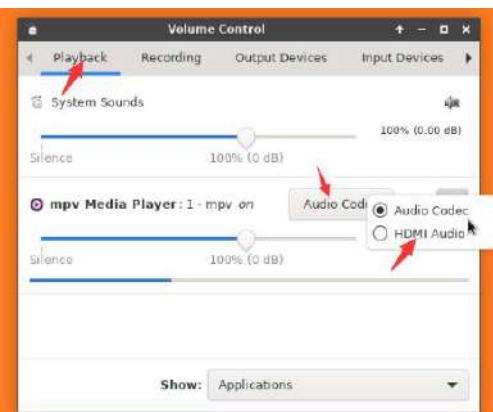
- a. First open the volume control interface



- b. Select **Output Devices**, then you can select the output audio device



- c. In addition, when playing audio, the audio setting options of the playback software will be displayed in **Playback**, as shown in the figure below, you can also set which audio device to play to here.



### 3. 13. 3. MIC recording test

1) The recording command is as follows

a. Linux4.9

```
root@orangepi:~# arecord -D hw:1,0 -d 5 -f cd -t wav test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

b. Linux5.1x

```
root@orangepi:~# arecord -D hw:0,0 -d 5 -f cd -t wav test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

2) After the recording is completed, a recording file named test.wav will be generated in the current path. Use the aplay command to play test.wav to check whether there is sound output. If there is sound output, the recording is normal.

```
root@orangepi:~# ls -lh
```



```
total 864K  
-rw-r--r-- 1 root root 862K Dec 1 06:25 test.wav
```

### 3. 14. Infrared receiving test

- 1) Install ir-keytable infrared test software

```
root@orangepi:~$ sudo apt update  
root@orangepi:~$ sudo apt-get install -y ir-keytable
```

- 2) Then execute ir-keytable to view the information of infrared devices

- a. linux4.9 system output is as follows

```
root@orangepi:~$ ir-keytable  
Found /sys/class/rc/rc0/ (/dev/input/event4) with:  
    Driver sunxi-rc-recv, table rc_map_sunxi  
    Supported protocols: nec  
    Enabled protocols: nec  
    Name: sunxi-ir  
    bus: 25, vendor/product: 0001:0001, version: 0x0100  
    Repeat delay = 500 ms, repeat period = 125 ms
```

- b. linux5.1x system output is as follows

```
root@orangepi:~$ ir-keytable  
Found /sys/class/rc/rc0/ (/dev/input/event1) with:  
    Name: sunxi-ir  
    Driver: sunxi-ir, table: rc-empty  
    LIRC device: /dev/lirc0  
    Attached BPF protocols: Operation not supported  
    Supported kernel protocols: other lirc rc-5 rc-5-sz jvc sony nec sanyo mce_kb  
    Enabled kernel protocols: lirc  
    bus: 25, vendor/product: 0001:0001, version: 0x0100  
    Repeat delay = 500 ms, repeat period = 125 ms
```

- 3) Before testing the infrared reception function, you need to prepare an infrared remote control (**note: only the remote control provided by the orange pi is supported, and the remote control of other TVs or air conditioners cannot be used...**)



- 4) Then enter the **ir-keytable -t** command in the terminal, and then use the infrared remote control to press the button on the infrared receiving head of the Orange Pi development board to see the received key code in the terminal

a. linux4.9 system output is as follows

```
root@orangepi:/# ir-keytable -t
```

Testing events. Please, press CTRL-C to abort.

1598339152.260376: event type EV\_MSC(0x04): scancode = 0xfb0413

1598339152.260376: event type EV\_SYN(0x00)

1598339152 914715: event type EV\_MSC(0x04): scancode = 0xfb0410

b linux5.1x system output is as follows

```
root@orangepi:~# ir-keytable -c -n NEC -t
```

| Old keytable cleared

#### Protocols changed to nec

Testing events. Please press CTRL-C to abort

3161 772262: lirc protocol(nec): scancode == 0x45c

3161 772323: event type EV\_MSC(0x04); scancode = 0x45c

3161 772323: event type EV\_SYN(0x00)

### 3.15. Temperature sensor

The displayed temperature value needs to be divided by 1000, the unit is Celsius.

- 1) H6 has a total of 2 temperature sensors, the command to view the temperature is as follows

a sensor0: CPU



```
orangeipi@orangeipi:~$ cat /sys/class/thermal/thermal_zone0/type  
cpu_thermal_zone  
orangeipi@orangeipi:~$ cat /sys/class/thermal/thermal_zone0/temp  
57734
```

b. sensor1: GPU

```
orangeipi@orangeipi:~$ cat /sys/class/thermal/thermal_zone1/type  
gpu_thermal_zone  
orangeipi@orangeipi:~$ cat /sys/class/thermal/thermal_zone1/temp  
57410
```

### 3. 16. How to install Docker

Docker official provided installation documentation link is as follows:

**Debian system:** <https://docs.docker.com/engine/install/debian/>  
**Ubuntu system:** <https://docs.docker.com/engine/install/ubuntu/>

1) The old version of the Docker installation package is called docker, docker.io or docker-engine. If these packages are installed, they need to be uninstalled first. The command is as follows:

```
orangeipi@orangeipi:~$ sudo apt-get remove -y docker docker-engine docker.io \  
containerd runc
```

2) Then add the official docker software repository

a. The commands used by the Debian system are as follows

```
orangeipi@orangeipi:~$ sudo apt update  
orangeipi@orangeipi:~$ sudo apt-get install -y ca-certificates curl gnupg lsb-release  
orangeipi@orangeipi:~$ curl -fsSL https://download.docker.com/linux/debian/gpg | \  
sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg  
orangeipi@orangeipi:~$ echo "deb [arch=$(dpkg --print-architecture) \  
signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \  
https://download.docker.com/linux/debian \  
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

b. The command used by the Ubuntu system is as follows

```
orangeipi@orangeipi:~$ sudo apt update
```



```
orangeipi@orangeipi:~$ sudo apt-get install -y ca-certificates curl gnupg lsb-release
orangeipi@orangeipi:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
orangeipi@orangeipi:~$ echo "deb [arch=$(dpkg --print-architecture) \
signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

3) Then install Docker Engine

```
orangeipi@orangeipi:~$ sudo apt-get update
orangeipi@orangeipi:~$ sudo apt-get install -y docker-ce docker-ce-cli containerd.io
```

**Note: If an error is reported after Debian Buster is installed, please enter the following command to solve it:**

```
orangeipi@orangeipi:~$ echo 1 | update-alternatives --config iptables > /dev/null
orangeipi@orangeipi:~$ sudo systemctl restart docker
```

4) Then you can add the current user to the docker user group, so that you can run docker commands without sudo

```
orangeipi@orangeipi:~$ sudo usermod -aG docker $USER
```

**Note: You need to log out and log in again for the system to take effect, or restart the system.**

5) Verify the status of docker

```
orangeipi@orangeipi:~$ systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2020-08-24 10:29:22 UTC; 26min ago
    Docs: https://docs.docker.com
   Main PID: 3145 (dockerd)
      Tasks: 15
     CGroup: /system.slice/docker.service
             └─3145 /usr/bin/dockerd -H fd://
```



```
--containerd=/run/containerd/containerd.sock
```

- 6) You can use the following command to test docker, if you can run hello-world, it means that docker can be used normally

```
orangeipi@orangeipi:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
256ab8fe8778: Pull complete
Digest:
sha256:7f0a9f93b4aa3022c3a4c147a449ef11e0941a1fd0bf4a8e6c9408b2600777c5
Status: Downloaded newer image for hello-world:latest
```

**Hello from Docker!**

**This message shows that your installation appears to be working correctly.**

- 7) Method of setting docker warehouse as domestic source

- Create the **/etc/docker/daemon.json** file and add the following configuration to it

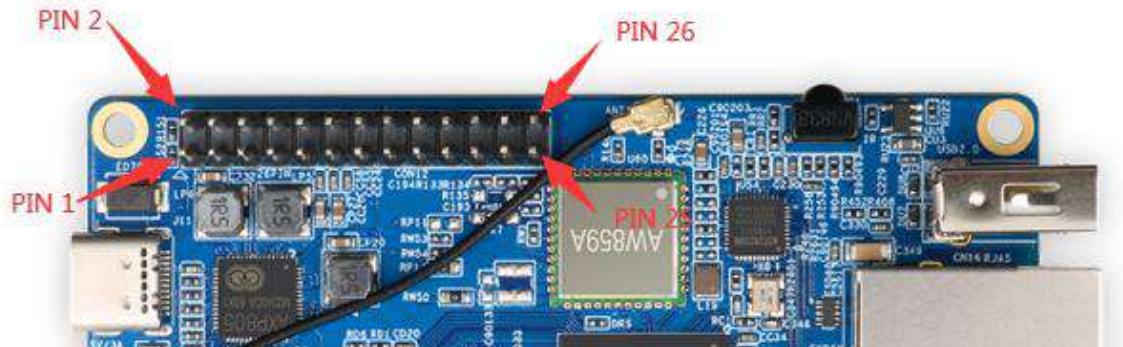
```
orangeipi@orangeipi:~$ sudo vim /etc/docker/daemon.json
{
  "registry-mirrors": [
    "https://docker.mirrors.ustc.edu.cn"
  ]
}
```

- Then enter the following command to restart the docker service (or restart the system)

```
orangeipi@orangeipi:~$ sudo systemctl restart docker
```

### 3. 17. Pin Interface Pin Description

- 1) Please refer to the following figure for the order of the 26 pin interface pins of the Orange Pi 3 LTS development board



- 2) The functions of the 26 pin interface pins of the Orange Pi 3 LTS development board are shown in the table below

| GPIO serial number | GPI O | Function  | pin |
|--------------------|-------|-----------|-----|
|                    |       | 3.3V      | 1   |
| 122                | PD26  | TWI0-SDA  | 3   |
| 121                | PD25  | TWI0-SCK  | 5   |
| 118                | PD22  | PWM0      | 7   |
|                    |       | GND       | 9   |
| 120                | PD24  | UART3_RX  | 11  |
| 119                | PD23  | UART3_TX  | 13  |
| 362                | PL10  | PL10      | 15  |
|                    |       | 3.3V      | 17  |
| 229                | PH5   | SPI1_MOSI | 19  |
| 230                | PH6   | SPI1_MISO | 21  |
| 228                | PH4   | SPI1_CLK  | 23  |
|                    |       | GND       | 25  |

| pin | Function | GPI O | GPIO serial number |
|-----|----------|-------|--------------------|
| 2   | 5V       |       |                    |
| 4   | 5V       |       |                    |
| 6   | GND      |       |                    |
| 8   | PL2      | PL2   | 354                |
| 10  | PL3      | PL3   | 355                |
| 12  | PD18     | PD18  | 114                |
| 14  | GND      |       |                    |
| 16  | PD15     | PD15  | 111                |
| 18  | PD16     | PD16  | 112                |
| 20  | GND      |       |                    |
| 22  | PD21     | PD21  | 117                |
| 24  | SPI1_CS  | PH3   | 227                |
| 26  | PL8      | PL8   | 360                |

- 3) There are a total of 17 GPIO ports in the 26pin interface, and the voltage of all GPIO ports is **3.3v**

### 3. 18. How to install wiringOP

- 1) Download the code of wiringOP

```
orangepi@orangepi:~$ sudo apt update
```



```
orangeipi@orangeipi:~$ sudo apt install -y git  
orangeipi@orangeipi:~$ git clone https://github.com/orangeipi-xunlong/wiringOP
```

## 2) Compile wiringOP

```
orangeipi@orangeipi:~$ cd wiringOP  
orangeipi@orangeipi:~/wiringOP$ sudo ./build clean  
orangeipi@orangeipi:~/wiringOP$ sudo ./build
```

3) The output of the test gpio readall command is as follows, in which the physical pins 1 to 26 correspond one-to-one with the 26 Pin pins on the development board

```
root@orangeipi3:~/wiringOP# gpio readall  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| GPIO | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | GPIO |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
122	0	SDA.0	OFF	0	3	4		5V			
121	1	SCL.0	OFF	0	5	6		5V			
118	2	PWM.0	OFF	0	7	8	0	OFF	PL02	3	354
		GND			9	10	0	OFF	PL03	4	355
120	5	RXD.3	ALT4	0	11	12	0	OFF	PD18	6	114
119	7	TXD.3	ALT4	0	13	14		GND			
362	8	PL10	OFF	0	15	16	0	OFF	PD15	9	111
		3.3V			17	18	0	OFF	PD16	10	112
229	11	MOSI.1	ALT2	0	19	20		GND			
230	12	MISO.1	ALT2	0	21	22	0	OFF	PD21	13	117
228	14	SCLK.1	ALT2	0	23	24	0	ALT2	CE.1	15	227
		GND			25	26	0	OFF	PL08	16	360
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+											
GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO	
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
root@orangeipi3:~/wiringOP#
```

## 3. 19. 26pin interface GPIO, I2C, UART, SPI and PWM test

wiringOP has been adapted to the Orange Pi development board, and wiringOP can be used to test the functions of GPIO, I2C, UART and SPI.

Before starting the test, make sure you have compiled and [installed wiringOP](#) by referring to the section [Installing wiringOP](#).

### 3. 19. 1. 26pin GPIO port test

1) The following is an example of how to set the high and low levels of the GPIO port by taking pin No. 7—the corresponding GPIO is PD22—the corresponding wPi serial number is 2—



| OPi 3 |     |       |      |   |          |    |      |      |      |       |
|-------|-----|-------|------|---|----------|----|------|------|------|-------|
| GPIO  | wPi | Name  | Mode | V | Physical | V  | Mode | Name | wPi  | GPIO  |
|       |     | 3.3V  |      |   | 1        | 2  |      | 5V   |      |       |
| 122   | 0   | SDA.0 | OFF  | 0 | 3        | 4  |      | 5V   |      |       |
| 121   | 1   | SCL.0 | OFF  | 0 | 5        | 6  |      | GND  |      |       |
| 118   | 2   | PWM.0 | OFF  | 0 | 7        | 8  | 0    | OFF  | PL02 | 3 354 |
|       |     | GND   |      |   | 9        | 10 | 0    | OFF  | PL03 | 4 355 |
| 120   | 5   | RXD.3 | ALT4 | 0 | 11       | 12 | 0    | OFF  | PD18 | 6 114 |

- 2) First set the GPIO port as output mode, where the third parameter requires the serial number of the wPi corresponding to the input pin

```
root@orangeipi:~/wiringOP# gpio mode 2 out
```

Use gpio readall to see that the mode of pin 7 has changed to out

| OPi 3 |     |       |      |   |          |    |      |      |      |       |
|-------|-----|-------|------|---|----------|----|------|------|------|-------|
| GPIO  | wPi | Name  | Mode | V | Physical | V  | Mode | Name | wPi  | GPIO  |
|       |     | 3.3V  |      |   | 1        | 2  |      | 5V   |      |       |
| 122   | 0   | SDA.0 | OFF  | 0 | 3        | 4  |      | 5V   |      |       |
| 121   | 1   | SCL.0 | OFF  | 0 | 5        | 6  |      | GND  |      |       |
| 118   | 2   | PWM.0 | OUT  | 0 | 7        | 8  | 0    | OFF  | PL02 | 3 354 |
|       |     | GND   |      |   | 9        | 10 | 0    | OFF  | PL03 | 4 355 |

- 3) Then set the GPIO port to output a low level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 0v, it means that the low level is successfully set.

```
root@orangeipi:~/wiringOP# gpio write 2 0
```

Use gpio readall to see that the value of pin 7 (V) has become 0

| OPi 3 |     |       |      |   |          |    |      |      |      |       |
|-------|-----|-------|------|---|----------|----|------|------|------|-------|
| GPIO  | wPi | Name  | Mode | V | Physical | V  | Mode | Name | wPi  | GPIO  |
|       |     | 3.3V  |      |   | 1        | 2  |      | 5V   |      |       |
| 122   | 0   | SDA.0 | OFF  | 0 | 3        | 4  |      | 5V   |      |       |
| 121   | 1   | SCL.0 | OFF  | 0 | 5        | 6  |      | GND  |      |       |
| 118   | 2   | PWM.0 | OUT  | 0 | 7        | 8  | 0    | OFF  | PL02 | 3 354 |
|       |     | GND   |      |   | 9        | 10 | 0    | OFF  | PL03 | 4 355 |

- 4) Then set the GPIO port to output a high level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is 3.3v, it means that the high level is successfully set.



```
root@orangepi:~/wiringOP# gpio write 2 1
```

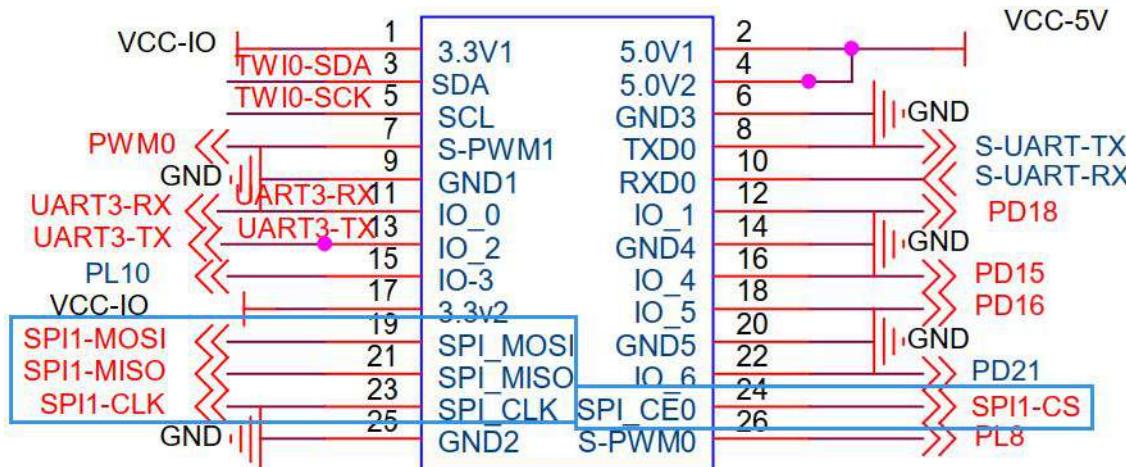
Use gpio readall to see that the value of pin 7 (V) has changed to 1

| GPIO | wPi | Name | Mode  | V   | OPI 3    |    | Mode | Name | wPi  | GPIO |
|------|-----|------|-------|-----|----------|----|------|------|------|------|
|      |     |      |       |     | Physical | V  |      |      |      |      |
| 122  | 0   | 3.3V | SDA.0 | OFF | 0        | 1  | 2    | 5V   |      |      |
| 121  | 1   | 3.3V | SCL.0 | OFF | 0        | 3  | 4    | 5V   |      |      |
| 118  | 2   | 3.3V | PWM.0 | OUT | 1        | 5  | 6    | GND  |      |      |
|      |     |      |       | GND |          | 7  | 8    | 0    | PL02 | 3    |
|      |     |      |       |     | 9        | 10 | 0    | OFF  | PL03 | 354  |
|      |     |      |       |     |          |    |      | OFF  |      | 4    |
|      |     |      |       |     |          |    |      |      |      | 355  |

- 5) The setting method of other pins is similar, just modify the serial number of wPi to the corresponding serial number of the pin.

### 3.19.2. 26pin SPI test

- 1) It can be seen from the schematic diagram of 26pin that the spi available for the development board is spi1



If you are using a Linux5.1x kernel system, spi1 is turned off by default and needs to be manually turned on to use it.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then restart the Linux system to open spi1.

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=spi-spidev1
```



- 2) After the system starts, if you can see the SPI device node under /dev, it means the configuration is correct

```
root@orangepi:~# ls /dev/spi*
/dev/spidev1.0
```

- 3) Compile the spidev\_test test program in the examples of wiringOP

```
root@orangepi:~/wiringOP/examples# make spidev_test
[CC] spidev_test.c
[link]
```

- 4) Do not short the mosi and miso pins of SPI1 first. The output result of running spidev\_test is as follows. You can see that the data of TX and RX are inconsistent

```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF
FF FF FF FF FF F0 0D | .....@..... █ ..... █
RX | FF FF
FF FF FF FF FF FF F0 FF | .....
```

- 5) Then short the two pins of SPI1 mosi (pin 19 in the 26pin interface) and miso (pin 21 in the 26pin interface) and then run the output of spidev\_test as follows, you can see the sent and received the same data

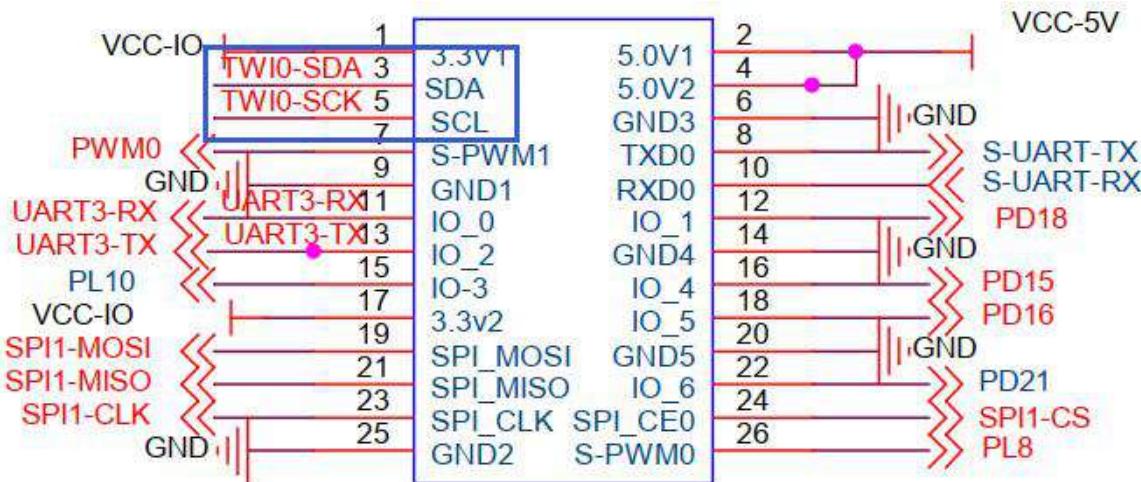
```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF
FF FF FF FF FF F0 0D | .....@..... █ ..... █
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF
FF FF FF FF FF F0 0D | .....@..... █ ..... █
```

### 3. 19. 3. 26pin I2C test

- 1) It can be seen from the schematic diagram of 26pin that the available i2c for the



development board is i2c0



If you are using a Linux5.1x kernel system, i2c0 is disabled by default and needs to be manually opened to use.

Add the configuration in the red font part below to `/boot/orangepiEnv.txt`, and then restart the Linux system to open i2c0.

```
orangeipi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=i2c0
```

2) After the system is started, if there are more i2c device nodes under `/dev`, it means the configuration is correct

```
root@orangepi:~# ls /dev/i2c*
/dev/i2c-0  /dev/i2c-1  /dev/i2c-2  /dev/i2c-3
```

- The corresponding relationship of different i2c device nodes is as follows, where
  - i2c0 in 26pin corresponds to `/dev/i2c-0`

```
root@orangepi3: # ls /sys/class/i2c-adapter/i2c-* .lh
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-0 -> ../../devices/platform/soc/5002000.i2c/i2c-0
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-1 -> ../../devices/platform/soc/5002c00.i2c/i2c-1
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-2 -> ../../devices/platform/soc/7081400.i2c/i2c-2
lrwxrwxrwx 1 root root 0 Dec  7 12:57 /sys/class/i2c-adapter/i2c-3 -> ../../devices/platform/soc/6000000.hdmi/i2c-3
```

3) Then start testing i2c, first install i2c-tools

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y i2c-tools
```



- 4) Then connect an i2c device to the i2c0 pin of the 26pin header

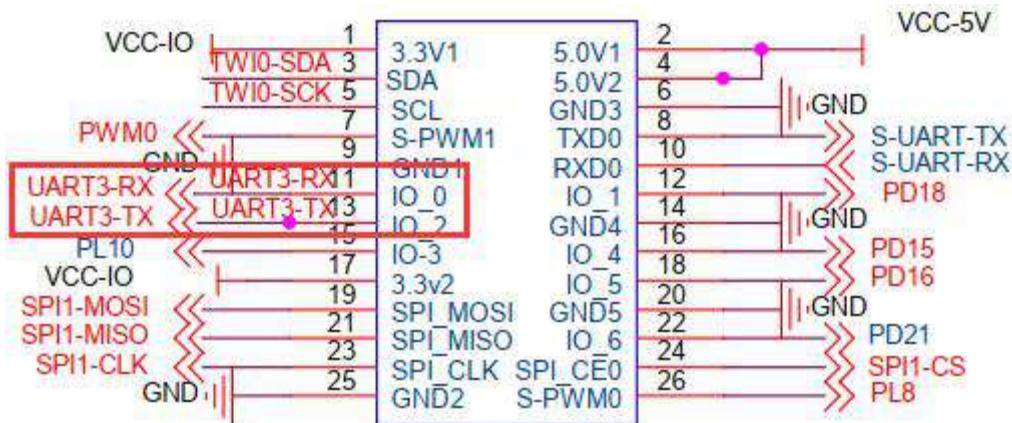
|         | i2c0                 |
|---------|----------------------|
| Sda pin | Corresponds to pin 3 |
| Sck pin | Corresponds to pin 5 |
| Vcc pin | Corresponds to pin 1 |
| Gnd pin | Corresponds to pin 6 |

- 5) Then use the `i2cdetect -y 0` command. If the address of the connected i2c device can be detected, it means that the i2c can be used normally

```
root@orangepi:~# i2cdetect -y 3
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: --
10: --
20: --
30: --
40: --
50: [50] -
60: --
70: --
```

### 3.19.4. 26pin UART test

- 1) It can be seen from the schematic diagram of 26pin that the uart available for the development board is uart3



If you are using a Linux5.1x kernel system, uart3 is disabled by default and needs to be manually opened to use.



Add the configuration in the red font part below to **/boot/orangepiEnv.txt**, and then restart the Linux system to open uart3.

```
orangeipi@orangeipi:~$ sudo vim /boot/orangepiEnv.txt  
overlays=uart3
```

- 2) After the system starts, you can see the information of ttyS3 under /sys/class/tty, and uart3 in 26pin corresponds to **/dev/ttys3**

```
root@orangeipi:~# ls /sys/class/tty/ttys* -lh  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys0 -> ../../devices/platform/soc/5000000.serial/tty/ttys0  
lrwxrwxrwx 1 root root 0 Dec 7 12:38 /sys/class/tty/ttys1 -> ../../devices/platform/soc/5000400.serial/tty/ttys1  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys2 -> ../../devices/platform/serial8250/tty/ttys2  
lrwxrwxrwx 1 root root 0 Dec 7 12:38 /sys/class/tty/ttys3 -> ../../devices/platform/soc/5000c00.serial/tty/ttys3  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys4 -> ../../devices/platform/serial8250/tty/ttys4  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys5 -> ../../devices/platform/serial8250/tty/ttys5  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys6 -> ../../devices/platform/serial8250/tty/ttys6  
lrwxrwxrwx 1 root root 0 Jan 1 1970 /sys/class/tty/ttys7 -> ../../devices/platform/serial8250/tty/ttys7
```

- 3) Then start to test the uart interface, first use the DuPont line to short-circuit the rx and tx of the uart3 interface to be tested

|        |                       |
|--------|-----------------------|
|        | uart3                 |
| Tx pin | Corresponds to pin 13 |
| Rx pin | Corresponds to pin 11 |

- 4) Use the **gpio** command in wiringOP to test the loopback function of the serial port as shown below. If you can see the following print, it means that the serial port communication is normal

```
orangeipi@orangeipi:~$ gpio serial /dev/ttys3
```

```
Out: 0: -> 0  
Out: 1: -> 1  
Out: 2: -> 2  
Out: 3: -> 3^C
```

- 5) You can also use the **serialTest.c** program in wiringOP to test the loopback function of the serial port. The specific steps are as follows

- First modify the serial port device node name opened by the serial port test program serialTest in wiringOP to **/dev/ttys3**

```
root@orangeipi:~/wiringOP/examples# vim serialTest.c
```



```
int main ()
{
    int fd ;
    int count ;
    unsigned int nextTime ;

    if ((fd = serialOpen ("/dev/ttyS3", 115200)) < 0)
    {
        fprintf (stderr, "Unable to open serial device: %s\n", strerror (errno)) ;
        return 1 ;
    }
```

b. Then compile the serial test program serialTest in wiringOP

```
root@orangepi:~/wiringOP/examples# make serialTest
[CC] serialTest.c
[link]
root@orangepi:~/wiringOP/examples#
```

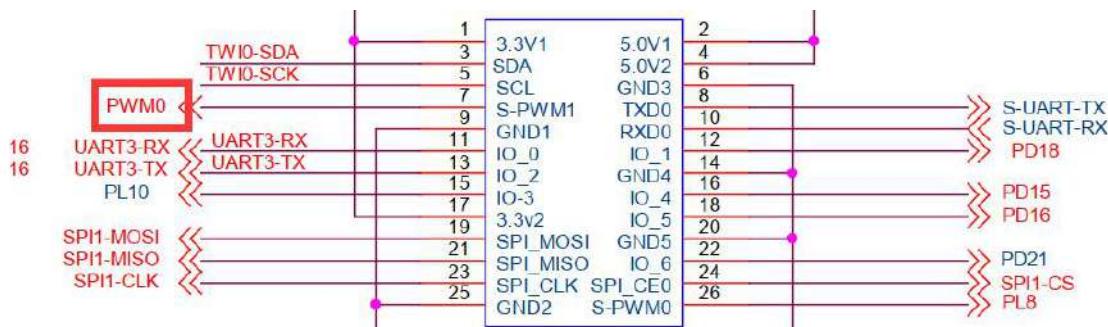
c. Finally run serialTest, if you can see the following print, it means the serial communication is normal

```
root@orangepi:~/wiringOP/examples# ./serialTest
```

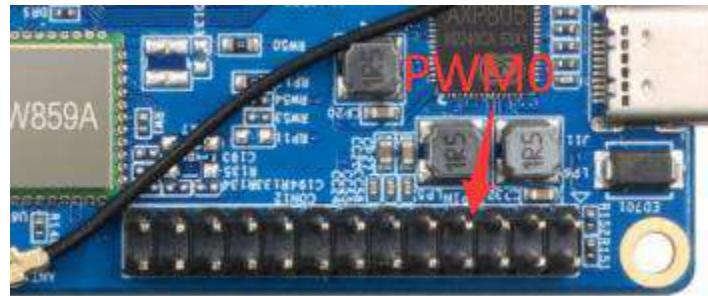
```
Out: 0: -> 0
Out: 1: -> 1
Out: 2: -> 2
Out: 3: -> 3^C
```

### 3.19.5. PWM test method

1) From the schematic diagram of the 26pin interface, the pwm available for the development board is pwm0



2) The corresponding pins on the development board are



If you are using a Linux5.1x kernel system, pwm is disabled by default and needs to be manually opened to use it.

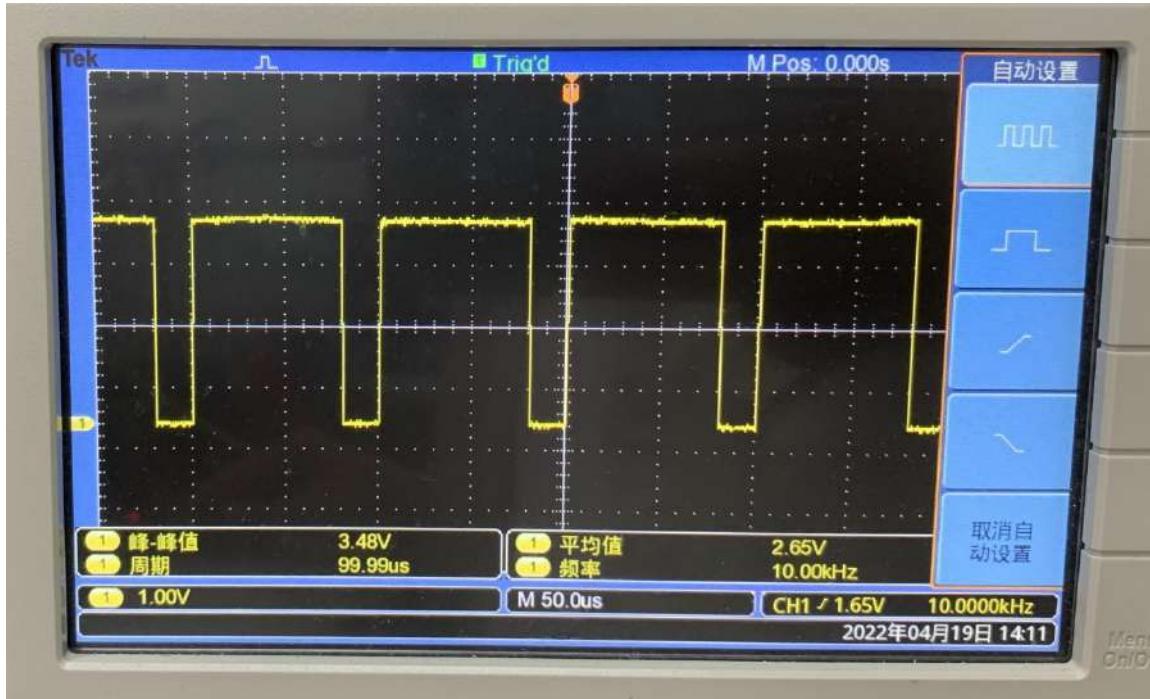
Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then **restart** the Linux system to open pwm.

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt  
overlays=pwm
```

3) Enter the following command in the Linux system to make pwm0 output a 10kHz rectangular wave

```
root@orangepi:~# echo 0 > /sys/class/pwm/pwmchip0/export  
root@orangepi:~# echo 100000 > /sys/class/pwm/pwmchip0/pwm0/period  
root@orangepi:~# echo 20000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle  
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
```

4) Use an oscilloscope to measure the pin corresponding to pwm0 and you can see the following waveform output



### 3.19.6. How to use 0.96-inch OLED module with I2C interface

If you are using a Linux5.1x kernel system, i2c0 is disabled by default and needs to be manually opened to use.

Add the configuration in the red font part below to `/boot/orangepiEnv.txt`, and then **restart** the Linux system to open i2c0.

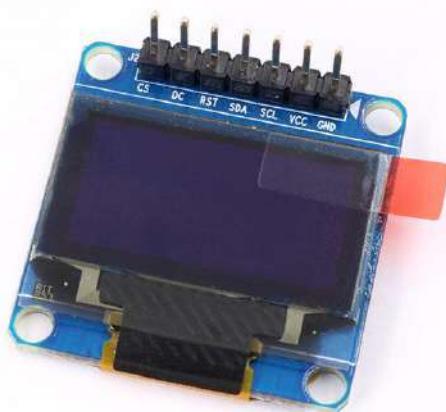
```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt  
overlays=i2c0
```

- 1) The 0.96-inch OLED module of orange pi is shown in the figure below, and its 7-bit i2c slave address is 0x3c



2) First, connect the 0.96-inch OLED module to the 26pin interface of the Orange Pi development board through the DuPont cable. The wiring method is as follows

| OLED module pins | describe        | Development board 26pin interface corresponding pin |
|------------------|-----------------|-----------------------------------------------------|
| GND              | power ground    | pin 6                                               |
| VCC              | 5V              | pin 2                                               |
| SCL              | I2C clock line  | pin 5                                               |
| SDA              | I2C data line   | pin 3                                               |
| RST              | Connect to 3.3V | pin 1                                               |
| DC               | Connect to GND  | pin 9                                               |
| CS               | Connect to GND  | pin 25                                              |





- 3) After connecting the OLED module to the development board, first use the i2c-tools tool to check whether the address of the OLED module can be scanned

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt install -y i2c-tools  
orangepi@orangepi:~$ sudo i2cdetect -y 0
```

```
root@orangepi3-lts:~# i2cdetect -y 0  
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  
00: -----  
10: -----  
20: -----  
30: -- -- -- -- -- -- -- 3c --  
40: -----  
50: -----  
60: -----  
70: -----
```

- 4) Then you can use the **oled\_demo** in wiringOP to test the OLED module. The test steps are as follows

```
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP  
root@orangepi:~# cd wiringOP  
root@orangepi:~/wiringOP# ./build clean && ./build  
root@orangepi:~/wiringOP# cd examples  
root@orangepi:~/wiringOP/examples# make oled_demo  
root@orangepi:~/wiringOP/examples# ./oled_demo /dev/i2c-0  
-----start-----  
-----end-----
```

- 5) After running oled\_demo, you can see the following output on the OLED screen





### 3. 19. 7. The method of outputting the kernel print information to the 26pin serial port

The kernel console is output to `ttyS0` by default, which is the 3pin debug serial port on the development board. We can also set the kernel console output to be redirected to `UART3` in the 26pin interface. Please refer to the following steps for the specific method.

If you are using a Linux5.1x kernel system, `uart3` is disabled by default and needs to be manually opened to use.

Add the configuration in the red font part below to `/boot/orangepiEnv.txt`, and then **restart** the Linux system to open `uart3`.

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt  
overlays=uart3
```

1) Then modify `console=ttyS0` in `/boot/boot.cmd` to `console=ttyS3`

```
root@orangepi:~# vim /boot/boot.cmd
```

```
if test "${console}" = "display" || test "${console}" = "both"; then setenv consoleargs "console=ttyS3,115200 console=tty1"; fi  
if test "${console}" = "serial"; then setenv consoleargs "console=ttyS3,115200"; fi  
if test "${bootlogo}" = "true"; then setenv consoleargs "bootsplash,bootfile=bootsplash.orangepi ${consoleargs}"; fi
```

2) Then recompile `/boot/boot.cmd` to `/boot/boot.scr` (operate in the linux system of the development board)

```
root@orangepi:~# mkimage -C none -A arm -T script -d /boot/boot.cmd /boot/boot.scr
```

Image Name:

Created: Tue Dec 8 02:35:43 2020

Image Type: ARM Linux Script (uncompressed)

Data Size: 2448 Bytes = 2.39 KiB = 0.00 MiB

Load Address: 00000000

Entry Point: 00000000

Contents:

```
Image 0: 2440 Bytes = 2.38 KiB = 0.00 MiB
```

3) Then connect the USB to TTL module to the `UART3` pin of the 26pin interface through the DuPont line



- a. The GND of the USB to TTL module is connected to the GND of the 26pin interface of the development board
- b. The RX of the USB to TTL module is connected to **the TX of the UART3 of the development board**
- c. The TX of the USB to TTL module is connected to **the RX of the UART3 of the development board**



- 4) Then restart the development board, you can see that the kernel console is output to ttyS3 by default. Note that the output log of u-boot is still output to ttyS0 at this time, and will not be output to ttyS3

```
Orange Pi 2.1.6 Buster ttyS3  
orange pi3-lts login:
```

### 3. 20. How to use SPI LCD display

**Note: This method is only applicable to the system with the linux4.9 kernel, and the system with the linux5.1x kernel is not adapted.**

#### 3. 20. 1. 2.4 inch SPI LCD display

- 1) The details page of the tested LCD display is linked as follows

[http://www.lcdwiki.com/2.4inch\\_SPI\\_Module\\_ILI9341\\_SKU:MSP2402](http://www.lcdwiki.com/2.4inch_SPI_Module_ILI9341_SKU:MSP2402)

- 2) The wiring method of the LCD display and the development board is as follows

| TFT SPI Module pins | The corresponding pin of the development board<br>26pin | GPIO -- GPIO num |
|---------------------|---------------------------------------------------------|------------------|
|                     |                                                         |                  |



|           |        |             |
|-----------|--------|-------------|
| VCC       | pin 1  |             |
| GND       | pin 6  |             |
| CS        | pin 24 |             |
| RESET     | pin 12 | PD18 -- 114 |
| D/C       | pin 16 | PD15 -- 111 |
| SDI(MOSI) | pin 19 |             |
| SCK       | pin 23 |             |
| LED       | pin 18 | PD16 -- 112 |
| SDO(MISO) | pin 21 |             |

- 3) After connecting the display to the development board, use the following command to load the **fbtft\_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb ili9341 \
busnum=1 cs=0 gpios=reset:114,dc:111,led:112 rotate=90 \
speed=65000000 bgr=1 txbuflen=65536
```

- 4) When the **fbtft\_device** kernel module is loaded, the correct output log of the dmesg command is as follows, and the log can know that the framebuffer used by the LCD display is **fb8**

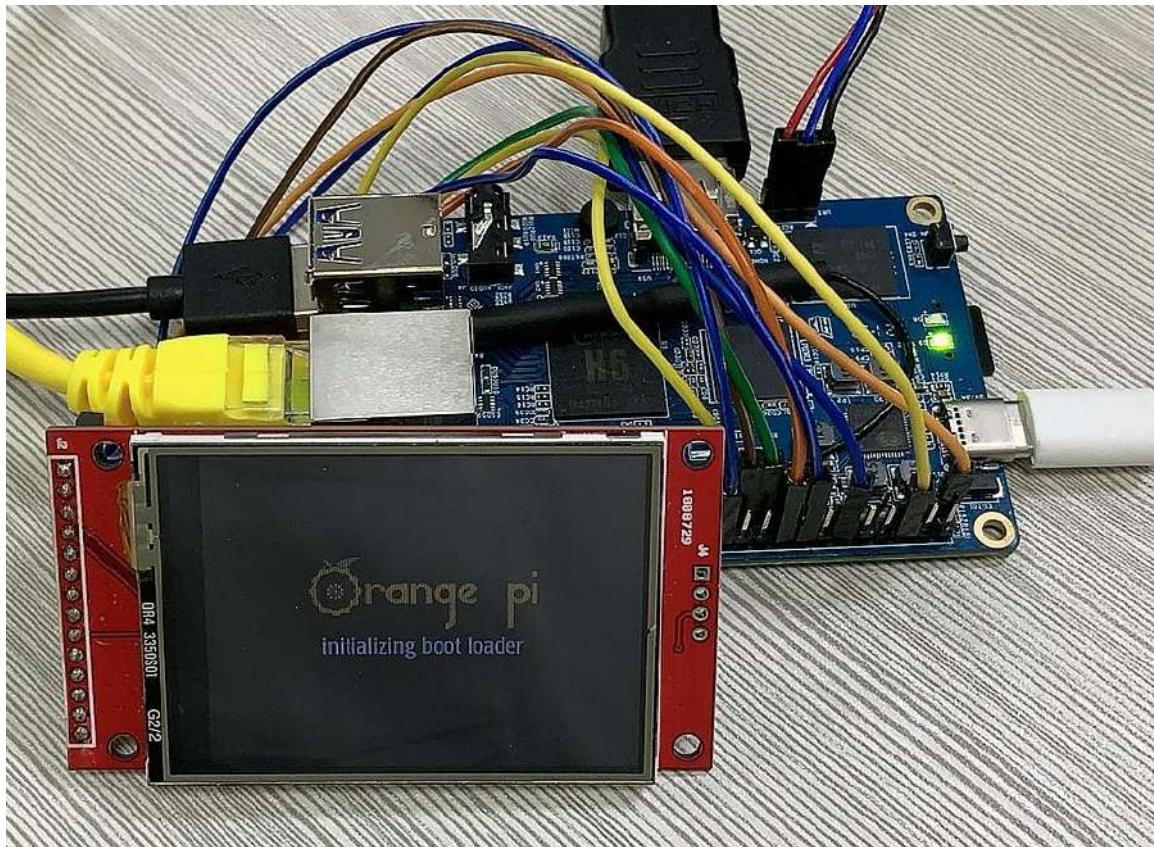
```
root@orangepi:~# dmesg | tail
[ 191.788887] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
[ 191.789343] spidev spi1.0: Deleting spi1.0
[ 191.793283] fbtft_device: GPIOs used by 'fb ili9341':
[ 191.793301] fbtft_device: 'reset' = GPIO114
[ 191.793313] fbtft_device: 'dc' = GPIO111
[ 191.793324] fbtft_device: 'led' = GPIO112
[ 191.793351] spi spi1.0: fb ili9341 spi1.0 65000kHz 8 bits mode=0x00
[ 191.807788] fb ili9341: module is from the staging directory, the quality is unknown,
you have been warned.
[ 192.083355] graphics fb8: fb ili9341 frame buffer, 320x240, 150 KiB video memory,
64 KiB buffer memory, fps=20, spi1.0 at 65 MHz
```

- 5) Then use the following command to display the Orange Pi logo image on the LCD display

```
root@orangepi:~# apt update
```



```
root@orangeipi:~# apt -y install fbi  
root@orangeipi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



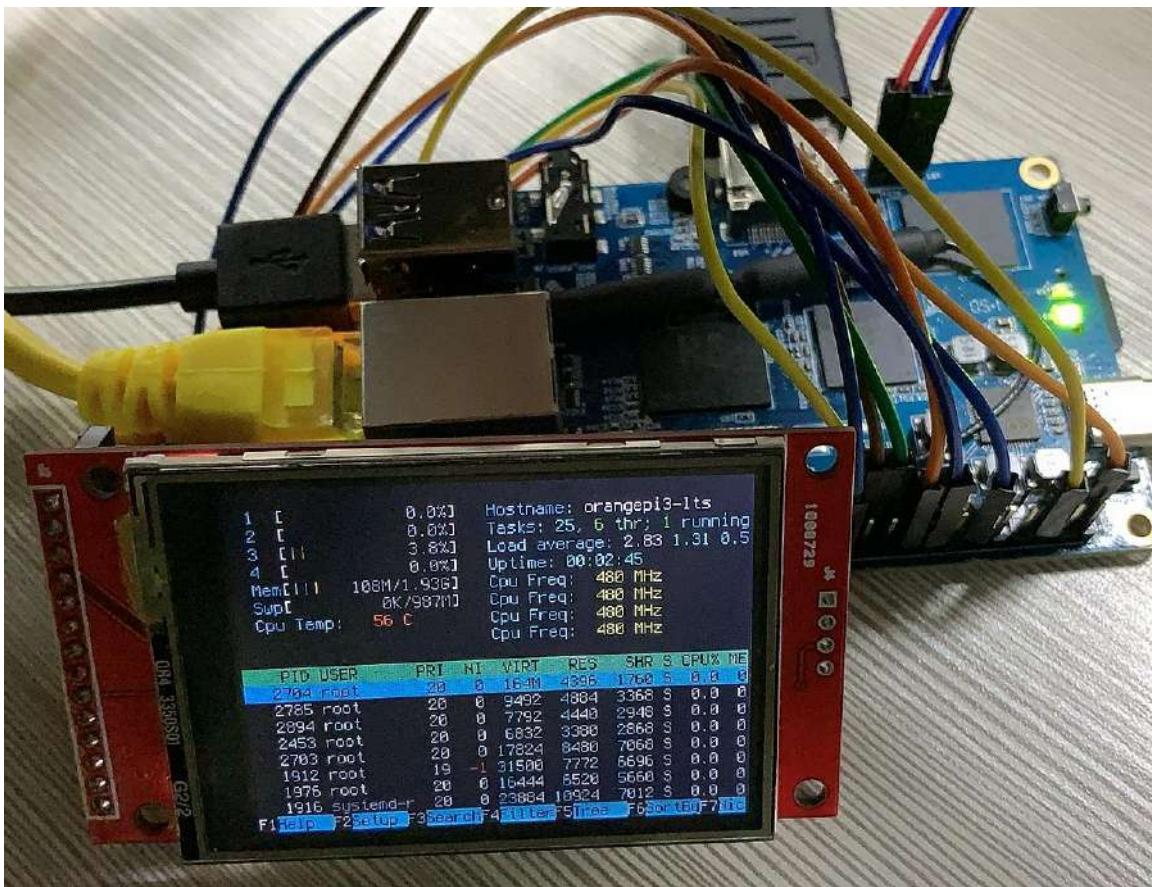
- 6) You can also map the output of tty1 to the fb device of the LCD display - **fb8**. After the mapping, there will be no more image output from HDMI

```
root@orangeipi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display use below command

```
root@orangeipi:~# con2fbmap 1 0
```

Below is the output of running the htop command



7) Since the default terminal font is too large, the display screen cannot display too much content. The following method can be used to reduce the font of the terminal

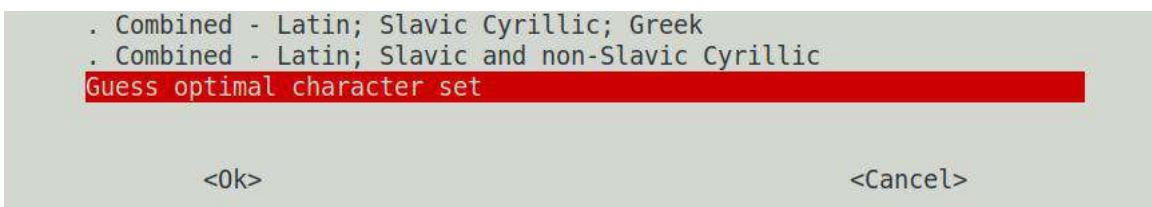
- First run **dpkg-reconfigure console-setup**

```
root@orangepi:~# apt-get update  
root@orangepi:~# apt-get install -y kbd  
root@orangepi:~# dpkg-reconfigure console-setup
```

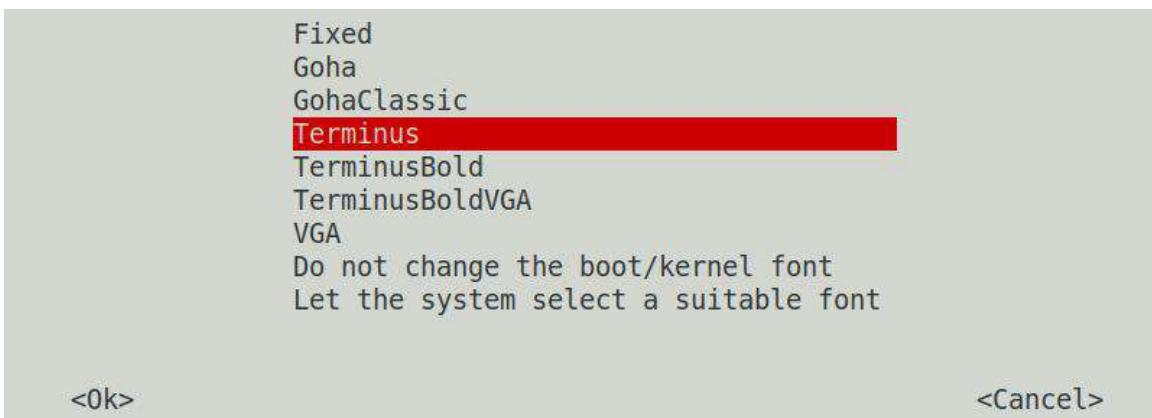
- Terminal code selection **UTF-8**



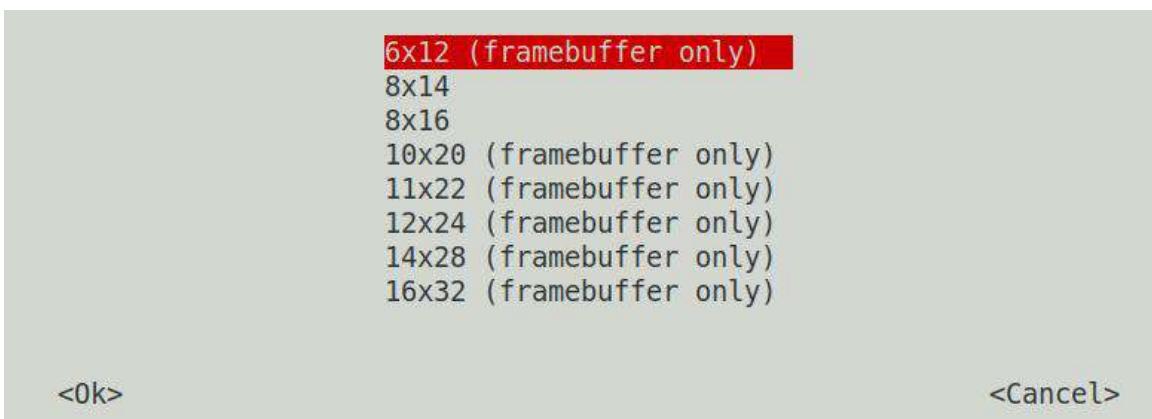
- Then select **Guess optimal character set**



d. Then select **Terminus**



e. The final font size is 6x12



f. After setting, you can see that the font on the LCD display becomes smaller

8) Set the method to automatically load the fbtft\_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9341 busnum=1 cs=0
```



```
gpios=reset:114,dc:111,led:112 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

- c. Then restart the linux system to see that the kernel modules related to **fbtft\_device** have been automatically loaded

- 9) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to **/boot/orangepiEnv.txt**, and then restart the system to see the LCD display output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"  
extraargs=cma=25M fbcon=map:8
```

### 3. 20. 2. 3.2 inch RPi SPI LCD display

- 1) The details page of the tested LCD display is linked as follows

[http://www.lcdwiki.com/3.2inch\\_RPi\\_Display](http://www.lcdwiki.com/3.2inch_RPi_Display)

- 2) The wiring method of LCD display and development board is as follows



- 3) After connecting the LCD display to the development board, use the following command to load the **fbtft\_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb ili9341 \  
busnum=1 cs=0 gpios=reset:119,dc:362 rotate=90 \\  
speed=65000000 bgr=1 txbuflen=65536
```

- 4) When the **fbtft\_device** kernel module is loaded, the correct output log of the dmesg command is as follows, and the log can know that the framebuffer used by the LCD screen is **fb8**

```
root@orangepi:~# dmesg | tail  
[ 97.884472] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
```



```
[ 97.884933] spidev spi1.0: Deleting spi1.0
[ 97.888354] fbtft_device: GPIOS used by 'fb_ili9341':
[ 97.888373] fbtft_device: 'reset' = GPIO119
[ 97.888385] fbtft_device: 'dc' = GPIO362
[ 97.889430] spi spi1.0: fb_ili9341 spi1.0 65000kHz 8 bits mode=0x00
[ 97.907191] fb_ili9341: module is from the staging directory, the quality is unknown,
you have been warned.
[ 98.181286] Console: switching to colour frame buffer device 40x30
[ 98.181938] graphics fb8: fb_ili9341 frame buffer, 320x240, 150 KiB video memory,
64 KiB buffer memory, fps=20, spi1.0 at 65 MHz
```

- 5) Then use the following command to display the Orange Pi logo image on the LCD screen

```
root@orangeipi:~# apt update
root@orangeipi:~# apt -y install fbi
root@orangeipi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



- 6) You can also map the output of tty1 to the fb device of the LCD screen - **fb8**. After the mapping, there will be no more image output from HDMI

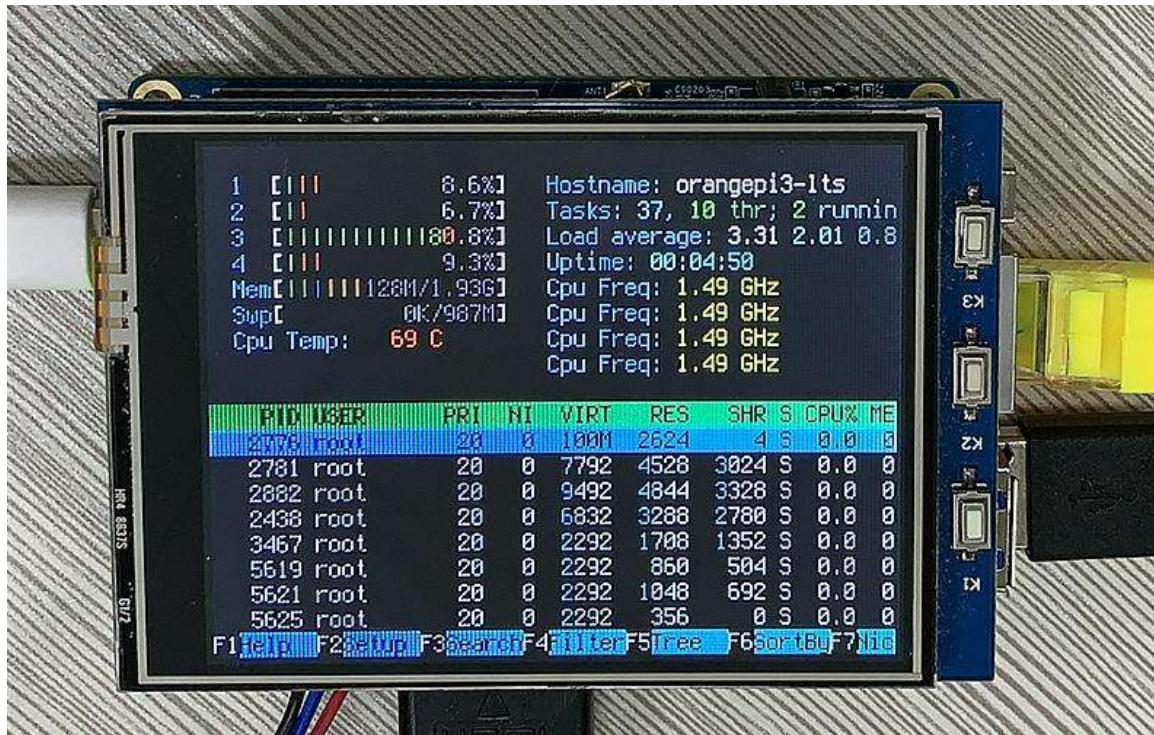
```
root@orangeipi:~# con2fbmap 1 8
```



If you want to switch back to HDMI display use below command

```
root@orangeipi:~# con2fbmap 1 0
```

Below is the output of running the htop command

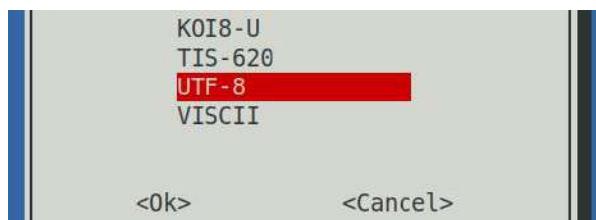


7) Since the default terminal font is too large, the screen cannot display too much content. You can reduce the terminal font by the following method

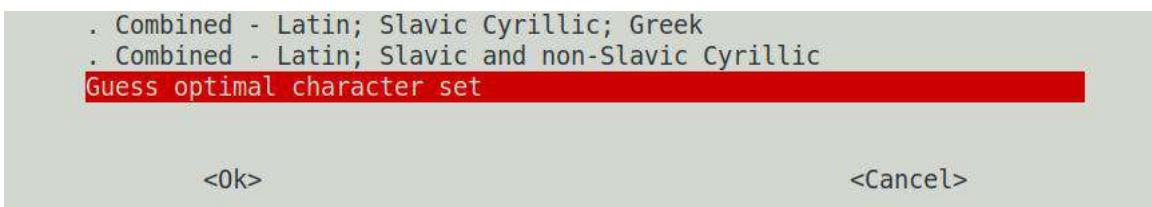
a. First run **dpkg-reconfigure console-setup**

```
root@orangeipi:~# apt-get update  
root@orangeipi:~# apt-get install -y kbd  
root@orangeipi:~# dpkg-reconfigure console-setup
```

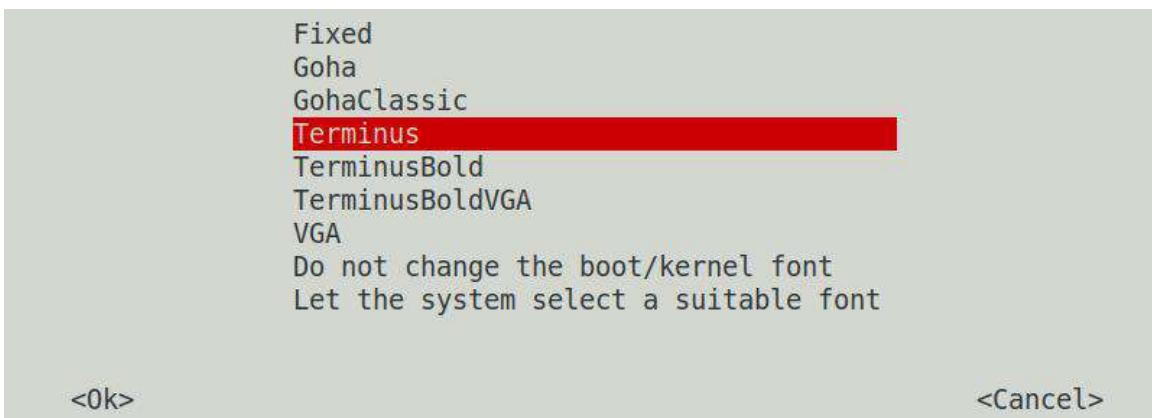
b. Terminal code selection **UTF-8**



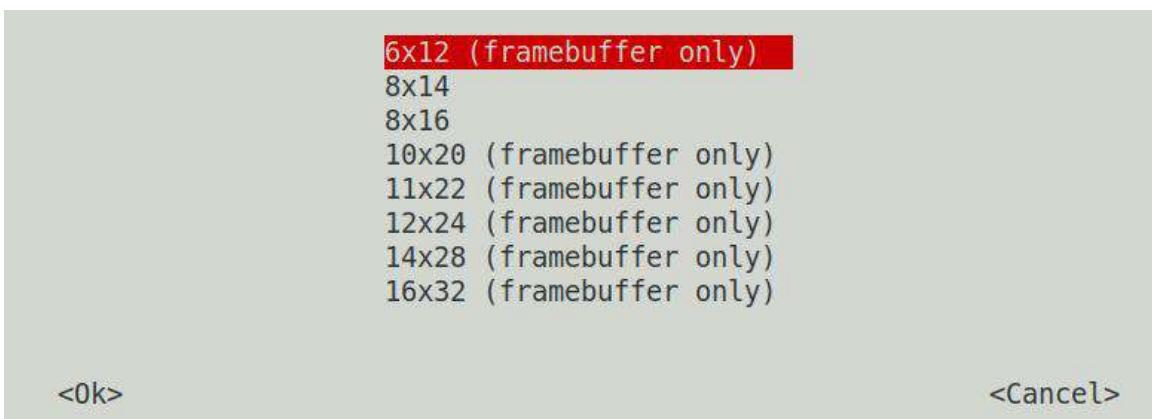
c. Then select **Guess optimal character set**



d. Then select **Terminus**



e. The final font size is 6x12



f. After setting, you can see that the font on the LCD screen has become smaller

8) Set the method to automatically load the fbtft\_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb ili9341 busnum=1 cs=0 gpios=reset:119,dc:362
```



```
rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

- c. Then restart the linux system to see that the kernel modules related to **fbtft\_device** have been automatically loaded

- 9) If you want the console to be automatically mapped to the LCD screen after the linux system starts, please add the following configuration to **/boot/orangepiEnv.txt**, and then restart the system to see the LCD screen output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"  
extraargs=cma=25M fbcon=map:8
```

### 3. 20. 3. 3.5 inch SPI LCD display

- 1) The details page of the tested LCD display is linked as follows

[http://www.lcdwiki.com/3.5inch\\_SPI\\_Module\\_ILI9488\\_SKU:MSP3520](http://www.lcdwiki.com/3.5inch_SPI_Module_ILI9488_SKU:MSP3520)

- 2) The wiring method of the LCD display and the development board is as follows

| TFT SPI Module pins | The corresponding pin of the development board<br>26pin | GPIO -- GPIO num |
|---------------------|---------------------------------------------------------|------------------|
| VCC                 | pin 1                                                   |                  |
| GND                 | pin 6                                                   |                  |
| CS                  | pin 24                                                  |                  |
| RESET               | pin 12                                                  | PD18 -- 114      |
| DC/RS               | pin 16                                                  | PD15 -- 111      |
| SDI(MOSI)           | pin 19                                                  |                  |
| SCK                 | pin 23                                                  |                  |
| LED                 | pin 18                                                  | PD16 -- 112      |
| SDO(MISO)           | pin 21                                                  |                  |

- 3) After connecting the display to the development board, use the following command to load the **fbtft\_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb ili9488 \
busnum=1 cs=0 gpios=reset:114,dc:111,led:112 rotate=270 \
speed=65000000 bgr=1 txbuflen=65536
```

- 4) When the **fbtft\_device** kernel module is loaded, the correct output log of the dmesg



command is as follows, and the log can know that the framebuffer used by the LCD display is **fb8**

```
root@orangepi:~# dmesg | tail
[ 36.897250] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
[ 36.897709] spidev spi1.0: Deleting spi1.0
[ 36.900560] fbtft_device: GPIOS used by 'fb ili9488':
[ 36.900579] fbtft_device: 'reset' = GPIO114
[ 36.900591] fbtft_device: 'dc' = GPIO111
[ 36.900602] fbtft_device: 'led' = GPIO112
[ 36.900629] spi spi1.0: fb ili9488 spi1.0 65000kHz 8 bits mode=0x00
[ 36.918499] fb ili9488: module is from the staging directory, the quality is unknown,
you have been warned.
[ 37.263271] graphics fb8: fb ili9488 frame buffer, 480x320, 300 KiB video memory,
64 KiB buffer memory, fps=60, spi1.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo image on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install -y fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



6) You can also map the output of tty1 to the fb device of the LCD display screen - **fb8**. After mapping, the LCD screen will display the output of the terminal, and there will be



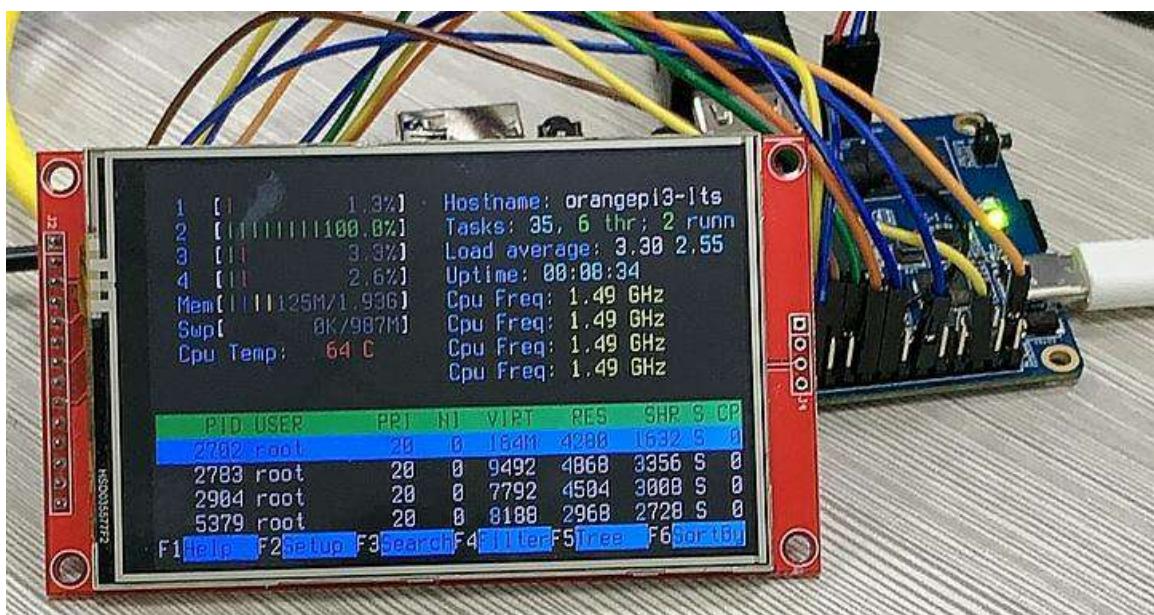
no more image output from HDMI

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display use below command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command



7) Set the method to automatically load the fbtft\_device module at system startup

- Create a new **/etc/modules-load.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

- Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb ili9488 busnum=1 cs=0
gpios=reset:114,dc:111,led:112 rotate=270 speed=65000000 bgr=1 txbuflen=65536
```

- Then restart the linux system to see that the kernel modules related to fbtft\_device have been automatically loaded



8) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to **/boot/orangepiEnv.txt**, and then restart the system to see that the LCD display has output

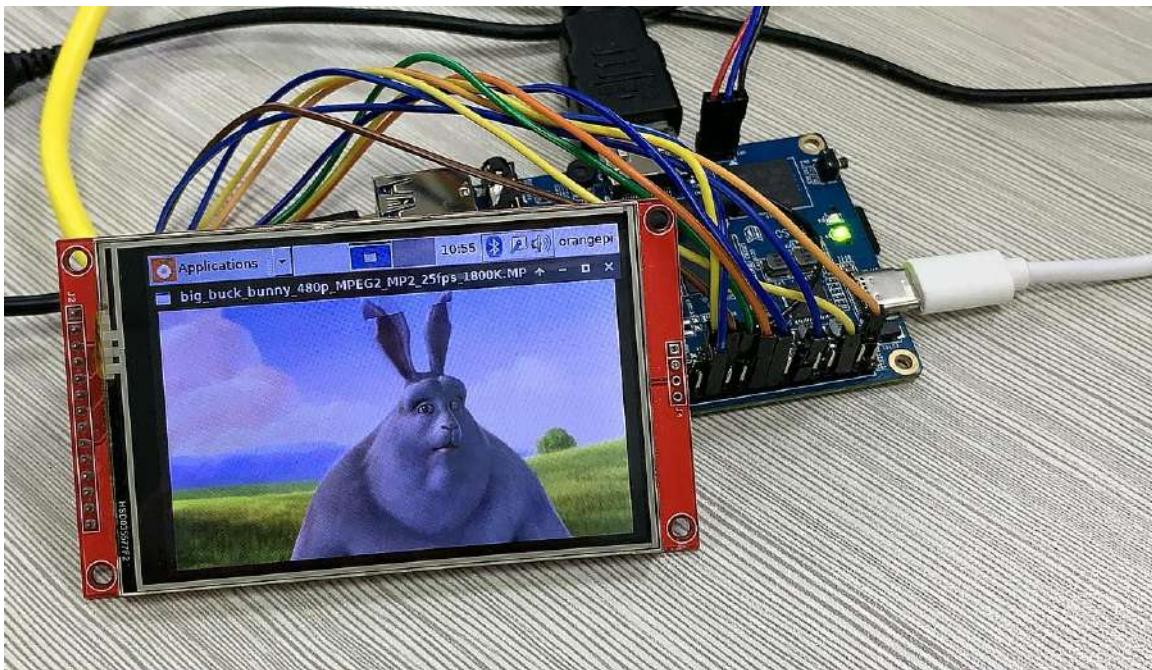
```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
```

```
extraargs=cma=25M fbcon=map:8
```

9) If you need to display the desktop version system to the LCD screen, you can execute the following command, after a few seconds, the LCD screen can see the desktop of the Linux system

```
root@orangepi:~# FRAMEBUFFER=/dev/fb8 startx
```





- 10) If you want to automatically display the desktop to the LCD display after the linux system starts, please add the following configuration file to the linux system, and then restart the system to see that the LCD display has display output

```
root@orangeipi:~# cat /usr/share/X11/xorg.conf.d/99-fbdev.conf
```

```
Section "Device"
```

```
    Identifier "myfb"
```

```
    Driver "fbdev"
```

```
    Option "fbdev" "/dev/fb8"
```

```
EndSection
```

### 3. 21. Hardware watchdog test

- 1) Download the code of wiringOP

```
orangeipi@orangeipi:~$ sudo apt update
```

```
orangeipi@orangeipi:~$ sudo apt install -y git
```

```
orangeipi@orangeipi:~$ sudo git clone https://github.com/orangeipi-xunlong/wiringOP
```

- 2) Compile the watchdog test program

```
orangeipi@orangeipi:~$ cd wiringOP/examples/
```

```
orangeipi@orangeipi:~/wiringOP/examples$ gcc watchdog.c -o watchdog
```



### 3) Run the watchdog test program

- a. The second parameter 10 represents the counting time of the watchdog. If the dog is not fed within this time, the system will restart
- b. We can feed the dog by pressing any key on the keyboard (except ESC), after feeding the dog, the program will print a line of keep alive to indicate that the dog was fed successfully

```
orangepi@orangepi:~/wiringOP/examples$ sudo ./watchdog 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
put_usr return,if 0,success:0
keep alive
keep alive
keep alive
```

## 3. 22. Set up Chinese environment and install Chinese input method

Note that before installing the Chinese input method, please make sure that the Linux system used by the development board is the desktop version system.

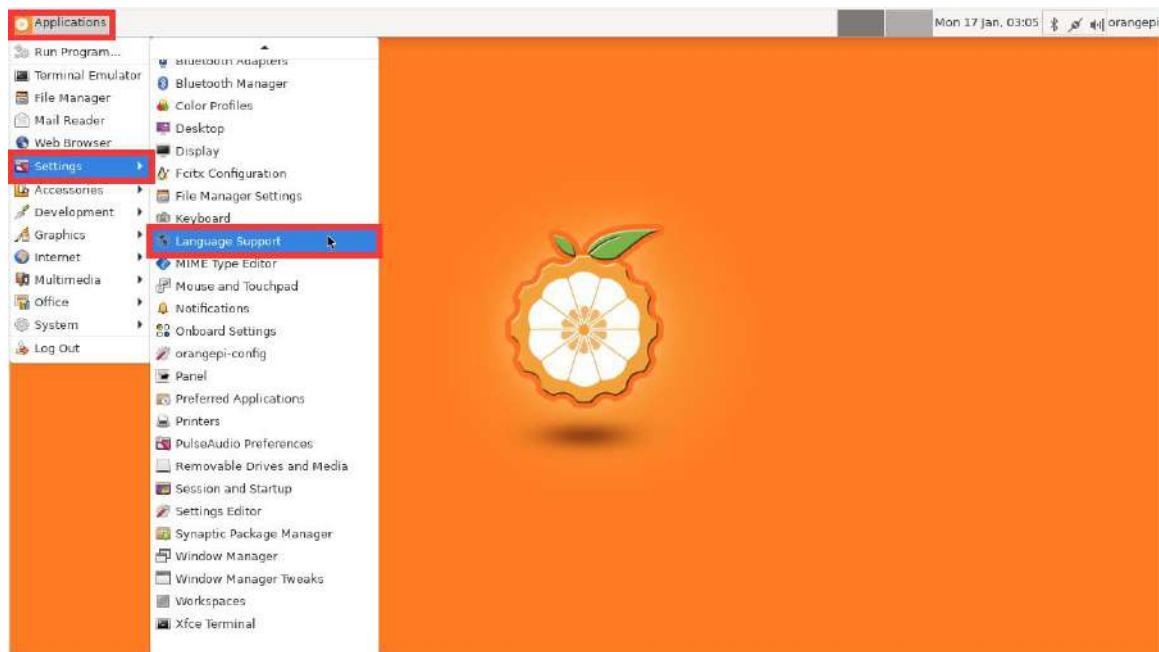
### 3. 22. 1. Installation method of Ubuntu system

- 1) First update the software source of the system

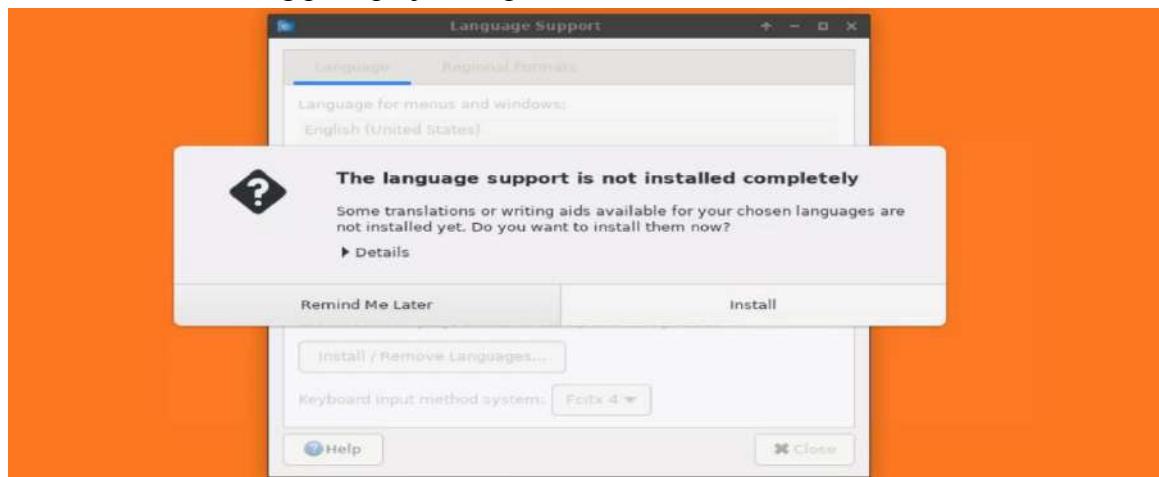
Note that an error may be reported during the installation process without executing the following command, so please do not ignore this step.

```
orangepi@orangepi:~$ sudo apt update
```

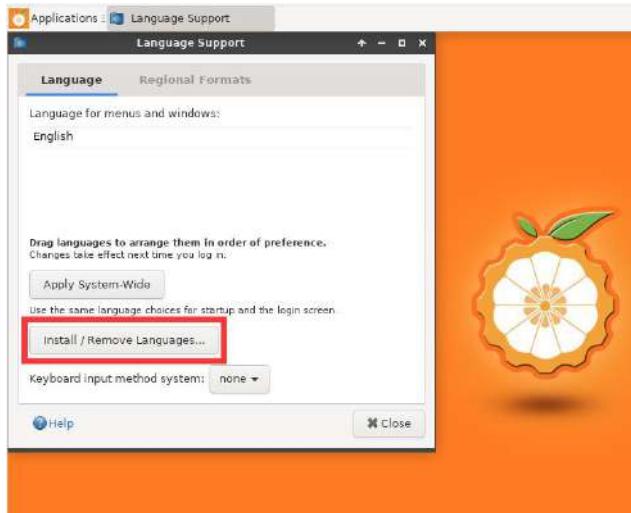
- 2) Then open **Language Support**



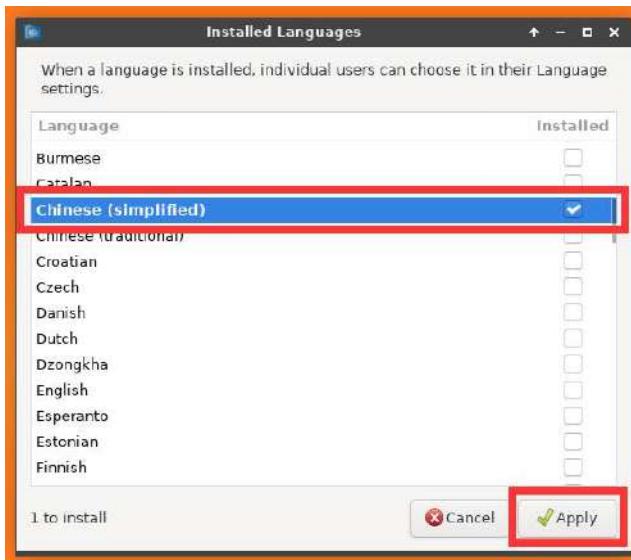
- 3) If the following information is prompted, click **Install** to repair it. Some systems will not have the following prompt, just skip it.



- 4) Then open **Install/Remove Languages...**



- 5) Then find **Chinese (simplified)**, click the box on the right to select it, and then click **Apply** in the lower right corner

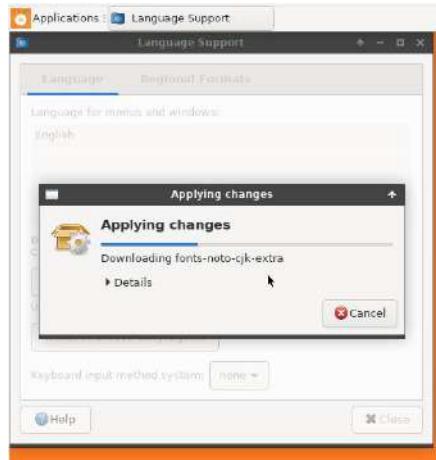


- 6) Then enter the password of the Linux system in the pop-up password input interface, the default is **orangepi**



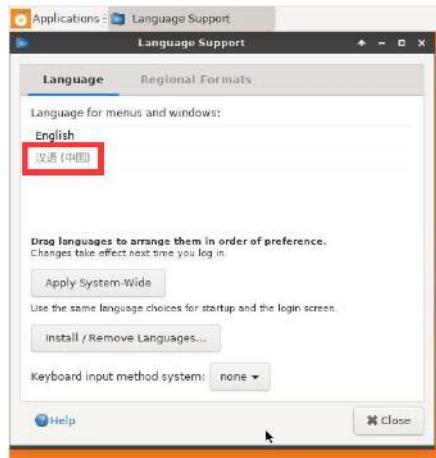


- 7) Then it will start to install the required software packages. At this time, wait patiently for the installation to complete.

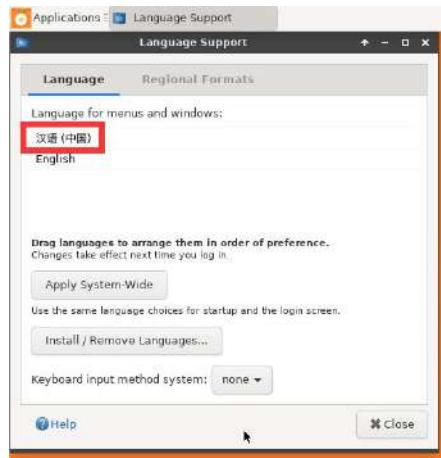


If the software installation fails, it is generally because the apt update command was not executed at the beginning. If the apt update command is executed and the prompt fails, it is generally because an error occurred when the apt update command was executed, but it was not executed successfully.

- 8) After the installation is complete, you can see the **Chinese (China)** option

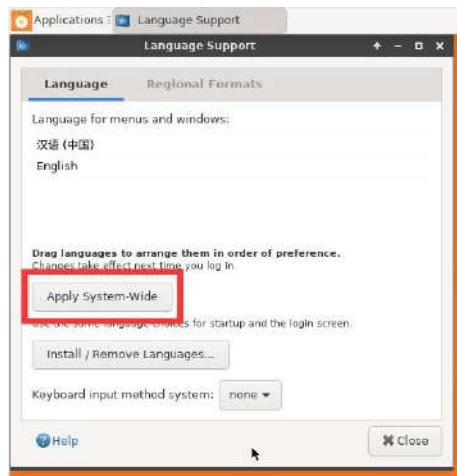


- 9) Then please use the left mouse button to select **Chinese (China)** and hold it down, then drag it up to the first position, the display after dragging is as shown below

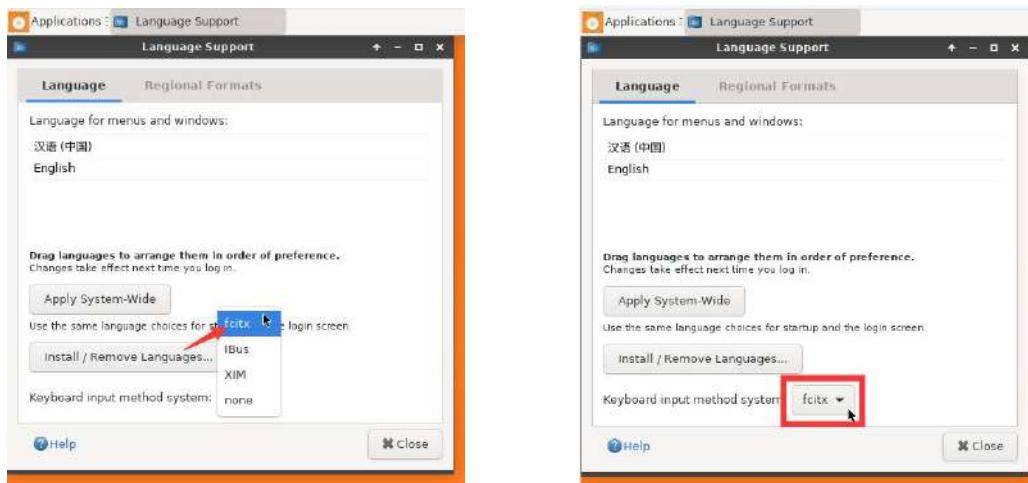


Note that this step is not very easy to drag, please be patient and try a few more times.

- 10) Then select **Apply System-Wide** to apply Chinese settings to the entire system



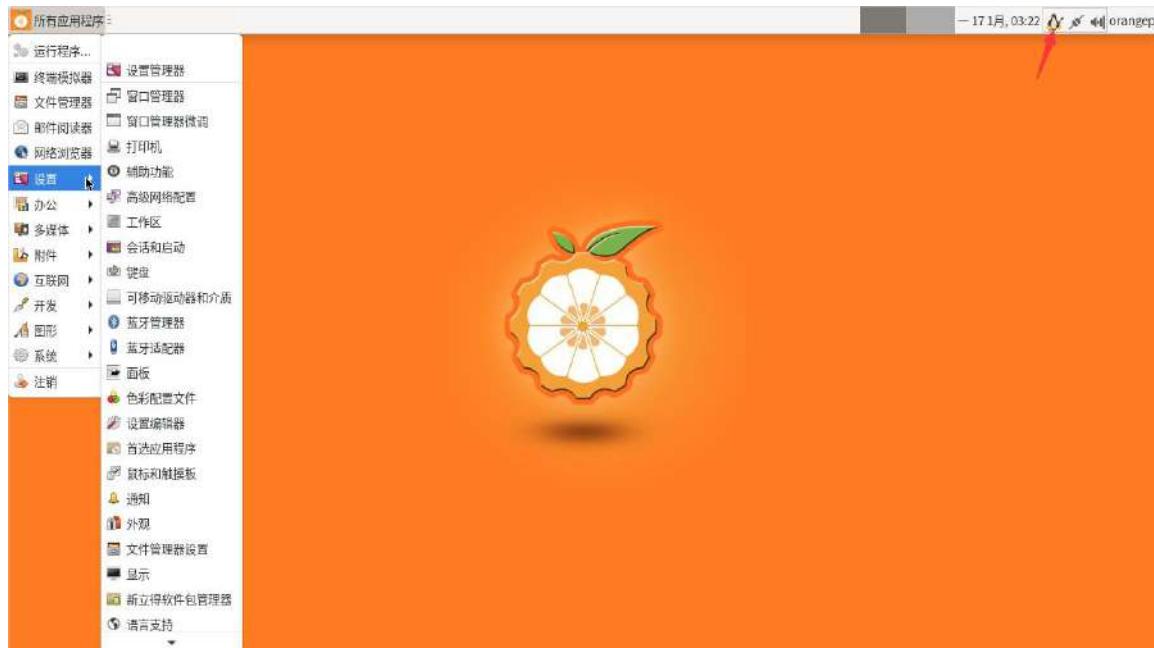
- 11) Then select **Keyboard input method system** as **fcitx**



## 12) Then restart the Linux system to make the configuration take effect

13) After re-entering the system, you can see that the desktop is displayed in Chinese, and you can also see a penguin in the upper right corner of the system

**Note that Ubuntu 22.04 shows a black keyboard icon in the upper right corner.**



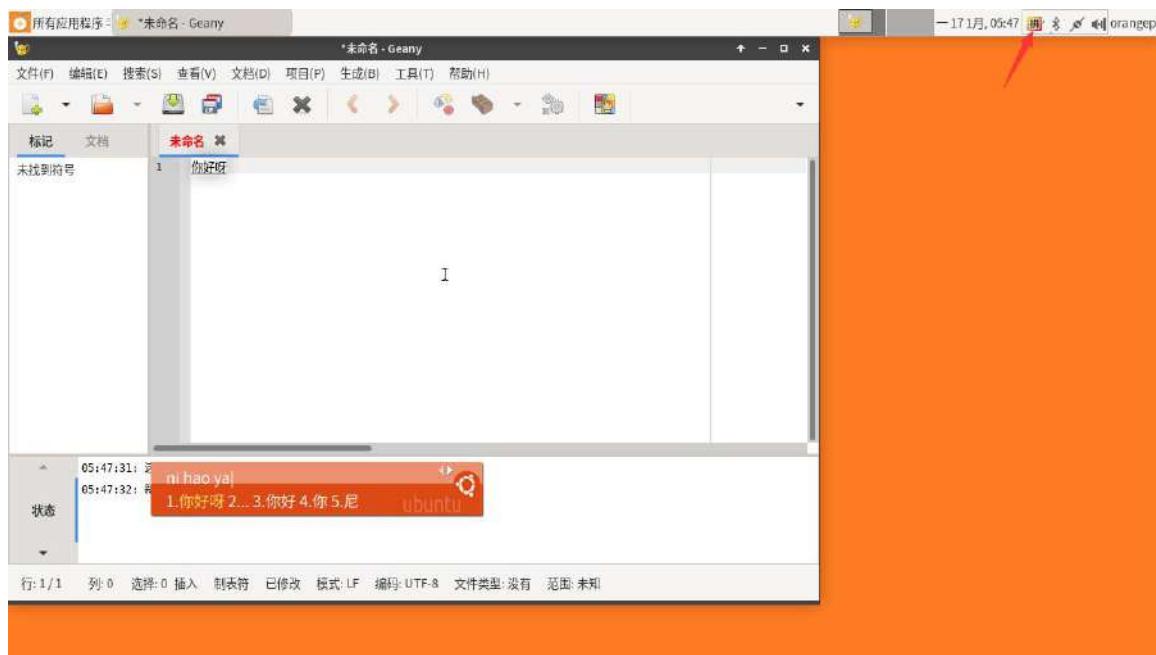
14) Then we can open the Chinese input method under **Geany** test, the opening method is as shown in the figure below



Note that if **Geany** is not installed on the system, feel free to open another application for testing.



15) After opening **Geany**, the default is English input method, we can switch to Chinese input method by **Ctrl+Space** shortcut key, and then we can input Chinese



16) In addition, you can place the mouse on the penguin in the upper right corner of the desktop, and then click the **right mouse button** to view all input methods supported by the system, or select the input method to be used.



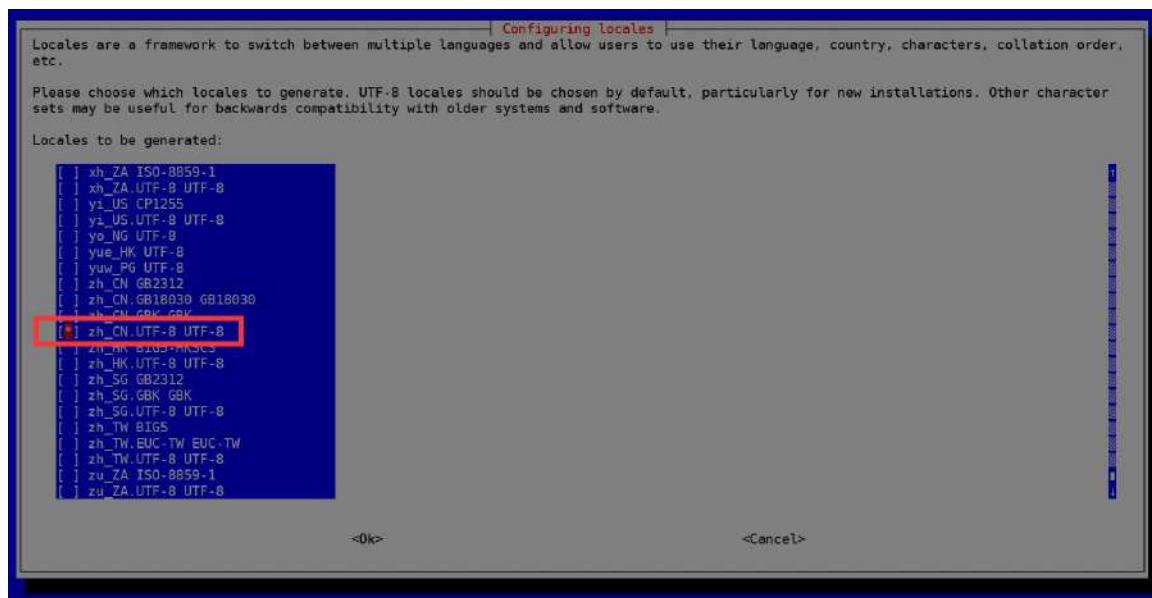
### 3.22.2. Installation method of Debian system

1) First set the default **locale** to Chinese

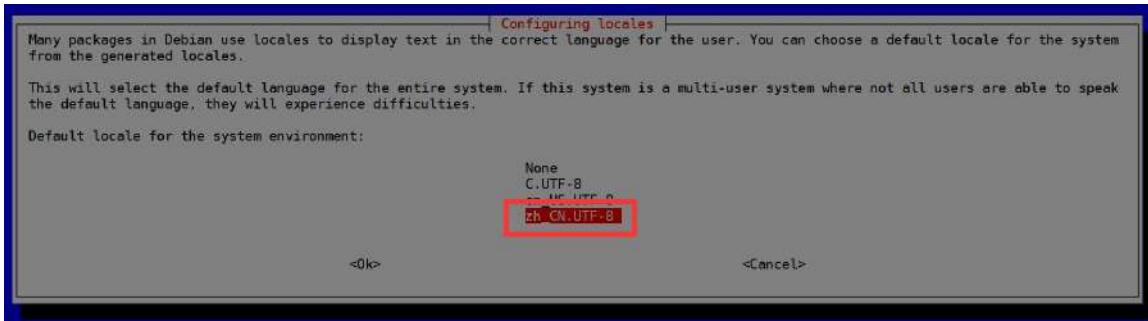
a. Enter the following command to start configuring **locale**

```
orangeipi@orangeipi:~$ sudo dpkg-reconfigure locales
```

b. Then select **zh\_CN.UTF-8 UTF-8** in the pop-up interface (move up and down through the up and down direction keys on the keyboard, select through the space bar, and finally use the Tab key to move the cursor to **<OK>**, and then return to car)



c. Then set the default **locale** to **zh\_CN.UTF-8**



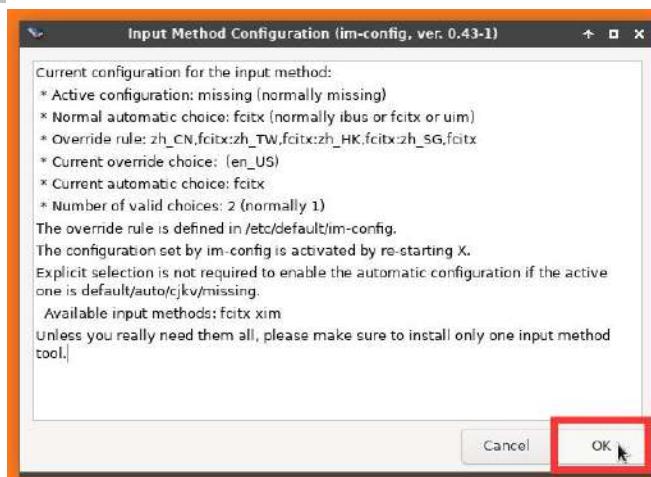
- d. After exiting the interface, the **locale** setting will start, and the output displayed on the command line is as follows

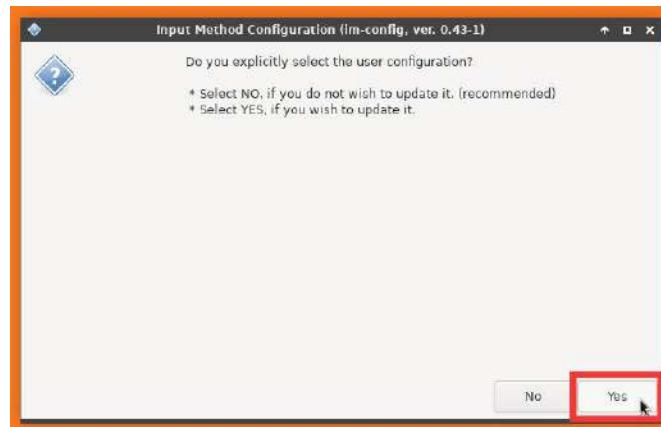
```
orangeipi@orangeipi:~$ sudo dpkg-reconfigure locales
Generating locales (this might take a while)...
en_US.UTF-8... done
zh_CN.UTF-8... done
Generation complete.
```

- 2) Then install the following packages

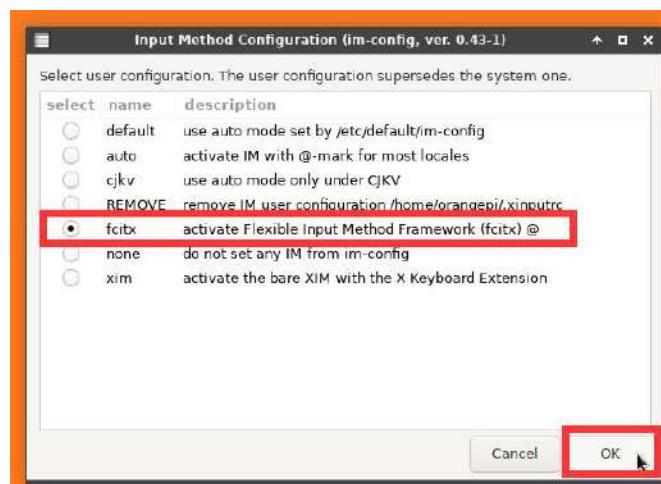
```
orangeipi@orangeipi:~$ sudo apt update
orangeipi@orangeipi:~$ sudo apt install -y fonts-archic-bsmi00lp \
fonts-archic-gbsn00lp fonts-archic-gkai00mp fcitx fcitx-table* \
fcitx-frontend-gtk* fcitx-frontend-qt* fcitx-config-gtk* im-config \
fcitx-googlepinyin fcitx-ui* zentity geany
```

- 3) Then open **Input Method**

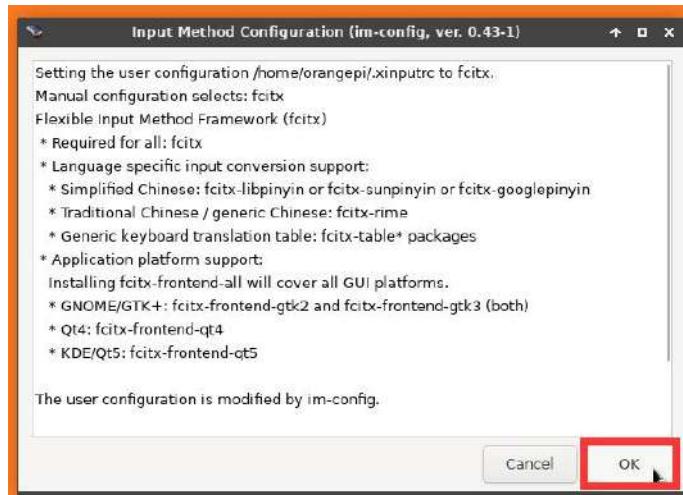
4) Then select **OK**5) Then select **Yes**



6) Then select **fcitx**



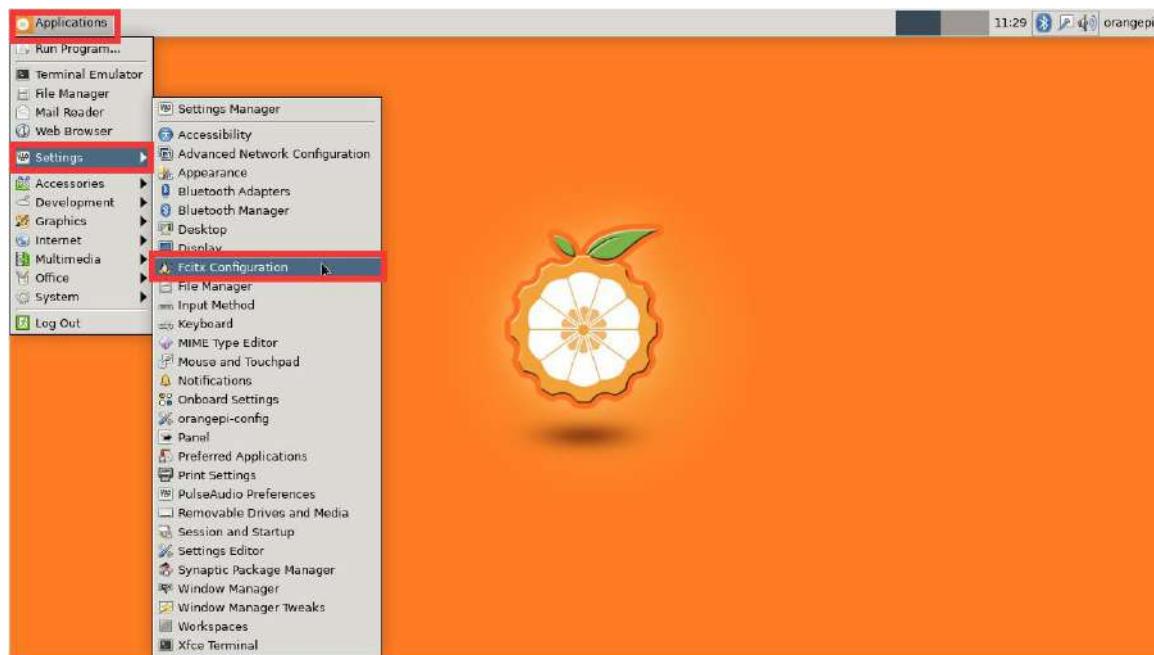
7) Then select **OK**



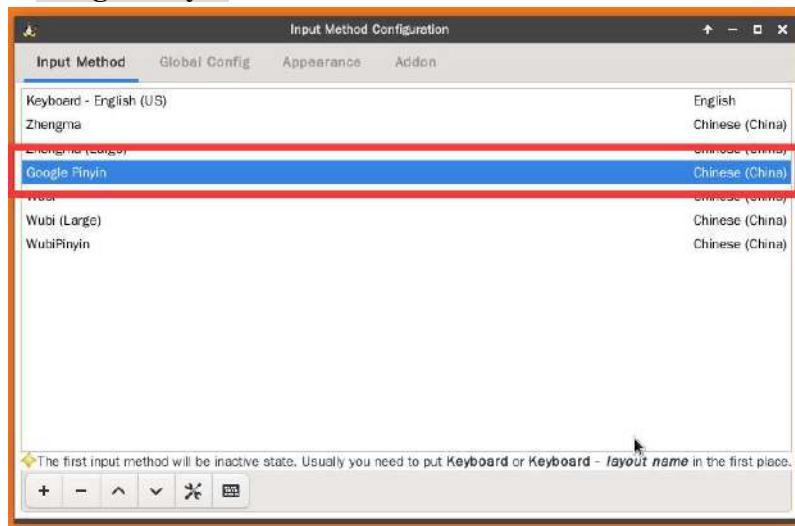


8) Then restart the Linux system for the configuration to take effect

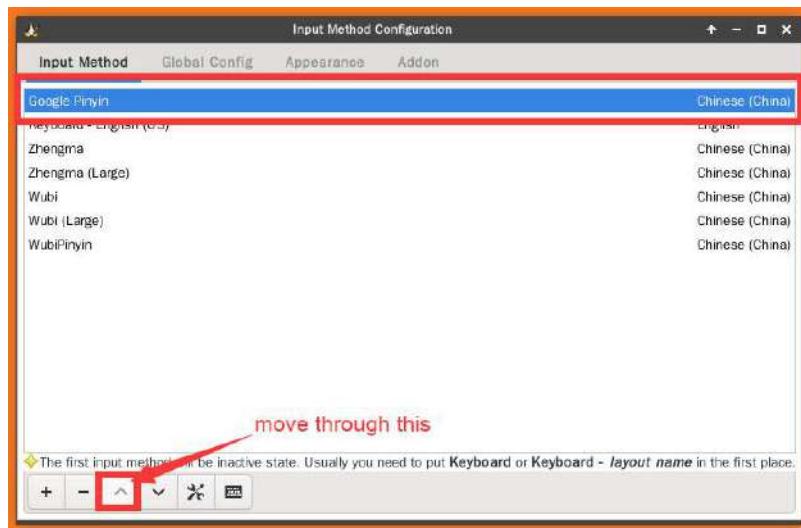
9) Then open **Fcitx Configuration**



10) Then select **Google Pinyin**



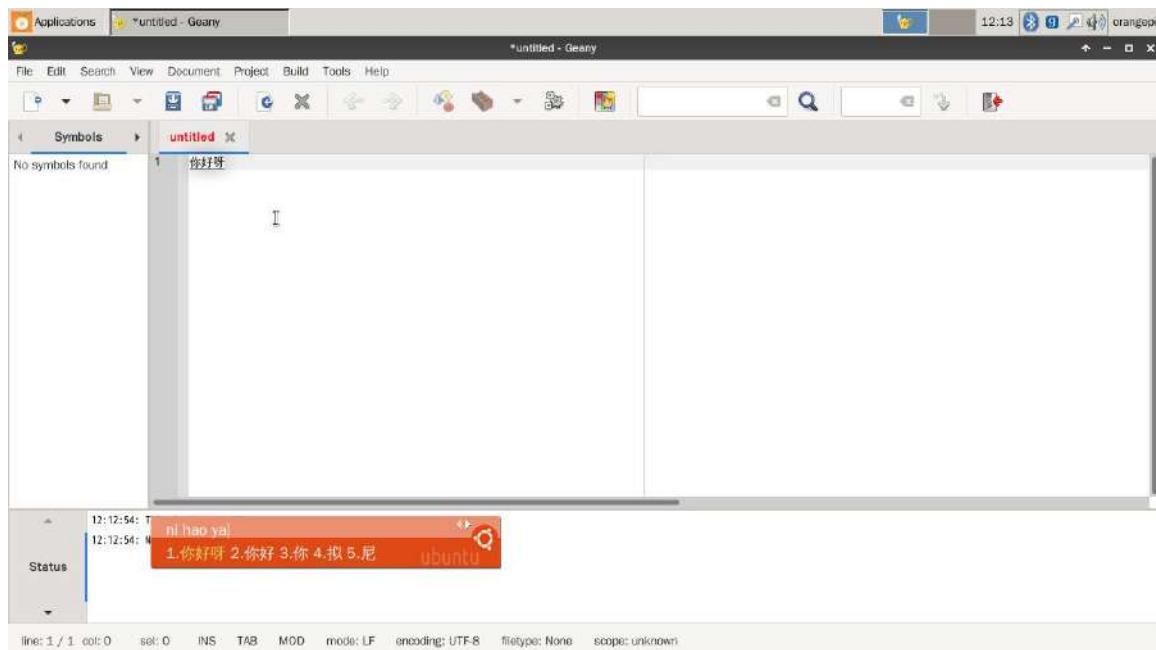
11) Then bring **Google Pinyin** to the front, and then close the configuration interface in the upper right corner



12) Then open the **Geany** editor to test the Chinese input method



13) The Chinese input test is as follows



- 14) You can switch back to English input in the upper right corner of the desktop
- First place the mouse cursor on the penguin position as shown in the figure below



- Then click the **right mouse button** to see the following options, and then select **Keyboard-English (US)** to switch back to English input

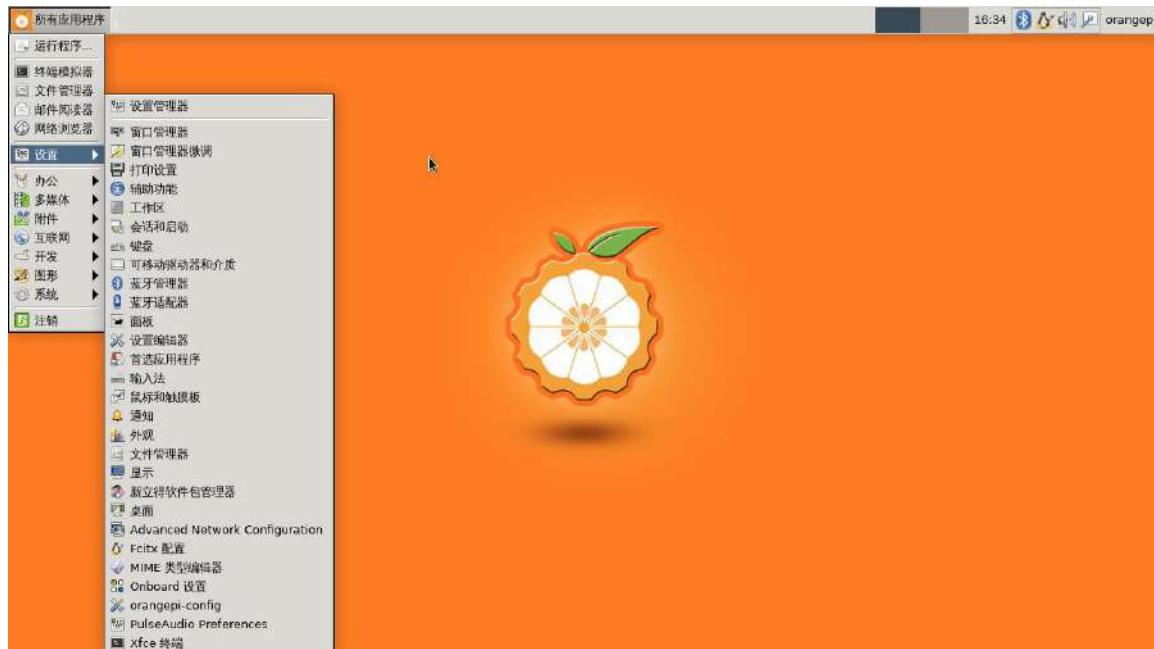


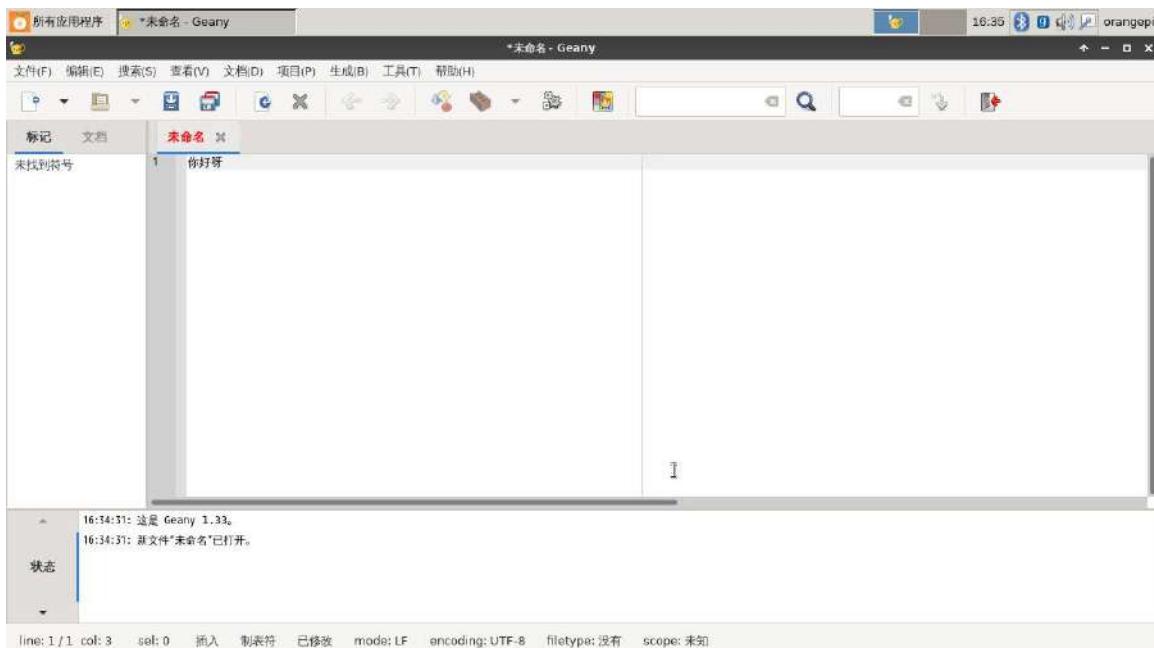
15) At this point, you can switch between Chinese and English input methods through the **Ctrl+Space** shortcut key

16) If you need the entire system to be displayed in Chinese, you can set the variables in **/etc/default/locale** to **zh\_CN.UTF-8**

```
orangeipi@orangeipi:~$ sudo vim /etc/default/locale
# File generated by update-locale
LC_MESSAGES=zh_CN.UTF-8
LANG=zh_CN.UTF-8
LANGUAGE=zh_CN.UTF-8
```

17) Then **restart the system** to see that the system is displayed in Chinese





### 3. 23. View the chipid of the H6 chip

1) The command to view the chipid of the h6 chip is as follows, the chipid of each chip is different, so you can use the chipid to distinguish multiple development boards

```
orangepi@orangepi:~$ cat /sys/class/sunxi_info/sys_info | grep "chipid"
sunxi_chipid : 38641f0f0141410900004c0000000000
```

### 3. 24. How to program linux image to eMMC

If there is a problem with startup after burning, you can use the following command to clear the eMMC first, and then try burning again:

1. First insert the TF card, then start the system, and then use the `ls /dev/mmcblk*` command to view all mmc device nodes in the system, eMMC generally has two boot partitions, and there is no TF card, through `/dev/mmcblk2boot1` You can know that the block device of eMMC is `/dev/mmcblk2` (not all systems must be mmcblk2, please refer to what you actually see). `/dev/mmcblk2p1` represents the partition in eMMC (the Android system in eMMC generally has more than a dozen partitions), the partition is not used here, just know it.



```
orangepi@orangepi:~$ ls /dev/mmcblk*
/dev/mmcblk0  /dev/mmcblk0p1  /dev/mmcblk2  /dev/mmcblk2boot0
/dev/mmcblk2boot1  /dev/mmcblk2p1
```

**2. Now that we know that the block device of eMMC is `/dev/mmcblk2`, we can use the following command to clear eMMC:**

```
orangepi@orangepi:~$ sudo dd bs=1M if=/dev/zero of=/dev/mmcblk2  \
count=1000
```

The above command `of=` specifies the device node of the eMMC, not a partition. If the eMMC is an Android system, all the partitions of the Android system will be cleared. Please do not have the illusion that you need to clear more than a dozen Android partitions one by one. . `count=1000` will clear the first 1GB of eMMC space. If you think it is not enough, you can increase this value.

**Note that the development board can be started through TF card or eMMC. The priority of TF card is higher than that of eMMC. That is to say, if a TF card is inserted into the development board, and there is a system in the TF card, the system in the TF card will be activated by default, but the system in eMMC will not be activated.**

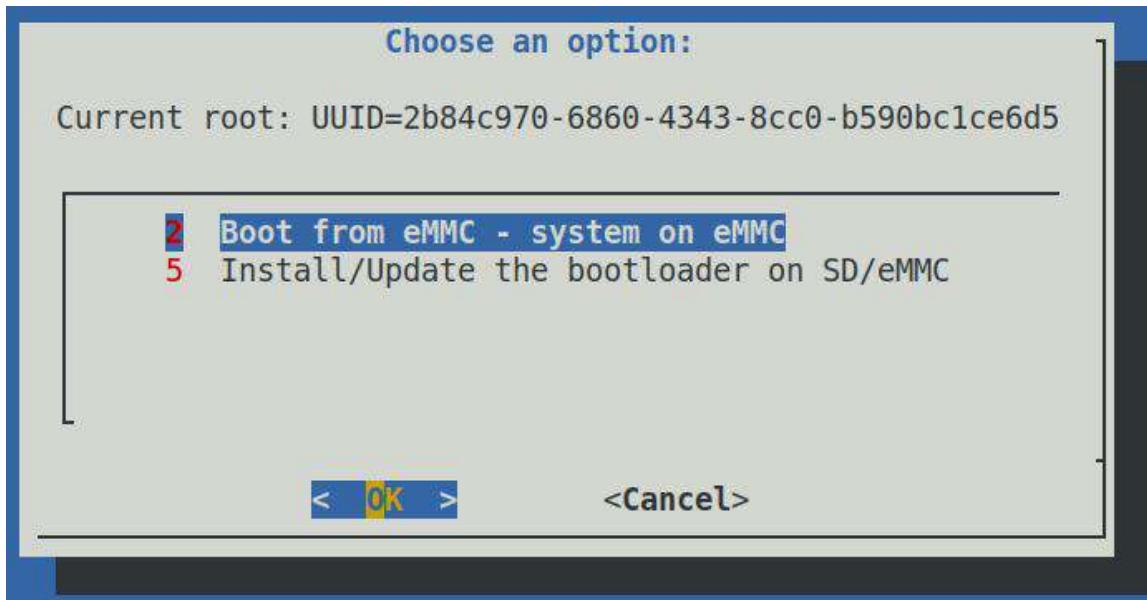
1) Burning the linux image to eMMC needs to be done with the help of a TF card, first burn the linux image to the TF card, and then start the development board to enter the linux system

2) Then run the `nand-sata-install` script, **remember to add sudo permissions**

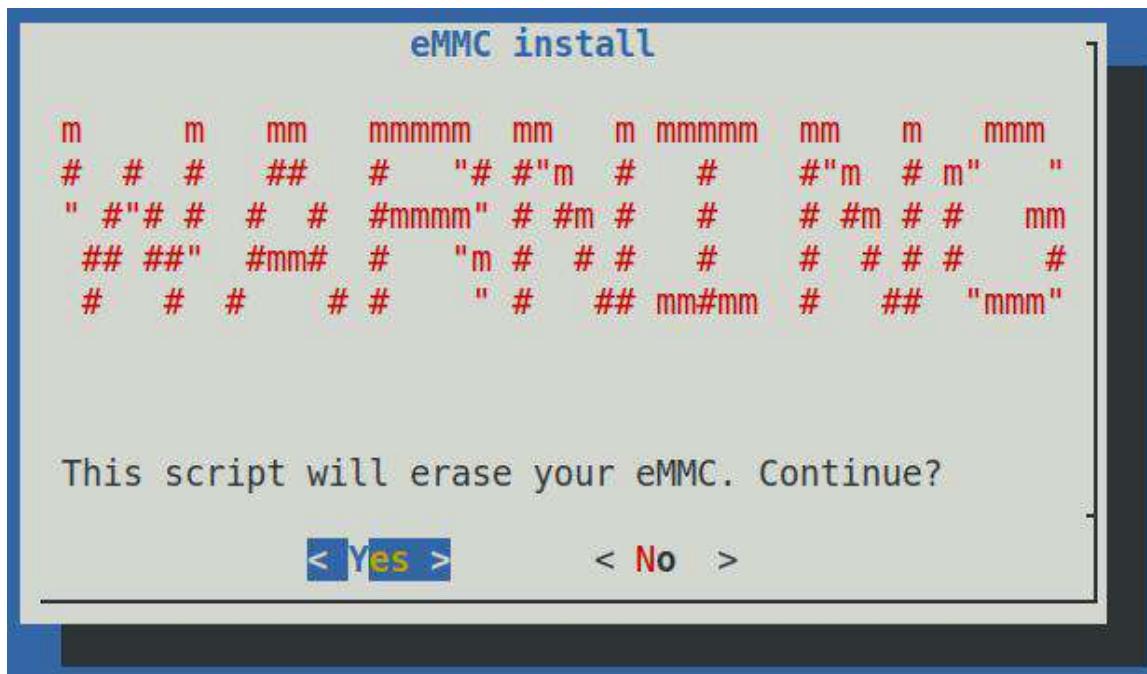
```
orangepi@orangepi:~$ sudo nand-sata-install
```

**If you are prompted that the command cannot be found, please check whether sudo is added, or switch to the root user to run.**

3) Then select **2 Boot from eMMC - sysystem on eMMC**



- 4) Then a warning will pop up, the script will erase all data on the eMMC, select <Yes> to continue

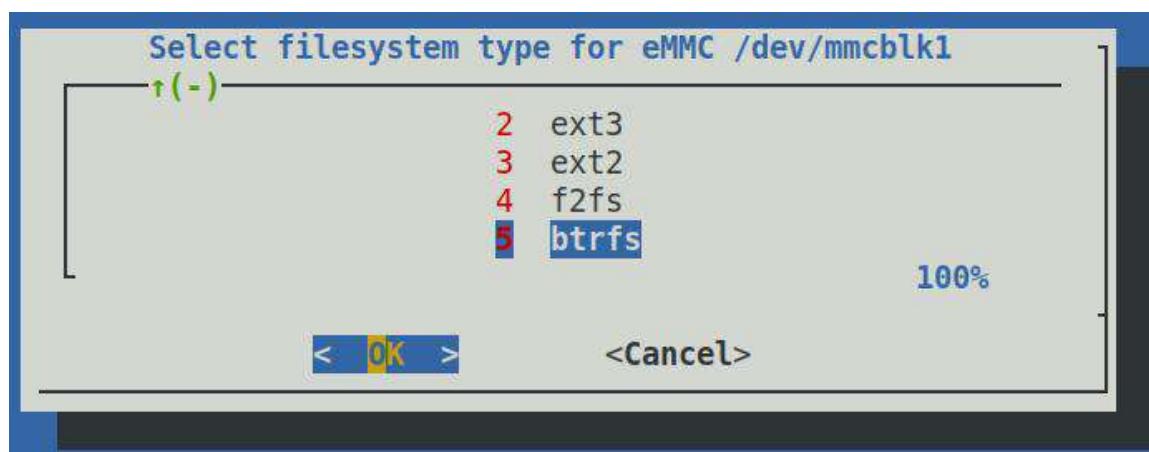
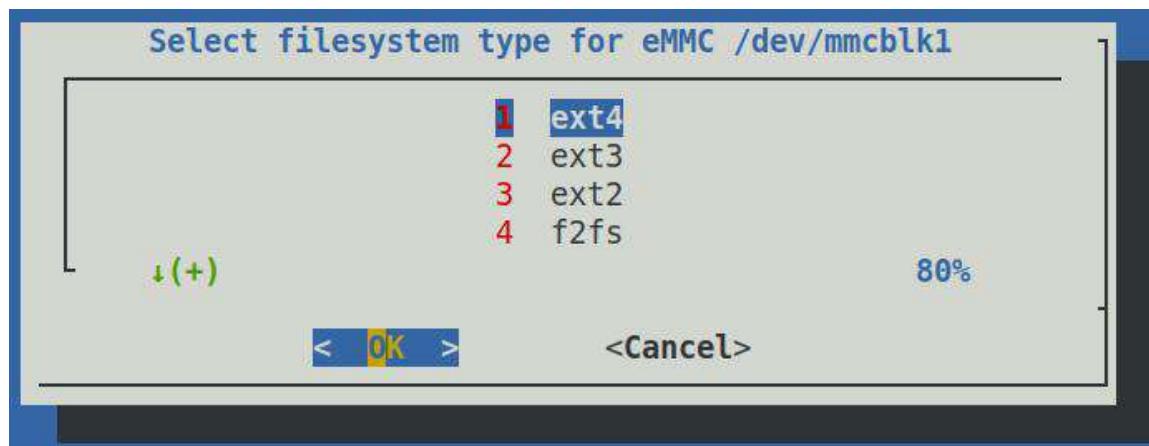


- 5) Then you will be prompted to select the type of file system, which supports ext2/3/4, f2fs and btrfs five file systems

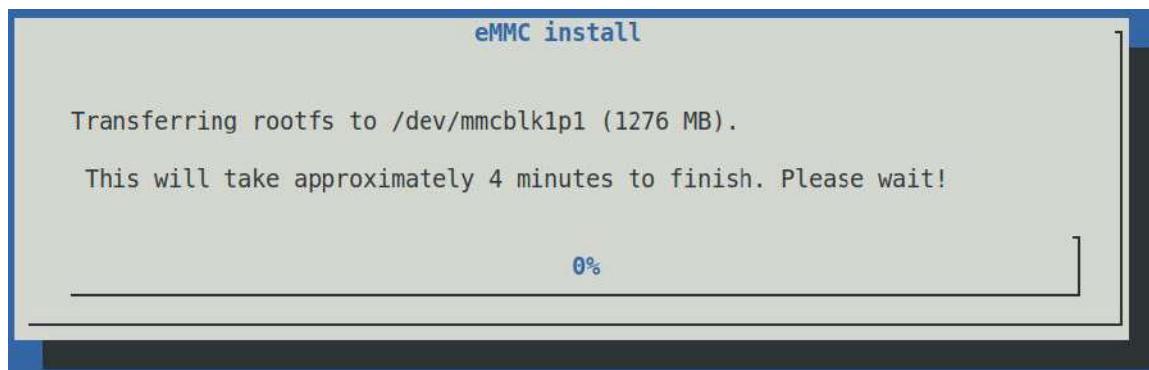
**Note that you can choose the ext4 file system without special requirements. If you are using the image of the linux4.9 kernel, please do not choose btrfs and f2fs**



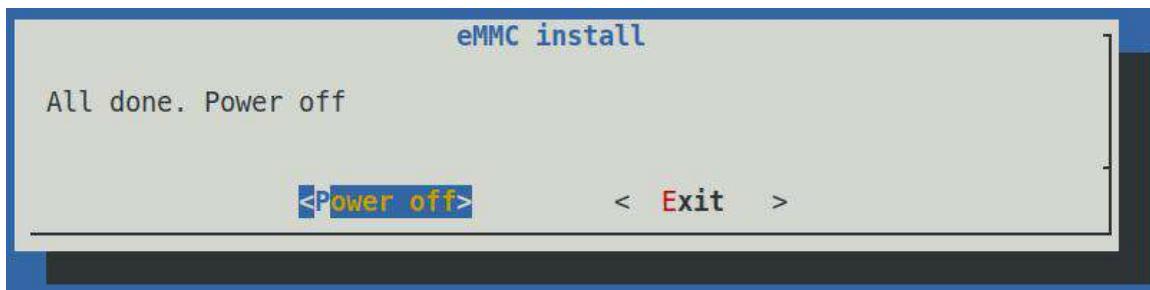
## file systems.



- 6) Then it will start to format eMMC. After formatting eMMC, it will start to burn the linux image into eMMC.



- 7) After burning, the following options will be prompted, you can select <Power off> to directly shut down



- 8) Then pull out the TF card, and then power on again, it will start the linux system in eMMC

### 3. 25. The usage of eMMC storage space and memory after the Linux system is started

- 1) For the server version image, the usage of eMMC storage space and memory after the system is started is as follows

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            925M    0  925M  0% /dev
tmpfs           199M  5.5M  194M  3% /run
/dev/mmcblk2p1  7.1G  1.3G  5.4G  20% /
tmpfs           995M    0  995M  0% /dev/shm
tmpfs            5.0M  4.0K  5.0M  1% /run/lock
tmpfs           995M    0  995M  0% /sys/fs/cgroup
tmpfs           995M  4.0K  995M  1% /tmp
/dev/zram0       49M   1.4M   44M   4% /var/log
tmpfs           199M    0  199M  0% /run/user/1000
orangepi@orangepi:~$
orangepi@orangepi:~$ free -h
              total        used        free      shared  buff/cache available
Mem:      1.9Gi     118Mi     1.7Gi      5.0Mi     102Mi
          1.8Gi
Swap:     994Mi        0B     994Mi
orangepi@orangepi:~$
orangepi@orangepi:~$ lsb_release -a
```



No LSB modules are available.

Distributor ID: Debian

Description: Debian GNU/Linux 10 (buster)

Release: 10

Codename: buster

orangeipi@orangeipi:~\$

orangeipi@orangeipi:~\$ **uname -r**

5.10.75-sunxi64

- 2) For the desktop version image, the usage of eMMC storage space and memory after the system is started is as follows

orangeipi@orangeipi:~\$ **df -h**

| Filesystem            | Size        | Used        | Avail       | Use%       | Mounted on     |
|-----------------------|-------------|-------------|-------------|------------|----------------|
| udev                  | 968M        | 0           | 968M        | 0%         | /dev           |
| tmpfs                 | 198M        | 8.8M        | 189M        | 5%         | /run           |
| <b>/dev/mmcblk0p1</b> | <b>7.1G</b> | <b>3.5G</b> | <b>3.3G</b> | <b>52%</b> | <b>/</b>       |
| tmpfs                 | 987M        | 0           | 987M        | 0%         | /dev/shm       |
| tmpfs                 | 5.0M        | 4.0K        | 5.0M        | 1%         | /run/lock      |
| tmpfs                 | 987M        | 0           | 987M        | 0%         | /sys/fs/cgroup |
| tmpfs                 | 987M        | 32K         | 987M        | 1%         | /tmp           |
| /dev/zram0            | 49M         | 2.8M        | 43M         | 7%         | /var/log       |
| tmpfs                 | 198M        | 12K         | 198M        | 1%         | /run/user/1000 |

orangeipi@orangeipi:~\$

orangeipi@orangeipi:~\$ **free -h**

|              | total        | used         | free         | shared      | buff/cache   |
|--------------|--------------|--------------|--------------|-------------|--------------|
| available    |              |              |              |             |              |
| Mem:         | <b>1.9Gi</b> | <b>350Mi</b> | <b>1.3Gi</b> | <b>12Mi</b> | <b>287Mi</b> |
| <b>1.5Gi</b> |              |              |              |             |              |
| Swap:        | 986Mi        | 0B           | 986Mi        |             |              |

orangeipi@orangeipi:~\$

orangeipi@orangeipi:~\$ **lsb\_release -a**

No LSB modules are available.

Distributor ID: Ubuntu

Description: Ubuntu 20.04.3 LTS

Release: 20.04

Codename: focal



```
orangeipi@orangeipi:~$  
orangeipi@orangeipi:~$ uname -r  
4.9.118-sun50iw6
```

### 3. 26. The method of modifying the picture displayed in the startup phase of the Linux 4.9 system

- 1) The following picture will be displayed on the HDMI display during the U-boot startup phase of the Linux 4.9 system



- 2) The location of this picture in the linux system is as follows

```
root@orangeipi:~$ ls /boot/boot.bmp  
/boot/boot.bmp
```

- 3) The location in the source code of orangepi-build is as follows. When compiling the linux system, the script in orangepi-build will copy orangepi-u-boot.bmp to the **/boot** directory of the final linux system and rename it to **boot.bmp**

```
orangepi-build/external/packages/blobs/splash/orangepi-u-boot.bmp
```

- 4) The relevant properties of the boot.bmp picture are as follows. If you need to replace the boot.bmp, you only need to make the corresponding picture according to the following property values, and then replace the boot.bmp in the linux system in the TF card. If it is To compile the linux system by yourself, you only need to replace orangepi-u-boot.bmp in orangepi-build



### 3.27. The method of remotely logging in to the Linux system desktop

Compared with VNC, it is more recommended to use NoMachine to remotely log in to the Linux system desktop

#### 3.27.1. Remote login using NoMachine

Please make sure that the Ubuntu or Debian system installed on the development board is the **desktop version**. In addition, NoMachine also provides detailed usage documentation. It is strongly recommended to read through this documentation to familiarize yourself with the usage of NoMachine. The documentation links are as follows:

<https://knowledgebase.nomachine.com/DT10R00166>

NoMachine supports Windows, Mac, Linux, iOS and Android platforms, so we can remotely log in to control the Orange Pi development board through NoMachine on a variety of devices. The following demonstrates how to remotely log in to the Linux system desktop of the Orange Pi development board through NoMachine in Windows. For installation methods on other platforms, please refer to the official documentation of NoMachine.

Before operation, please make sure that the Windwos computer and the development board are in the same local area network, and can log in to the Ubuntu or Debian system of the development board normally through ssh.

- 1) First download the installation package of the Linux **arm64** deb version of the



NoMachine software, and then install it into the Linux system of the development board

- a. Since H6 is the SOC of the ARMv8 architecture, the system we use is Ubuntu or Debian, so here we need to download the **NoMachine for ARM ARMv8 DEB** installation package. The download link is as follows:

**Note that this download link may change, please look for the Armv8/Arm64 version of the deb package.**

<https://www.nomachine.com/download/download&id=116&s=ARM>

Home / Download / NoMachine for ARM - arm64

### NoMachine for ARM - arm64



Version: 7.9.2\_1  
Package size: 4223 MB  
Package type: DEB  
MD5 signature: 5d4c4b4a1f17569fc5918296fe39156  
For: Ubuntu 14.04/16.04/18.04/20.04, Debian 8/9/10



Although your ARMv8 device may not be listed here, we encourage you to try the packages. Please consult the installation and configuration notes about Linux for ARM packages for more details about devices and specific distributions we have tested.

[Download](#)

- b. In addition, the **NoMachine** installation package can also be downloaded in the official tool

## Downloads



Ubuntu Image

[Downloads](#)



Debian Image

[Downloads](#)



Android Image

[Downloads](#)



Android Source Code

[Downloads](#)



Linux Source code

[Downloads](#)



User Manual

[Downloads](#)



Office Tools

[Downloads](#)



|                                     |  |                                  |       |
|-------------------------------------|--|----------------------------------|-------|
| <input type="checkbox"/>            |  | VNC-Viewer-6.21.1109-Windows.exe | 11.3M |
| <input checked="" type="checkbox"/> |  | nomachine_7.9.2_1_arm64.deb      | 42.2M |
| <input type="checkbox"/>            |  | nomachine_7.9.2_1_amd64.deb      | 45.6M |
| <input type="checkbox"/>            |  | nomachine_7.9.2_1.exe            | 34.4M |
| <input type="checkbox"/>            |  | nomachine_7.9.2_1.dmg            | 45.2M |

Please download the installation package of the Arm64 version

- c. Then upload the downloaded **nomachine\_7.9.2\_1\_arm64.deb** to the Linux system of the development board
- d. Then use the following command to install **NoMachine** in the Linux system of the development board

```
orangeipi@orangeipi:~$ sudo dpkg -i nomachine_7.9.2_1_arm64.deb
```

```
[sudo] password for orangeipi:
```

```
Selecting previously unselected package nomachine.
```

```
(Reading database ... 182635 files and directories currently installed.)
```

```
Preparing to unpack nomachine_7.9.2_1_arm64.deb ...
```

```
Unpacking nomachine (7.9.2-1) ...
```

```
Setting up nomachine (7.9.2-1) ...
```

```
NX> 700 Starting install at: Sun Apr 17 10:52:07 2022.
```

```
NX> 700 Installing: nxclient version: 7.9.2.
```

```
NX> 700 Using installation profile: Debian.
```

```
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
```

```
NX> 700 Compiling the USB module.
```

```
NX> 700 Installing: nxplayer version: 7.9.2.
```

```
NX> 700 Using installation profile: Debian.
```

```
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
```

```
NX> 700 To connect the remote printer to the local desktop,
```

```
NX> 700 the user account must be a member of the CUPS System Group: lpadmin.
```

```
NX> 700 Installing: nxnode version: 7.9.2.
```

```
NX> 700 Using installation profile: Debian.
```

```
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
```

```
NX> 700 Creating configuration in: /usr/NX/etc/node.cfg.
```

```
NX> 700 Installing: nxserver version: 7.9.2.
```

```
NX> 700 Using installation profile: Debian.
```

```
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
```

```
NX> 700 Creating configuration in: /usr/NX/etc/server.cfg.
```

```
NX> 700 Install completed at: Sun Apr 17 10:53:00 2022.
```



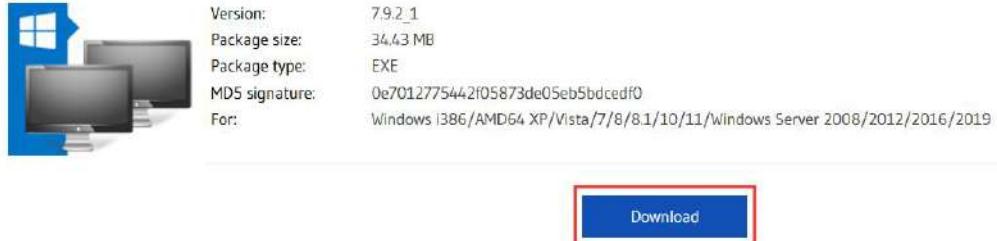
NX> 700 NoMachine was configured to run the following services:  
NX> 700 NX service on port: 4000

- 2) Then download the installation package of the Windows version of the NoMachine software, the download address is as follows

<https://www.nomachine.com/download/download&id=8>

Home / Download / NoMachine for Windows

NoMachine for Windows



Version: 7.9.2\_1  
Package size: 34.43 MB  
Package type: EXE  
MD5 signature: 0e7012775442f05873de05eb5bdcedf0  
For: Windows (386/AMD64 XP/Vista/7/8/8.1/10/11/Windows Server 2008/2012/2016/2019)

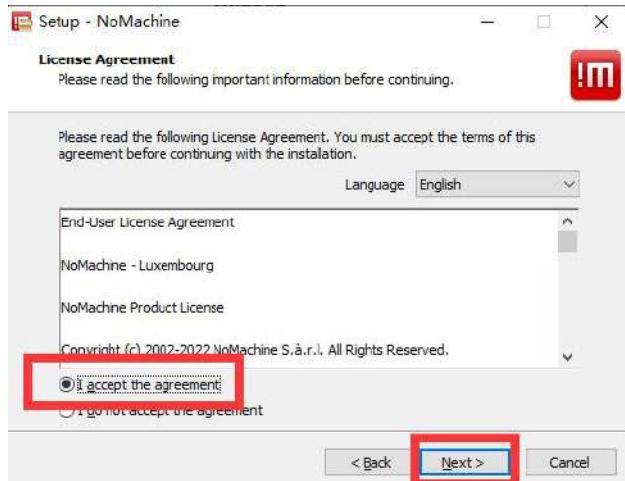
**Download**

- 3) Then install NoMachine in Windows

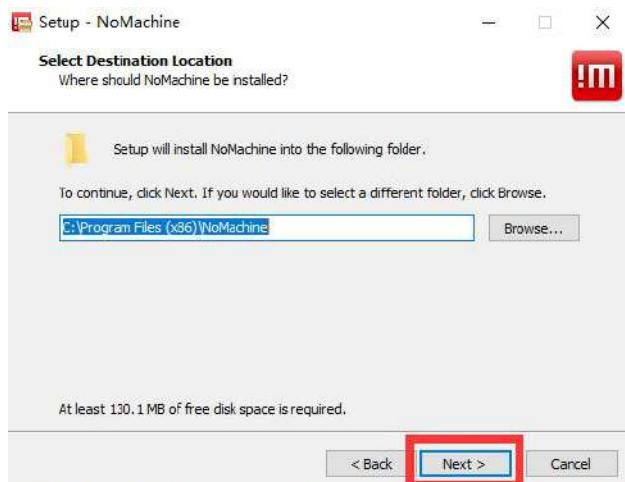
- a. Double-click NoMachine in Windows to start the installation of NoMachine, and then select **Next**



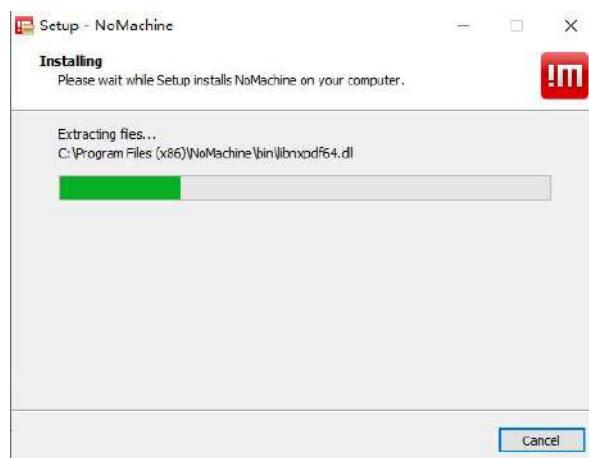
- b. Then select **I accept the agreement**, and then select **Next**



c. Then click **Next**



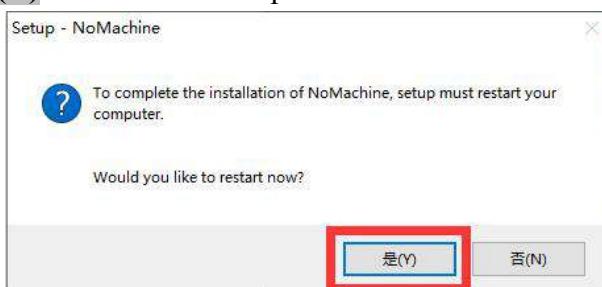
d. The installation process will then begin



e. After the installation is complete, the display is as shown in the figure below, and then click **Finish**.



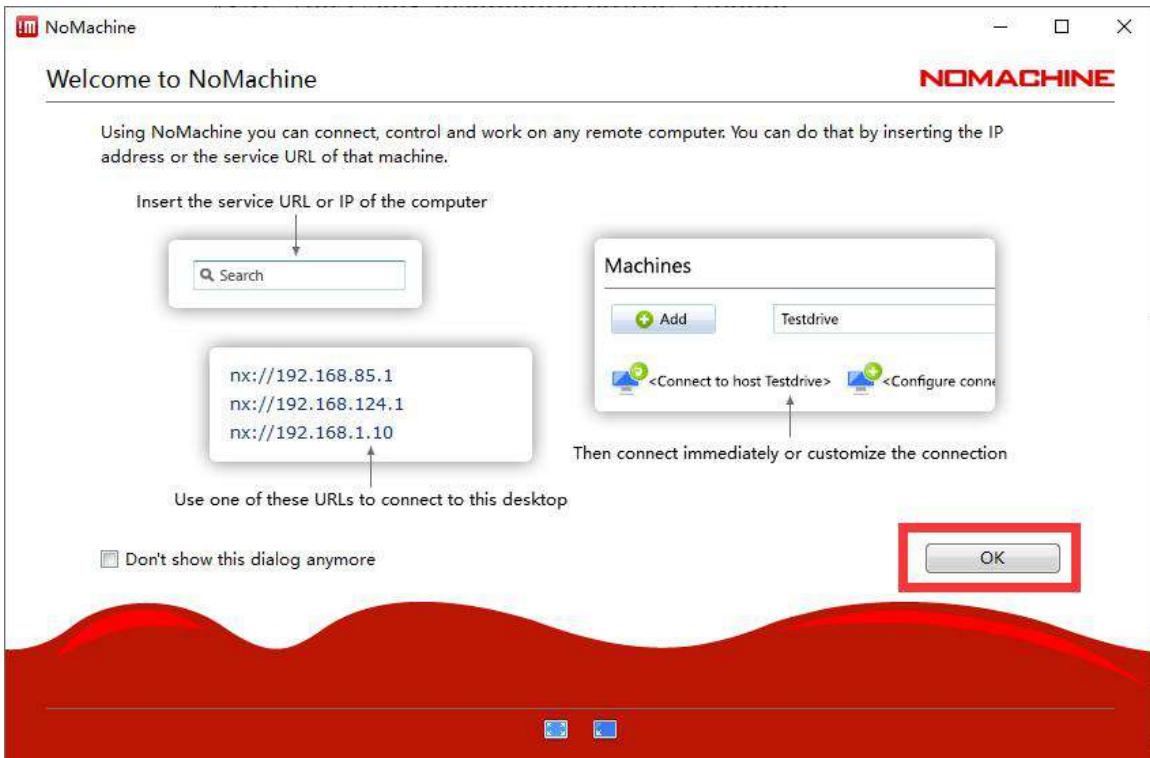
- f. Then NoMachine will prompt you to restart to complete the installation, here we choose **Yes (Y)** to restart the computer



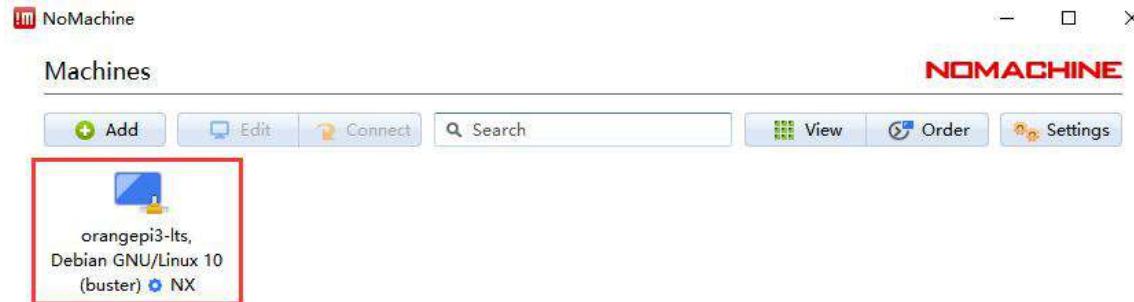
- 4) Then open **NoMachine** in Window double click



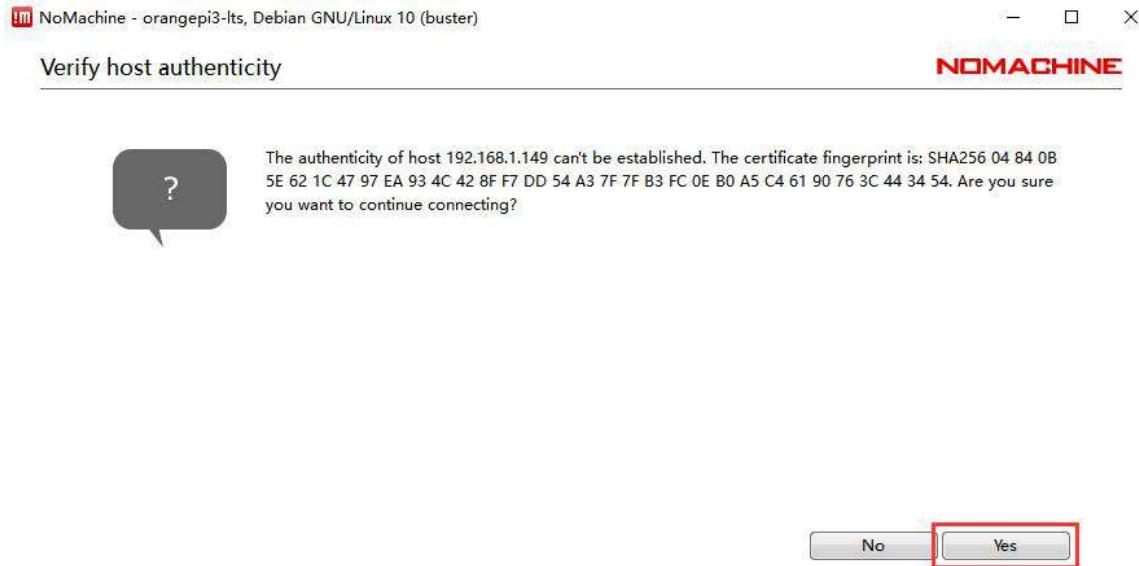
- 5) Then click **OK**



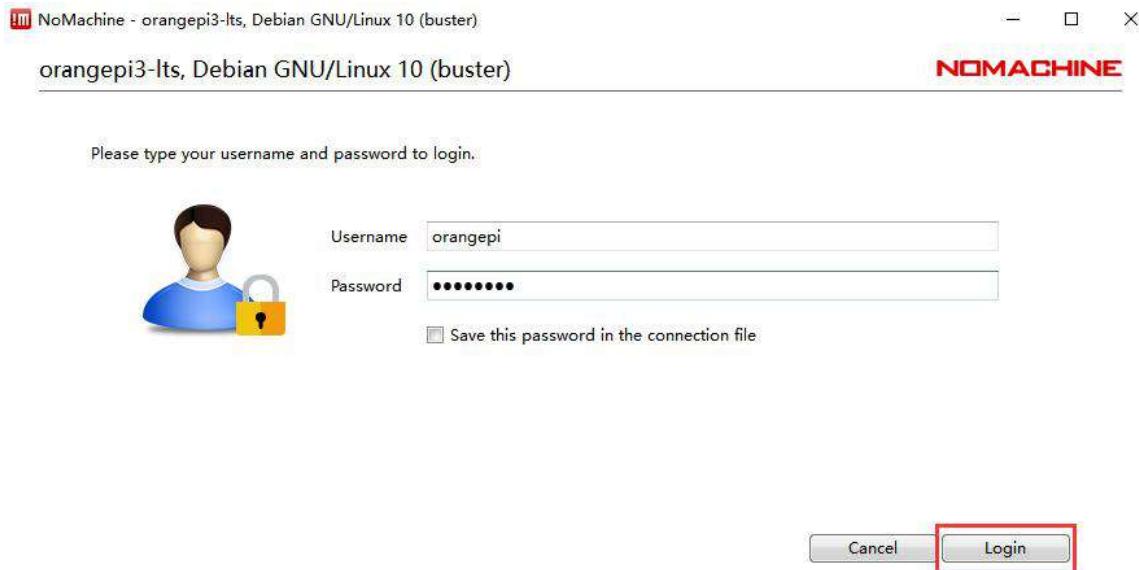
- 6) After NoMachine starts, it will automatically scan other devices with NoMachine installed in the local area network. After entering the main interface of NoMachine, you can see that the development board is already in the list of connectable devices, and then click the position shown in the red box in the figure below. You can start to log in to the Linux system desktop of the development board



- 7) Then click **Yes**



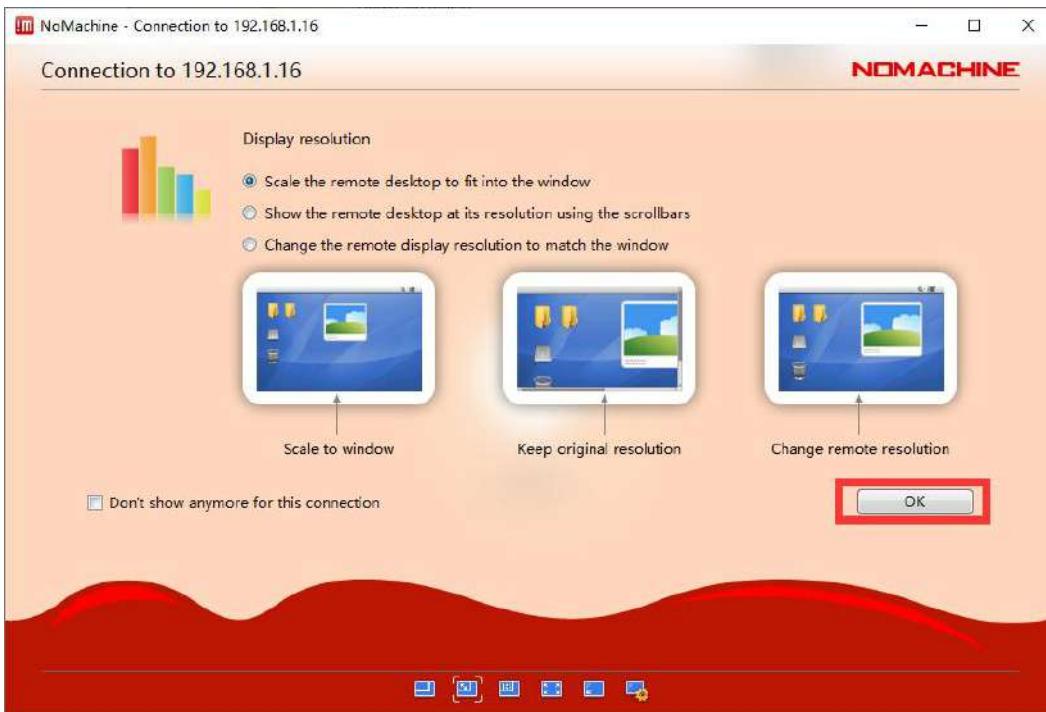
- 8) Then enter the user name **orangepi** and password **orangepi** of the Linux system of the development board in the corresponding position in the figure below, and then click **Login** to start logging in

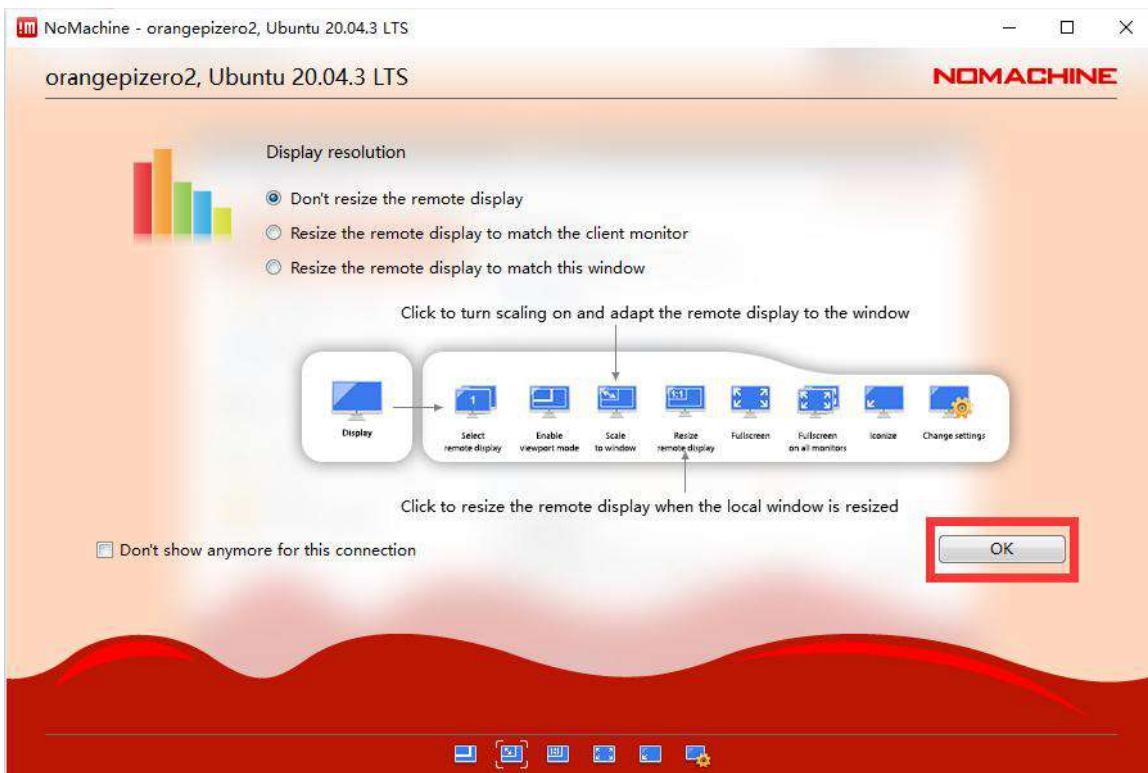


- 9) Then click OK



- 10) Then you can set the displayed resolution, select it as needed, and then click OK





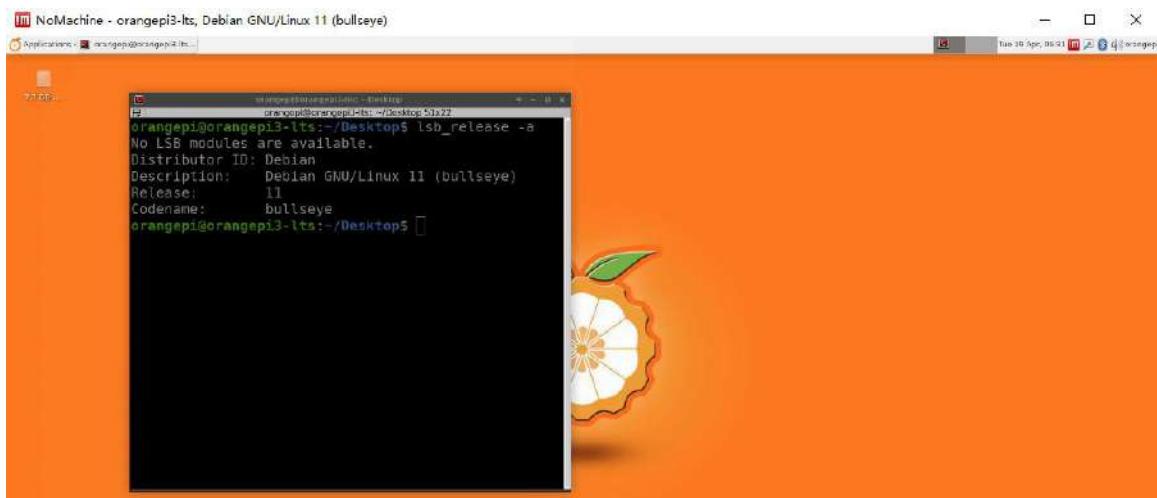
11) Finally, you can see the desktop of the Linux system of the development board

a. Debian Buster

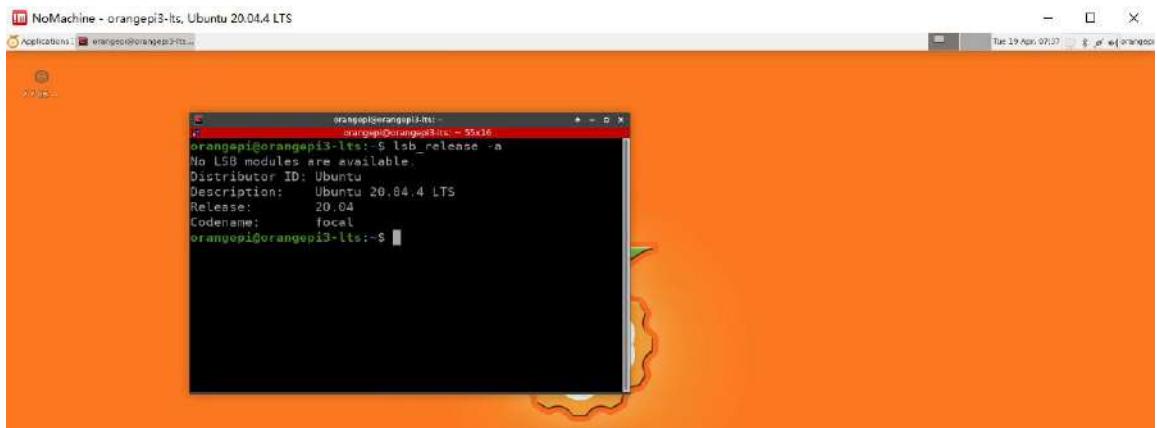




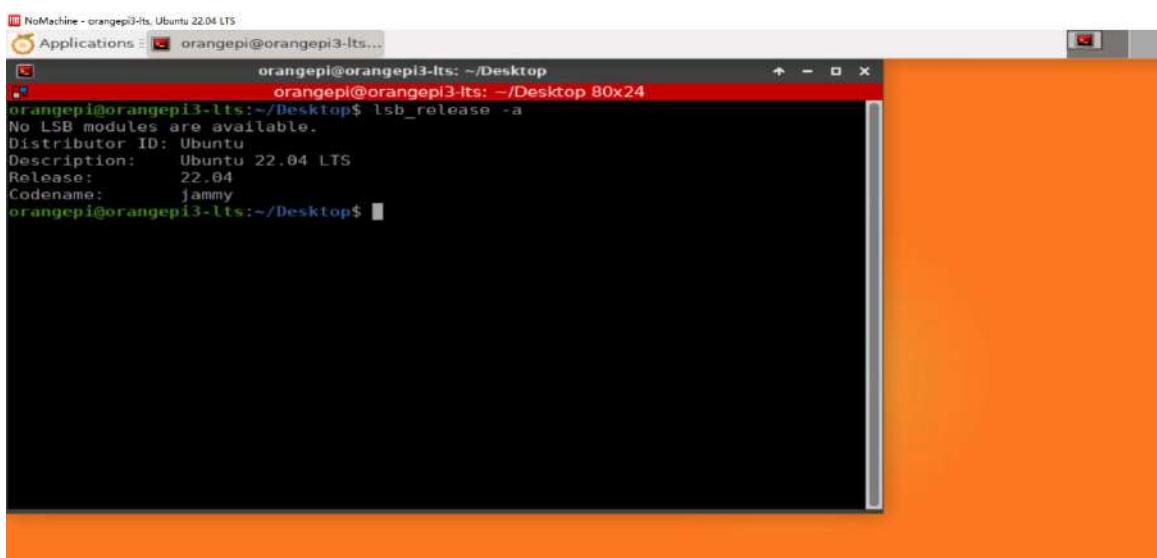
## b. Debian Bullseye



## c. Ubuntu Focal



## d. Ubuntu Jammy





### 3. 27. 2. Remote login using VNC

**Before operation, please make sure that the Windwos computer and the development board are in the same local area network, and can log in to the Ubuntu or Debian system of the development board normally through ssh**

- 1) First execute the following commands in the Linux system of the development board to install **tightvncserver** and **xrdp**

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt -y install tightvncserver xrdp
```

- 2) Then run the **vncserver** command in the Linux system of the development board to set the password (**the password is 6 to 8 characters**) and create the initial configuration file

```
orangepi@orangepi:~$ vncserver
```

You will require a password to access your desktops.

Password:

Verify:

Would you like to enter a view-only password (y/n)? **n**

New 'X' desktop is orangepi:1

```
Creating default startup script /home/orangepi/.vnc/xstartup  
Starting applications specified in /home/orangepi/.vnc/xstartup  
Log file is /home/orangepi/.vnc/orangepi:1.log
```

- 3) Then set the xstartup configuration file of vnc

- a. First deactivate the vnc server instance

```
orangepi@orangepi:~$ vncserver -kill :1
```

```
Killing Xtightvnc process ID 5337
```

- b. Then back up the xstartup configuration file

```
orangepi@orangepi:~$ mv ~/.vnc/xstartup ~/.vnc/xstartup.bak
```

- c. Then create a new xstartup configuration file and enter the following content in it

```
orangepi@orangepi:~$ vim ~/.vnc/xstartup
```



```
#!/bin/bash  
xrdb $HOME/.Xresources  
startxfce4 &
```

d. Then add executable permissions to xstartup

```
orangeipi@orangeipi:~$ chmod +x ~/.vnc/xstartup
```

e. Finally restart the vncserver server, then you can log in to the Linux desktop remotely through the vnc client

```
orangeipi@orangeipi:~$ vncserver
```

New 'X' desktop is orangeipi:1

Starting applications specified in /home/orangeipi/.vnc/xstartup

Log file is /home/orangeipi/.vnc/orangeipi:1.log

4) The steps to use the VNC Viewer client to connect to the Linux system desktop of the development board are as follows

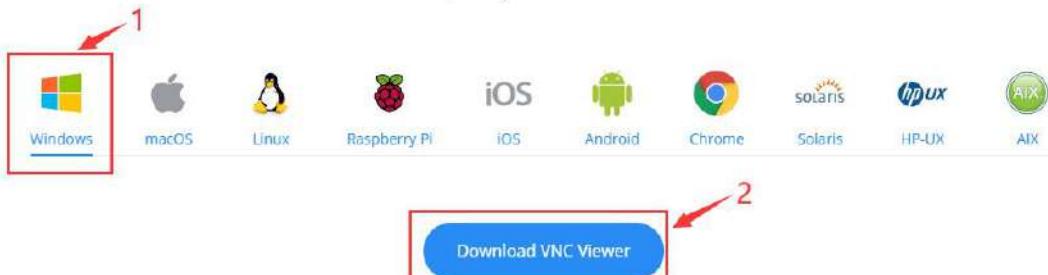
a. First download and install the VNC Viewer client on the Windows PC, the download link is as follows

<https://www.realvnc.com/en/connect/download/viewer/>

The screenshot shows the RealVNC website's 'VNC Connect' download page. At the top, there are navigation links for Service status, Products, Company, Contact us, EN, and Sign in. Below that is a main menu with options Discover, Pricing, Download, Support, and Partners. On the left, there's a 'VNC CONNECT' section. The main content area features a heading 'VNC® Connect consists of VNC® Viewer and VNC® Server'. Below it, a text instruction says: 'Download VNC® Viewer to the device you want to control from, below. Make sure you've installed VNC® Server on the computer you want to control.' To the right of the text are icons for various platforms: Windows, macOS, Linux, Raspberry Pi, iOS, Android, Chrome, Solaris, HP-UX, and AIX. A large red arrow labeled '1' points to the Windows icon. Another red arrow labeled '2' points to a blue 'Download VNC Viewer' button.

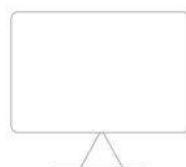
VNC® Connect consists of VNC® Viewer and VNC® Server

Download VNC® Viewer to the device you want to control from, below. Make sure you've [installed VNC® Server](#) on the computer you want to control.





- b. After installing the **VNC Viewer** client on the Windows PC, open the **VNC Viewer**, and then enter [**the IP address of the development board: 5901**] in the search bar of the VNC Viewer

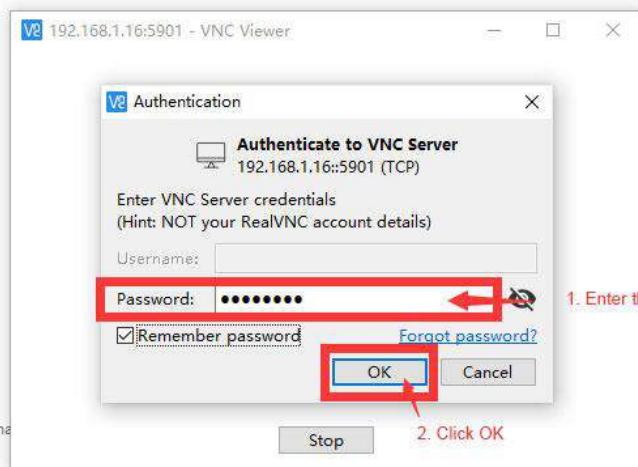


There are no computers in your address book at present.

Sign in to your RealVNC account to automatically discover team computers.

Alternatively, enter the VNC Server IP address or hostname in the Search bar to connect directly.

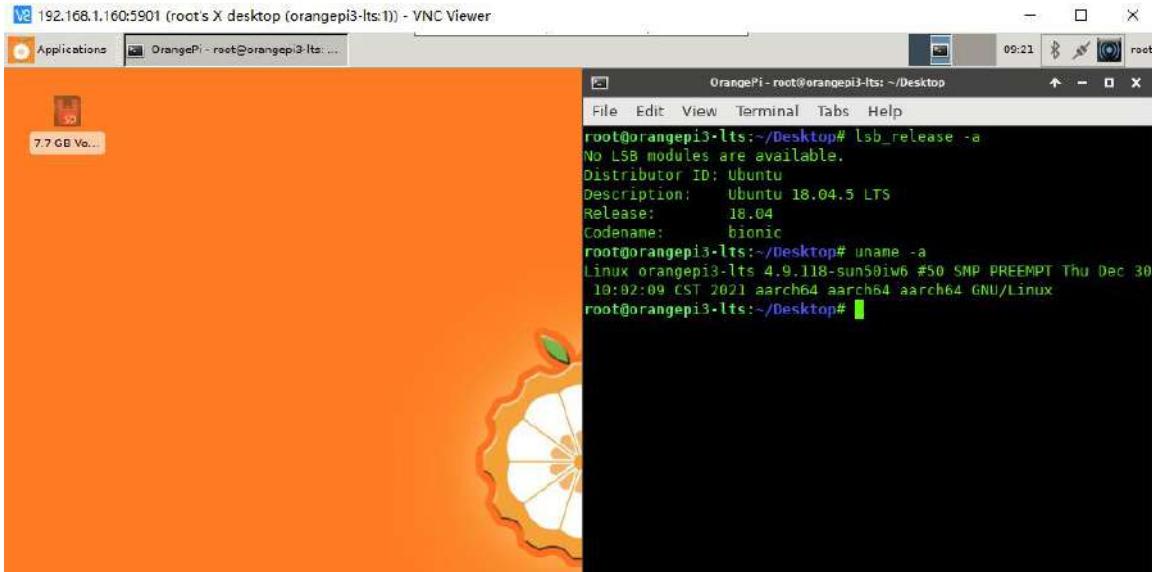
- c. Then enter the **password** set when running the vncserver command in step 2 in the Passowrd column, and then click **OK** to remotely log in to the Linux system desktop of the development board



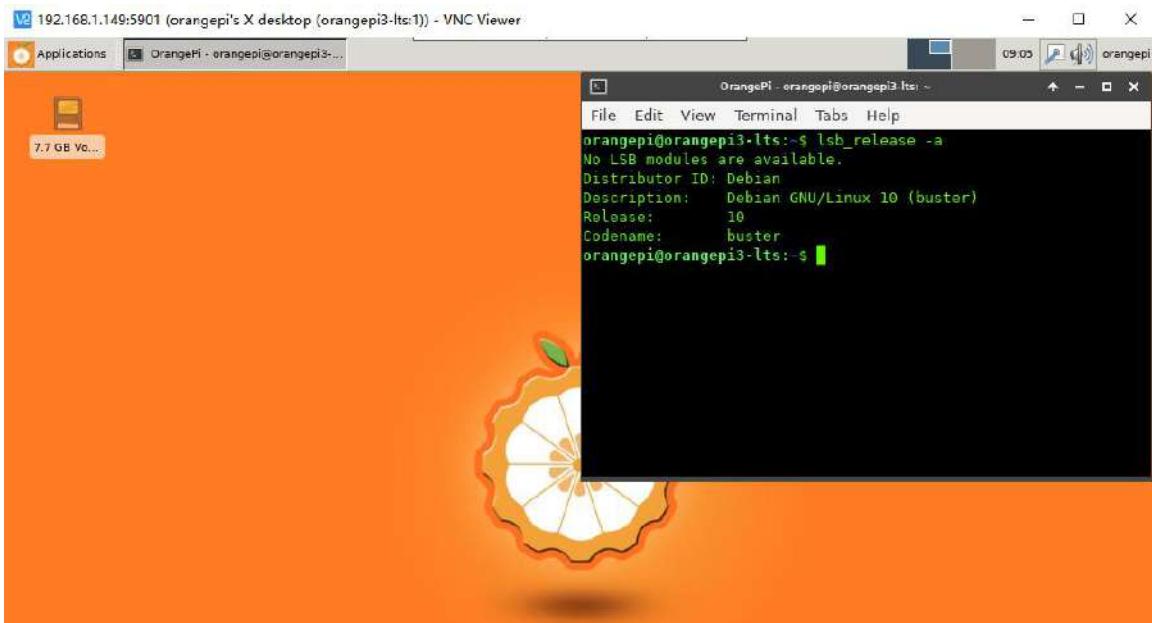


d. After the login is successful, the interface is displayed as shown in the figure below, and then the desktop of the Linux system of the development board can be remotely operated.

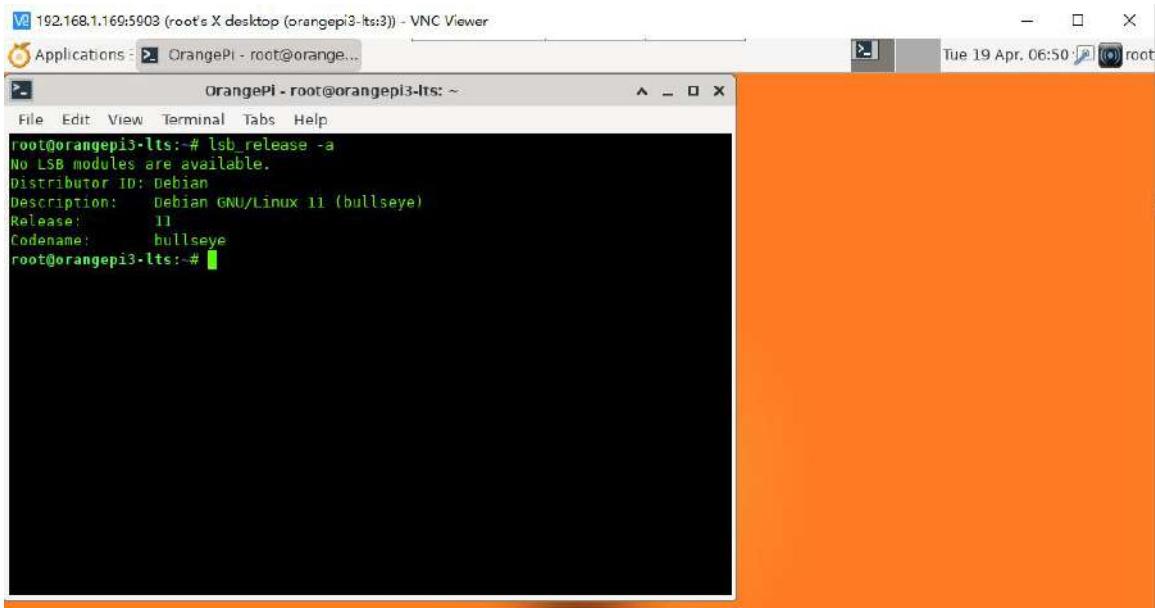
a) Ubuntu18.04 login display as shown below



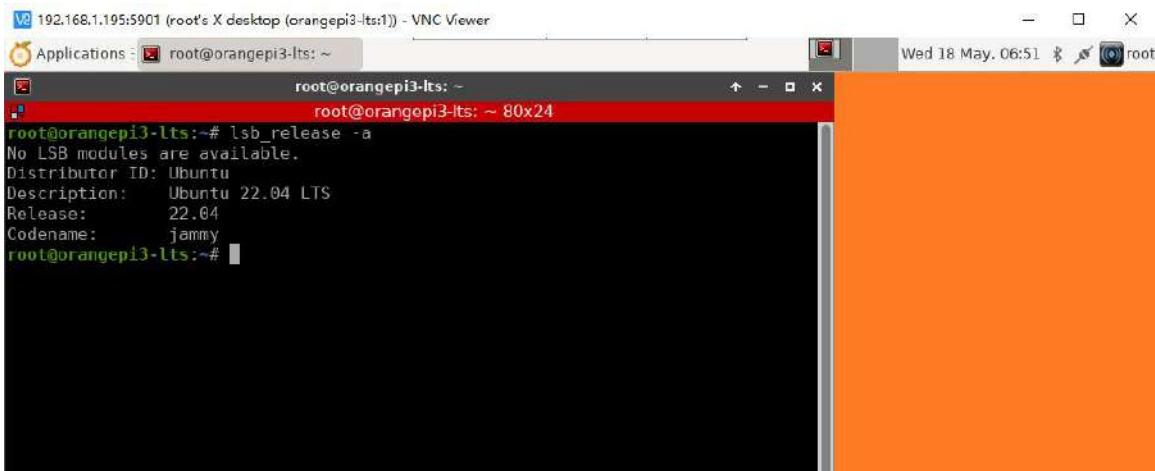
b) Debian10 login shows as below



c) Debian 11 is currently tested and can only log in with the root user



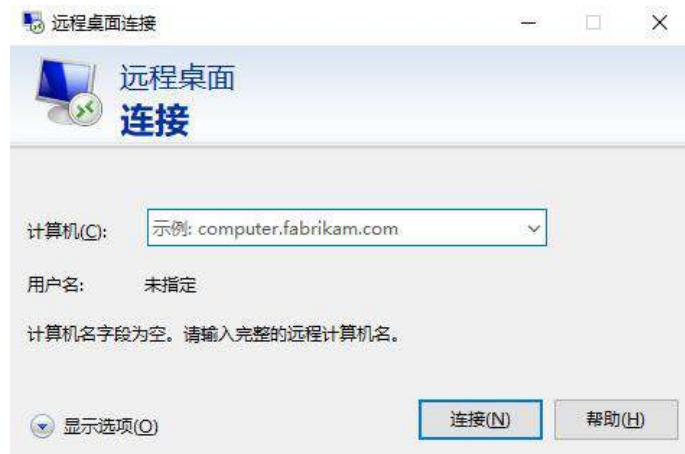
d) Ubuntu22.04 login display as shown below



e) Ubuntu20.04 currently has some problems in testing, please use NoMachine first

5) The steps to use the **remote desktop connection** application that comes with Windows to log in to the Linux system desktop of the development board are:

- First open the **remote desktop connection** that comes with Windows

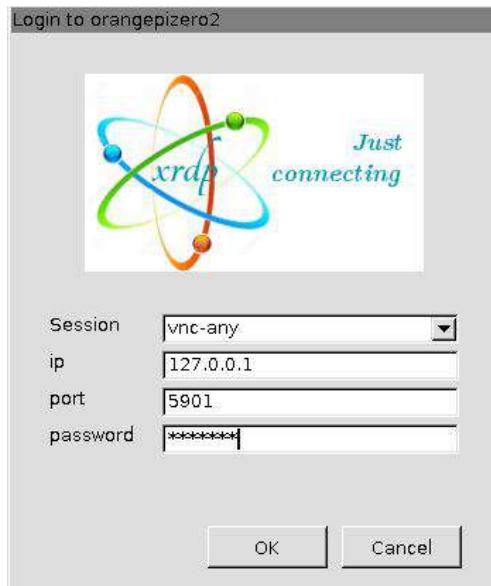


- b. Then enter the IP address of the development board

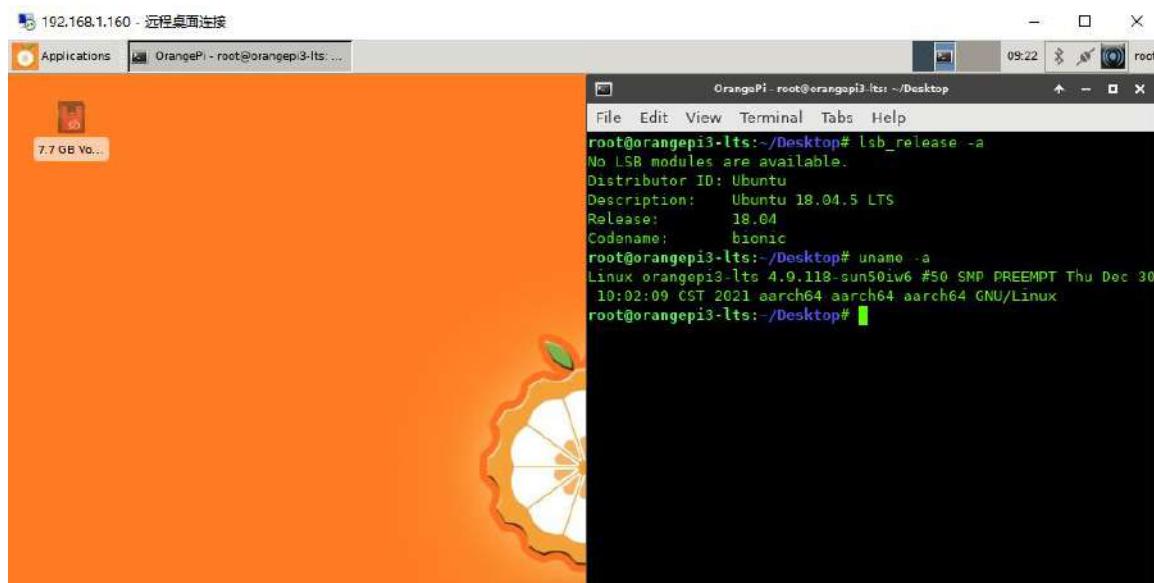


- c. Then set the connection information according to the following figure

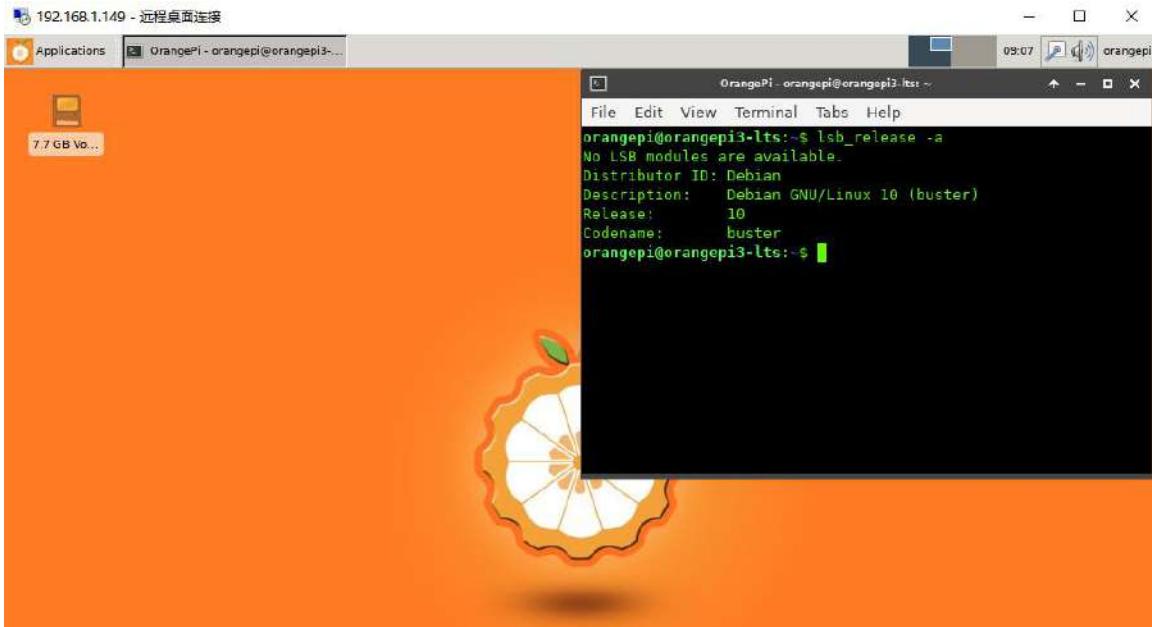
- a) **Session:** Need to select vnc-any
- b) **ip:** You can enter 127.0.0.1 or the IP address of the development board
- c) **port:** Usually 5901
- d) **password :** You need to enter the password you set when you ran the vncserver command in step 2



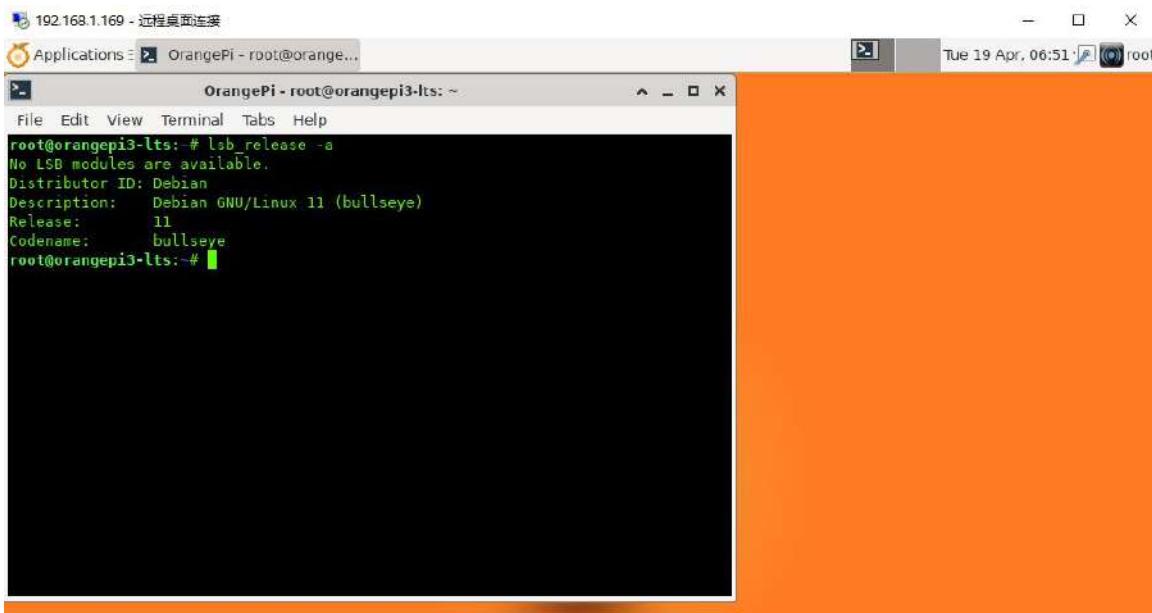
- d. The display of successfully logging in to the Linux system desktop of the development board is shown in the following figure
- a) Ubuntu18.04 login display as shown below



- b) Debian10 login shows as below



- c) The Debian11 login display is as follows, and the current test can only log in with the root user



### 3. 28. How to install Klipper firmware host computer using Kiauh

**Klipper is a 3D printer firmware. It can combine the functionality of the Orange Pi development board with one or more microcontrollers. For more information about Klipper, please refer to the [official documentation of Klipper](#).**

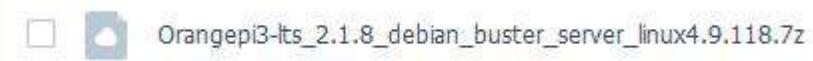


Kiauh is the abbreviation of **Klipper Installation And Update Helper**, which provides a very intuitive selection interface, and can complete the installation of Klipper and other software through simple selection. For more information about Kiauh, please refer to the [README](#) document in the Kiauh source code

This section will only demonstrate the process of using Kiauh to install the Klipper firmware host computer in the Linux system of the Orange Pi development board, and does not involve how to use the microcontroller or 3D printer.

Please make sure that the Linux system used by the development board is **Ubuntu Focal or Debian Buster**. Ubuntu Bionic will have problems because the default Python3 version does not meet the requirements.

If there are no special requirements, it is recommended to use the following images, which will encounter the least problems:



If you have never used a Linux system before, please read the [linux command format description section of this manual](#) carefully before starting to install Klipper, and then start the installation after figuring out which commands you really need to enter.

The default apt software source of the Linux system provided by Orange Pi has been set to Tsinghua source, so there is no need to replace the software source.

1) First, please make sure that the Linux system of the development board has been connected to WIFI or wired network, and then log in to the Linux system remotely through ssh. In Windows system, it is recommended to use the software MobaXterm to remotely log in to the Linux system of the development board.

- a. First download MobaXterm, this software can be downloaded in the **official tool** to

<http://www.orangepi.cn/downloadresourcescn/>



## Downloads



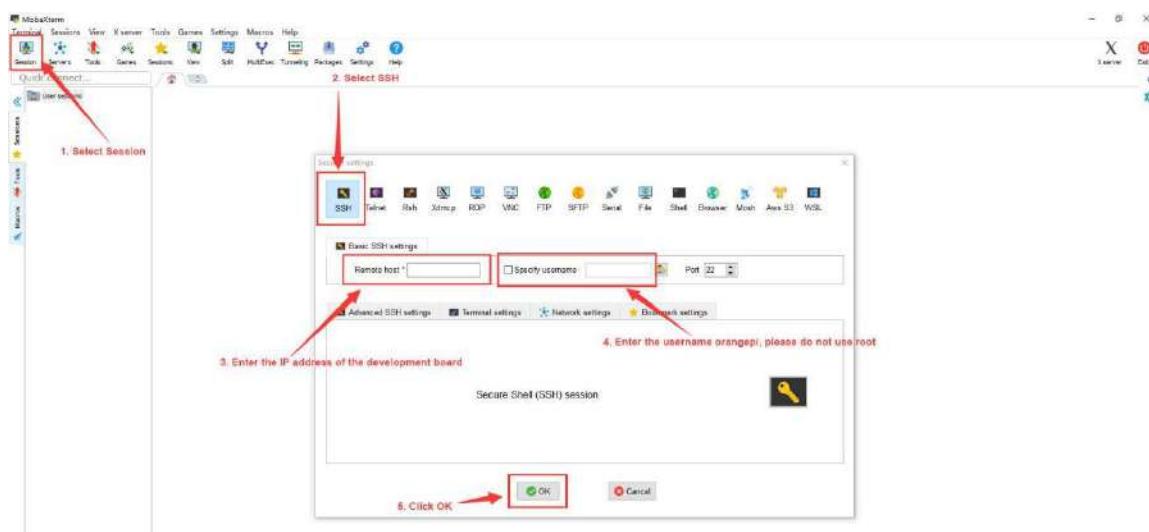
|                                                       |       |                  |
|-------------------------------------------------------|-------|------------------|
| <input type="checkbox"/> vcredit_x86.exe              | 4.3M  | 2021-04-25 21:25 |
| <input type="checkbox"/> security.targz               | 2.3M  | 2021-06-16 14:07 |
| <input type="checkbox"/> SDCardFormatter5_WhEN.zip    | 0M    | 2022-03-09 15:33 |
| <input type="checkbox"/> MobaXterm_Portable_v20.3.zip | 24.9M | 2020-11-04 13:48 |

- b. After downloading, use the decompression software to decompress the downloaded compressed package, you can get the executable software of MobaXterm, and then double-click to open it

| 名称                      | 修改日期           | 类型        | 大小        |
|-------------------------|----------------|-----------|-----------|
| CygUtils.plugin         | 2020/5/21 4:06 | PLUGIN 文件 | 15,570 KB |
| MobaXterm_Personal_20.3 | 2020/6/5 4:30  | 应用程序      | 14,104 KB |

- c. Then create a new ssh session in MobaXterm
- Open **Session**
  - Then select **SSH** in **Session Setting**
  - Then enter the IP address of the development board in **Remote host**
  - Then enter the username **orangeipi** of the Linux system of the development board in **Specify username, please do not use the root user**
  - Finally click **OK**

**Kiauh does not support the root user, so please use the orangeipi user to log in to the Linux system.**

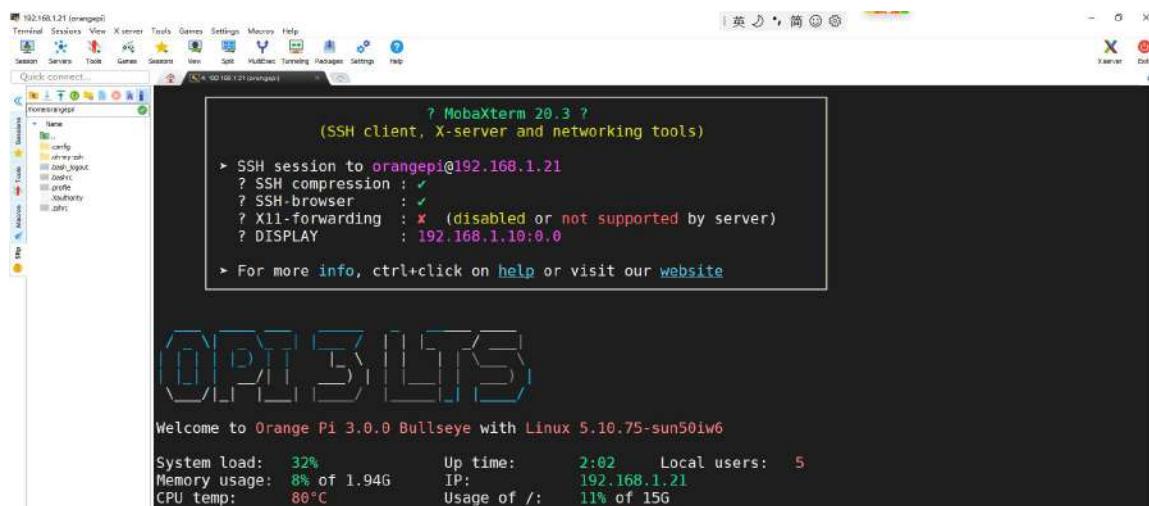


- d. Then you will be prompted to enter a password. The default password of the orangepi user in the Linux system of the development board is **orangepi**

**Note that when entering the password, the specific content of the entered password will not be displayed on the screen, please do not think that there is any fault, just press Enter after entering it.**



- e. The display after successfully logging in to the system is as shown below





## 2) Then download the source code of Kiauh

**Here are two ways to download the kiauh source code:**

**The first is to download the kiauh source code compressed package provided by Orange Pi from Google Cloud Disk for testing. The kiauh source code compression package provided by Orange Pi solves the problem of installation failure due to inaccessibility of github, and also fixes the problem of KlipperScreen installation failure.**

**The second is to download from kiauh's github repository. However, using this method to download the source code, if it does not solve the problem of accessing github from the development board linux system, it is basically difficult to install klipper successfully, and the installation speed is extremely slow.**

- a. The first method: the method of downloading the source code compressed package of kiauh provided by Orange Pi from Google Cloud Disk
  - a) From the Google cloud disk link below, you can download the kiauh source code compressed package provided by Orange Pi. You can see it in the **kiauh\_klipper** folder

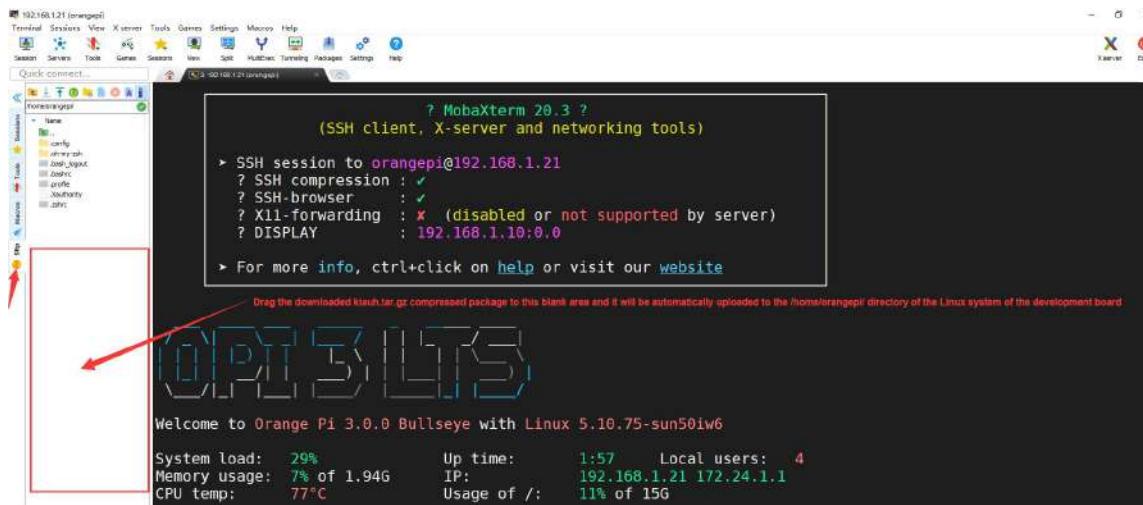
**<https://drive.google.com/drive/folders/1fie2GcF-6zfWMqBDufnLUV3B9qMnbUkO?usp=sharing>**

| 该目录上一级: 全部文件 - orangepi-build |       |                  |    |
|-------------------------------|-------|------------------|----|
| 文件名                           | 大小    | 修改日期             | 操作 |
| uboot                         | 7     | 2020-11-05 12:54 |    |
| watchers                      | 7     | 2021-04-02 11:53 |    |
| orangepi-build-hs             | 7     | 2021-12-27 18:12 |    |
| orangepi-build                | 7     | 2020-12-09 10:37 |    |
| OpenWRT                       | 7     | 2021-04-18 14:43 |    |
| linux编译用的临时压缩包                | 7     | 2020-11-05 12:08 |    |
| <b>Kiauh_Mirror</b>           | 7     | 2022-01-08 10:21 |    |
| kernel                        | 7     | 2020-11-05 12:54 |    |
| orangepi-firmware-gt.tar.gz   | 33.9M | 2020-11-05 14:52 |    |

|                      |       |                  |
|----------------------|-------|------------------|
| Kiauh.tar.gz_and5sum | 47B   | 2022-03-21 15:20 |
| <b>Kiauh.tar.gz</b>  | 74.3M | 2022-03-21 15:20 |

- b) After downloading the **kiauh.tar.gz** compressed package, first upload **kiauh.tar.gz** to the **/home/orangepi** directory of the Linux system of the development board



c) Then use the following command to unzip **kiauh.tar.gz**

After running the **tar zxf kiauh.tar.gz** command, normally there will be no print output (no output is the best result), and then run the **ls** command to see that there is an additional file named **kiauh** in the current directory folder.

The **l** in the **ls** command is the lowercase L, not the number 1, nor the lowercase i.

```
orangepi@orangepi:~$ tar zxf kiauh.tar.gz
orangepi@orangepi:~$ ls
kiauh kiauh.tar.gz
```

d) The difference between the compressed package of Kiauh source code provided by Orange Pi and the Kiauh source code downloaded from github is as follows:

If you don't understand what you are saying about the modification instructions of the source code in the following part, please ignore it directly, and it will not affect the subsequent installation.

- i. There is a **github\_src** folder in the Kiauh source code provided by Orange Pi, which caches the source code and compressed package required during the installation of Klipper and other software, as well as the configuration files required for the use of the USB camera

```
orangepi@orangepi:~$ cd kiauh
orangepi@orangepi:~/kiauh$ ls github_src
DuetWebControl-SD.zip    fluidd.zip   klipper   mainsail.zip   moonraker
pgcode      webcam.txt   dwc2-for-klipper-socket  KlipperScreen mjpg-streamer
moonraker-telegram-bot  webcamd
```



- ii. Modified part of the kiauh script code, you can see all the modified files through the **git status** . command

```
orangeipi@orangeipi:~/kiauh$ git status .
```

- iii. The functions implemented by the modified code mainly include:
- i) Disable the automatic update of Kiauh code. Every time **kiauh.sh** runs, it will automatically synchronize the code with github. If there is a problem with the network, it will cause it to get stuck.
  - ii) Change the installation source of pip to **https://pypi.tuna.tsinghua.edu.cn/simple** to speed up the installation of the python library
  - iii) Since the source code and other files that need to be downloaded from github are cached in the **github\_src** folder, part of the code in the Kiauh script that needs to download the source code from github is blocked, and modified to use the cp command to copy from **github\_src** to the corresponding location. In this way, there is no need to download the source code needed to install Klipper and other software from github through the network.
- b. The second: The command to download the source code of kiauh from github is as follows (**if you have no experience in installing klipper in kiauh, it will not solve the network problem of github, please do not use this method to download the source code of kiauh, it is difficult to install successfully**)

```
orangeipi@orangeipi:~$ sudo apt update  
orangeipi@orangeipi:~$ sudo apt install -y git  
orangeipi@orangeipi:~$ git clone https://github.com/th33xitus/kiauh.git
```

3) Enter the source code directory of Kiauh, you can see the files and folders it contains are as follows

- a. kiauh.sh: kiauh's startup script
- b. scripts: Contains installation scripts for software such as klipper

**The `cd kiauh` command means to enter the kiauh source code. If you are already in the kiauh source code directory, you do not need to execute it again.**

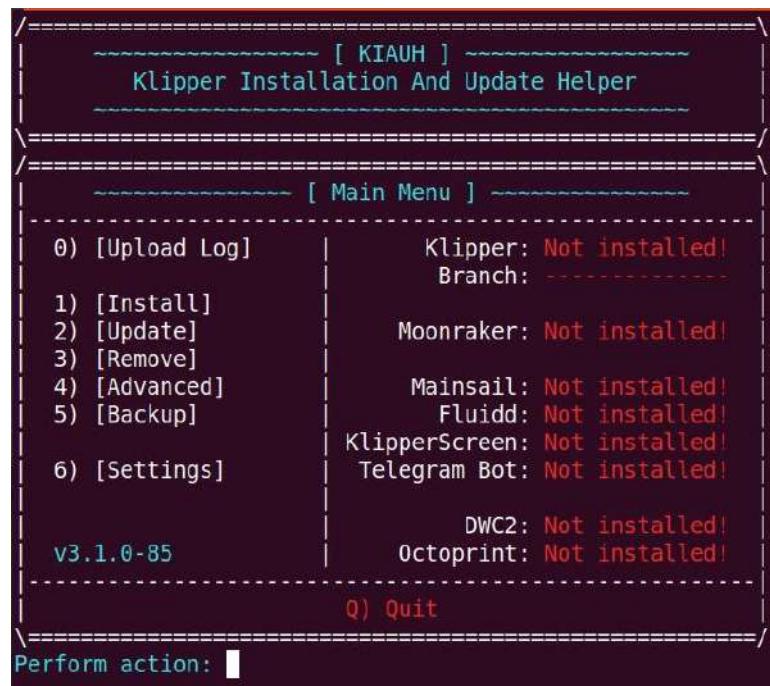
```
orangeipi@orangeipi:~$ cd kiauh/  
orangeipi@orangeipi:~/kiauh$ ls  
docs  github_src  kiauh.sh  LICENSE  README.md  resources  scripts
```



- 4) Then you can run the kiauh.sh script in the source directory of Kiauh

```
orangepi@orangepi:~/kiauh$ ./kiauh.sh
```

- 5) After running kiauh.sh, the following selection interface will pop up. You can see that Kiauh provides many operation options, and can also display the installation status of Klipper and other related software



- 6) Then you can enter **1** after **Perform action:** and press Enter, select **1) [Install]**



```
/=====
          [ KIAUH ]
          Klipper Installation And Update Helper
\=====

/===== [ Main Menu ] =====/
| 0) [Upload Log]           Klipper: Not installed!
|                                Branch: -----
| 1) [Install]               Moonraker: Not installed!
| 2) [Update]                Mainsail: Not installed!
| 3) [Remove]                Fluidd: Not installed!
| 4) [Advanced]              KlipperScreen: Not installed!
| 5) [Backup]                Telegram Bot: Not installed!
| 6) [Settings]
| v3.1.0-85                 DWC2: Not installed!
|                                Octoprint: Not installed!
|                               Q) Quit
\=====

Perform action: 1
```

7) Then the following installation selection interface will pop up

```
/=====
          [ KIAUH ]
          Klipper Installation And Update Helper
\=====

/===== [ Installation Menu ] =====/
| You need this menu usually only for installing
| all necessary dependencies for the various
functions on a completely fresh system.
Firmware:             Touchscreen GUI:
1) [Klipper]          5) [KlipperScreen]
-----
Klipper API:          Other:
2) [Moonraker]        6) [Duet Web Control]
-----
Klipper Webinterface: 7) [OctoPrint]
3) [Mainsail]         8) [PrettyGCode]
4) [Fluidd]           9) [Telegram Bot]
Webcam:
10) [MJPG-Streamer]
-----
B) « Back
\=====

Perform action: 1
```

8) We first enter 1 after **Perform action:** and press Enter to start the process of installing Klipper



```
/===== [ KIAUH ] =====\n      Klipper Installation And Update Helper\n\\===== [\n  Installation Menu ]\n-----\nYou need this menu usually only for installing\nall necessary dependencies for the various\nfunctions on a completely fresh system.\n-----\nFirmware:\n  1) [Klipper]           Touchscreen GUI:\n                                5) [KlipperScreen]\n\nKlipper API:\n  2) [Moonraker]          Other:\n                                6) [Duet Web Control]\n                                7) [OctoPrint]\nKlipper Webinterface:\n  3) [Mainsail]           8) [PrettyGCode]\n  4) [Fluidd]              9) [Telegram Bot]\n\n                                Webcam:\n                                10) [MJPG-Streamer]\n-----\nB) << Back\n\\=====
```

Perform action: 1 ←

- 9) After installing Klipper, you will first be reminded to set the path of the folder where the Klipper configuration file is located. The default is **/home/orangepi/klipper\_config**. If you do not need to modify it, just press Enter.



```
/=====
|----- [ KIAUH ] -----
|----- Klipper Installation And Update Helper
|-----\

##### Initializing Klipper installation ...

/=====
|----- !!! WARNING !!!
|----- No Klipper configuration directory set!

|----- Before we can continue, KIAUH needs to know where
|----- you want your printer configuration to be.

|----- Please specify a folder where your Klipper configu-
|----- ration is stored or, if you don't have one yet, in
|----- which it should be saved after the installation.
|-----\

/=====

|----- IMPORTANT:
|----- Please enter the new path in the following format:
|----- /home/orangepi/your_config_folder

|----- By default 'klipper_config' is recommended!
|-----\

##### Please set the Klipper config directory:
/home/orangepi/klipper_config
```

Then the following prompt message will pop up, the default is Y, just press Enter

```
/=====
|----- [ KIAUH ] -----
|----- Klipper Installation And Update Helper
|-----\

##### Initializing Klipper installation ...

/=====
|----- !!! WARNING !!!
|----- No Klipper configuration directory set!

|----- Before we can continue, KIAUH needs to know where
|----- you want your printer configuration to be.

|----- Please specify a folder where your Klipper configu-
|----- ration is stored or, if you don't have one yet, in
|----- which it should be saved after the installation.
|-----\

/=====

|----- IMPORTANT:
|----- Please enter the new path in the following format:
|----- /home/orangepi/your_config_folder

|----- By default 'klipper_config' is recommended!
|-----\

##### Please set the Klipper config directory:
/home/orangepi/klipper_config

##### Set config directory to '/home/orangepi/klipper_config' ? (Y/n): 
```

- 10) Then you will be prompted to enter the password of the Linux system, the default is orangepi



```
##### Please set the Klipper config directory:  
/home/orangepi/klipper_config  
  
##### Set config directory to '/home/orangepi/klipper_config' ? (Y/n):  
##### > Yes  
  
##### Create KIAUH backup directory ...  
>>>>> Directory created!  
>>>>> No config directory found! Skipping backup ...  
  
##### Directory set to '/home/orangepi/klipper_config'!  
[sudo] password for orangepi: █ ← Enter the password of the Linux system here: orangepi
```

- 11) Then you will be prompted to set the required number of Klipper instances, here we can enter 1

```
##### Directory set to '/home/orangepi/klipper_config'!  
[sudo] password for orangepi:  
  
>>>>> Config directory changed!  
/=====\\  
| Please select the number of Klipper instances to set  
| up. The number of Klipper instances will determine  
| the amount of printers you can run from this machine.  
|  
| WARNING: There is no limit on the number of instances  
| you can set up with this script.  
\\=====/  
##### Number of Klipper instances to set up: 1 ←
```

Then continue to press Enter to confirm

```
>>>>> Config directory changed!  
/=====\\  
| Please select the number of Klipper instances to set  
| up. The number of Klipper instances will determine  
| the amount of printers you can run from this machine.  
|  
| WARNING: There is no limit on the number of instances  
| you can set up with this script.  
\\=====/  
##### Number of Klipper instances to set up: 1  
  
##### Install 1 instance(s)? (Y/n): █ ←
```



## 12) Then the Klipper installation process will officially begin

- a. The script in Kiauh will first automatically download the source code of Klipper from GitHub. This step is easy to download failure due to network problems. Please pay special attention to the following output log and the following figure, no error is reported (**using the kiauh source code provided by Orange Pi will not have The process of downloading the code, so there will be no errors due to network problems**)

```
##### Installing 1 Klipper instance(s) ...

##### Checking for the following dependencies:
● git
>>>> Dependencies already met! Continue...

##### Downloading Klipper ...
Cloning into 'Klipper'...
remote: Enumerating objects: 28306, done.
remote: Counting objects: 100% (76/76), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 28306 (delta 49), reused 67 (delta 44), pack-reused 28230
Receiving objects: 100% (28306/28306), 20.32 MiB | 396.00 KiB/s, done.
Resolving deltas: 100% (21377/21377), done.
Updating files: 100% (1526/1526), done.

##### Download complete!
```

- b. Then the dependency package will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons

```
##### Installing packages...
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'python-dev-is-python2' instead of 'python-dev'
Note, selecting 'libusb-1.0-0' for regex 'libusb-1.0'
Note, selecting 'libusb-1.0-0-dev' for regex 'libusb-1.0'
Note, selecting 'libusb-1.0-0-doc' for regex 'libusb-1.0'
libusb-1.0-0 is already the newest version (2:1.0.23-2build1).
libusb-1.0-0 set to manually installed.
build-essential is already the newest version (12.8ubuntu1.1).
```

- c. Then the virtual environment of Python will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons



```
##### Installing python virtual environment...
created virtual environment CPython2.7.18.final.0-64 in 1637ms
  creator CPython2Posix(dest=/home/orangepi/klippy-env, clear=False, global=
    seeder FromAppData(download=False, pip=latest, setuptools=latest, wheel=la
ed-app-data/v1.0.1.debian.1)
  activators BashActivator,CShellActivator,FishActivator,PowerShellActivator
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Pl
l drop support for Python 2.7. More details about Python 2 support in pip, c
Collecting cffi==1.14.6
  Downloading cffi-1.14.6.tar.gz (475 kB)
    |████████| 475 kB 401 kB/s
Collecting pyserial==3.4
  Downloading pyserial-3.4-py2.py3-none-any.whl (193 kB)
    |████████| 193 kB 328 kB/s
```

- d. After the installation is complete, the following information will be displayed, and it will automatically return to the Kiauh installation interface

```
##### Creating Klipper Service ...
Created symlink /etc/systemd/system/multi-user.target.wants/klipper.service → /etc/systemd/system/klipper.service.
>>>> Single Klipper instance created!

##### Launching Klipper instance ...

#####
Klipper has been set up!
#####

/===== [ Installation Menu ] =====\
|-----+
| You need this menu usually only for installing
| all necessary dependencies for the various
| functions on a completely fresh system.
|-----+
| Firmware:           | Touchscreen GUI:
| 1) [Klipper]        | 5) [KlipperScreen]
|-----+
| Klipper API:       | Other:
| 2) [Moonraker]      | 6) [Duet Web Control]
|-----+
| Klipper Webinterface: | 7) [OctoPrint]
| 3) [Mainsail]       | 8) [PrettyGCode]
| 4) [Fluidd]         | 9) [Telegram Bot]
|-----+
|                               | Webcam:
|                               | 10) [MPJPG-Streamer]
|-----+
| B) < Back
\=====/
Perform action: █
```

- 13) Then start to install Klipper API - Moonraker, enter 2 after **Perform action:** and press Enter



```
/=====
~~~~~ [ Installation Menu ] ~~~~\

You need this menu usually only for installing
all necessary dependencies for the various
functions on a completely fresh system.

Firmware:
1) [Klipper] Touchscreen GUI:
5) [KlipperScreen]

Klipper API:
2) [Moonraker] Other:
6) [Duet Web Control]
7) [OctoPrint]
Klipper Webinterface:
3) [Mainsail] 8) [PrettyGCode]
4) [Fluidd] 9) [Telegram Bot]

Webcam:
10) [MJPG-Streamer]

B) << Back

\=====/
Perform action: 2
```

- 14) Then you will be prompted to set the number of Moonraker instances, which needs to correspond to the number of Klipper instances set earlier. Here we set it to **1** and press Enter.

```
/=====
~~~~~ [ KIAUH ] ~~~~\

Klipper Installation And Update Helper

\=====/
##### Initializing Moonraker installation ...

##### Your Python 3 version is: Python 3.8.10
/=====
| 1 Klipper instance was found!
| Usually you need one Moonraker instance per Klipper
| instance. Though you can install as many as you wish.
\=====/

##### Number of Moonraker instances to set up: 1
```

Then continue to press Enter to confirm



```
/-----\\
| [ KIAUH ] ~~~~~
| Klipper Installation And Update Helper
| ~~~~~
\\-----/\\
##### Initializing Moonraker installation ...

##### Your Python 3 version is: Python 3.8.10
/-----\\
| 1 Klipper instance was found!
| Usually you need one Moonraker instance per Klipper
| instance. Though you can install as many as you wish.
\\-----/\\

##### Number of Moonraker instances to set up: 1

##### Install 1 instance(s)? (Y/n):
```

Then enter the default password orangepi of the Linux system to start the official installation process

```
##### Installing Moonraker ...

##### Checking for the following dependencies:
● wget
● curl
● unzip
● dfu-util
● virtualenv
● libjpeg-dev
● zlib1g-dev

##### Installing the following dependencies:
● libjpeg-dev
● zlib1g-dev

[sudo] password for orangepi: | ← Enter the password of the Linux system: orangepi
```

- 15) The specific installation process of Moonraker is as follows
  - a. The script in Kiauh will automatically install the dependency package first, please make sure that the download process will not fail due to network reasons



```
##### Checking for the following dependencies:  
● wget  
● curl  
● unzip  
● dfu-util  
● virtualenv  
● libjpeg-dev  
● zlib1g-dev  
  
##### Installing the following dependencies:  
● libjpeg-dev  
● zlib1g-dev  
  
[sudo] password for orangepi:  
Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease  
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease  
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease  
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease  
Reading package lists... Done  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  libjpeg-turbo8-dev libjpeg8-dev  
The following NEW packages will be installed:  
  libjpeg-dev libjpeg-turbo8-dev libjpeg8-dev zlib1g-dev  
0 upgraded, 4 newly installed, 0 to remove and 124 not upgraded.
```

- b. Then the source code of Moonraker will be automatically downloaded from GitHub. This step is easy to fail due to network problems. Please pay special attention that the output log is consistent with the figure below, and no error is reported (**using the kiauh source code provided by Orange Pi will not have the process of downloading the code , so there will be no errors due to network problems**)

```
##### Downloading Moonraker ...  
Cloning into 'moonraker'...  
remote: Enumerating objects: 5413, done.  
remote: Counting objects: 100% (188/188), done.  
remote: Compressing objects: 100% (86/86), done.  
remote: Total 5413 (delta 118), reused 144 (delta 102), pack-reused 5225  
Receiving objects: 100% (5413/5413), 1.65 MiB | 1.38 MiB/s, done.  
Resolving deltas: 100% (3955/3955), done.  
>>>>> Download complete!
```

- c. The download error is as follows

```
##### Downloading Moonraker ...  
Cloning into 'moonraker'...  
fatal: unable to access 'https://github.com/Arksine/moonraker.git/': GnuTLS recv error (-110): The TLS connection was non-properly terminated.  
>>>>> Download complete!
```

- d. Then it will continue to install the dependent package, please make sure that the process of downloading and installing will not fail due to network reasons



```
##### Installing dependencies ...
##### Reading dependencies...
python3-virtualenv
python3-dev
libopenjp2-7
python3-libgpiod
curl
libcurl4-openssl-dev
libssl-dev
liblmdb-dev
libsodium-dev
zlib1g-dev
libjpeg-dev

##### Running apt-get update...
[sudo] password for orangepi:
Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease
Reading package lists... Done

##### Installing packages...
```

- e. Then the virtual environment of Python will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons

```
##### Installing python virtual environment...
created virtual environment CPython3.8.10.final.0-64 in 1075ms
  creator CPython3Posix(dest=/home/orangepi/moonraker-env, clear=False, global=False)
  seeder FromAppData(download=False, pip=latest, setuptools=latest, wheel=latest, pkg_resources=eed-app-data/v1.0.1.debian.1)
  activators BashActivator,CShellActivator,FishActivator,PowerShellActivator,PythonActivator,Xo
Collecting tornado==6.1.0
  Downloading tornado-6.1-cp38-cp38-manylinux2014_aarch64.whl (427 kB)
    |██████████| 427 kB 609 kB/s
Collecting pyserial==3.4
  Using cached pyserial-3.4-py2.py3-none-any.whl (193 kB)
Collecting pillow==8.3.2
  Downloading Pillow-8.3.2-cp38-cp38-manylinux_2_17_aarch64.manylinux2014_aarch64.whl (3.0 MB)
    |██████████| 3.0 MB 7.9 MB/s
```

- f. After the installation is complete, the following information will be displayed, and it will automatically return to the Kiauh installation interface

```
##### Enabling moonraker.service ...
Created symlink /etc/systemd/system/multi-user.target.wants/moonraker.service → /etc/systemd/system/moonraker.service.
>>>> moonraker.service enabled!
>>>> Single Moonraker instance created!

##### Starting moonraker.service ...
>>>> moonraker.service started!

#####
Moonraker has been set up!
#####

● Instance 1: 192.168.1.11:7125
```

- 16) Then start to install **Klipper**'s web interface, the default options are **3) [Mainsail]** and **4) [Fluidd]**, etc. Here we test the process of installing **4) [Fluidd]**, to install **4) [Fluidd]** in Perform action: Enter **4** after that and press Enter



```
/=====
|----- [ KIAUH ] -----
|----- Klipper Installation And Update Helper -----
\-----
|----- [ Installation Menu ] -----
|----- You need this menu usually only for installing
|----- all necessary dependencies for the various
|----- functions on a completely fresh system.
|----- Firmware: | Touchscreen GUI:
|----- 1) [Klipper] | 5) [KlipperScreen]
|----- Klipper API: | Other:
|----- 2) [Moonraker] | 6) [Duet Web Control]
|----- Klipper Webinterface: | 7) [OctoPrint]
|----- 3) [Mainsail] | 8) [PrettyGCode]
|----- 4) [Fluidd] | 9) [Telegram Bot]
|----- | Webcam:
|----- | 10) [MJPG-Streamer]
|----- B) « Back
\-----
|----- Perform action: 4 | ←
```

- 17) Klipper's Web interface - the specific installation process of **Fluidd** is as follows
- The script in Kiauh will automatically download and install the dependency package first, please make sure that the download process will not fail due to network reasons

```
/=====
|----- [ KIAUH ] -----
|----- Klipper Installation And Update Helper -----
\-----

##### Checking for the following dependencies:
● nginx

##### Installing the following dependencies:
● nginx

Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail
```

- Then it will prompt whether you need to install MJGP-Streamer to display the output video of the camera. If you need to press Enter directly, if not, you can



enter **n** in the place shown in the figure below and press Enter.

```
##### Initializing Fluidd installation ...  
/=====\\  
| Install MJGP-Streamer for webcam support? |  
\\=====/  
##### Install MJPG-Streamer? (Y/n): █
```

- c. Here we choose to install MJGP-Streamer, and then the following information will be prompted, just press Enter

```
/=====\\  
| Install MJGP-Streamer for webcam support? |  
\\=====/  
##### Install MJPG-Streamer? (Y/n):  
##### > Yes  
  
/=====\\  
| It is recommended to have some important macros set  
| up in your printer configuration to have Fluidd  
| fully functional and working.  
  
Those macros are:  
● [gcode_macro PAUSE]  
● [gcode_macro RESUME]  
● [gcode_macro CANCEL_PRINT]  
  
If you already have these macros in your config file  
you can skip this step and choose 'no'.  
Otherwise you should consider to answer with 'yes' to  
add the recommended example macros to your config.  
\\=====/  
##### Add the recommended macros? (Y/n): █
```

- d. Then it will download **fluidd.zip** from GitHub and decompress and install it. This step is easy to download failure due to network problems. Please pay special attention to the following output log and the following figure, no error is reported (**using the kiauh source code provided by Orange Pi will not download fluidd .zip process, so there will be no errors due to network problems**)



```
##### Downloading Fluidee ...  
-- 13:33:33 - https://github.com/fluidd-core/fluidee/releases/download/v1.16.2/fluidee.zip  
Resolving github.com (github.com)... 149.82.112.4  
Connecting to github.com (github.com)|149.82.112.4|:443... connected.  
HTTP request sent, awaiting response... 302 Found  
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/295836951/5d248200-e0dd-11eb-9d39-b4a76797fd0a?X-Amz-Algorithm=AWS4-HMAC-SHA256  
X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220101%2Fs3%2Faws4%2Frequest&X-Amz-Date=20220101T133334Z&X-Amz-Expires=3000X-Amz-Signature=76e8c6f5c7a375dc1f037b  
56c9b6f5c87b37e2665db233401a412361e2a746X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=295836951&response-content-disposition=attachment%3B&filename=3Dfluidee.zip  
zip-response-content-type=application%2Foctet-stream [following]  
-- 13:33:34 - https://objects.githubusercontent.com/github-production-release-asset-2e65be/295836951/5d248200-e0dd-11eb-9d39-b4a76797fd0a?X-Amz-Algorithm=AWS4-HMAC-SHA256  
X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220101%2Fs3%2Faws4%2Frequest&X-Amz-Date=20220101T133334Z&X-Amz-Expires=3000X-Amz-Signature=76e8c6f5c7a375dc1f037b  
56c9b6f5c87b37e2665db233401a412361e2a746X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=295836951&response-content-disposition=attachment%3B&filename=3Dfluidee.zip  
Resolving objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.  
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 9518655 (9.1M) [application/octet-stream]  
Saving to: 'fluidee.zip'  
  
fluidee.zip 100%[=====] 9.08M 15.3KB/s in 10m 54s  
  
13:44:30 (14.2 kB/s) - 'fluidee.zip' saved [9518655/9518655]  
  
>>>> Download complete!  
>>>> Done!  
>>>> Remove downloaded archive ...  
>>>> Done!
```

- e. Then the dependency package will be installed, please make sure that the download process will not fail due to network reasons

```
##### Checking for the following dependencies:  
● git  
● cmake  
● build-essential  
● imagemagick  
● libv4l-dev  
● ffmpeg  
● libjpeg8-dev  
  
##### Installing the following dependencies:  
● cmake  
● imagemagick  
● libv4l-dev  
● ffmpeg  
  
Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease  
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease  
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease  
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease  
Reading package lists... Done  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:
```

- f. Then MJPG-Streamer will be downloaded, compiled and installed. In this step, it is easy to fail to download due to network problems. Please pay special attention that the output log is consistent with the figure below, and no error is reported (**using the kiauh source code provided by Orange Pi will not have the process of downloading the code, So there will be no errors due to network problems**)



```
##### Downloading MJPG-Streamer ...
Cloning into 'mjpg-streamer'...
remote: Enumerating objects: 2964, done.
remote: Total 2964 (delta 0), reused 0 (delta 0), pack-reused 2964
Receiving objects: 100% (2964/2964), 3.48 MiB | 1.96 MiB/s, done.
Resolving deltas: 100% (1885/1885), done.
>>>> Download complete!

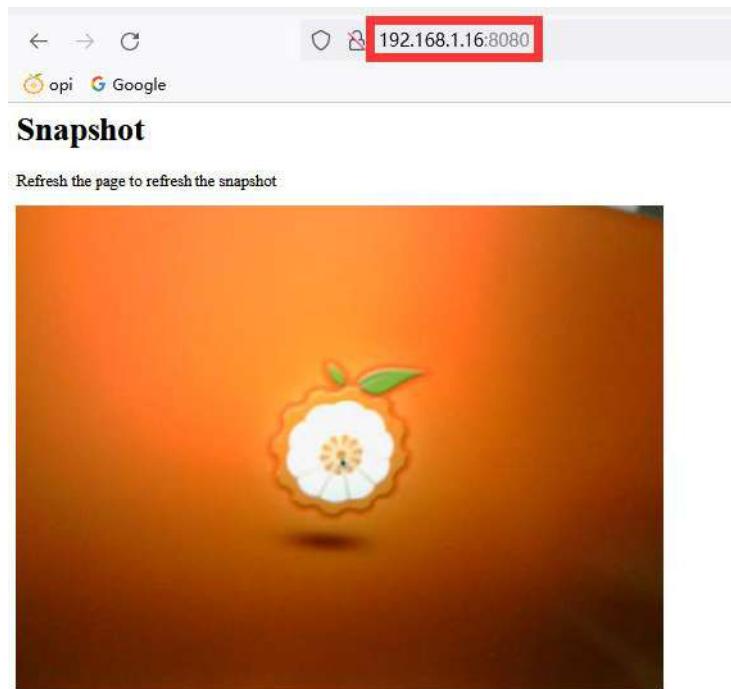
##### Compiling MJPG-Streamer ...
[ -d _build ] || mkdir _build
[ -f _build/Makefile ] || (cd _build && cmake -DCMAKE_BUILD_TYPE=Release ..)
-- The C compiler identification is GNU 9.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
```

- g. The access address of g.MJPG-Streamer is shown below. If the development board is connected to a USB camera, enter the following address in the browser to see the video output of the USB camera.

```
#####
# MJPG-Streamer has been set up!
#####

● Webcam URL: http://192.168.1.11:8080/?action=stream
● Webcam URL: http://192.168.1.11/webcam/?action=stream
```

The output of the camera looks like this



- h. The output after installation is as shown below



```
#####
# Fludd has been set up!
#####

/===== [ Installation Menu ] =====\

You need this menu usually only for installing
all necessary dependencies for the various
functions on a completely fresh system.

Firmware:           Touchscreen GUI:
1) [Klipper]        5) [KlipperScreen]

Klipper API:        Other:
2) [Moonraker]      6) [Duet Web Control]
                    7) [OctoPrint]
Klipper Webinterface: 8) [PrettyGCode]
3) [Mainsail]       9) [Telegram Bot]
4) [Fludd]          Webcam:
                    10) [MJPG-Streamer]

B) << Back

\=====/
Perform action: 
```

- 18) Then enter **B** after **Perform action:** to return to the main interface of the installation

```
=====
[ Installation Menu ]

You need this menu usually only for installing
all necessary dependencies for the various
functions on a completely fresh system.

Firmware:           Touchscreen GUI:
1) [Klipper]        5) [KlipperScreen]

Klipper API:        Other:
2) [Moonraker]      6) [Duet Web Control]
                    7) [OctoPrint]
Klipper Webinterface: 8) [PrettyGCode]
3) [Mainsail]       9) [Telegram Bot]
4) [Fludd]          Webcam:
                    10) [MJPG-Streamer]

B) << Back

\=====
Perform action: b  ←
```



```
[----- [ KIAUH ] -----]
Klipper Installation And Update Helper

[----- [ Main Menu ] -----]

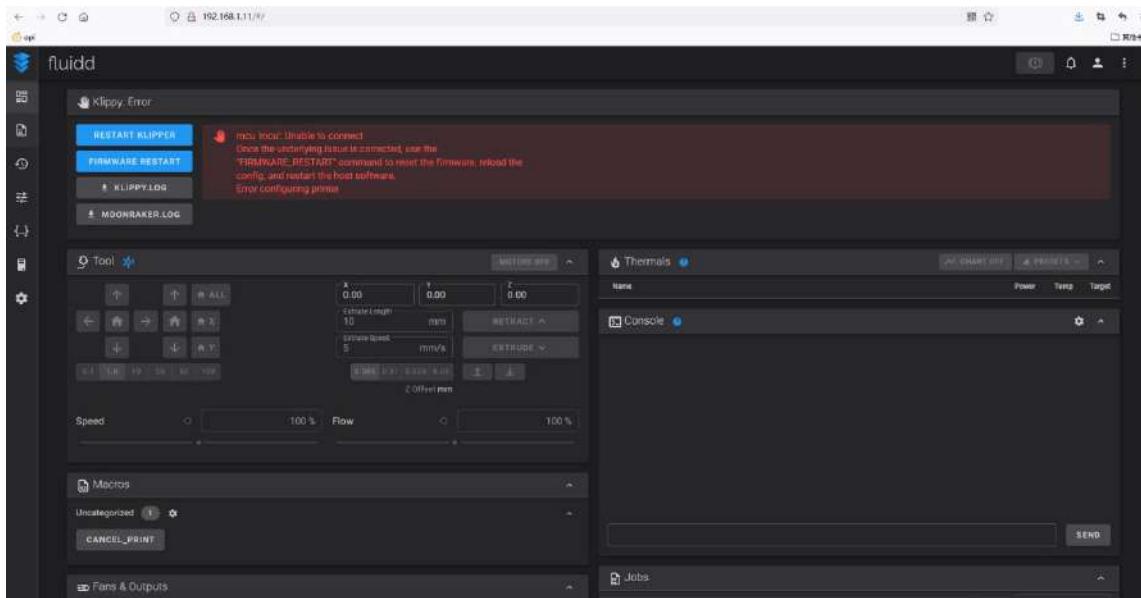
0) [Upload Log] | Klipper: Installed: 1
                  Branch: master
1) [Install]     | Moonraker: Installed: 1
2) [Update]      | Mainsail: Not installed!
3) [Remove]      | KlipperScreen: NOT INSTALLED!
4) [Advanced]    | Telegram Bot: Not installed!
5) [Backup]      | Fluidd: Installed!
6) [Settings]   | DWC2: Not installed!
                  Octoprint: Not installed!

v3.1.0-85
          0) Quit

Perform action: [
```

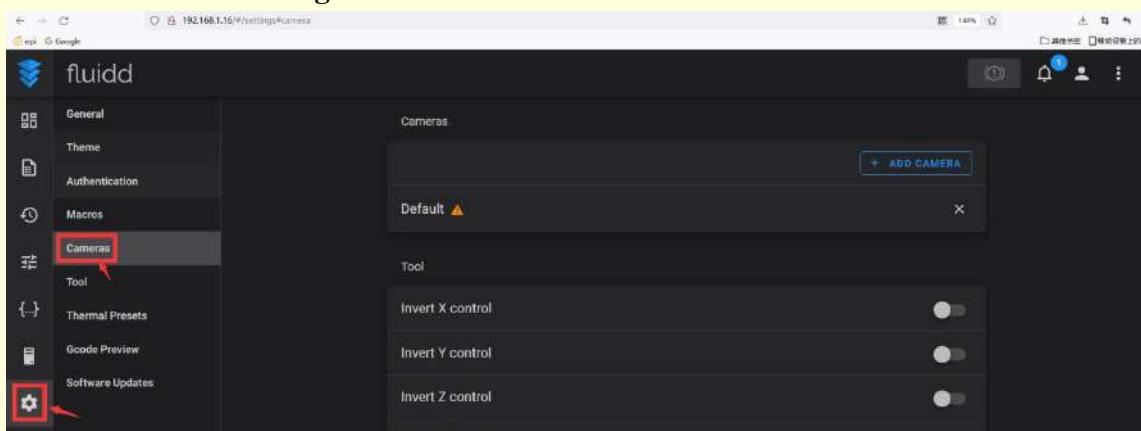
After returning to the main interface, you can see that Klipper, Moonraker and Fluidd are all displayed as Installed. However, it should be noted that even if Klipper, Moonraker and Fluidd are all displayed as Installed, it does not ensure that all software has been installed successfully. During the test, it was found that even after the code download failed due to network problems, Klipper, Moonraker and Fluidd were all displayed as Installed when returning to the main interface. So the best way to ensure that the installation is no problem is to ensure that the output log of the installation process does not report an error that the source code download fails due to network problems.

- 19) Finally, enter the IP address of the development board in the browser to see the web control interface of fluidd

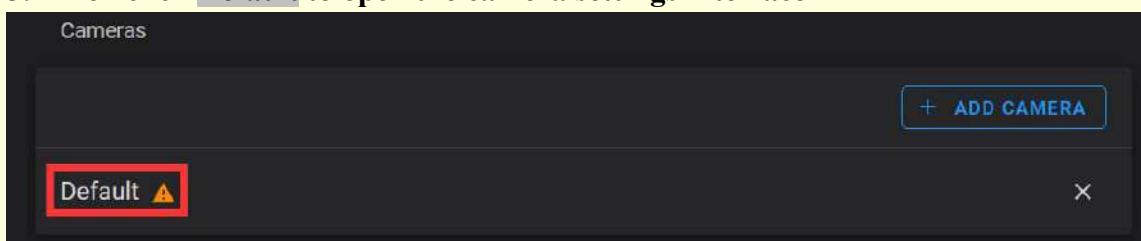


The steps to set the camera in the web control interface of flidd are as follows:

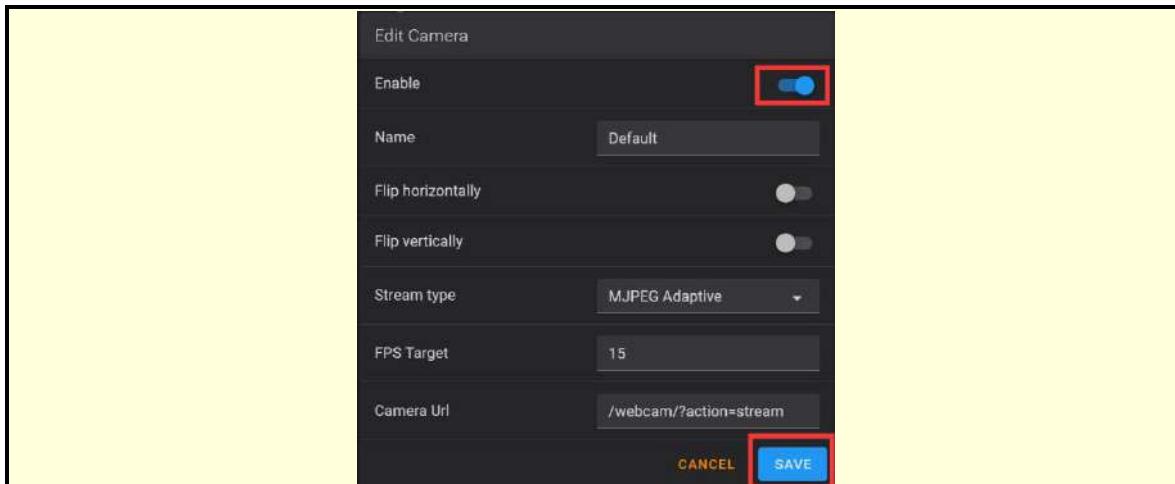
1. First of all, please make sure that MJPG-Streamer has been installed normally, and you can open the camera in the browser to see the output screen of the camera
2. Then find the setting interface of the camera in Flidd



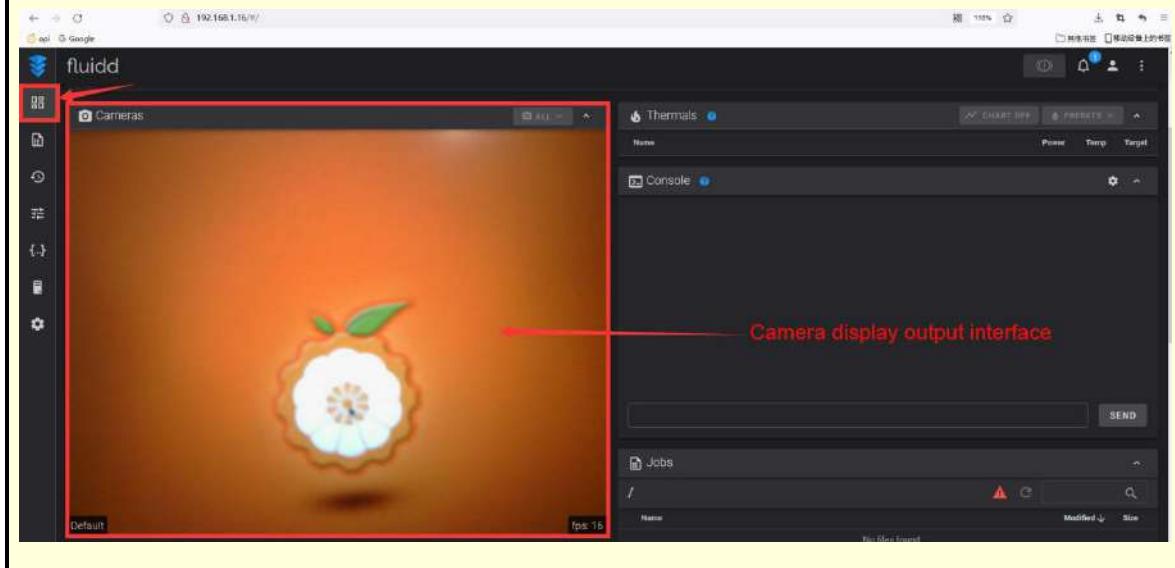
3. Then click Default to open the camera settings interface



4. Then Enable the camera, and then click Save to save



##### 5. Then go back to the main interface to see the output screen of the camera



20) Install the Mainsail example using the Kiauh source code archive provided by Orange Pi

- a. First please install 1) [Klipper] and 2) [Moonraker]
- b. Install 3) The [Mainsail] option is as follows



```
/=====
           [ Installation Menu ] =====\

You need this menu usually only for installing
all necessary dependencies for the various
functions on a completely fresh system.

Firmware:          Touchscreen GUI:
1) [Klipper]       5) [KlipperScreen]

Klipper API:      Other:
2) [Moonraker]     6) [Duet Web Control]
                  7) [OctoPrint]
Klipper Webinterface: 8) [PrettyGCode]
3) [Mainsail]      9) [Telegram Bot]
4) [Fluidd]         Webcam:
                  10) [MJPG-Streamer]

B) « Back
\=====/
Perform action: █
```

- c. If Fluidd has been installed before, port 80 is already occupied. First, you need to set the port number, for example, set it to 85

```
##### Initializing Mainsail installation ...

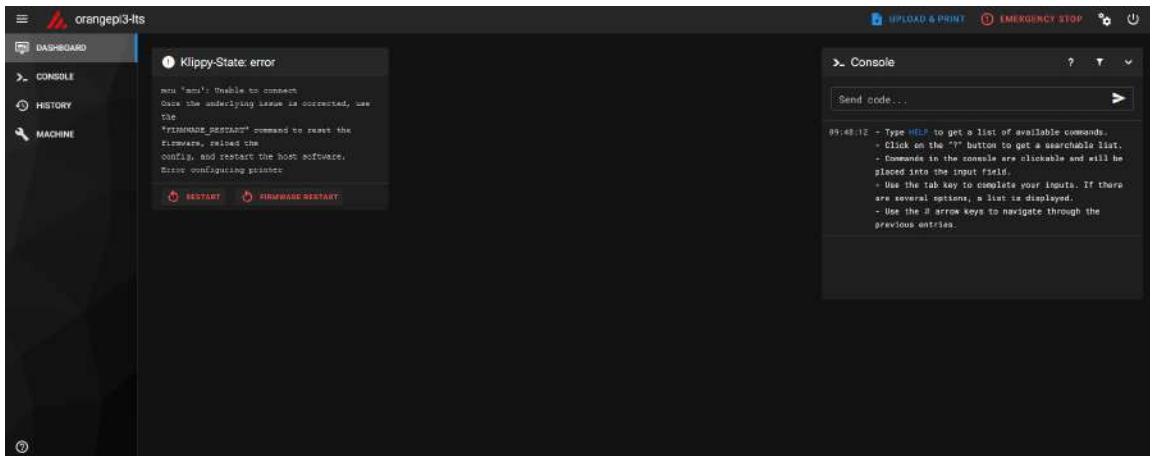
##### Detected other enabled interfaces:
● Fluidd - Port: 80

/=====
           !!!WARNING!!!
You need to choose a different port for Mainsail!
The following web interface is listening at port 80:
● Fluidd

Make sure you don't choose a port which was already
assigned to one of the other webinterfaces and do NOT
use ports in the range of 4750 or above!

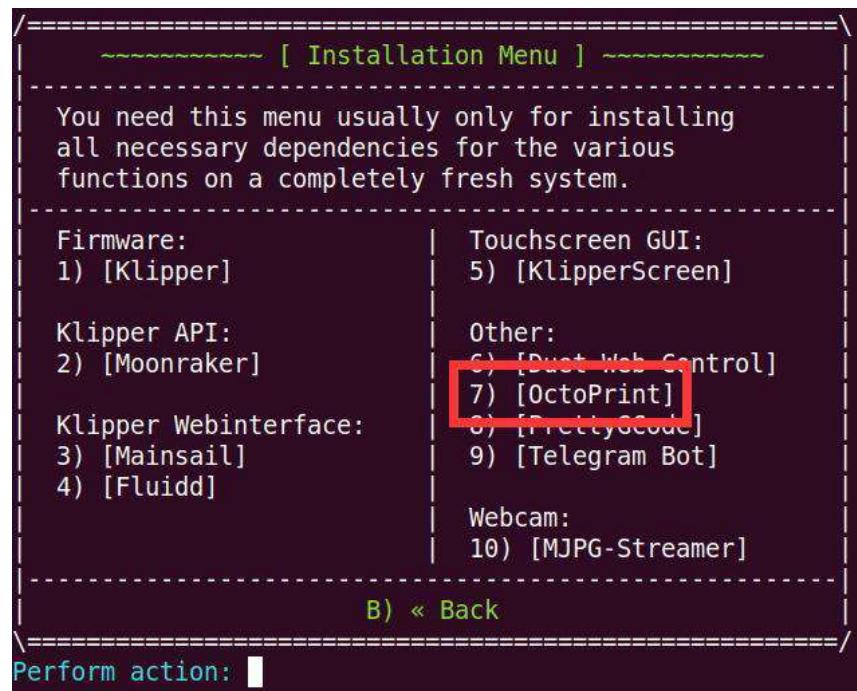
Be aware: there is NO sanity check for the following
input. So make sure to choose a valid port!
\=====/
Please enter a new Port: █
```

- d. After installation, enter the **IP address of the development board: port number** in the browser to see the Mainsail web control interface



21) Install the OctoPrint example using the Kiauh source code archive provided by Orange Pi

- First please install **1) [Klipper]** and **2) [Moonraker]**
- Install 7) The [OctoPrint]** option is shown below

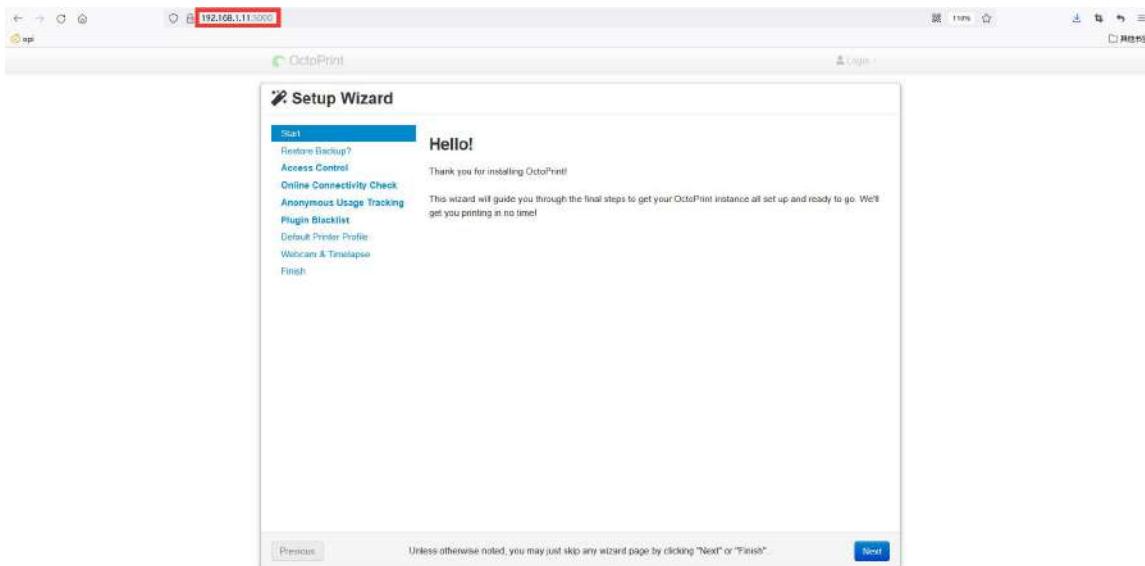


- After installation, you can see the access address of OctoPrint's web interface, the port number is 5000

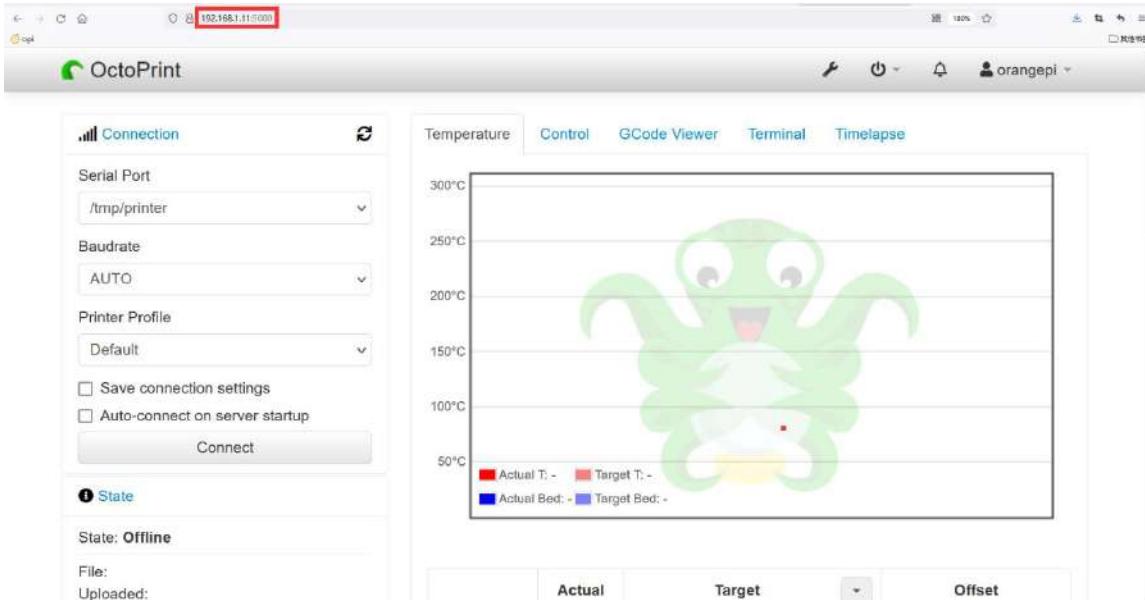




- d. Then enter the address shown above in the browser to see the OctoPrint web control interface



- e. The interface after setting according to the OctoPrint installation wizard is as follows



- 22) Install the KlipperScreen example using the Kiauh source code archive provided by Orange Pi

If you need to use kiauh to install KlipperScreen, please make sure that the Ubuntu or Debian system used is the linux4.9 server version system, and the kiauh source code used is the Kiauh source code compressed package downloaded from

**the Google network disk of Orange Pi. Installing KlipperScreen from the kiauh source code downloaded from github will not work properly**

- a. Orange Pi's modification to the KlipperScreen source code can be viewed using the following command

**If you are not interested in the source code, please skip it directly, it will not affect the installation and use of KlipperScreen.**

```
orangepi@orangepi:~$ cd kiauh/github_src/KlipperScreen  
orangepi@orangepi:~/kiauh/github_src/KlipperScreen$ git status .  
orangepi@orangepi:~/kiauh/github_src/KlipperScreen$ git diff .
```

- b. Before installing KlipperScreen, please connect the development board to the HDMI display, and ensure that the HDMI can be displayed normally
- c. If you must use the **desktop version** of Ubuntu or Debian, first close the desktop according to the instructions in the section on how to disable the desktop in the **Linux desktop version**.
- d. Then select **5) [KlipperScreen]** in Kiauh to install KlipperScreen, **please carefully check the printing information output during the installation process to ensure that no errors are reported**



- e. After e.KlipperScreen is installed, Kiauh is displayed as shown below. If there is a problem, please uninstall and reinstall



```
/=====
~~~~~ [ KIAUH ] ~~~~
Klipper Installation And Update Helper
\=====

~~~~~ [ Main Menu ] ~~~~

0) [Upload Log] Klipper: Installed: 1
Branch: master
1) [Install]
2) [Update]
3) [Remove]
4) [Advanced]
5) [Backup]
Mainsail: Not installed!
  ADD: Not installed!
KlipperScreen: Installed!
Telegram Bot: Not installed!
DW2: Not installed!
Octoprint: Not installed!

v3.1.0-102
\=====

Q) Quit

Perform action: |
```

- f. After KlipperScreen is installed, the HDMI monitor can see the interface shown in the figure below (if you can't see it, you can restart it and try it)



- g. The method of setting KlipperScreen to display the mouse cursor is:

- a) First run the following command to set **show\_cursor** to True in  
~/klipper\_config/KlipperScreen.conf

```
orangeipi@orangeipi:~$ cat <<EOF > ~/klipper_config/KlipperScreen.conf
[main]
show_cursor: True
```

**EOF**

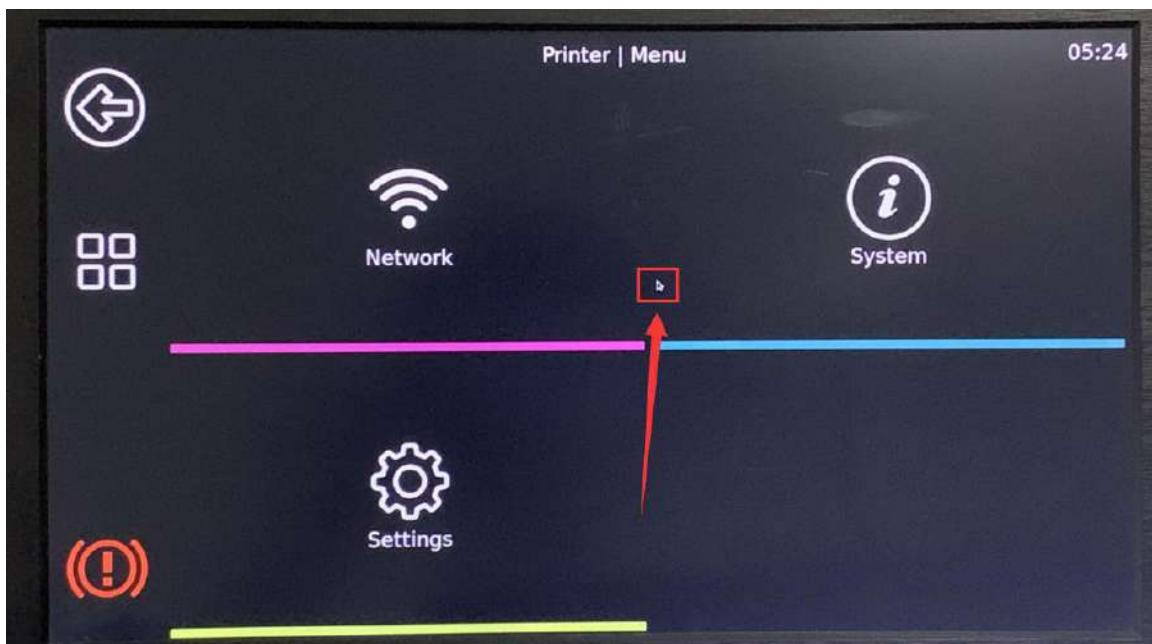
- b) Then use the following command to check to make sure the configuration is ok

```
orangepi@orangepi:~$ cat ~/klipper_config/KlipperScreen.conf
```

**[main]****show\_cursor: True**

- c) Then restart the **KlipperScreen** service and move the mouse to see the cursor

```
orangepi@orangepi:~$ systemctl restart KlipperScreen.service
```



### 23) Related information

Klipper source code: <https://github.com/Klipper3d/klipper>

Klipper official documentation: <http://www.klipper3d.org>

Kiauh source code: <https://github.com/th33xitus/kiauh>

**How to connect the Klipper to the microcontroller and the use of the 3D printer cannot be tested in this section. Please find the relevant information or video for testing and debugging.**



### 3. 29. Installation method of pagoda Linux panel

Pagoda Linux panel is a server management software that improves operation and maintenance efficiency. It supports more than 100 server management functions such as one-click LAMP/LNMP/cluster/monitoring/website/FTP/database/JAVA  
(excerpted from Pagoda's official website)

- 1) The recommended order for Pagoda Linux system compatibility is

**Debian10 > Ubuntu 20.04 > Ubuntu 18.04 > Ubuntu 22.04 > Other systems**

- 2) Log in to the linux system and enter the following command to start the installation of the pagoda

```
orangeipi@orangeipi:~$ wget -O install.sh  
http://download.bt.cn/install/install-ubuntu\_6.0.sh && sudo bash install.sh
```

- 3) Then the pagoda installer will remind whether to install **Bt-Panel** to the **/www** folder, then enter **y**

```
+-----  
| Bt-WebPanel FOR CentOS/Ubuntu/Debian  
+-----  
| Copyright © 2015-2099 BT-SOFT(http://www.bt.cn) All rights reserved.  
+-----  
| The WebPanel URL will be http://SERVER\_IP:8888 when installed.  
+-----
```

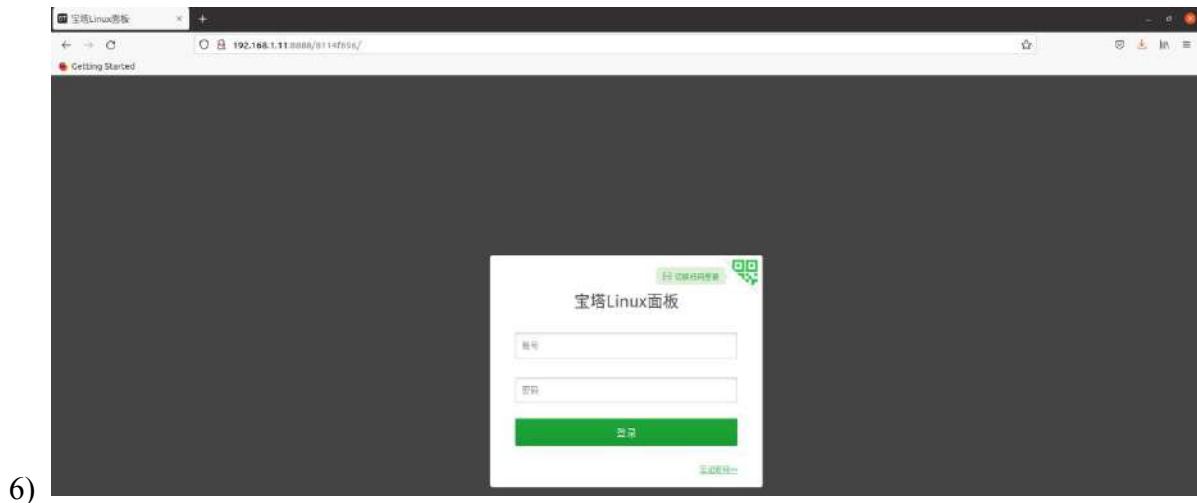
Do you want to install Bt-Panel to the /www directory now?(y/n): **y**

- 4) Then all you have to do is wait patiently. When you see the following print information output from the terminal, it means that the pagoda has been installed. The whole installation process takes about 28 minutes, and there may be some differences depending on the network speed.



```
=====
Congratulations! Installed successfully!
=====
外网面板地址: http://113.116.159.170:8888/21454a93
内网面板地址: http://192.168.1.132:8888/21454a93
username: mld01aeb
password: 9b3e5a85
If you cannot access the panel,
release the following panel port [8888] in the security group
若无法访问面板, 请检查防火墙/安全组是否有放行面板[8888]端口
=====
Time consumed: 28 Minute!
orangepi@orangepi3-lts:~$
```

- 5) At this time, enter **the panel address** shown above in the browser to open the login interface of the pagoda Linux panel, and then enter the **username** and **password** shown in the above figure in the corresponding position to log in to the pagoda



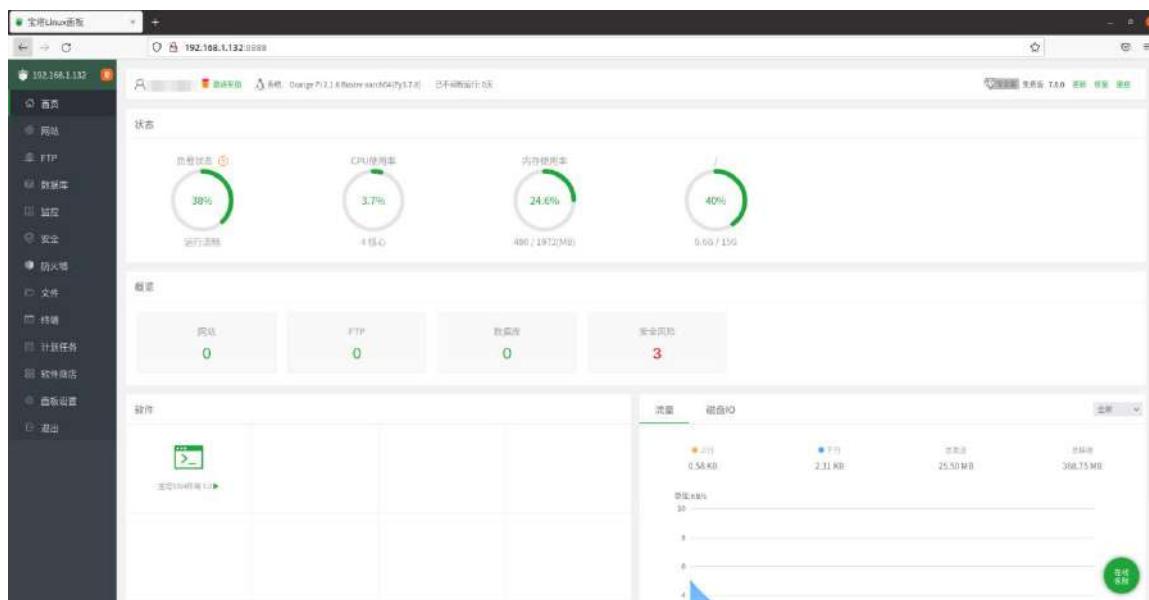
- 6)
- 7) After successfully logging in to the pagoda, the following welcome interface will pop up. First, please read the user instructions in the middle and drag it to the bottom, then you can select "I have agreed and read the "User Agreement", and then click "Enter the panel" You can enter the pagoda



8) After entering the pagoda, you will first be prompted to bind an account on the official website of the pagoda. If you do not have an account, you can go to the official website of the pagoda (<https://www.bt.cn>) to register one.

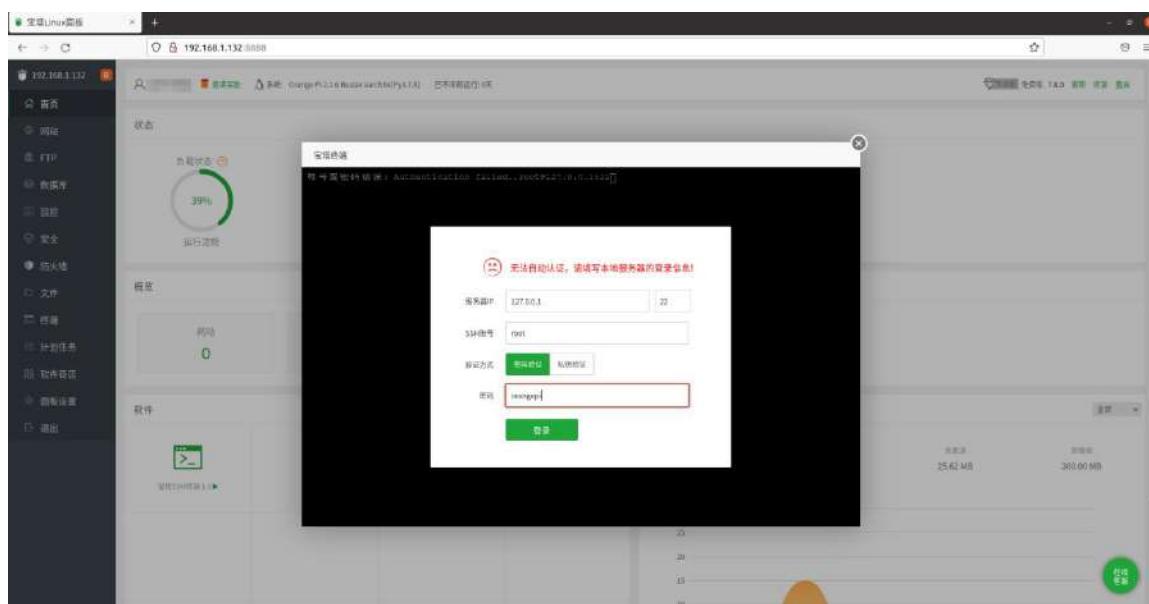


9) The final displayed interface is shown in the figure below. You can intuitively see some status information of the Linux system on the development board, such as load status, CPU usage, memory usage, and storage space usage, etc.

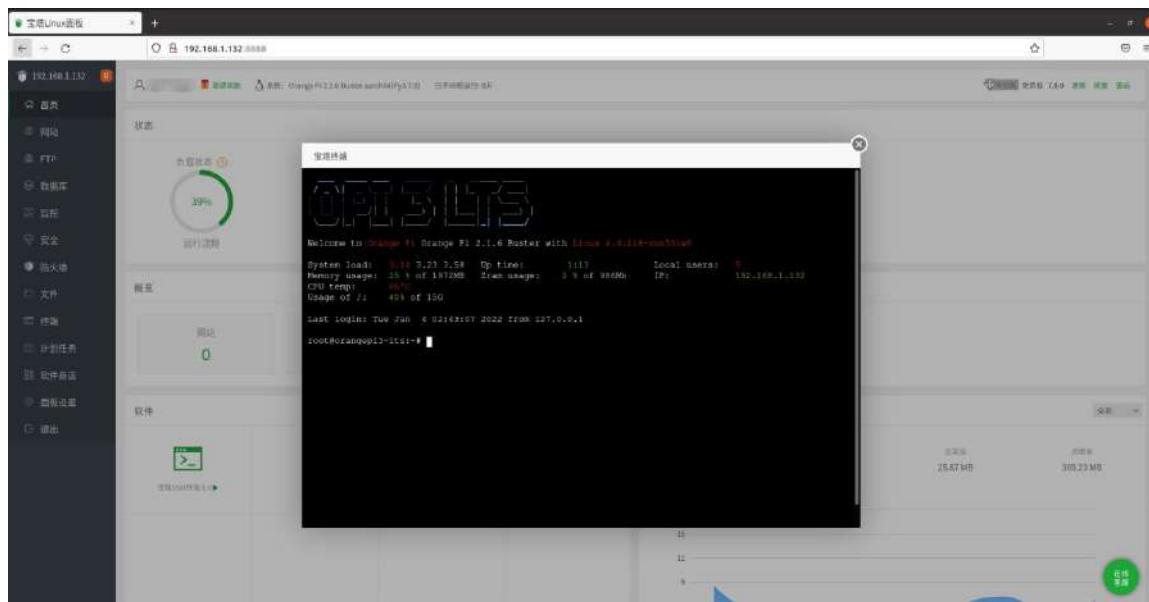


## 10) Test the SSH terminal login of the pagoda

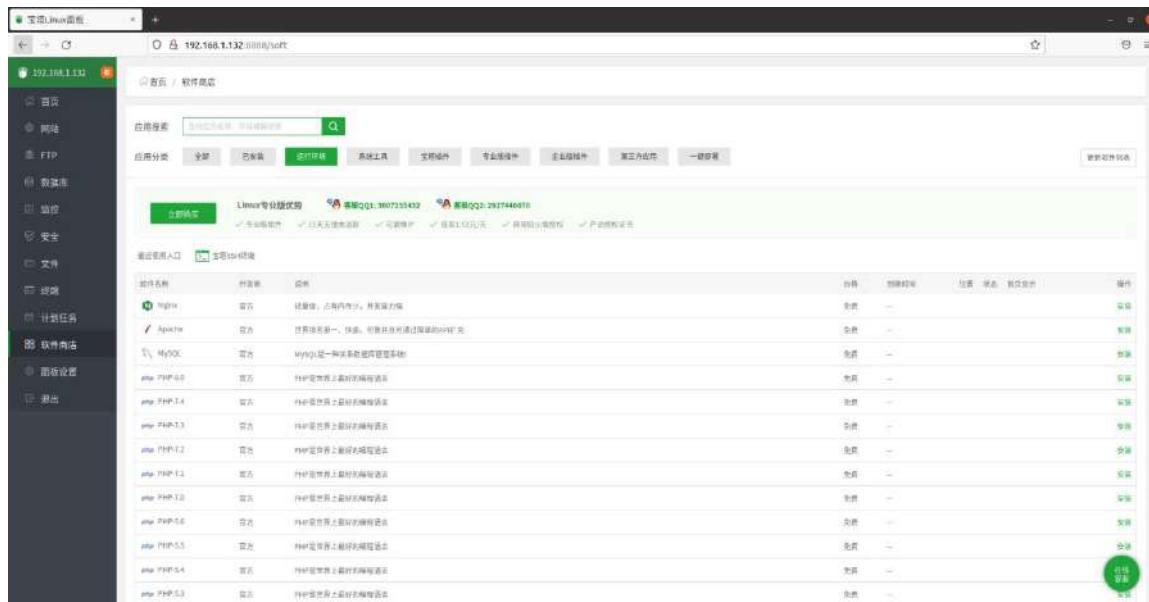
- After opening the SSH terminal of the pagoda, you will first be prompted to enter the password of the development board system. At this time, enter **orangeipi** in the password box (the default password, if there is any modification, please fill in the modified one)



- The display after successful login is as shown below



11) Software such as Apache, MySQL and PHP can be installed in the software store of the pagoda, and various applications can also be deployed with one click. Please explore these functions by yourself, and I will not demonstrate them one by one here.



12) Pagoda command line tool test



```
orangepi@orangepi3-lts:~$ sudo bt
[sudo] password for orangepi:
=====
宝塔面板命令行=====
(1) 重启面板服务          (8) 改面板端口
(2) 停止面板服务          (9) 清除面板缓存
(3) 启动面板服务          (10) 清除登录限制
(4) 重载面板服务          (11) 取消入口限制
(5) 修改面板密码          (12) 取消域名绑定限制
(6) 修改面板用户名        (13) 取消IP访问限制
(7) 强制修改MySQL密码      (14) 查看面板默认信息
(22) 显示面板错误日志      (15) 清理系统垃圾
(23) 关闭BasicAuth认证      (16) 修复面板(检查错误并更新面板文件到最新版)
(24) 关闭动态口令认证      (17) 设置日志切割是否压缩
(25) 设置是否保存文件历史副本 (18) 设置是否自动备份面板
(0) 取消                  (29) 取消访问设置验证
=====
请输入命令编号 : 14
=====
正在执行(14)...
=====
BT-Panel default info!
=====
外网面板地址: http://113.116.159.170:8888/21454a93
内网面板地址: http://192.168.1.132:8888/21454a93
*以下仅为初始默认账户密码，若无法登录请执行bt命令重置账户/密码登录
username: mld01aeb
password: 9b3e5a85
If you cannot access the panel,
release the following panel port [8888] in the security group
若无法访问面板，请检查防火墙/安全组是否有放行面板[8888]端口
=====
orangepi@orangepi3-lts:~$
```

- 13) For more functions of the pagoda, you can refer to the following information to explore by yourself

manual: <http://docs.bt.cn>

Forum address: <https://www.bt.cn/bbs>

GitHub: <https://github.com/aaPanel/BaoTa>

### 3. 30. Python related instructions

#### 3. 30. 1. Method of compiling and installing Python source code

If the Python version in the Ubuntu or Debian system software repository does not meet the development requirements, and you want to use the latest version of Python, you can use the following method to download the Python source package to compile and install the latest version of Python.

The following demonstration is to compile and install the latest version of



**Python 3.9. If you want to compile and install other versions of Python, the method is the same (you need to download the source code corresponding to the Python you want to install).**

- 1) First install the dependencies needed to compile Python

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt install -y build-essential zlib1g-dev \  
libncurses5-dev libgdbm-dev libnss3-dev libssl-dev libsqlite3-dev \\  
libreadline-dev libffi-dev curl libbz2-dev
```

- 2) Then download the latest version of Python3.9 source code and unzip it

```
orangepi@orangepi:~$ wget \  
https://www.python.org/ftp/python/3.9.10/Python-3.9.10.tgz  
orangepi@orangepi:~$ tar xvf Python-3.9.10.tgz
```

- 3) Then run the configure command

```
orangepi@orangepi:~$ cd Python-3.9.10  
orangepi@orangepi:~$ ./configure --enable-optimizations
```

- 4) Then compile and install Python3.9, the compilation time will take about half an hour

```
orangepi@orangepi:~$ make -j4  
orangepi@orangepi:~$ sudo make altinstall
```

- 5) After installation, you can use the following command to check the version number of the Python you just installed

```
orangepi@orangepi:~$ python3.9 --version  
Python 3.9.10
```

- 6) Then update pip

```
orangepi@orangepi:~$ /usr/local/bin/python3.9 -m pip install --upgrade pip
```

### 3. 30. 2. How to replace pip source in Python

The default source used by Linux system pip is the official source of Python, but the speed of accessing the official source of Python in China is very slow, and the installation of Python packages often fails due to network reasons. So when using pip to install the Python library, please remember to replace the pip source.



### 1) First install **python3-pip**

```
orangeipi@orangeipi:~$ sudo apt install -y python3-pip
```

### 2) The method of permanently replacing the pip source under Linux

- First create a new **~/.pip** directory, then add the **pip.conf** configuration file, and set the source of pip to Tsinghua source in it

```
orangeipi@orangeipi:~$ mkdir -p ~/.pip
orangeipi@orangeipi:~$ cat <<EOF > ~/.pip/pip.conf
[global]
timeout = 6000
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
trusted-host = pypi.tuna.tsinghua.edu.cn
EOF
```

- Then using pip3 to install the Python library will be very fast

### 3) The method of temporarily replacing the pip source under Linux, where **<packagename>** needs to be replaced with a specific package name

```
orangeipi@orangeipi:~$ pip3 install <packagename> -i \
https://pypi.tuna.tsinghua.edu.cn/simple --trusted-host pypi.tuna.tsinghua.edu.cn
```

## 3. 31. Installation method of OpenCV

### 3. 31. 1. Using apt to install OpenCV

#### 1) The installation command is as follows

```
orangeipi@orangeipi:~$ sudo apt update
orangeipi@orangeipi:~$ sudo apt install -y libopencv-dev python3-opencv
```

#### 2) Then use the following command to print the version number of OpenCV and the output is normal, indicating that the installation of OpenCV is successful

- The version of OpenCV in Ubuntu22.04 is as follows:

```
orangeipi@orangeipi:~$ python3 -c "import cv2; print(cv2.__version__)"
4.5.4
```

- The version of OpenCV in Ubuntu20.04 is as follows:

```
orangeipi@orangeipi:~$ python3 -c "import cv2; print(cv2.__version__)"
```

**4.2.0**

- c. The version of OpenCV in Ubuntu18.04 is as follows:

```
orangeipi@orangeipi:~$ python3 -c "import cv2; print(cv2.__version__)"
```

**3.2.0**

- d. The version of OpenCV in Debian10 is as follows:

```
orangeipi@orangeipi:~$ python3 -c "import cv2; print(cv2.__version__)"
```

**3.2.0**

- e. The version of OpenCV in Debian11 is as follows:

```
orangeipi@orangeipi:~$ python3 -c "import cv2; print(cv2.__version__)"
```

**4.5.1**

### 3. 32. How to install Home Assistant

**Note that only the method of installing Home Assistant in Ubuntu or Debian system is provided here. For detailed usage of Home Assistant, please refer to the official documentation or corresponding books.**

#### 3. 32. 1. Installation via docker

1) Please install docker first, and make sure that docker can run normally. For the installation steps of Docker, please refer to the instructions in the section How to Install Docker.

2) Then you can search for the docker image of Home Assistant

```
orangeipi@orangeipi:~$ docker search homeassistant
```

3) Then use the following command to download the docker image of Home Assistant to the local, the image size is about 1GB, the download time will be longer, please wait patiently for the download to complete

```
orangeipi@orangeipi:~$ docker pull homeassistant/home-assistant
```

Using default tag: latest

latest: Pulling from homeassistant/home-assistant

be307f383ecc: Downloading

5fbc4c07ac88: Download complete

..... (omit part of the output)

3cc6a1510c9f: Pull complete

7a4e4d5b979f: Pull complete



Digest:

sha256:81d381f5008c082a37da97d8b08dd8b358dae7ecf49e62ce3ef1eeaefc4381bb

Status: Downloaded newer image for homeassistant/home-assistant:latest

docker.io/homeassistant/home-assistant:latest

**If the network connected to the development board is relatively fast, but the download of the docker image is very slow, please check if you forgot to configure the download address of the docker image as a domestic source. The configuration method is described in the section on how to install Docker.**

- 4) Then you can use the following command to view the docker image of Home Assistant just downloaded

```
orangeipi@orangeipi:~$ docker images homeassistant/home-assistant
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
homeassistant/home-assistant    latest   bfa0ab9e1cf5  2 months ago  1.17GB
```

- 5) At this point, you can run the docker container of Home Assistant

```
orangeipi@orangeipi:~$ docker run -d \
--name homeassistant \
--privileged \
--restart=unless-stopped \
-e TZ=Asia/Shanghai \
-v /home/orangeipi/home-assistant:/config \
--network=host \
homeassistant/home-assistant:latest
```

- 6) Then enter [IP address of the development board: 8123] in the browser to see the interface of Home Assistant

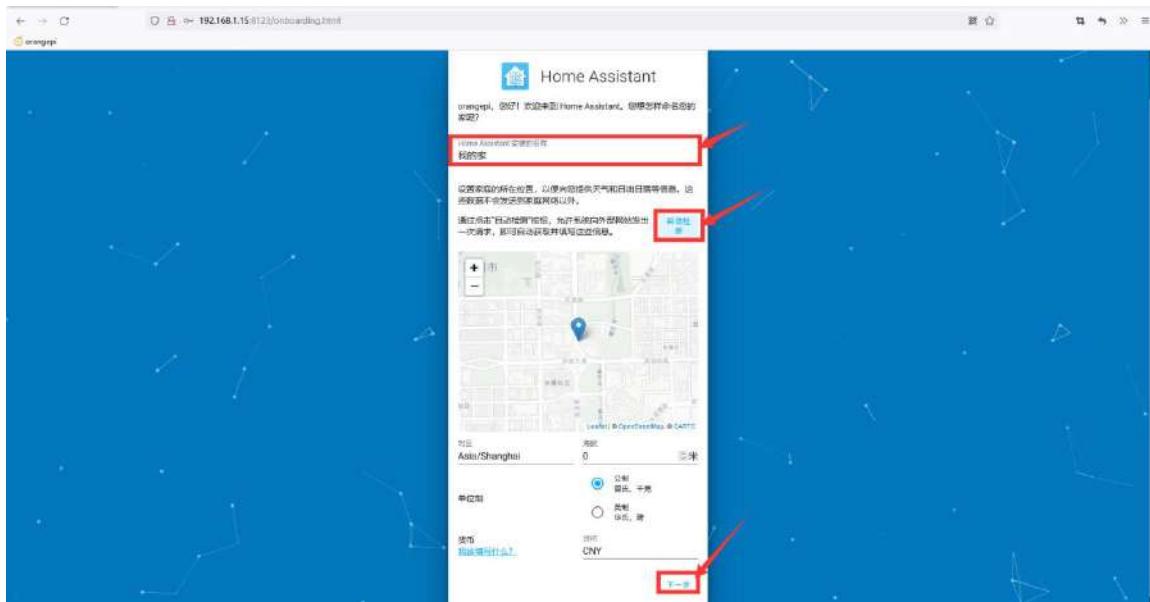
**The startup of the Home Assistant container takes a while, if the interface below does not appear normally, please wait a few seconds before refreshing. If the following interface is not displayed after waiting for more than a minute, it means that there is a problem with the Home Assistant installation. At this time, you need to check whether there is a problem with the previous installation and setting process.**



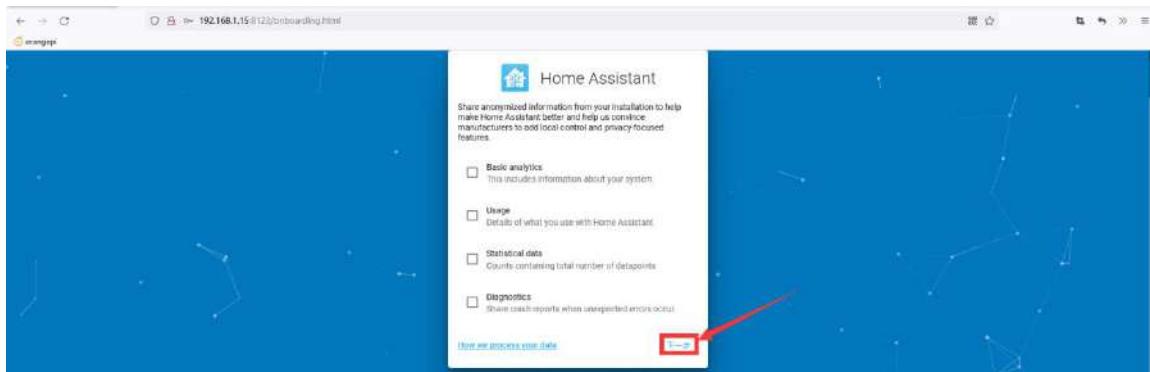
7) Then enter **your name, username** and password and click **Create Account**



8) Then follow the interface prompts to set according to your own preferences, and then click Next



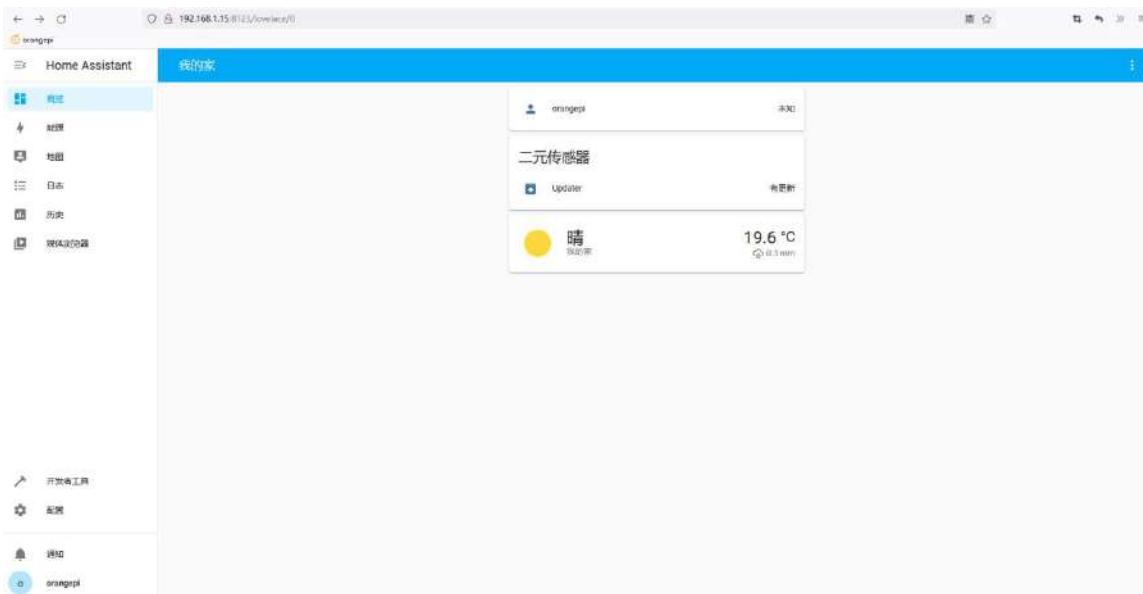
### 9) Then click Next



### 10) Then click Finish



### 11) The main interface finally displayed by Home Assistant is shown in the figure below



## 12) Ways to stop the Home Assistant container

- The command to view the docker container is as follows

```
orangepi@orangepi:~$ docker ps -a
```

- The command to stop the Home Assistant container is as follows

```
orangepi@orangepi:~$ docker stop homeassistant
```

- The command to delete the Home Assistant container is as follows

```
orangepi@orangepi:~$ docker rm homeassistant
```

### 3. 32. 2. Install via python

Before installation, please change the source of pip to a domestic source to speed up the installation of Python packages. For the configuration method, see the description in the section [How to replace pip source in Python](#).

#### 1) First install the dependency package

```
orangepi@orangepi:~$ sudo apt-get update  
orangepi@orangepi:~$ sudo apt-get install -y python3 python3-dev python3-venv  
python3-pip libffi-dev libssl-dev libjpeg-dev zlib1g-dev autoconf build-essential  
libopenjp2-7 libtiff5 libturbojpeg0-dev tzdata
```

#### 2) Then you need to compile and install Python 3.9. For the method, please refer to the section on [compiling and installing Python source code](#).

**The default Python version of Debian Bullseye is Python3.9, so there is no need**

**to compile and install.**

**The default Python version of Ubuntu Jammy is Python3.10, so there is no need to compile and install.**

## 3) Then create a Python virtual environment

```
orangepi@orangepi:~$ sudo mkdir /srv/homeassistant
orangepi@orangepi:~$ sudo chown orangepi:orangepi /srv/homeassistant
orangepi@orangepi:~$ cd /srv/homeassistant
orangepi@orangepi:~$ python3.9 -m venv .
orangepi@orangepi:~$ source bin/activate
(homeassistant) orangepi@orangepi:/srv/homeassistant$
```

**The fourth command Ubuntu Jammy needs to be modified to python3.10 -m venv .**

## 4) Then install the required Python packages

```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ python3 -m pip install wheel
```

## 5) Then you can install Home Assistant Core

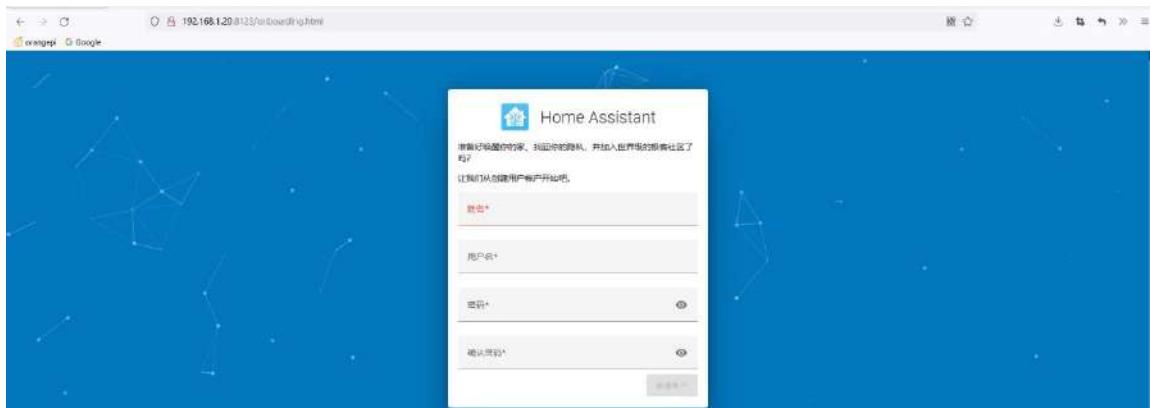
```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ pip3 install homeassistant
```

## 6) Then enter the following command to run Home Assistant Core

```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ hass
```

## 7) Then enter [IP address of the development board: 8123] in the browser to see the interface of Home Assistant

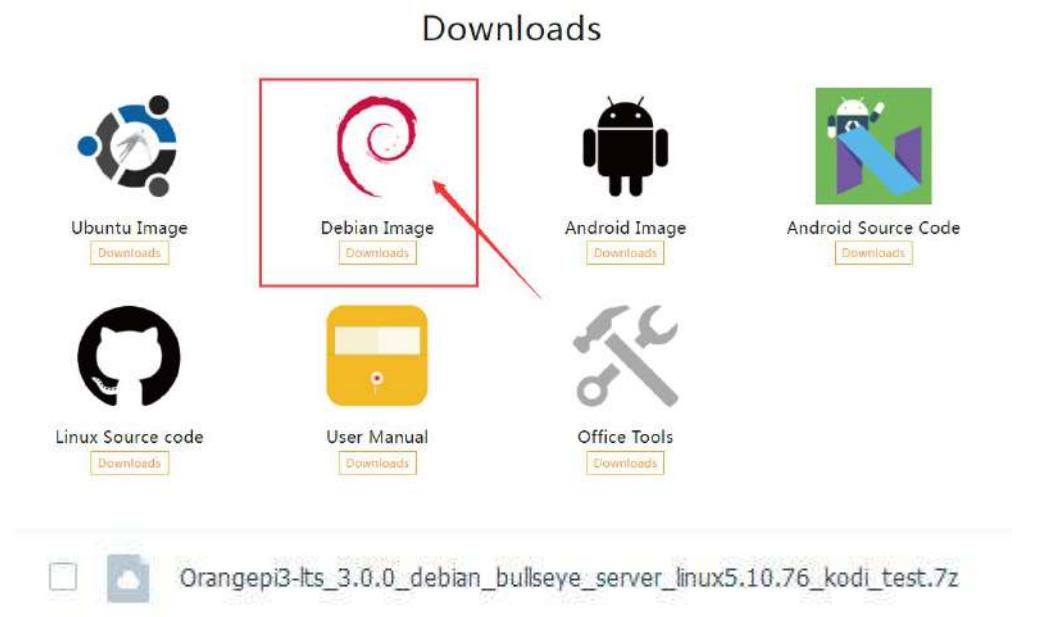
**When you run the hass command for the first time, it will download, install and cache some necessary libraries and dependencies. This process may take a few minutes. Note that the interface of Home Assistant cannot be seen in the browser at this time, please wait for a while before refreshing.**



### 3. 33. Debian11 Kodi image usage instructions

- 1) The images that Kodi needs to use to play videos are as follows, **only this image supports hard-to-play video through Kodi**

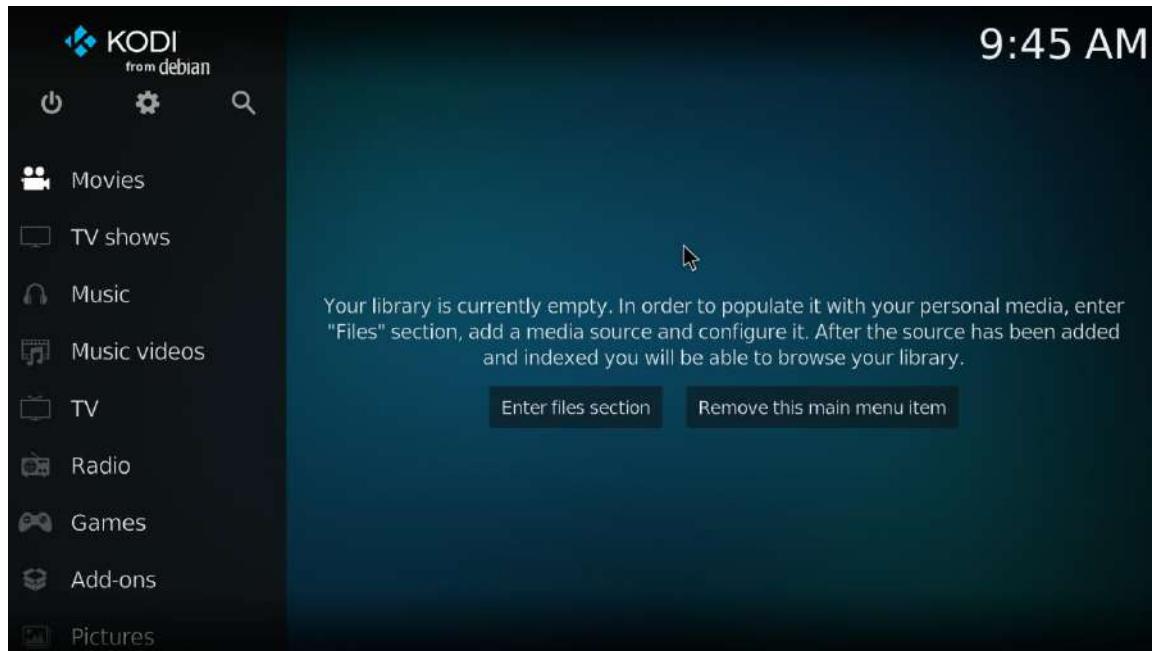
**Orangepi3-lts\_3.0.0\_debian\_bullseye\_server\_linux5.10.76\_kodi\_test.7z**



- 2) First, please connect the development board to the HDMI monitor, keyboard and mouse, and then perform the following operations on the HDMI monitor

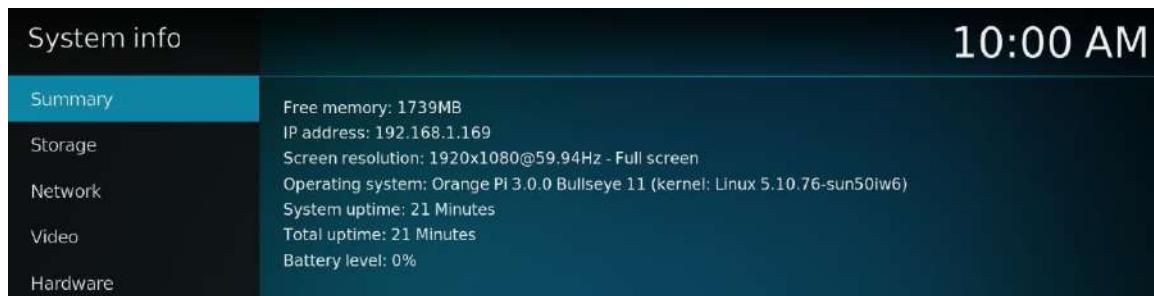
- 3) The command to open Kodi looks like this

**orangeipi@orangeipi:~\$ kodi**



#### 4) The basic information of the system is as follows

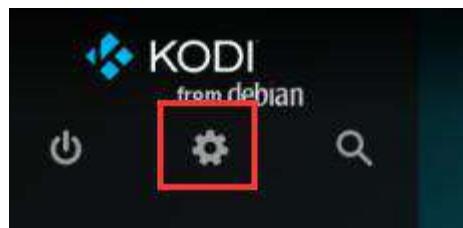




10:00 AM

5) The audio is output to the headset by default, and the method to set the audio output to HDMI is as follows

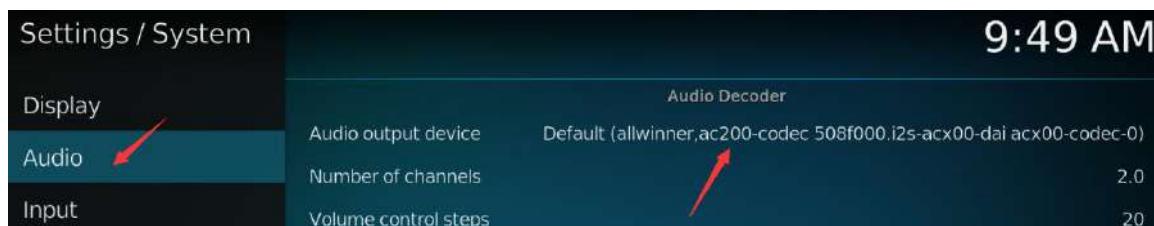
- Click the Settings button



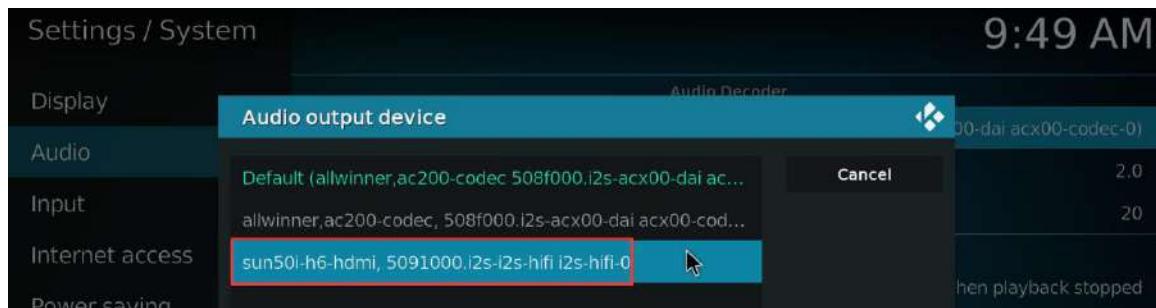
- Then select **System**



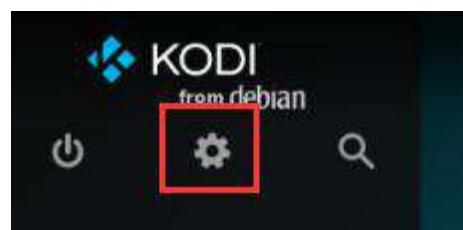
- Then select **Audio**, and then click the **Audio output device** column



- Then select the audio device corresponding to HDMI in the pop-up selection box.



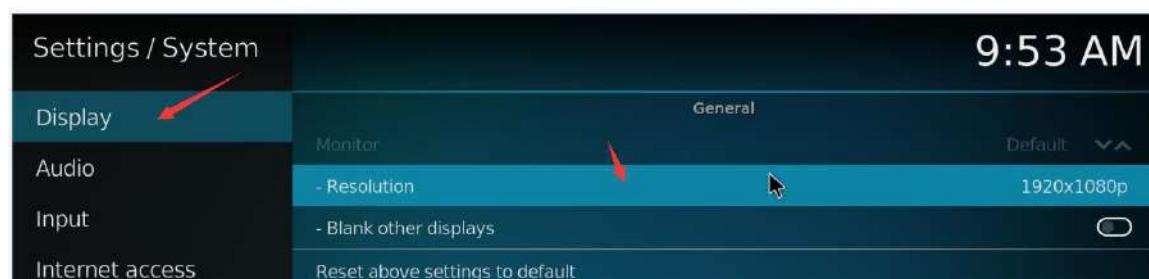
- 6) How to set HDMI resolution  
a. Click the Settings button



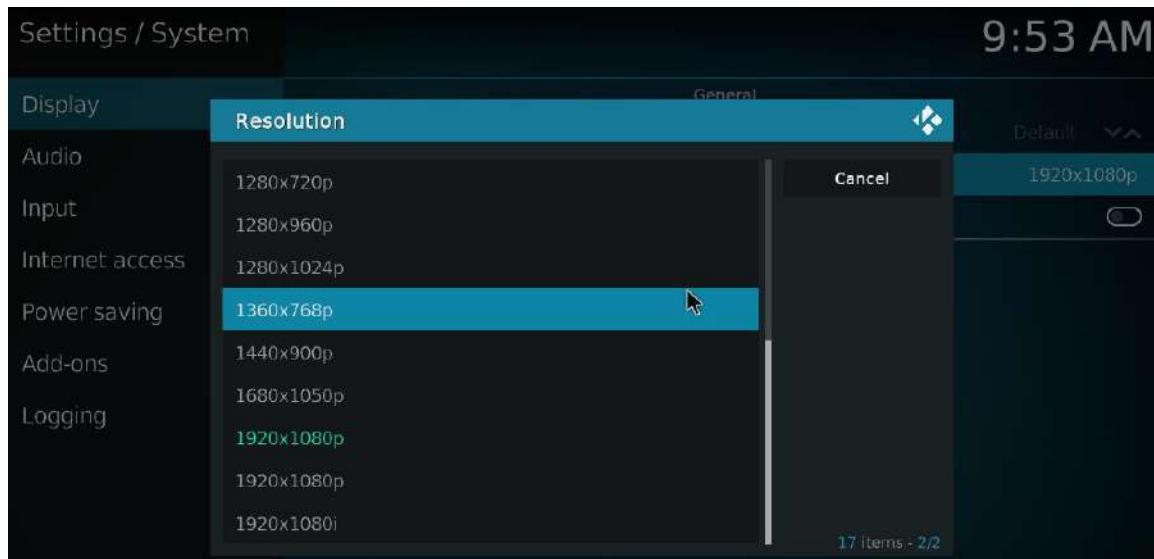
- b. Then select **System**



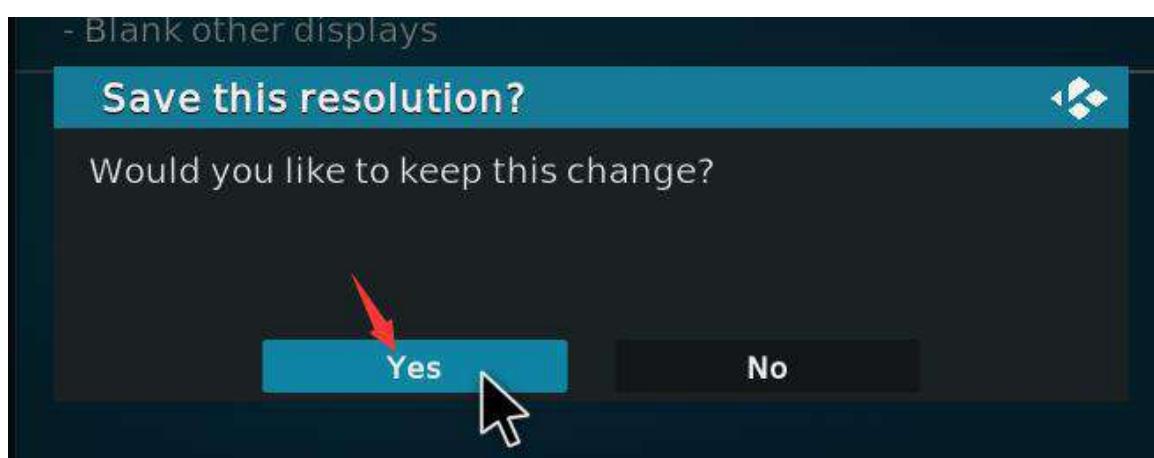
- c. Then select **Display**, then click **Resolution**



- d. Then you can choose the resolution you want to set



- e. Then click Yes to save the settings



## 7) Method to play video

- Currently tested supported video formats are **H264**、**H265**、**VP8** and **MPEG2**
- The tested video can be downloaded in Google cloud disk, from which you can see more detailed information of the test video, the link is as follows:

<https://drive.google.com/drive/folders/1KtVFs4o4VQbyhMlkMkmiMW6CWyTG-tKU?usp=sharing>





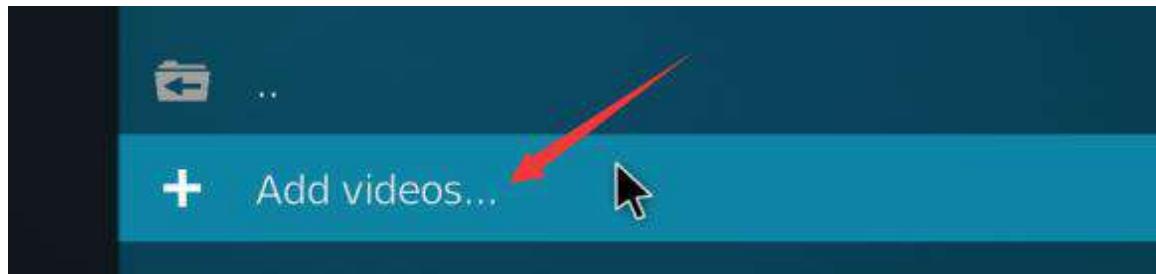
返回上一级 | 全部文件 > orangepi-build > test\_video

| □ 文件名                                          | 大小     |
|------------------------------------------------|--------|
| Jellyfish_1080_10s_30MB_vp8.webm               | 30.1M  |
| Jellyfish_1080_10s_30MB.mkv                    | 29.9M  |
| Big_Buck_Bunny_1080_10s_30MB_h265.mp4          | 30M    |
| Big_Buck_Bunny_1080_10s_30MB.mp4               | 29.2M  |
| big_buck_bunny_1080p_MPEG2_MP2_25fps_6600K.MPG | 38.2M  |
| bbb_sunflower_1080p_60fps_normal.mp4           | 339.4M |
| bbb_sunflower_1080p_30fps_normal.mp4           | 263.3M |

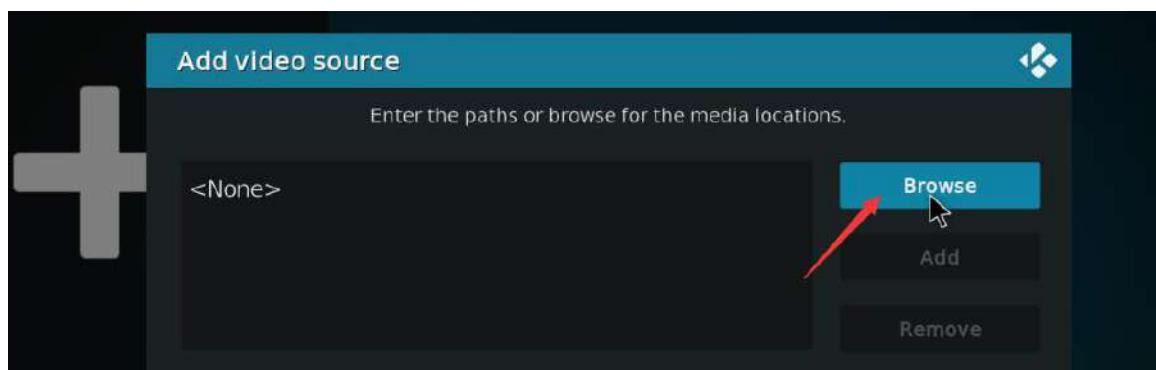
- c. After downloading the **test\_video** folder, please upload it to the **/home/orangepi** directory of the Linux system of the development board
- d. Then go back to the Kodi interface and select **Movies**



- e. Then click **Add videos**



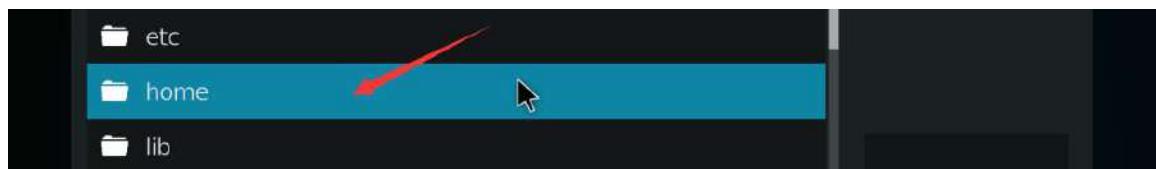
- f. Then click **Browse**



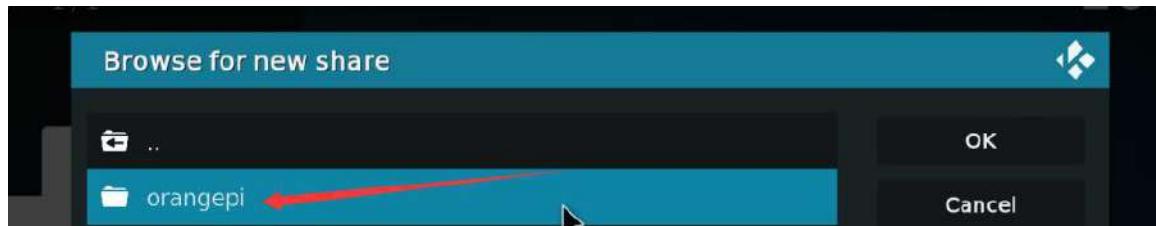
- g. Then select **Root filesystem**



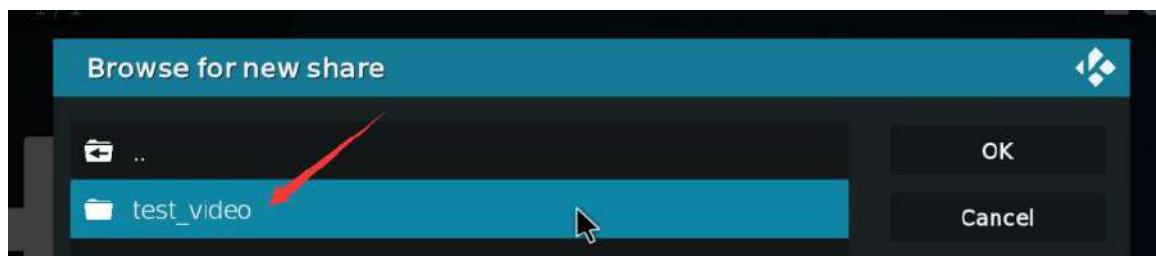
h. Then select the **home** directory



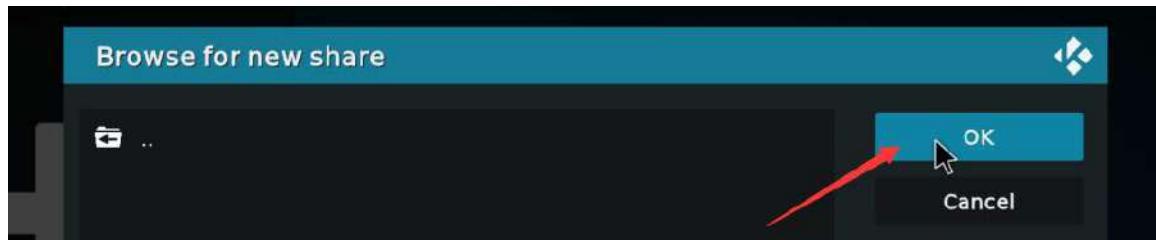
i. Then select the **orangeipi** directory



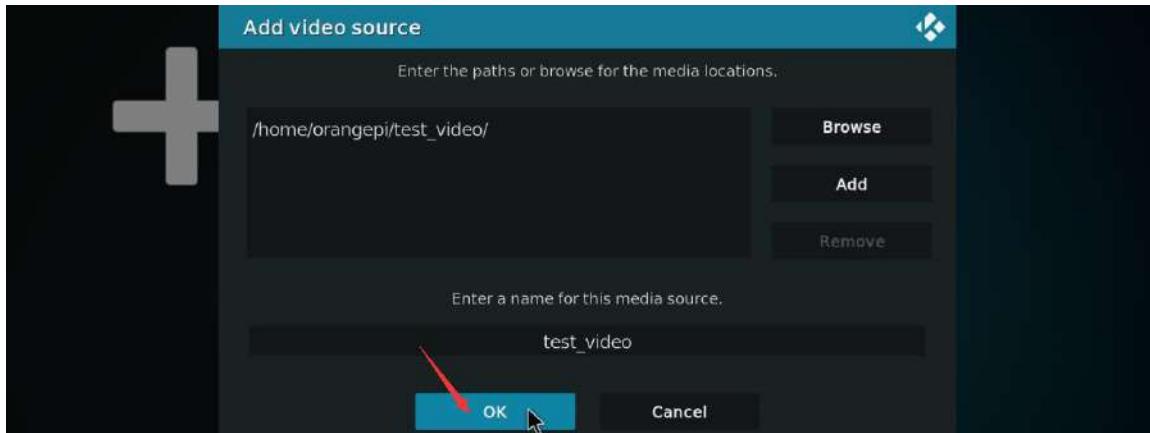
j. Then select the **test\_video** test video folder you just uploaded



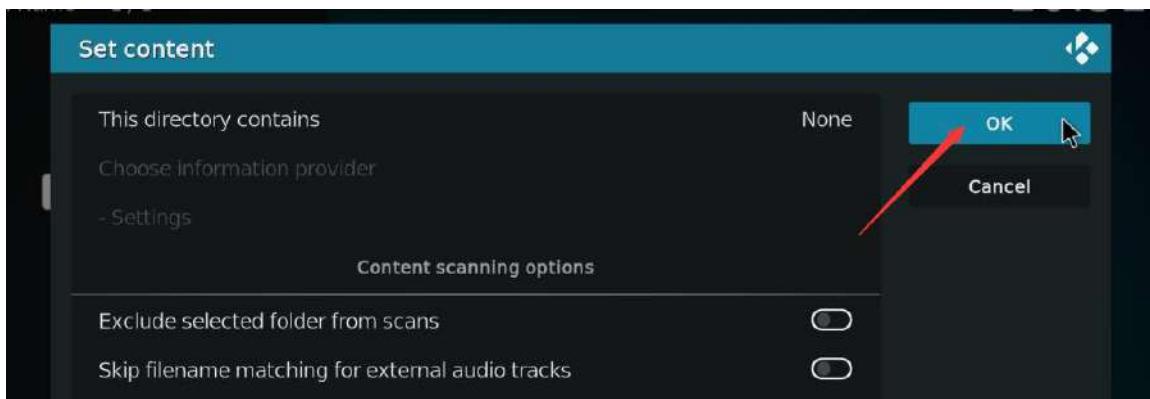
k. Then click **OK**



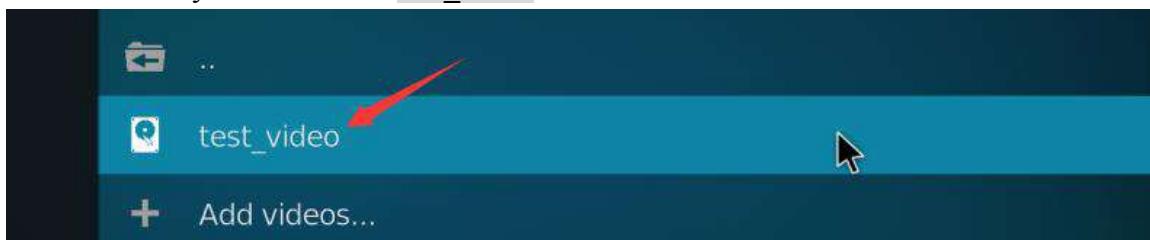
l. Continue to click **OK**



m. Continue to click **OK**



n. Then you can see the **test\_video** folder



o. Click the **test\_video** playback folder again to view the videos playing in it



p. When playing a video, you can open a terminal window and use the top



command to check the CPU usage. The actual CPU usage is generally only  
**15%~30%**

```
orangeipi@orangeipi:~$ top
```

```
top - 10:27:41 up 50 min, 4 users, load average: 2.22, 2.08, 1.75
Tasks: 167 total, 1 running, 166 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.9 us, 2.3 sy, 0.1 ni, 93.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1989.4 total, 548.9 free, 775.4 used, 665.1 buff/cache
MiB Swap: 994.7 total, 994.7 free, 0.0 used. 1111.5 avail Mem

      PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  2379 orangeipi  20   0  78.7g 413432 126292 D 25.7 20.3  8:52.19 kodi.bin
  4671 root       20   0  9976   3220   2744 R  0.7  0.2  0:01.78 top
    11 root       20   0      0      0      0 I  0.3  0.0  0:02.61 rcu_sch+
    21 root       20   0      0      0      0 S  0.3  0.0  0:00.50 ksoftirq+
```

### 3. 34. Tencent ncnn high-performance neural network forward computing framework test

If you don't know what ncnn is, you can read the introduction of ncnn in the README on github.

The ncnn github repository address is: <https://github.com/Tencent/ncnn>

#### ncnn

license BSD-3-Clause downloads 89k codecov 93% code quality: c/c++ A+

ncnn is a high-performance neural network inference computing framework optimized for mobile platforms. ncnn is deeply considerate about deployment and uses on mobile phones from the beginning of design. ncnn does not have third party dependencies. It is cross-platform, and runs faster than all known open source frameworks on mobile phone cpu. Developers can easily deploy deep learning algorithm models to the mobile platform by using efficient ncnn implementation, create intelligent APPs, and bring the artificial intelligence to your fingertips. ncnn is currently being used in many Tencent applications, such as QQ, Qzone, WeChat, Pitu and so on.

ncnn 是一个为手机端极致优化的高性能神经网络前向计算框架。ncnn 从设计之初深刻考虑手机端的部署和使用。无第三方依赖，跨平台，手机端 cpu 的速度快于目前所有已知的开源框架。基于 ncnn，开发者能够将深度学习算法轻松移植到手机端高效执行，开发出人工智能 APP，将 AI 带到你的指尖。ncnn 目前已在腾讯多款应用中使用，如 QQ，Qzone，微信，天天P图等。

- 1) Tencent ncnn source code download command is as follows
  - a. The first method: Download the **ncnn.tar.gz** compressed package provided in



the Orange Pi Google cloud disk

- a) From the Google cloud disk link below, you can download the **ncnn.tar.gz** source code compressed package. Go to the ncnn folder and you can see

```
https://drive.google.com/drive/folders/1JAj7xT1ViZKH51BZ-dUqM9bBySQILpZ1?usp=sharing
```

|                          |                  |
|--------------------------|------------------|
| orangepi-build           | 2020-12-09 20:37 |
| ncnn                     | 2022-04-11 10:19 |
| ncnn_test_demo_1g.tar.gz | 2022-04-11 10:19 |
| ncnn_test_demo.tar.gz    | 2022-04-11 10:19 |
| ncnn.tar.gz              | 2022-04-11 10:19 |

- b) After downloading the **ncnn.tar.gz** compressed package, first upload **ncnn.tar.gz** to the Linux system of the development board
- c) Then extract **ncnn.tar.gz** using the following command

```
orangeipi@orangeipi:~$ tar zxf ncnn.tar.gz  
orangeipi@orangeipi:~$ ls  
ncnn ncnn.tar.gz
```

b. The second method: use the git command to download the source code directly, but if the problem of accessing github from the development board is not solved, it is difficult to download successfully. If there is no problem accessing github, it is recommended to use this method, because this way the code is guaranteed to be up-to-date.

```
orangeipi@orangeipi:~$ git clone https://github.com/Tencent/ncnn.git  
orangeipi@orangeipi:~$ cd ncnn  
orangeipi@orangeipi:~/ncnn$ git submodule update --init
```

2) Then install the dependency package

```
orangeipi@orangeipi:~$ sudo apt update  
orangeipi@orangeipi:~$ sudo apt install -y build-essential git cmake \\\nlibprotobuf-dev protobuf-compiler libopencv-dev
```

3) Then start compiling, the ncnn compilation command is as follows

```
orangeipi@orangeipi:~$ cd ncnn  
orangeipi@orangeipi:~/ncnn$ mkdir build  
orangeipi@orangeipi:~/ncnn$ cd build  
orangeipi@orangeipi:~/ncnn/build$ cmake \\\n
```



```
-DCMAKE_TOOLCHAIN_FILE=../toolchains/aarch64-linux-gnu.toolchain.cmake \
-DNCNN_SIMPLEOCV=ON -DNCNN_BUILD_EXAMPLES=ON ..
orangeipi@orangeipi:~/ncnn/build$ make -j$(nproc)
[ 0%] Built target ncnn-generate-spirv
[ 1%] Generating source absval_arm_arm82.h
[ 1%] Generating source convolution1d_arm_arm82.cpp
.....
[ 99%] Linking CXX executable squeezenetssd
[ 99%] Built target squeezenetssd
[100%] Linking CXX executable shufflenetv2
[100%] Built target shufflenetv2
```

**Without any cooling measures, it takes about 15 minutes to compile ncnn directly on the development board, please wait patiently for the compilation to complete. If there is a fan to cool the development board, the speed should be faster.**

- 4) There are some test examples in ncnn, such as **squeeze** test commands and results are as follows

```
orangeipi@orangeipi:~/ncnn/build$ cd ../examples
orangeipi@orangeipi:~/ncnn/examples$ ./build/examples/squeeze
./images/256-ncnn.png
532 = 0.165950
920 = 0.094098
716 = 0.062193
```

- 5) **benchncnn** can be used to test the reasoning performance of the neural network. The test method is as follows

- a. The **benchncnn** executable file generated by compilation is in the following path.  
**Note that the execution path of the following command is the top-level directory of the ncnn source code**

```
orangeipi@orangeipi:~/ncnn$ ls build/benchmark/
benchncnn CMakeFiles cmake_install.cmake Makefile
```

- b. First, you need to copy **benchmarkcnn** to the **benchmark** directory

```
orangeipi@orangeipi:~/ncnn$ cp build/benchmark/benchncnn benchmark/
```

- c. The usage of **benchcnn** is as follows



**./benchncnn [loop count] [num threads] [powersave] [gpu device] [cooling down]**

#### Parameter

| param        | options                                            | default       |
|--------------|----------------------------------------------------|---------------|
| loop count   | 1~N                                                | 4             |
| num threads  | 1~N                                                | max_cpu_count |
| powersave    | 0=all cores, 1=little cores only, 2=big cores only | 0             |
| gpu device   | -1=cpu-only, 0=gpu0, 1=gpu1 ...                    | -1            |
| cooling down | 0=disable, 1=enable                                | 1             |

<https://github.com/Tencent/ncnn/blob/9d0c36358cec2d1da471574064d4abd8787b45a8/benchmark/README.md>

d. **benchncnn** using cpu test results are as follows

```
orangeipi@orangeipi:~/ncnn$ cd benchmark
```

```
orangeipi@orangeipi:~/ncnn/benchmark$ ./benchncnn 4 $(nproc) 0 -1
```

a) Debian Bullseye Linux 5.16 server version system test results



```
orangepi@orangepi3-lts:~/ncnn_test_demo/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
    squeezenet min = 80.49 max = 81.51 avg = 81.19
    squeezenet_int8 min = 64.92 max = 65.26 avg = 65.02
    mobilenet min = 100.09 max = 143.02 avg = 113.31
    mobilenet_int8 min = 56.67 max = 59.17 avg = 57.70
    mobilenet_v2 min = 101.38 max = 101.84 avg = 101.53
    mobilenet_v3 min = 80.03 max = 81.62 avg = 81.10
    shufflenet min = 51.54 max = 52.98 avg = 52.41
    shufflenet_v2 min = 46.57 max = 47.20 avg = 46.90
    mnasnet min = 83.53 max = 84.68 avg = 84.21
    proxylessnasnet min = 89.09 max = 93.87 avg = 92.03
    efficientnet_b0 min = 129.80 max = 131.78 avg = 130.68
    efficientnetv2_b0 min = 146.08 max = 147.71 avg = 147.18
    regnet_y_400M min = 101.23 max = 102.08 avg = 101.59
    blazeface min = 18.76 max = 21.01 avg = 20.23
    googlenet min = 219.47 max = 227.17 avg = 224.13
    googlenet_int8 min = 166.44 max = 175.20 avg = 176.54
    resnet18 min = 264.23 max = 265.07 avg = 264.45
    resnet18_int8 min = 150.37 max = 153.43 avg = 151.72
    alexnet min = 218.41 max = 221.63 avg = 219.33
    vgg16 min = 1290.36 max = 1315.25 avg = 1301.47
    vgg16_int8 min = 679.03 max = 688.06 avg = 683.09
    resnet50 min = 518.51 max = 524.94 avg = 523.31
    resnet50_int8 min = 348.23 max = 363.29 avg = 357.71
    squeezenet_ssd min = 276.15 max = 278.01 avg = 277.53
    squeezenet_ssd_int8 min = 182.26 max = 183.13 avg = 182.72
    mobilenet_ssd min = 222.64 max = 232.63 avg = 227.07
    mobilenet_ssd_int8 min = 117.66 max = 126.97 avg = 122.18
    mobilenet_yolo min = 473.25 max = 513.59 avg = 485.23
    mobilenetv2_yolov3 min = 322.55 max = 331.23 avg = 326.91
    yolov4-tiny min = 421.52 max = 427.36 avg = 425.22
    nanodet_m min = 125.49 max = 128.88 avg = 127.30
    yolo-fastest-1.1 min = 71.44 max = 74.76 avg = 72.87
    yolo-fastestv2 min = 55.44 max = 56.23 avg = 55.84
orangepi@orangepi3-lts:~/ncnn_test_demo/benchncnn_demo$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux 11 (bullseye)
Release:        11
Codename:       bullseye
orangepi@orangepi3-lts:~/ncnn_test_demo/benchncnn_demo$ uname -r
5.16.17+sun50iw6
orangepi@orangepi3-lts:~/ncnn test demo/benchncnn demo$
```

6) NanoDet is an ultra-fast and lightweight mobile Anchor-free object detection model.

The test method is as follows

- The compiled **nanodet** executable file is in the following path. **Note that the execution path of the following command is the upper level directory of the ncnn source code.**

```
orangepi@orangepi:~$ ls ncnn/build/examples/nanodet
ncnn/build/examples/nanodet
```

- First create a new **nanonodet\_demo** folder

```
orangepi@orangepi:~$ mkdir nanodet_demo
```

- Then copy the compiled **nanodet** executable program to the **nanodet\_demo** folder

```
orangepi@orangepi:~$ cp ncnn/build/examples/nanodet nanodet_demo/
```

- Then you need to download the **nanodet** model file and upload it to the **nanodet\_demo** folder
  - The download address of the **nanodet** model file is as follows

```
https://github.com/nihui/ncnn-assets/tree/master/models
```



- b) Open the link above, find the two files **nanodet\_m.bin** and **nanodet\_m.param**, download them, and upload them to the **nanodet\_demo** folder of the Linux system of the development board

|                 |                      |               |
|-----------------|----------------------|---------------|
| nanodet_m.bin   | Add files via upload | 16 months ago |
| nanodet_m.param | Add files via upload | 16 months ago |

- c) At this point, there should be the following three files in the **nanodet\_demo** folder

```
orangeipi@orangeipi:~$ cd nanodet_demo  
orangeipi@orangeipi:~/nanodet_demo$ ls  
nanodet nanodet_m.bin nanodet_m.param
```

- e. Then you need to put the pictures you want to detect in the **nanodet\_demo** folder, such as the picture below with many cars (you can use your mobile phone to take a few pictures of traffic or animals)

```
orangeipi@orangeipi:~/nanodet_demo$ ls  
car.jpg nanodet nanodet_m.bin nanodet_m.param
```



- f. Then run the following command to use **nanodet** for target detection, please replace **car.jpg** with the name of your image

```
orangeipi@orangeipi:~/nanodet_demo$ ./nanodet car.jpg  
2 = 0.73488 at 536.36 408.79 103.68 x 79.63  
2 = 0.73003 at 74.47 530.85 184.61 x 131.70  
2 = 0.68989 at 724.94 305.76 58.30 x 49.73  
2 = 0.65828 at 412.10 348.38 80.33 x 64.65  
2 = 0.64167 at 152.09 257.67 61.52 x 49.91
```

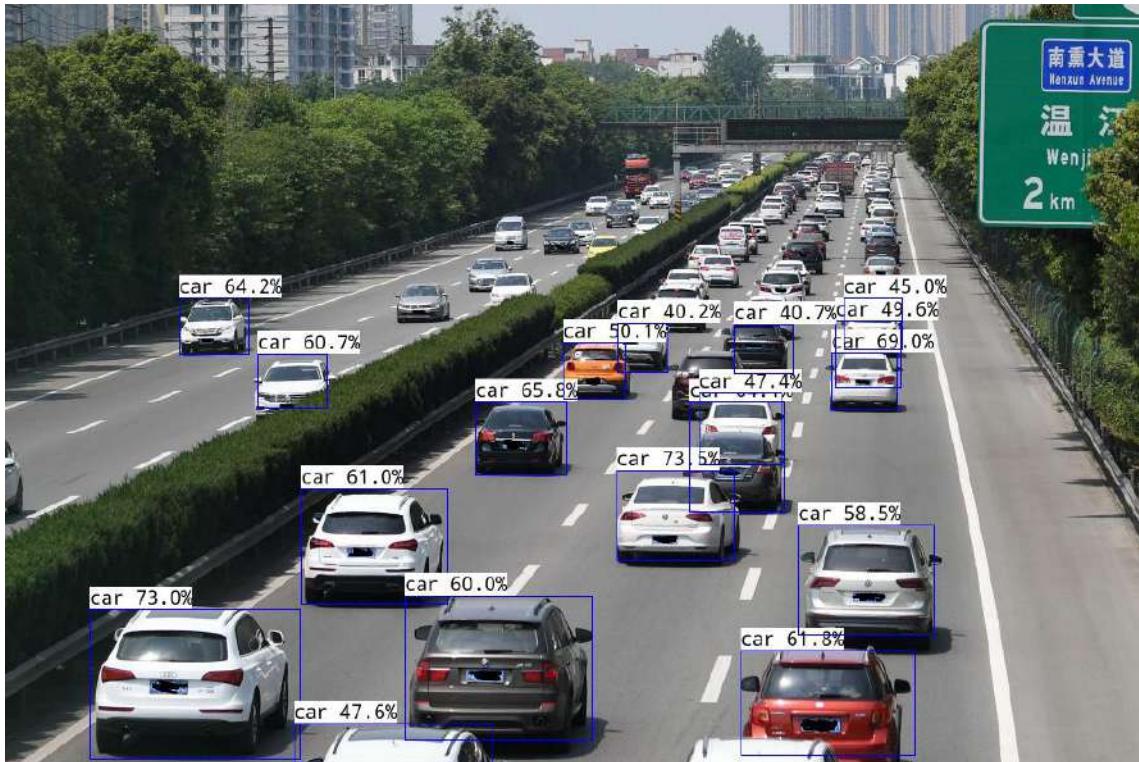


```
2 = 0.64124 at 600.63 348.06 83.82 x 96.93
2 = 0.61759 at 645.80 566.98 152.59 x 92.63
2 = 0.61004 at 259.78 424.55 128.62 x 101.88
2 = 0.60663 at 221.76 306.18 61.54 x 47.73
2 = 0.60043 at 350.11 518.50 164.37 x 126.59
2 = 0.58546 at 695.82 456.88 119.12 x 96.87
2 = 0.50075 at 489.94 296.66 54.39 x 49.01
2 = 0.49616 at 728.32 277.73 57.70 x 58.59
2 = 0.47553 at 253.21 630.12 174.18 x 35.88
2 = 0.47408 at 608.36 340.09 72.34 x 63.40
2 = 0.44989 at 735.15 257.97 51.55 x 45.79
2 = 0.40665 at 639.82 280.51 52.43 x 43.13
2 = 0.40169 at 537.50 279.67 44.98 x 43.54
imshow save image to image.png
waitKey stub
```

- g. The result of the detection will be saved in an image named **image.png**

```
orangepi@orangepi:~/nanodet_demo$ ls
car.jpg  image.png  nanodet  nanodet_m.bin  nanodet_m.param
```

- h. If you are using a desktop version of Linux system, you can directly open **image.png** to view it. If you are using a server version of Linux system, you can copy **image.png** to your computer for viewing. The content of **image.png** is shown in the figure below. You can see that the upper left corner of the recognized object will display the type of object and the percentage of reliability



7) In order to facilitate the testing of **benchcnn** and **nanodet**, I have compiled an executable file containing only **benchcnn** and **nanodet** and the model files required for the test, packaged into a **ncnn\_test\_demo.tar.gz** compressed package and placed it on Google cloud disk, no need to download Compile the source code of **ncnn**, use this executable program to start the test directly

- a. You can download the **ncnn\_test\_demo.tar.gz** compressed package from the Google cloud disk link below. Go to the ncnn folder and you can see

<https://drive.google.com/drive/folders/1JAj7xT1ViZKH51BZ-dUqM9bBySQILpZ1?usp=sharing>

|                          |        |                  |
|--------------------------|--------|------------------|
| orangepi-build           |        | 2020-12-09 20:37 |
| ncnn                     |        | 2022-04-11 10:19 |
| ncnn_test_demo_1g.tar.gz | 5.9M   | 2022-04-11 10:19 |
| ncnn_test_demo.tar.gz    | 5.9M   | 2022-04-11 10:19 |
| ncnn.tar.gz              | 106.6M | 2022-04-11 10:19 |

- b. After downloading the **ncnn\_test\_demo.tar.gz** compressed package, first upload the **ncnn\_test\_demo.tar.gz** compressed package to the Linux system of the development board
- c. Then use the following command to extract **ncnn\_test\_demo.tar.gz**



```
orangeipi@orangeipi:~$ tar ncnn_test_demo.tar.gz
```

- d. After decompression, enter the **ncnn\_test\_demo** directory and you can see that it contains two subfolders, **benchncnn\_demo** and **nanonet\_demo**, which are used to test benchncnn and nanonet respectively

```
orangeipi@orangeipi:~$ cd ncnn_test_demo
```

```
orangeipi@orangeipi:~/ncnn_test_demo$ ls
```

```
benchncnn_demo nanonet_demo
```

- e. Enter the **benchncnn\_demo** folder, and then run the command **./benchncnn 4 \$(nproc) 0 -1** to directly test the inference performance of the neural network

```
orangeipi@orangeipi:~/ncnn_test_demo$ cd benchncnn_demo
```

```
orangeipi@orangeipi:~/ncnn_test_demo/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
```

- f. Enter the **nanonet\_demo** folder, and then run the **./nanonet car.jpg** command to directly use **nanonet** to detect objects in the **car.jpg** image. You can also put the image you want to detect in the **nanonet\_demo** folder, and then Use nanonet to detect

```
orangeipi@orangeipi:~/ncnn_test_demo$ cd nanonet_demo
```

```
orangeipi@orangeipi:~/ncnn_test_demo/nanonet_demo$ ./nanonet car.jpg
```

Note that the content demonstrated in this section is mainly to prove that ncnn can be compiled and run normally on Orange Pi's development board and system. If there is any problem with the content demonstrated in this section, you can provide feedback and provide technical support (such as ncnn source code download failure, ncnn There are problems with compilation, problems with benchncnn and nanonet tests), but other things beyond this section cannot provide technical support, please study by yourself.

### 3. 35. Installation and testing method of face\_recognition face recognition library

Note that the content in this section is tested on the desktop version of Linux system, so please make sure that the system used by the development board is the desktop version system.

In addition, the following installation tests are carried out under the orangeipi user, please keep the environment consistent.



The address of the face\_recognition source code repository is:

[https://github.com/ageitgey/face\\_recognition](https://github.com/ageitgey/face_recognition)

The documentation for the Chinese version of face\_recognition is:

[https://github.com/ageitgey/face\\_recognition/blob/master/README\\_Simplified\\_Chinese.md](https://github.com/ageitgey/face_recognition/blob/master/README_Simplified_Chinese.md)

### 3. 35. 1. Automatic installation of face\_recognition using script

- 1) First open a terminal in the desktop and download face\_recognition\_install.sh

```
orangeipi@orangeipi:~/Desktop$ wget \
https://gitee.com/leeboby/face\_recognition\_install/raw/master/face\_recognition\_install.sh
```

```
File Edit View Terminal Help
orangeipi@orangeipi:~/Desktop$ wget \
-a https://gitee.com/leeboby/face_recognition_install/raw/master/face_recognition_install.sh
--2022-05-14 13:59:03 -- https://gitee.com/leeboby/face_recognition_install/raw/master/face_recognition_install.sh
Resolving gitee.com (gitee.com)... 125.208.125.208
Connecting to gitee.com (gitee.com)|125.208.125.208|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1800 (1.8K) [text/plain]
Saving to: "face_recognition_install.sh"
face_recognition_install.sh          100%[=====]  1.75M/ 1.8K  1m 3s
2022-05-14 13:59:03 155.8 MB/s
orangeipi@orangeipi:~/Desktop$
```

- 2) Then execute the following command to start installing face\_recognition

```
orangeipi@orangeipi:~/Desktop$ bash face_recognition_install.sh
```

- 3) After face\_recognition is installed, it will automatically download the source code of face\_recognition, and then automatically run some examples in face\_recognition. If you can see the following pictures pop up on the desktop at the end, it means that the face\_recognition installation test is successful



### 3.35.2. Manual installation of `face_recognition`

1) First create a new `~/.pip` directory, then add the `pip.conf` configuration file, and set the pip image source to Tsinghua source in it. The commands to be executed are as follows:

```
orangepi@orangepi:~$ mkdir -p ~/.pip
orangepi@orangepi:~$ cat <<EOF > ~/.pip/pip.conf
[global]
timeout = 6000
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
trusted-host = pypi.tuna.tsinghua.edu.cn
EOF
```

2) Then install the dependency package

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y python3-pip libopencv-dev \
python3-opencv imagemagick python3-scipy python3-setuptools python3-wheel \
python3-dev cmake python3-testresources
```

3) Then update pip3

```
orangepi@orangepi:~$ python3 -m pip install -U pip setuptools wheel
```

4) Before installing `face_recognition`, you first need to install the `dlib` library. Since the



dlib library is slow to compile and install on the development board, I saved a compiled dlib whl file on **gitee**, and it can be installed directly after downloading. The download address of the dlib whl file is as follows:

**[https://gitee.com/leeboby/python\\_whl](https://gitee.com/leeboby/python_whl)**

- a. First download the python\_whl repository to the Linux system of the development board

```
orangeipi@orangeipi:~$ git clone --depth=1 https://gitee.com/leeboby/python_whl
```

- b. In the python\_whl folder, you can see that there are multiple versions of the dlib installation package. The Linux systems corresponding to different versions of dlib are as follows:

|                    |                                                   |
|--------------------|---------------------------------------------------|
| <b>Ubuntu18.04</b> | <b>dlib-19.24.0-cp36-cp36m-linux_aarch64.whl</b>  |
| <b>Ubuntu20.04</b> | <b>dlib-19.24.0-cp38-cp38-linux_aarch64.whl</b>   |
| <b>Ubuntu22.04</b> | <b>dlib-19.24.0-cp310-cp310-linux_aarch64.whl</b> |
| <b>Debian10</b>    | <b>dlib-19.24.0-cp37-cp37m-linux_aarch64.whl</b>  |
| <b>Debian11</b>    | <b>dlib-19.24.0-cp39-cp39-linux_aarch64.whl</b>   |

- c. Then you can start to install dlib, the command is as follows

- a) Ubuntu18.04

```
orangeipi@orangeipi:~$ cd python_whl  
orangeipi@orangeipi:~/python_whl$ python3 -m pip install  
dlib-19.24.0-cp36-cp36m-linux_aarch64.whl
```

- b) Ubuntu20.04

```
orangeipi@orangeipi:~$ cd python_whl  
orangeipi@orangeipi:~/python_whl$ python3 -m pip install  
dlib-19.24.0-cp38-cp38-linux_aarch64.whl
```

- c) Ubuntu22.04

```
orangeipi@orangeipi:~$ cd python_whl  
orangeipi@orangeipi:~/python_whl$ python3 -m pip install  
dlib-19.24.0-cp310-cp310-linux_aarch64.whl
```

- d) Debian10

```
orangeipi@orangeipi:~$ cd python_whl  
orangeipi@orangeipi:~/python_whl$ python3 -m pip install  
dlib-19.24.0-cp37-cp37m-linux_aarch64.whl
```

- e) Debian11

```
orangeipi@orangeipi:~$ cd python_whl
```



```
orangeipi@orangeipi:~/python_whl$ python3 -m pip install  
dlib-19.24.0-cp39-cp39-linux_aarch64.whl
```

- d. After installation, if the version number of dlib can be printed normally by using the following command, it means that dlib is installed correctly

```
orangeipi@orangeipi:~/python_whl$ python3 -c "import dlib; print(dlib.__version__)"  
19.24.0
```

- 5) then install **face\_recognition\_models-0.3.0-py2.py3-none-any.whl**

```
orangeipi@orangeipi:~/python_whl$ python3 -m pip install \  
face_recognition_models-0.3.0-py2.py3-none-any.whl
```

- 6) then install **face\_recognition**

```
orangeipi@orangeipi:~$ python3 -m pip install face_recognition
```

- 7) Then you **need to reopen a terminal** to find and run the two commands **face\_detection** and **face\_recognition**

- face\_recognition command is used to identify whose face is in a single image or a folder of images
- face\_detection command is used to locate the position of the face in a single image or a folder of images

```
orangeipi@orangeipi:~$ which face_detection  
/usr/local/bin/face_detection  
orangeipi@orangeipi:~$ which face_recognition  
/usr/local/bin/face_recognition
```

Or run the following commands in the terminal, you can find the above two commands without reopening the terminal

```
orangeipi@orangeipi:~$ export PATH=/home/orangeipi/.local/bin:$PATH
```

### 3. 35. 3. Test method of face\_recognition

Note that the following operations are demonstrated on the desktop, so please connect the HDMI display first, or use NoMachine/VNC to log in to the Linux desktop to test.

- There are some sample codes in the source code of **face\_recognition**, which we can use for testing directly. The download address of the source code of face\_recognition is as



follows:

- a. GitHub official download address

```
orangeipi@orangeipi:~$ git clone https://github.com/ageitgey/face_recognition.git
```

- b. Gitee image download address

```
orangeipi@orangeipi:~$ git clone https://gitee.com/leeboby/face_recognition.git
```

2) The path of the face\_recognition sample code is as follows

```
face_recognition/examples
```

3) The link to the Chinese documentation of face\_recognition is as follows, please read it carefully before using face\_recognition

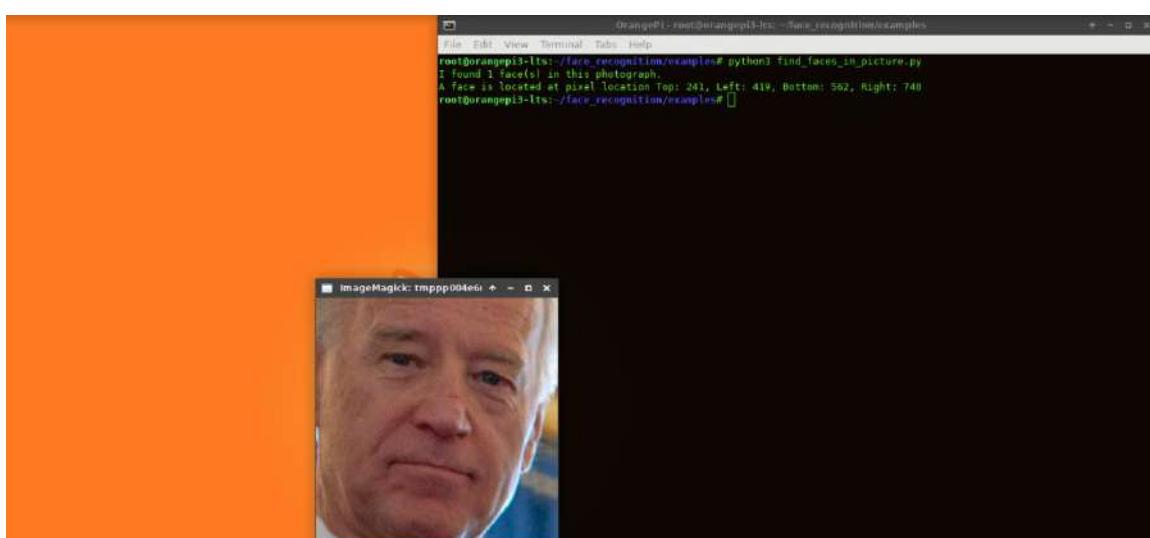
```
https://github.com/ageitgey/face\_recognition/blob/master/README\_Simplified\_Chinese.md
```

4) **find\_faces\_in\_picture.py** is used to locate the position of the face in the picture. The test steps are as follows

- a. Open a terminal on the desktop, enter the **face\_recognition/examples** directory, and execute the following command

```
orangeipi@orangeipi:~$ cd face_recognition/examples  
orangeipi@orangeipi:~/face_recognition/examples$ python3 find_faces_in_picture.py  
I found 1 face(s) in this photograph.  
A face is located at pixel location Top: 241, Left: 419, Bottom: 562, Right: 740
```

- b. Wait for a while and the following picture will pop up, this is the face located in the test picture



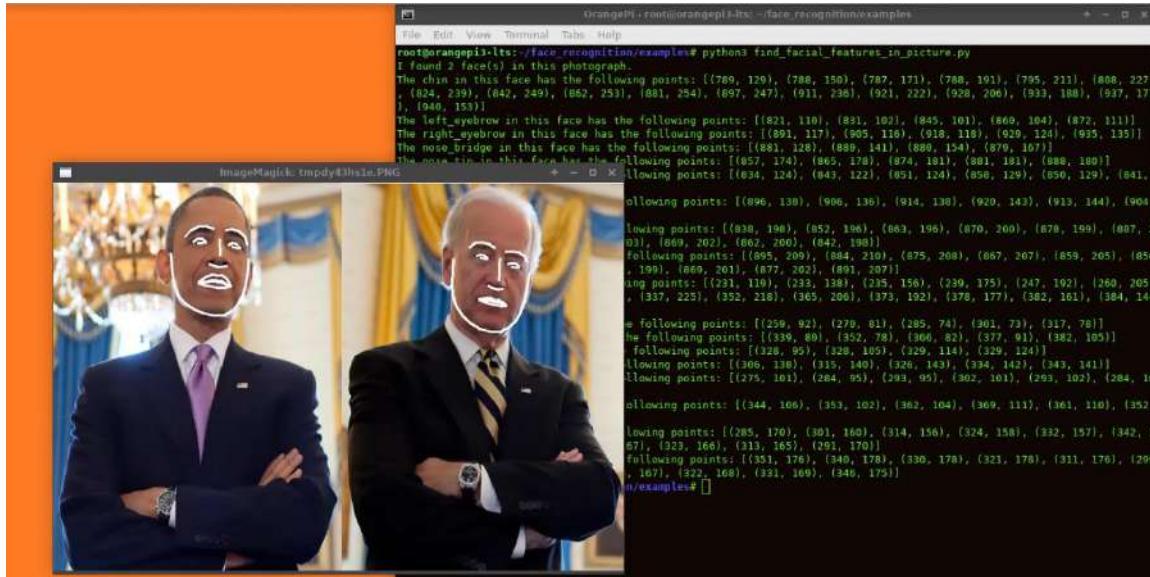


5) **find\_facial\_features\_in\_picture.py** is used to identify the key points of the face in a single picture. The test steps are as follows

- Open a terminal on the desktop, enter the **face\_recognition/examples** directory, and execute the following command

```
orangeipi@orangeipi:~/face_recognition/examples  
orangeipi@orangeipi:~/face_recognition/examples$ python3 \  
find_facial_features_in_picture.py
```

- After waiting for a while, the picture below will pop up, and you can see that the outlines of the faces are marked



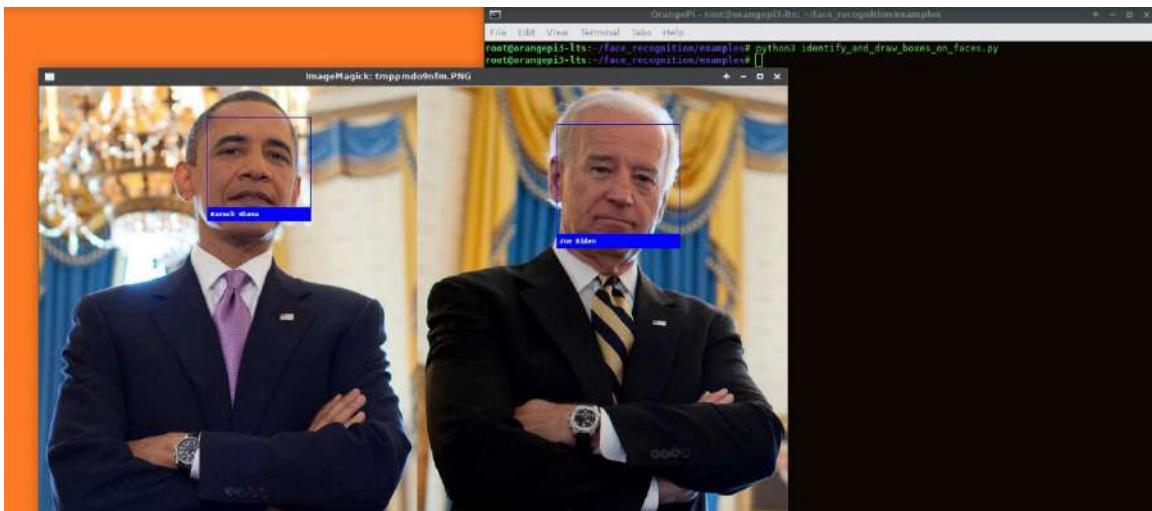
6) **identify\_and\_draw\_boxes\_on\_faces.py** is used to identify faces and use box labels.

The test steps are as follows

- Open a terminal on the desktop, enter the **face\_recognition/examples** directory, and execute the following command

```
orangeipi@orangeipi:~/face_recognition/examples  
orangeipi@orangeipi:~/face_recognition/examples$ python3 \  
identify_and_draw_boxes_on_faces.py
```

- After waiting for a while, the following picture will pop up. You can see that the faces in the picture are marked with boxes, and the names of the characters are displayed correctly.



7) **face\_distance.py** is used to compare whether two faces belong to the same person at different precisions. First open a terminal, then enter the **face\_recognition/examples** directory, and then execute the following command to see the output of the test

```
orangeipi@orangeipi:~$ cd face_recognition/examples
orangeipi@orangeipi:~/face_recognition/examples$ python3 face_distance.py
The test image has a distance of 0.35 from known image #0
- With a normal cutoff of 0.6, would the test image match the known image? True
- With a very strict cutoff of 0.5, would the test image match the known image? True

The test image has a distance of 0.82 from known image #1
- With a normal cutoff of 0.6, would the test image match the known image? False
- With a very strict cutoff of 0.5, would the test image match the known image?
  False
```

8) **recognize\_faces\_in\_pictures.py** is used to identify who the faces in unknown pictures are. First open a terminal, then enter the **face\_recognition/examples** directory, and then execute the following command, wait for one end to see the test results

```
orangeipi@orangeipi:~$ cd face_recognition/examples
orangeipi@orangeipi:~/face_recognition/examples$ python3 \
recognize_faces_in_pictures.py
Is the unknown face a picture of Biden? False
Is the unknown face a picture of Obama? True
Is the unknown face a new person that we've never seen before? False
```



9) **facerec\_from\_webcam\_faster.py** is used to identify the face in the USB camera. The test steps are as follows:

- a. First, please insert the USB camera into the USB interface of the development board, and then use the v4l2-ctl (**note that the l in v4l2 is a lowercase letter l, not the number 1**) command to check the serial number of the device node of the USB camera

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y v4l-utils
orangepi@orangepi:~$ v4l2-ctl --list-devices
cedrus (platform:cedrus):
    /dev/video0
```

USB2.0 UVC PC Camera: USB2.0 UV (usb-5311000.usb-1):

**/dev/video1**

**/dev/video2**

- b. Then open a terminal on the desktop, enter the **face\_recognition/examples** directory, and first modify the device serial number of the camera used in **facerec\_from\_webcam\_faster.py**. For example, through the v4l2-ctl --list-devices command above, you can see that the USB camera is /dev/video1, then modify **0** in **cv2.VideoCapture(0)** to **1**

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ vim \
facerec_from_webcam_faster.py
video_capture = cv2.VideoCapture(1)
```

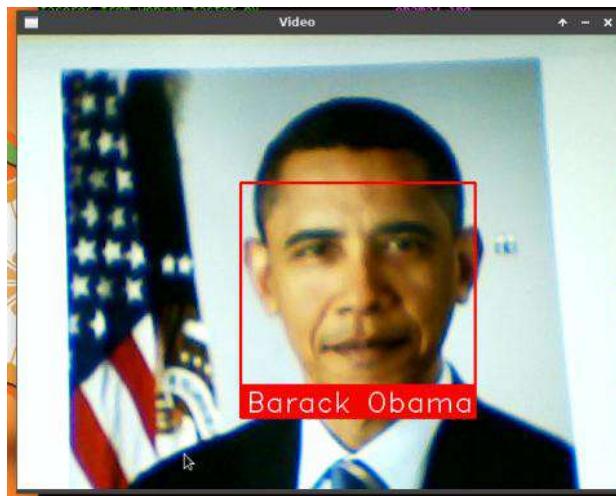
- c. Then execute the following command to run **facerec\_from\_webcam\_faster.py**

```
orangepi@orangepi:~/face_recognition/examples$ python3 \
facerec_from_webcam_faster.py
```

- d. Wait for a while and the display screen of the camera will pop up



- e. At this point, you can point the camera at yourself. When the camera detects a face, it will use a box to frame the detected face. **Note that when detecting faces, the picture displayed by the camera will be relatively stuck, please do not move too fast**
- f. You can also open a picture of Obama, and then use the camera to aim at the opened picture, you can see that not only the face can be marked, but also the name of the detected face can be displayed correctly. **Note that when detecting faces, the picture displayed by the camera will be relatively stuck, please do not move too fast**



- 10) **web\_service\_example.py** is a very simple case of using a web service to upload a picture to run face recognition. The back-end server will identify whether the picture is Obama, and output the recognition result as a json key-value pair. The test steps are as follows:



- a. Open a terminal on the desktop, then enter the **face\_recognition/examples** directory, and execute the following command (if it is face\_recognition that is automatically installed using a script, then you don't need to install flask)

```
orangepi@orangepi:~$ python3 -m pip install flask
orangepi@orangepi:~$ cd face_recognition/examples
root@orangepi:~/face_recognition/examples$ python3 web_service_example.py
* Serving Flask app 'web_service_example' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses (0.0.0.0)
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://127.0.0.1:5001
* Running on http://192.168.1.79:5001 (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 500-161-390
```

- b. Then run the following command to return the result of image recognition (note that the execution path of the following command is **face\_recognition/examples**)

```
orangepi@orangepi:~/face_recognition/examples$ curl -XPOST -F \
"file=@obama2.jpg" http://127.0.0.1:5001
{
  "face_found_in_image": true,
  "is_picture_of_obama": true
}
```

- c. We can also copy the picture **face\_recognition/examples/obama2.jpg** to other Linux computers, of course, we can also prepare a picture named **obama2.jpg** by ourselves, and then use the following command to remotely remote from the Linux computer Identify the face through the service running on the development board (note that the IP address in the command needs to be replaced with the IP address of the development board, and the file name after the file needs to be replaced with the name of the image you want to test)



```
test@test:~$ curl -XPOST -F "file=@obama2.jpg" http://192.168.1.79:5001
```

```
{  
  "face_found_in_image": true,  
  "is_picture_of_obama": true  
}
```

d. The method of using the browser test is as follows:

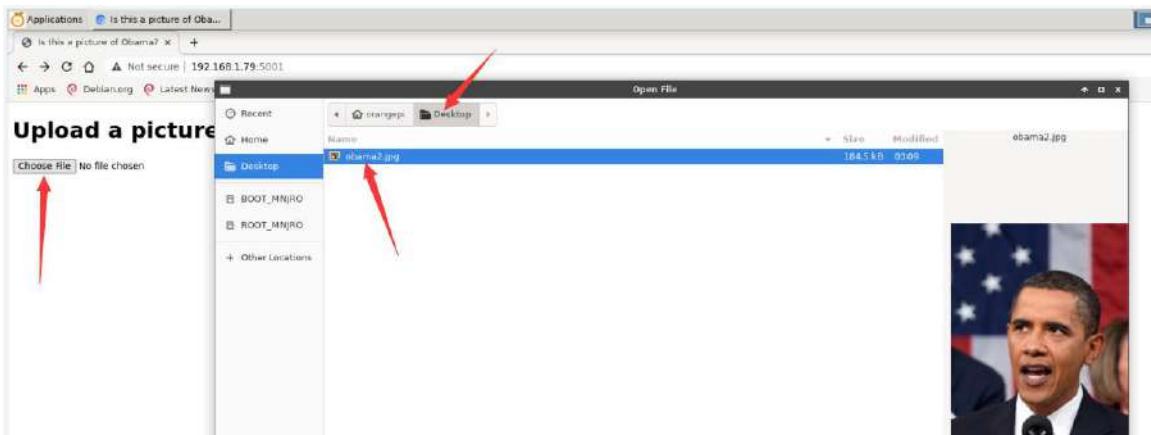
- First open the browser, then enter the IP address of the development board in the address bar of the browser: **5001**, and then you can see the following page



- Then copy obama2.jpg to the desktop

```
orangeipi@orangeipi:~/face_recognition/examples$ cp obama2.jpg \  
/home/orangeipi/Desktop/
```

- Then select the image you just copied in the browser



- Then click **Upload** to upload the image you just selected for face recognition



- e) After a period of time, the test results will be displayed



### 11) **face\_detection** command test example

- The a.face\_detection command-line tool can locate the face position (output pixel coordinates) in a single image or a folder of images. Use **face\_detection --help** to view the help information of the face\_detection command

```
orangeipi@orangeipi:~$ face_detection --help
Usage: face_detection [OPTIONS] IMAGE_TO_CHECK
```

Options:

- cpus INTEGER number of CPU cores to use in parallel. -1 means "use all in system"
- model TEXT Which face detection model to use. Options are "hog" or "cnn".
- help Show this message and exit.

- An example of detecting a single image is as follows:

```
orangeipi@orangeipi:~$ cd face_recognition/examples
orangeipi@orangeipi:~/face_recognition/examples$ face_detection obama2.jpg
obama2.jpg,302,474,611,164
```

- An example of using multiple cores to detect multiple images in parallel is as follows:
  - First go to the **face\_recognition/examples** folder



- b) Then create a new test folder
- c) Then copy the jpg images to the test folder
- d) Then use all cpus to run **face\_detection** in parallel to check the pictures in the test folder, where **--cpus -1** means use all cpus

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ mkdir test
orangepi@orangepi:~/face_recognition/examples$ cp *.jpg test
orangepi@orangepi:~/face_recognition/examples$ face_detection --cpus -1 test
test/obama-240p.jpg,29,261,101,189
test/obama_small.jpg,65,215,169,112
test/obama2.jpg,302,474,611,164
test/two_people.jpg,62,394,211,244
test/two_people.jpg,95,941,244,792
test/obama.jpg,136,624,394,366
test/obama-480p.jpg,65,507,189,383
test/obama-720p.jpg,94,751,273,572
test/obama-1080p.jpg,136,1140,394,882
test/biden.jpg,233,749,542,439
```

## 12) **face\_recognition** command test example

- a. **face\_recognition** command line tool can recognize whose face is in a single image or a folder of images. Use **face\_recognition --help** to view help information for the **face\_recognition** command

```
orangepi@orangepi:~$ face_recognition --help
Usage: face_recognition [OPTIONS] KNOWN_PEOPLE_FOLDER
IMAGE_TO_CHECK
```

Options:

|                         |                                                                                                                |
|-------------------------|----------------------------------------------------------------------------------------------------------------|
| --cpus INTEGER          | number of CPU cores to use in parallel (can speed up processing lots of images). -1 means "use all in system"  |
| --tolerance FLOAT       | Tolerance for face comparisons. Default is 0.6.<br>Lower this if you get multiple matches for the same person. |
| --show-distance BOOLEAN | Output face distance. Useful for tweaking tolerance setting.                                                   |



|        |                             |
|--------|-----------------------------|
| --help | Show this message and exit. |
|--------|-----------------------------|

- b. First create a new face image folder **known\_people** with a known name, then copy two images to **known\_people**, and then copy **obama2.jpg** as **unknown.jpg**, which is the image we want to identify

```
orangeipi@orangeipi:~/face_recognition/examples$ cd face_recognition/examples  
orangeipi@orangeipi:~/face_recognition/examples$ mkdir known_people  
orangeipi@orangeipi:~/face_recognition/examples$ cp biden.jpg obama.jpg \  
known_people  
orangeipi@orangeipi:~/face_recognition/examples$ cp obama2.jpg unkown.jpg
```

- c. Then you can use the following command to identify the name of the person in the **unknown.jpg** picture, you can see that the unknown.jpg picture is recognized as obama

```
orangeipi@orangeipi:~/face_recognition/examples$ face_recognition known_people \  
unkown.jpg  
unkown.jpg,obama
```

- d. If we identify an irrelevant picture, unknown\_person will be displayed

```
root@orangeipi:~/face_recognition/examples$ face_recognition known_people \  
alex-lacamoire.png  
alex-lacamoire.png,unknown_person
```

- e. We can also create a new test folder, and then put multiple pictures in it, and then we can use all the CPUs to recognize all the pictures in parallel

```
orangeipi@orangeipi:~/face_recognition/examples$ mkdir test  
orangeipi@orangeipi:~/face_recognition/examples$ cp *.jpg *.png test  
orangeipi@orangeipi:~/face_recognition/examples$ face_recognition --cpus -1 \  
known_people test  
test/obama-240p.jpg,obama  
test/alex-lacamoire.png,unknown_person  
test/obama_small.jpg,obama  
test/unkown.jpg,obama  
test/obama2.jpg,obama  
test/lin-manuel-miranda.png,unknown_person  
test/two_people.jpg,biden  
test/two_people.jpg,obama  
test/obama-720p.jpg,obama  
test/obama.jpg,obama
```



```
test/obama-480p.jpg,obama  
test/biden.jpg,biden  
test/obama-1080p.jpg,obama
```

### 3. 36. Installation method of Tensorflow

Note that before installing Tensorflow, please make sure that the Linux system used is **Debian Buster**. The installation method of Tensorflow demonstrated in this section cannot be guaranteed to work normally on other versions of Linux systems.

#### 3. 36. 1. The method of using script to automatically install Tensorflow

- 1) First download and install the tensorflow installation script provided by Orange Pi

```
orangepi@orangepi:~$ wget \  
https://gitee.com/leeboby/tensorflow/raw/master/install\_tensorflow.sh
```

- 2) Then run the **install\_tensorflow.sh** script to start installing tensorflow

```
orangepi@orangepi:~$ sudo bash install_tensorflow.sh
```

- 3) After the tensorflow installation is completed, the version number of tensorflow will be automatically tested and printed. If you can see the following output at the end, it means that the tensorflow installation is successful

```
##### Start Test Tensorflow #####  
Tensorflow version is : 2.4.0  
##### End Test Tensorflow #####
```

#### 3. 36. 2. Steps to manually install Tensorflow

- 1) First use the following command to set the source of pip to Tsinghua source to speed up the download speed of the Python package

```
orangepi@orangepi:~$ mkdir -p ~/.pip  
orangepi@orangepi:~$ cat <<EOF > ~/.pip/pip.conf  
[global]  
timeout = 6000  
index-url = https://pypi.tuna.tsinghua.edu.cn/simple  
trusted-host = pypi.tuna.tsinghua.edu.cn  
EOF
```



2) Then install the dependency package

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt install -y python3-pip gfortran \\\nlibopenblas-dev liblapack-dev libatlas-base-dev libblas-dev \\\nlibhdf5-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpeg62-turbo-dev \\\npython3-dev pkg-config python3-setuptools python3-wheel
```

3) Then download the whl package related to tensorflow

```
orangepi@orangepi:~$ git clone --depth=1 https://gitee.com/leeboby/tensorflow.git
```

4) Then enter the **tensorflow** directory to install the whl package that tensorflow depends on

```
orangepi@orangepi:~$ cd tensorflow  
orangepi@orangepi:~/tensorflow$ pip3 install \\\ntensorflow/grpcio-1.32.0-cp37-cp37m-linux_aarch64.whl  
orangepi@orangepi:~/tensorflow$ pip3 install \\\ntensorflow(numpy-1.19.5-cp37-cp37m-linux_aarch64.whl  
orangepi@orangepi:~/tensorflow$ pip3 install \\\ntensorflow(h5py-2.10.0-cp37-cp37m-linux_aarch64.whl
```

5) Then you can use the following command to install tensorflow

```
orangepi@orangepi:~/tensorflow$ pip3 install \\\ntensorflow-2.4.0-cp37-none-linux_aarch64.whl
```

6) After installing tensorflow, you can use the following command to print the version number of tensorflow. If the version number **2.4.0** of tensorflow can be printed out normally, it means that the installation of tensorflow is successful

```
orangepi@orangepi:~/tensorflow$ python3 -c \\\n"import tensorflow; print(tensorflow.__version__)"  
2.4.0
```

7) References

```
https://github.com/lhelontra/tensorflow-on-arm  
https://tf.kmtea.eu/whl/stable.html  
https://www.tensorflow.org
```



<https://repo.rock-chips.com/pypi/simple>

### 3. 37. ROS installation method

#### 3. 37. 1. How to install ROS 1 Noetic

- 1) The current active versions of ROS 1 are as follows, and the recommended version is **Noetic Ninjemys**

| Distro                                      | Release date   | Poster | Tuturtle, turtle in tutorial | EOL date                  |
|---------------------------------------------|----------------|--------|------------------------------|---------------------------|
| ROS Noetic Ninjemys<br><b>(Recommended)</b> | May 23rd, 2020 |        |                              | May, 2025<br>(Focal EOL)  |
| ROS Melodic Morenia                         | May 23rd, 2018 |        |                              | May, 2023<br>(Bionic EOL) |

<http://docs.ros.org>

<https://wiki.ros.org/Distributions>

- 2) ROS 1 **Noetic Ninjemys** The official installation documentation link is as follows:

- a. Ubuntu

<http://wiki.ros.org/noetic/Installation/Ubuntu>

- b. Debian



<http://wiki.ros.org/noetic/Installation/Debian>

3) Ubuntu 20.04 is recommended for Ubuntu Linux in the official installation documentation of ROS Noetic Ninjemys, so please make sure that the system used by the development board is **Ubuntu 20.04**. If you want to use the Debian system, please test it yourself, it will not be demonstrated here

<http://wiki.ros.org/noetic/Installation>

### Select Your Platform

Supported:



4) First add the software source of ros

```
orangeipi@orangeipi:~$ sudo sh -c 'echo \
"deb http://mirrors.ustc.edu.cn/ros/ubuntu ${lsb_release -sc} main" \
> /etc/apt/sources.list.d/ros-latest.list'
```

5) Then set Keys

```
orangeipi@orangeipi:~$ sudo apt-key adv \
--keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key \
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

```
Executing: /tmp/apt-key-gpghome.0zbZ9ucMdb/gpg.1.sh --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
gpg: key F42ED6FBAB17C654: public key "Open Robotics <info@osrfoundation.org>" imported
gpg: Total number processed: 1
gpg:          imported: 1
```

6) Then update the apt repository cache

```
orangeipi@orangeipi:~$ sudo apt update
```



7) Then install ros

```
orangeipi@orangeipi:~$ sudo apt install -y ros-noetic-desktop-full
```

8) Before running the commands in ros, you need to set the environment variables first

```
orangeipi@orangeipi:~$ source /opt/ros/noetic/setup.bash
```

9) If you don't want to manually set the environment variables before using the commands in ros, you can add the commands for setting the environment variables to `~/.bashrc`, so that every time you open a new terminal, the environment variables of ros will be automatically set

```
orangeipi@orangeipi:~$ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
orangeipi@orangeipi:~$ source ~/.bashrc
```

10) Now everything needed to run the core ros package has been installed. If you want to create and manage your own ros workspace, you also need to install some other tools. For example, rosinstall is a common command-line tool that lets you download the source tree of some ros packages. Run the following command to install this tool and other dependencies needed to build the ros package

```
orangeipi@orangeipi:~$ sudo apt install -y python3-rosdep python3-rosinstall \
python3-rosinstall-generator python3-wstool build-essential
```

11) Before using the ROS tool, you first need to initialize rosdep, and then you can quickly install some system dependencies and some core components in ROS when compiling the source code

**Note that running the following command needs to ensure that the development board can access github normally, otherwise an error will be reported due to network problems.**

```
orangeipi@orangeipi:~$ source /opt/ros/noetic/setup.bash
```

```
orangeipi@orangeipi:~$ sudo rosdep init
```

Wrote /etc/ros/rosdep/sources.list.d/20-default.list

Recommended: please run

```
        rosdep update
```

```
orangeipi@orangeipi:~$ rosdep update
```

```
reading in sources list data from /etc/ros/rosdep/sources.list.d
```



```
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml
Query rosdistro index
https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml
Skip end-of-life distro "ardent"
Skip end-of-life distro "bouncy"
Skip end-of-life distro "crystal"
Skip end-of-life distro "dashing"
Skip end-of-life distro "eloquent"
Add distro "foxy"
Add distro "galactic"
Skip end-of-life distro "groovy"
Add distro "humble"
Skip end-of-life distro "hydro"
Skip end-of-life distro "indigo"
Skip end-of-life distro "jade"
Skip end-of-life distro "kinetic"
Skip end-of-life distro "lunar"
Add distro "melodic"
Add distro "noetic"
Add distro "rolling"
updated cache in /home/orangepi/.ros/rosdep/sources.cache
```

## 12) How to verify if ROS is installed correctly

### a. First run **roscore**

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ roscore
... logging to
/home/orangepi/.ros/log/132132c4-c873-11ec-9b13-5099013e1a05/roslaunch-orangepi-1
8381.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.
```



```
started roslaunch server http://orangepi:44425/
ros_comm version 1.15.14
```

## SUMMARY

---

## PARAMETERS

- \* /rosdistro: noetic
- \* /rosversion: 1.15.14

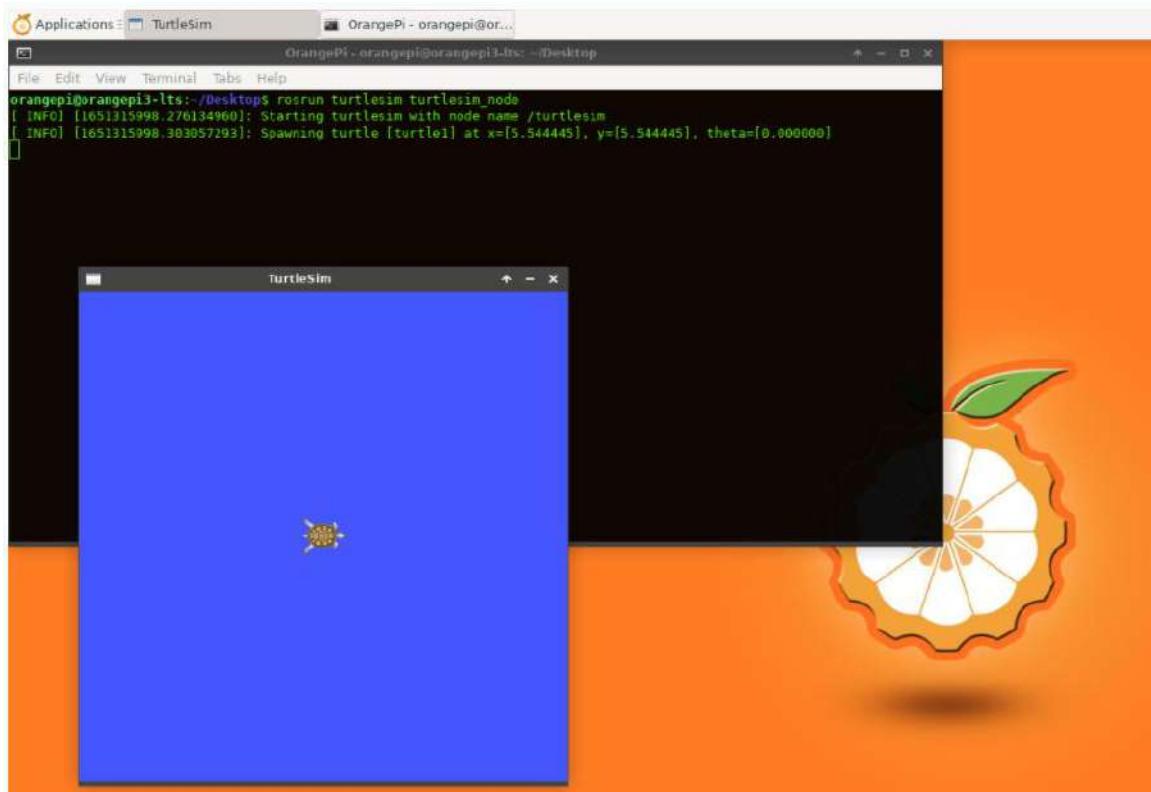
## NODES

```
auto-starting new master
process[master]: started with pid [18389]
ROS_MASTER_URI=http://orangepi:11311/
```

```
setting /run_id to 132132c4-c873-11ec-9b13-5099013e1a05
process[rosout-1]: started with pid [18399]
started core service [/rosout]
```

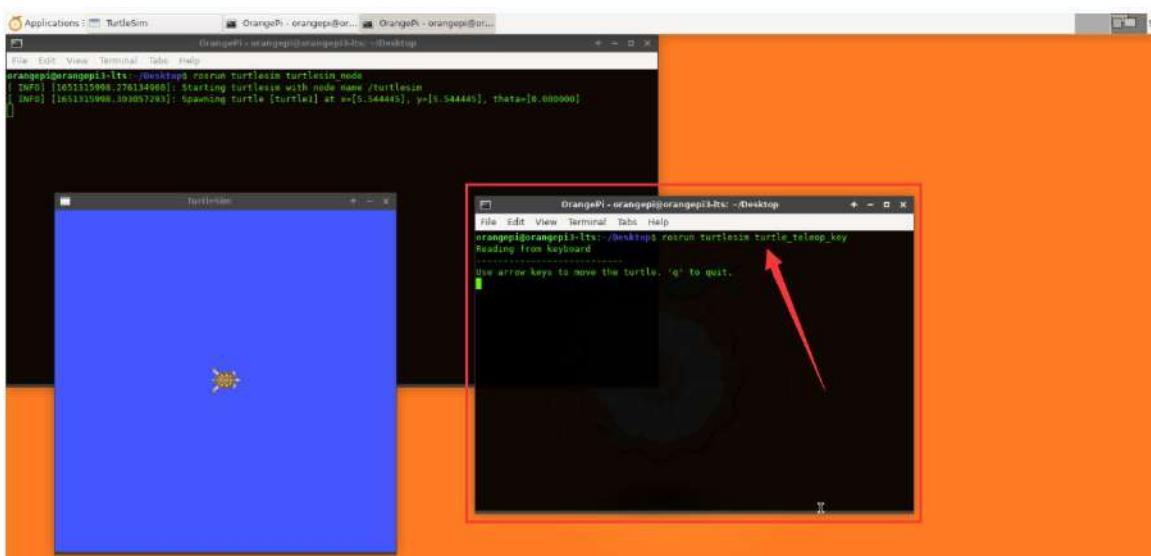
- b. Then start a small turtle routine to test whether ROS can be used normally
  - a) First open a command line terminal window in the desktop
  - b) Then enter the following command, a small turtle as shown in the figure below will pop up

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ rosrun turtlesim turtlesim_node
```

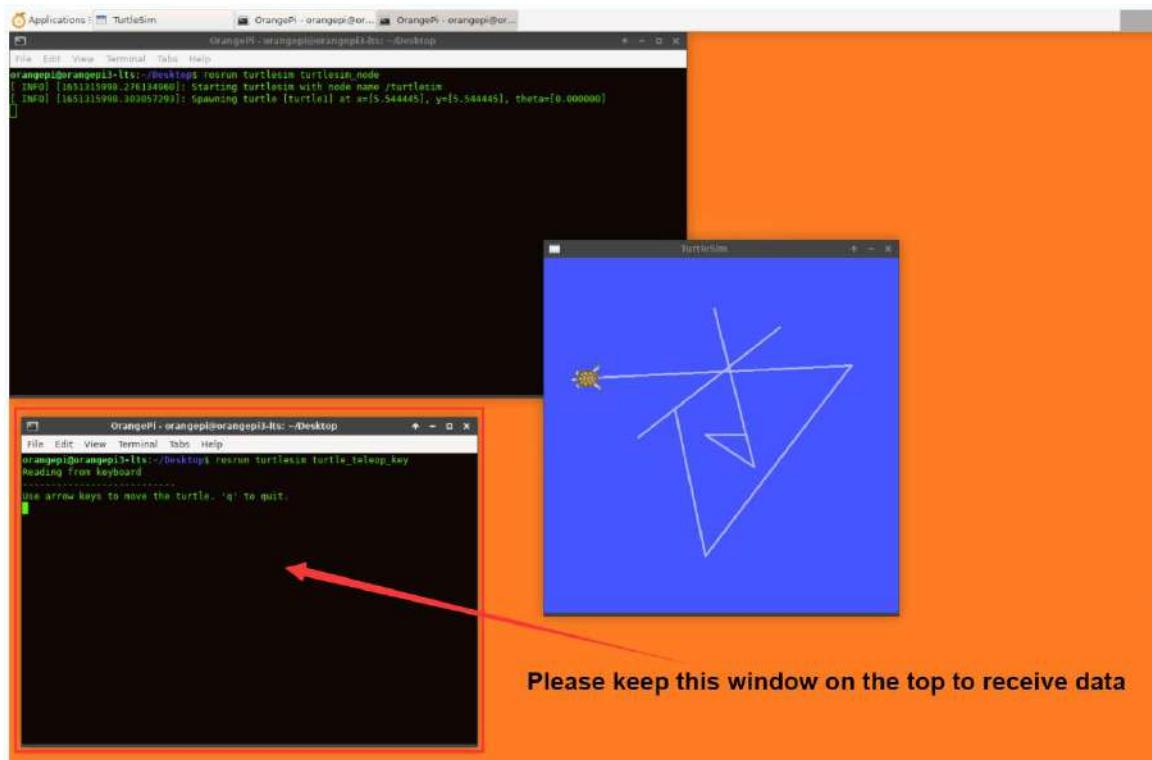


- c) Then open a terminal window and enter the following command to run the turtle control program

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash  
orangepi@orangepi:~$ rosrun turtlesim turtle_teleop_key
```



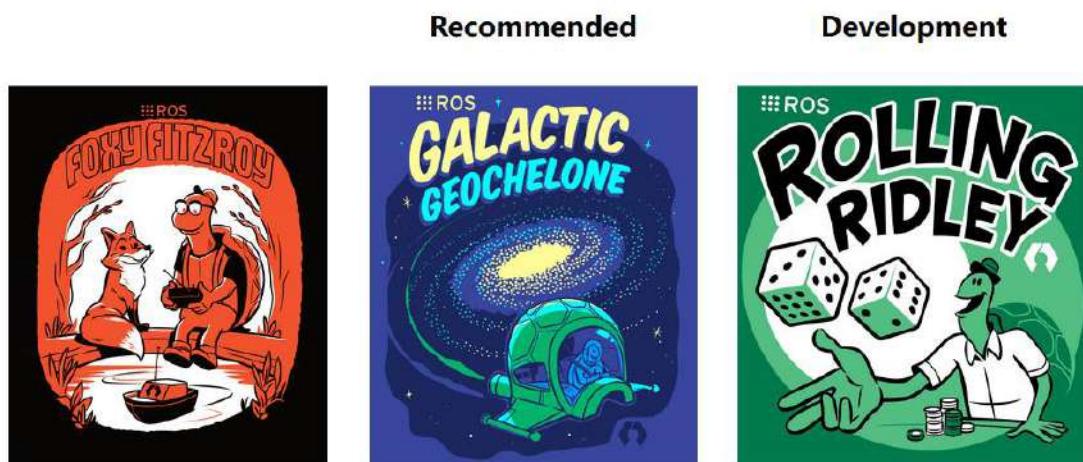
- d) Then please keep the terminal window of the turtle control program you just opened at the top. At this time, you can control the small turtle to move up, down, left and right by pressing the direction keys on the keyboard.



### 3.37.2. How to install ROS 2 Galactic

- 1) The current active versions of ROS 2 are as follows, the recommended version is **Galactic Geochelone**

#### Active ROS 2 distributions





| Distro              | Release date   | Logo                                                                               | EOL date      |
|---------------------|----------------|------------------------------------------------------------------------------------|---------------|
| Humble Hawksbill    | May 23rd, 2022 |                                                                                    | May 2027      |
| Galactic Geochelone | May 23rd, 2021 |  | November 2022 |
| Foxy Fitzroy        | June 5th, 2020 |  | May 2023      |

<http://docs.ros.org>

<http://docs.ros.org/en/galactic/Releases.html>

2) ROS 2 **Galactic Geochelone** The official installation documentation link is as follows:

[docs.ros.org/en/galactic/Installation.html](http://docs.ros.org/en/galactic/Installation.html)

<http://docs.ros.org/en/galactic/Installation/Ubuntu-Install-Debians.html>

3) Ubuntu 20.04 is recommended for Ubuntu Linux in the official installation document of ROS 2 **Galactic Geochelone**, so please make sure that the system used by the development board is **Ubuntu 20.04**. There are several ways to install ROS 2. The following demonstrates how to install ROS 2 **Galactic Geochelone** through **Debian packages**.

4) First add the key

```
orangepi@orangepi:~$ sudo apt-key adv --keyserver \
'hkp://keyserver.ubuntu.com:80' \
--recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

[sudo] password for orangepi:

```
Executing: /tmp/apt-key-gpghome.gNBSKx6Ums/gpg.1.sh --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```



```
gpg: key F42ED6FBAB17C654: public key "Open Robotics <info@osrfoundation.org>"  
imported  
gpg: Total number processed: 1  
gpg:          imported: 1
```

- 5) Then add the repository source of ROS 2 to the Ubuntu system

```
orangeipi@orangeipi:~$ echo "deb [arch=$(dpkg --print-architecture)] \\\nhttp://mirrors.ustc.edu.cn/ros2/ubuntu $(source /etc/os-release && echo \\$UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list \\>/dev/null
```

- 6) Then update the apt repository cache

```
orangeipi@orangeipi:~$ sudo apt update
```

- 7) Then you can install ROS 2 related packages. The following commands will install ROS, RViz, demos, tutorials

```
orangeipi@orangeipi:~$ sudo apt install -y ros-galactic-desktop
```

- 8) Before running the **ros2** command, you need to set the environment variable first

```
orangeipi@orangeipi:~$ source /opt/ros/galactic/setup.bash  
orangeipi@orangeipi:~$ ros2 -h  
usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...
```

ros2 is an extensible command-line tool for ROS 2.

optional arguments:

-h, --help show this help message and exit

Commands:

|           |                                            |
|-----------|--------------------------------------------|
| action    | Various action related sub-commands        |
| bag       | Various rosbag related sub-commands        |
| component | Various component related sub-commands     |
| daemon    | Various daemon related sub-commands        |
| doctor    | Check ROS setup and other potential issues |
| interface | Show information about ROS interfaces      |



|           |                                        |
|-----------|----------------------------------------|
| launch    | Run a launch file                      |
| lifecycle | Various lifecycle related sub-commands |
| multicast | Various multicast related sub-commands |
| node      | Various node related sub-commands      |
| param     | Various param related sub-commands     |
| pkg       | Various package related sub-commands   |
| run       | Run a package specific executable      |
| security  | Various security related sub-commands  |
| service   | Various service related sub-commands   |
| topic     | Various topic related sub-commands     |
| wtf       | Use `wtf` as alias to `doctor`         |

Call `ros2 <command> -h` for more detailed usage.

9) You can use the following method to test whether ROS 2 is successfully installed

- Open a terminal first, then use the following two commands to run a **C++ talker** that keeps sending Hello World

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
orangepi@orangepi:~$ ros2 run demo_nodes_cpp talker
[INFO] [1649599956.390893735] [talker]: Publishing: 'Hello World: 1'
[INFO] [1649599957.390812753] [talker]: Publishing: 'Hello World: 2'
[INFO] [1649599958.390890143] [talker]: Publishing: 'Hello World: 3'
.....
```

- Then open a terminal and use the following two commands to run a **Python listener**. If you can receive the Hello World sent above, it means that both the C++ and Python APIs of RSO 2 can work normally

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
orangepi@orangepi:~$ ros2 run demo_nodes_py listener
[INFO] [1649600109.504678962] [listener]: I heard: [Hello World: 154]
[INFO] [1649600110.393777793] [listener]: I heard: [Hello World: 155]
[INFO] [1649600111.393769143] [listener]: I heard: [Hello World: 156]
.....
```

10) For the usage of ROS, please refer to the documentation of ROS 2

<http://docs.ros.org/en/galactic/Tutorials.html>



11) Run the following command to uninstall ROS 2

```
orangepi@orangepi:~$ sudo apt remove ~nros-galactic-* && sudo apt autoremove  
orangepi@orangepi:~$ sudo rm /etc/apt/sources.list.d/ros2.list  
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt autoremove  
orangepi@orangepi:~$ sudo apt upgrade
```

### 3. 38. Installation method of OpenMediaVault

**OpenMediaVault is a Debian-based NAS operating system.**

**It can be known from the table below:**

**Debian10 can only install OpenMediaVault 5.x version;**

**Debian11 can only install OpenMediaVault 6.x version.**

Table 1: openmediavault historical releases

| Version | Codename    | Base Distro | Status         | Date Released |
|---------|-------------|-------------|----------------|---------------|
| 0.2     | Ix          | Debian 6    | EOL            | Oct 2011      |
| 0.3     | Omnious     | Debian 6    | EOL            | Jul 2012      |
| 0.4     | Fedaykin    | Debian 6    | EOL            | Sep 2012      |
| 0.5     | Sardoukar   | Debian 6    | EOL            | Aug 2013      |
| 1.0     | Kralizec    | Debian 7    | EOL            | Sept 2014     |
| 2.0     | Stoneburner | Debian 7    | EOL            | Jun 2015      |
| 3.0     | Erasmus     | Debian 8    | EOL            | Jun 2016      |
| 4.0     | Arrakis     | Debian 9    | EOL            | Apr 2018      |
| 5.0     | Usul        | Debian 10   | Stable         | Mar 2020      |
| 6.0     | Shaitan     | Debian 11   | In Development | est. Q2/2022  |

**So before installation, please select the version of OpenMediaVault you want to install, and then make sure that the Debian system used by the development board is the matching system.**

**In addition, OpenMediaVault officially recommends using the server version of the system, so please do not use the desktop version of the system to install OpenMediaVault.**

**Can I install openmediavault on top a running Debian system?** Yes, but it is recommended that the current running OS not to have a desktop environment installed.

#### 3. 38. 1. Install OpenMediaVault 5.x on Debian 10

Note that **Debian10** can only install OpenMediaVault 5.x.

1) The official documentation of OpenMediaVault is as follows:



a. Documentation for version 5.x (**stable version of OpenMediaVault**)

<https://openmediavault.readthedocs.io/en/5.x/>

2) The official documentation for installing OpenMediaVault in Debian10 is as follows:

[https://openmediavault.readthedocs.io/en/5.x/installation/on\\_debian.html](https://openmediavault.readthedocs.io/en/5.x/installation/on_debian.html)

3) First install the keyring of OpenMediaVault, note that the following commands are executed under the root user

```
root@orangeipi:~# apt-get install -y gnupg
root@orangeipi:~# wget -O  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"  \
https://packages.openmediavault.org/public/archive.key
root@orangeipi:~# apt-key add  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"
```

4) Then add the package repository of OpenMediaVault, pay attention to switch to the root user and enter the following command, the black font part is a complete command, please copy it directly

```
root@orangeipi:~# cat <<EOF > /etc/apt/sources.list.d/openmediavault.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul main
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul main
## Uncomment the following line to add software from the proposed repository.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul-proposed main
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul-proposed main
## This software is not part of OpenMediaVault, but is offered by third-party
## developers as a service to OpenMediaVault users.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul partner
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul partner
EOF
```

The Tsinghua source is used above. For related instructions, please refer to the following link

<https://mirrors.tuna.tsinghua.edu.cn/help/openmediavault/>

5) Then use the following command to install OpenMediaVault

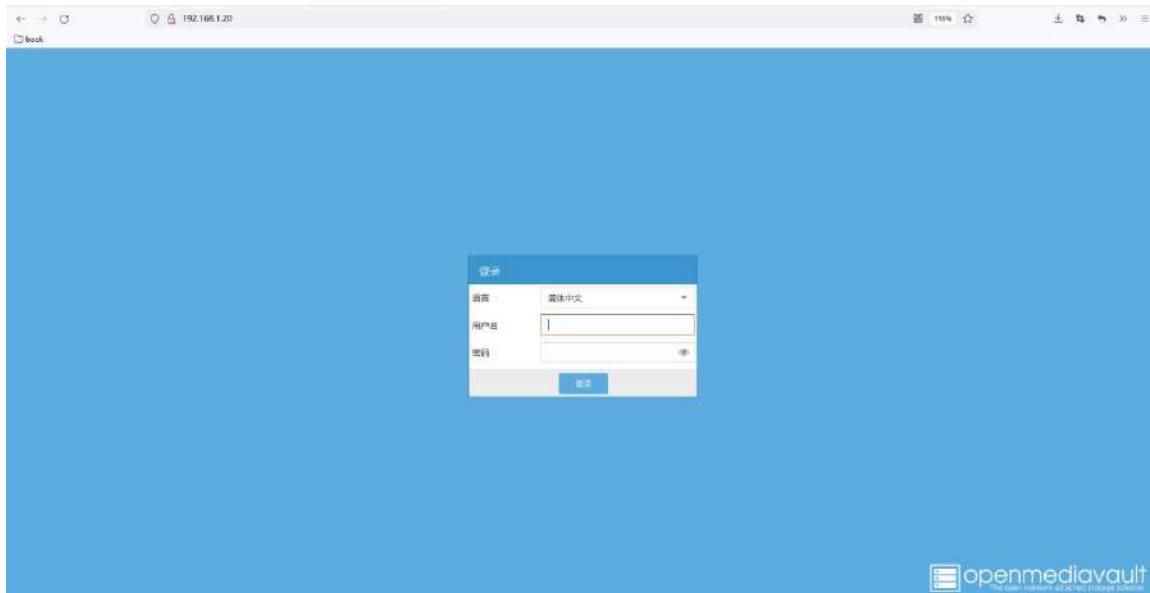


```
root@orangepi:~# export LANG=C.UTF-8
root@orangepi:~# export DEBIAN_FRONTEND=noninteractive
root@orangepi:~# export APT_LISTCHANGES_FRONTEND=none
root@orangepi:~# apt-get update
root@orangepi:~# apt-get --yes --auto-remove --show-upgraded \
--allow-downgrades --allow-change-held-packages \
--no-install-recommends \
--option DPkg::Options::="--force-confdef" \
--option DPkg::Options::="--force-confold" \
install openmediavault-keyring openmediavault
```

- 6) Then run the following command. After the operation is completed, enter the IP address of the development board in the browser to open the login page of OpenMediaVault

```
root@orangepi:~# omv-confdbadm populate
```

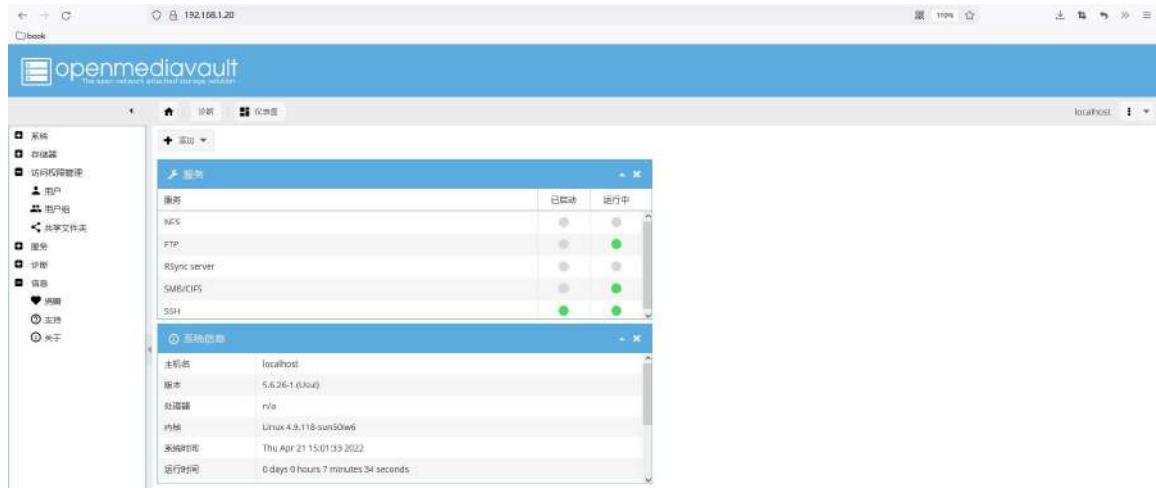
- 7) The login interface of OpenMediaVault is as follows



- 8) Then enter the default username **admin** and **password** openmediavault



9) The main interface displayed by OpenMediaVault login is as follows



### 3. 38. 2. Install OpenMediaVault 6.x on Debian11

Note that **Debian11** can only install OpenMediaVault 6.x.

1) The official documentation of OpenMediaVault is as follows:

<https://openmediavault.readthedocs.io/en/latest/>

2) The official documentation for installing OpenMediaVault in Debian looks like this:

[https://openmediavault.readthedocs.io/en/latest/installation/on\\_debian.html](https://openmediavault.readthedocs.io/en/latest/installation/on_debian.html)

3) First install the keyring of OpenMediaVault, note that the following commands are executed under the **root** user



```
root@orangepi:~# apt-get install -y gnupg
root@orangepi:~# wget -O  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"  \
https://packages.openmediavault.org/public/archive.key
root@orangepi:~# apt-key add  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"
```

- 4) Then add the package repository of OpenMediaVault, pay attention to switch to the **root** user and enter the following command, the black font part is a complete command, please copy it directly

```
root@orangepi:~# cat <<EOF >> /etc/apt/sources.list.d/openmediavault.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan main
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan main
## Uncomment the following line to add software from the proposed repository.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan-proposed main
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan-proposed main
## This software is not part of OpenMediaVault, but is offered by third-party
## developers as a service to OpenMediaVault users.
# https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan partner
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan partner
EOF
```

The Tsinghua source is used above. For related instructions, please refer to the following link

<https://mirrors.tuna.tsinghua.edu.cn/help/openmediavault/>

- 5) Then use the following command to install OpenMediaVault

```
root@orangepi:~# export LANG=C.UTF-8
root@orangepi:~# export DEBIAN_FRONTEND=noninteractive
root@orangepi:~# export APT_LISTCHANGES_FRONTEND=none
root@orangepi:~# apt-get update
root@orangepi:~# apt-get --yes --auto-remove --show-upgraded \
--allow-downgrades --allow-change-held-packages \
--no-install-recommends \
--option Dpkg::Options::="--force-confdef" \
```



```
--option DPkg::Options::="--force-confold" \
install openmediavault-keyring openmediavault
```

- 6) Then run the following command. After the operation is complete, enter the IP address of the development board in the browser to open the login page of OpenMediaVault

```
root@orangepi:~# omv-confdbadm populate
```

- 7) The login interface of OpenMediaVault is as follows



- 8) Then enter the default username **admin** and password **openmediavault**

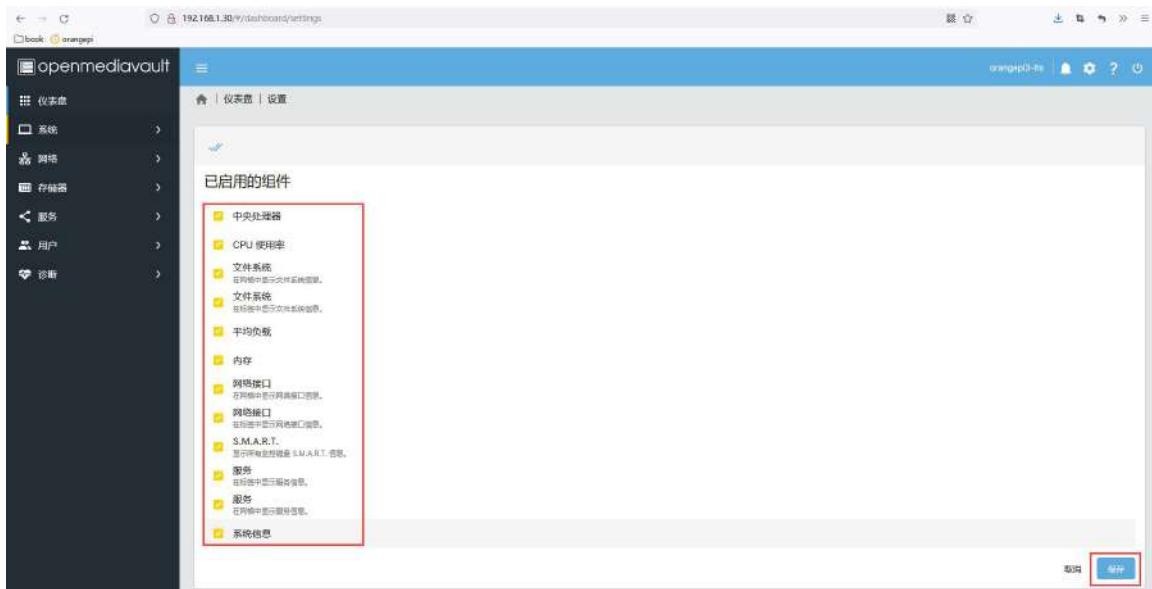
- 9) The main interface displayed by OpenMediaVault login is as follows



- 10) Then click on the **setting page**



11) Then select all these components, and then click the save button to **save**



12) Then you can see the system information displayed on the dashboard





- 13) If the interface displayed by the CPU is not normal, you can open the **diagnosis -> performance statistics -> CPU**, and then click the refresh button in the upper right corner to refresh the



- 14) How to install OMV plugin

- First open the following URL

<https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers/pool/main/o/openmediavault-omvextrasorg/>

- Then record the file name of the latest plugin package

| File Name ↓                                      | File Size ↓     | Date ↓                  |
|--------------------------------------------------|-----------------|-------------------------|
| Parent directory/                                | -               | -                       |
| openmediavault-omvextrasorg_4.1.16_all.deb       | 73.5 Kib        | 2021-03-21 09:00        |
| openmediavault-omvextrasorg_5.5.1_all.deb        | 63.7 Kib        | 2021-03-21 09:00        |
| openmediavault-omvextrasorg_5.6.6_all.deb        | 66.6 Kib        | 2022-02-09 20:15        |
| <b>openmediavault-omvextrasorg_6.0.8_all.deb</b> | <b>57.7 Kib</b> | <b>2022-02-23 02:14</b> |

- Then use the following command to download the plug-in package shown above in the Linux system of the development board (**if the following command cannot be downloaded, the name of the plug-in package may have changed, please replace it with the latest name**)

```
orangeipi@orangeipi:~$ wget \
https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers/pool/main/o/openmediavault-omvextrasorg/
```

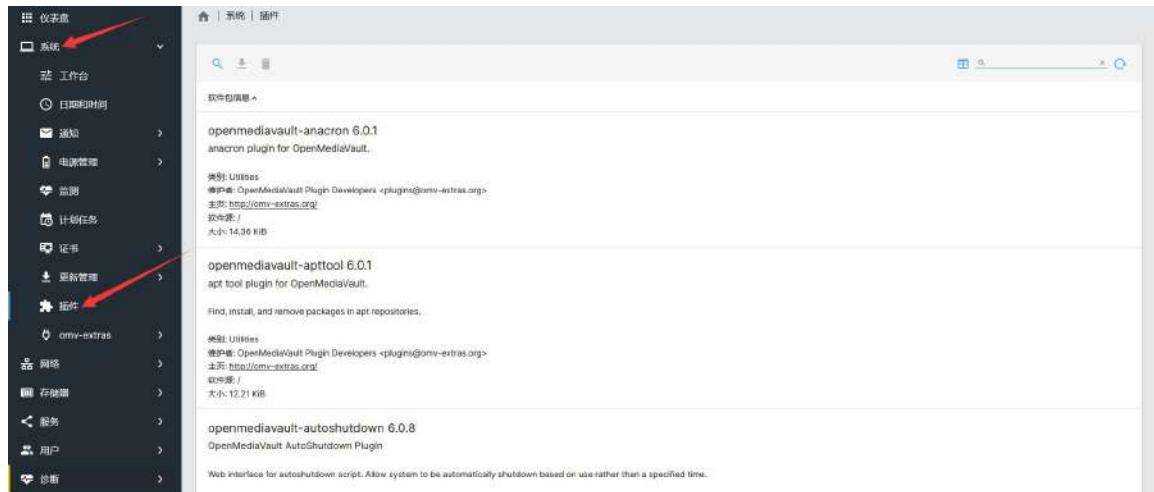


## opers/pool/main/o/openmediavault-omvextrasorg/openmediavault-omvextrasorg\_6.0.8\_all.deb

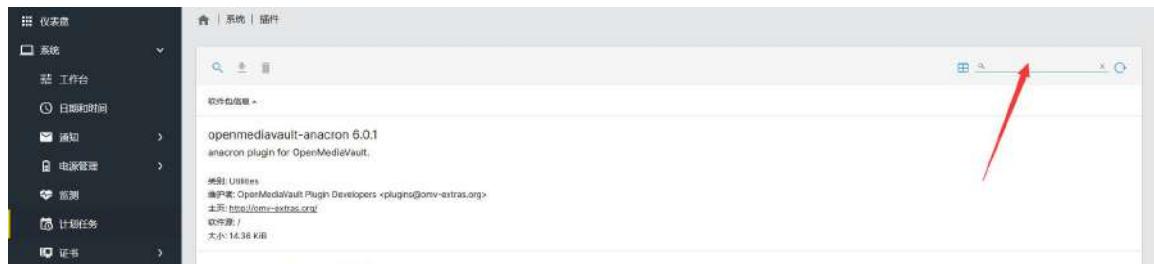
d. Then install the deb package just downloaded in the Linux system of the development board

```
orangeipi@orangeipi:~$ sudo dpkg -i openmediavault-omvextrasorg_6.0.8_all.deb
```

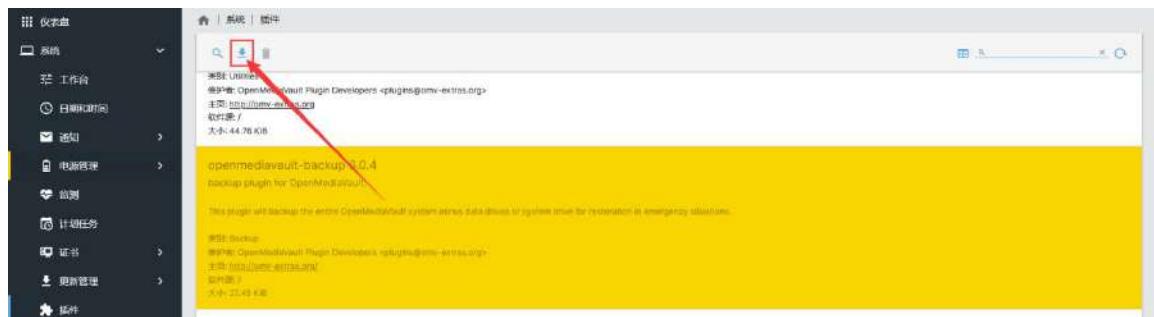
e. Then click **System -> Plugins** to see the following interface



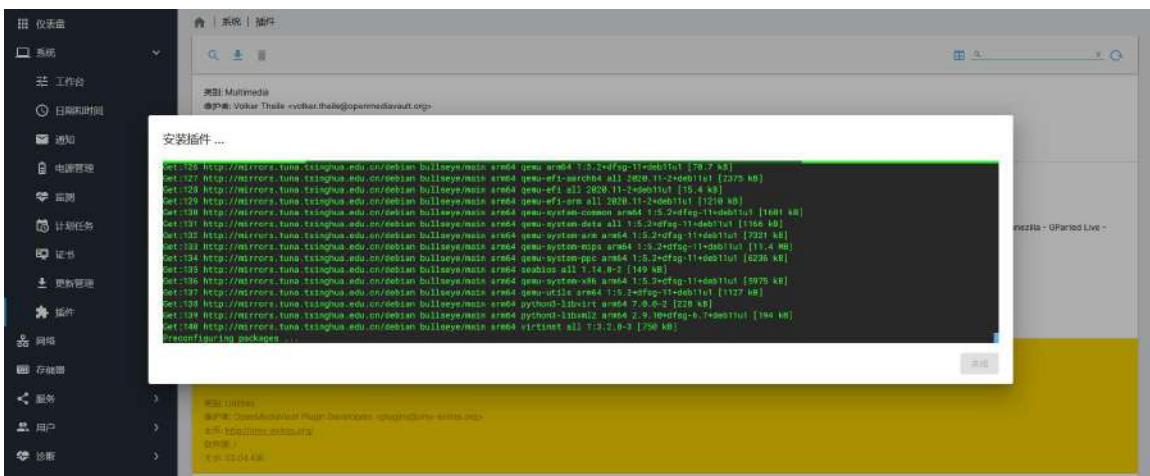
f. Then you can select or search for the plugin you want to install



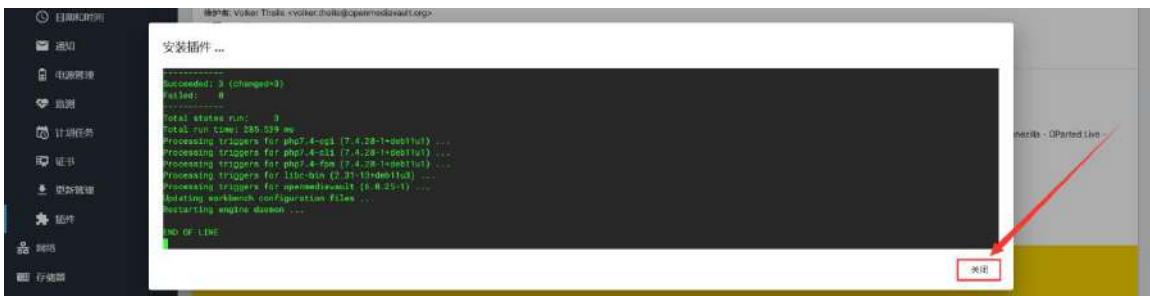
g. Then click the button shown in the figure below to start installing the selected plug-in



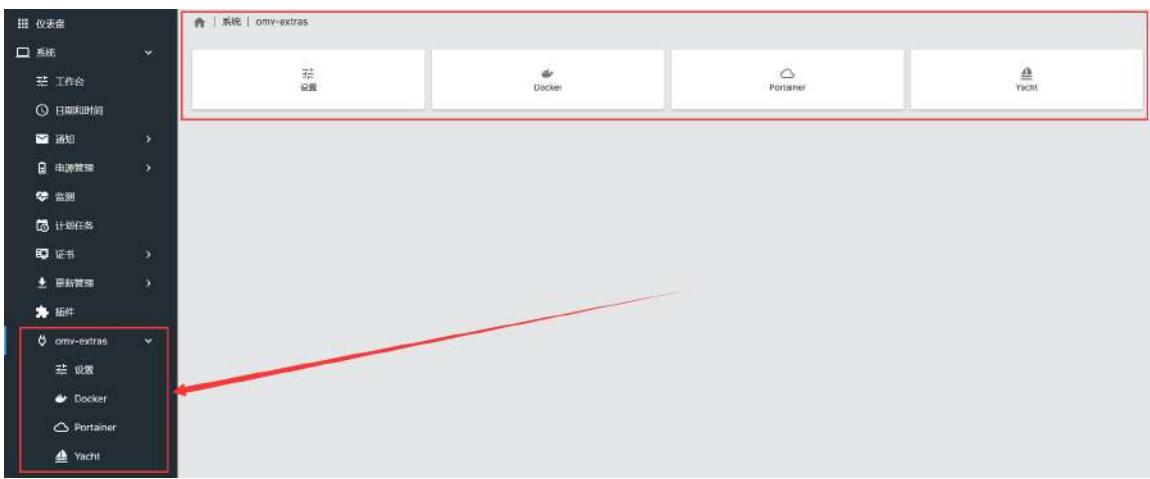
h. The plugin installation process is as follows



- i. After the installation is complete, click Close



- 15) After the plugin package is installed, there will be an **omv-extras** option on the right side of the web interface. **Docker**, **Portainer** and **Yacht** can be installed in omv-extras



- 16) Before installing docker, please replace the software source of docker. First, use the following command to open **omvextras.list**, and then replace the content in the blue font part. Finally, please remember to use **sudo apt-get update** to update the package index cache of the Linux system



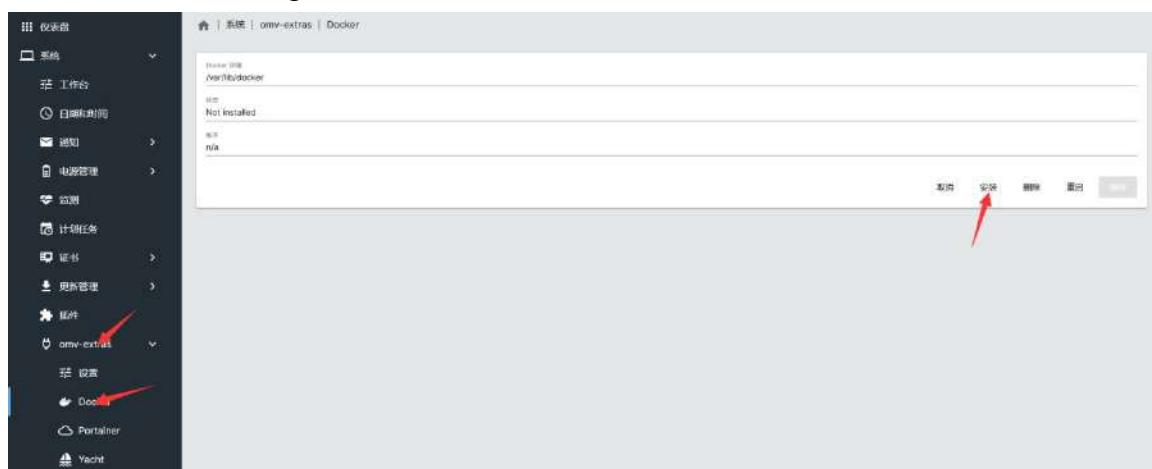
```
orangeipi@orangeipi:~$ sudo vim /etc/apt/sources.list.d/omvextras.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers shaitan main
deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/docker-ce/linux/debian bullseye stable
orangeipi@orangeipi:~$ sudo apt-get update
```

17) The way to install Docker in OMV is as follows

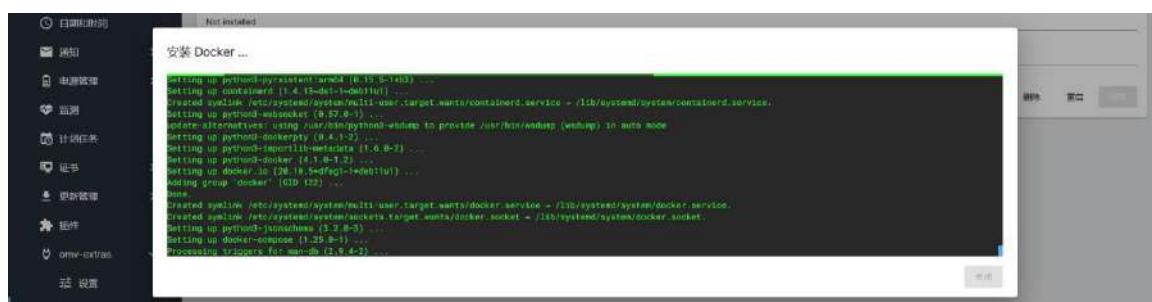
a. First install **apparmor**

```
orangeipi@orangeipi:~$ sudo apt install -y apparmor
```

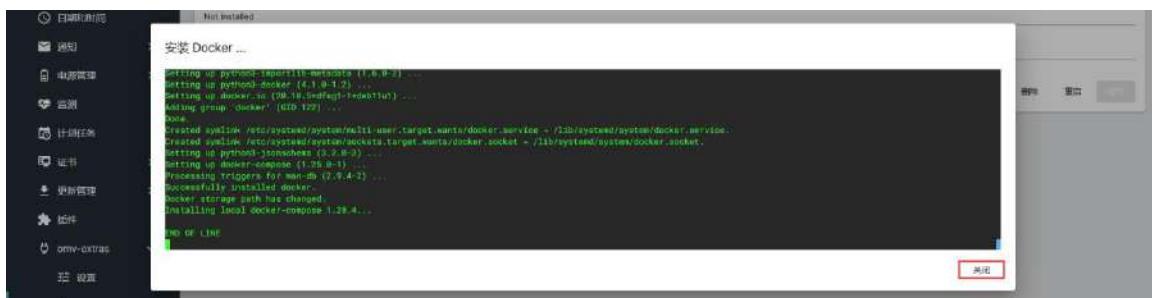
b. Then open the control interface of **Docker**, and then click the **install** button to start installing Docker



c. The display output of the Docker installation process is shown below



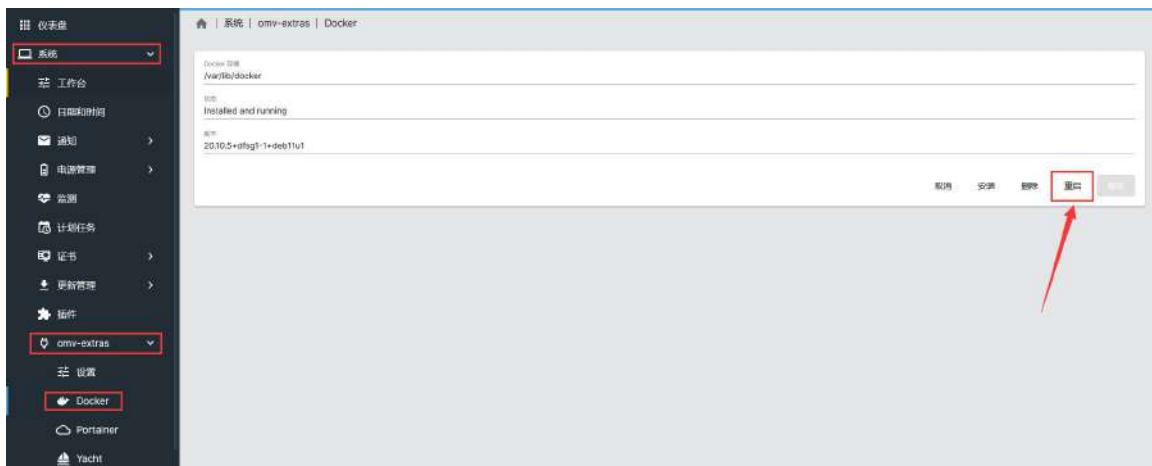
d. The display after the Docker installation is completed is as follows, and then click **Close**.



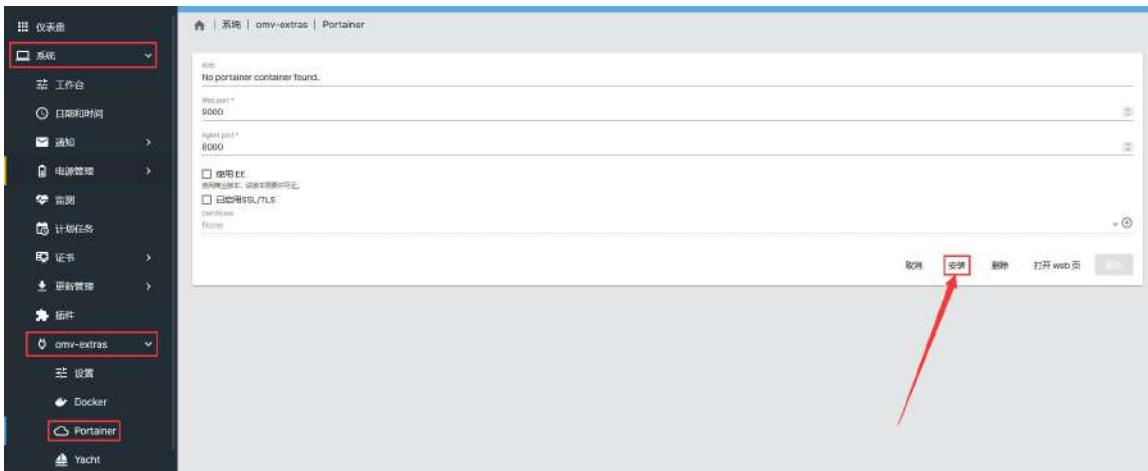
- e. Finally, you need to set the address of the Docker warehouse to the domestic address to speed up the download speed of the Docker container. The method looks like this:
- First open the **/etc/docker/daemon.json** file, and then add the configuration in the red font part below (note that **"data-root": "/var/lib/docker"** should be followed by a **,**)

```
orangeipi@orangeipi:~$ sudo vim /etc/docker/daemon.json
{
    "data-root": "/var/lib/docker",
    "registry-mirrors": ["https://docker.mirrors.ustc.edu.cn"]
}
```

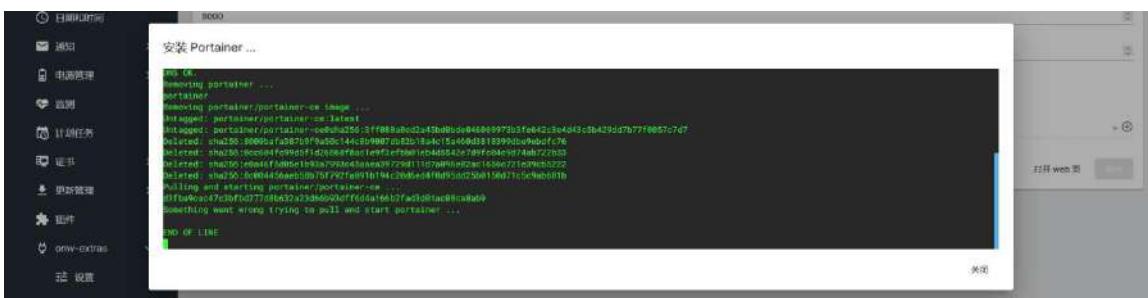
- Then click the **restart** button on the Docker control interface to restart the Docker service for the configuration to take effect (**if an error is reported, first check whether the above configuration is correct, then try a few more times, or restart the Debian11 system**)



- 18) **Portainer** is a docker visual management tool. The steps to install Portainer are:
- First open Portainer's control interface, and then click the **install** button to start installing Portainer



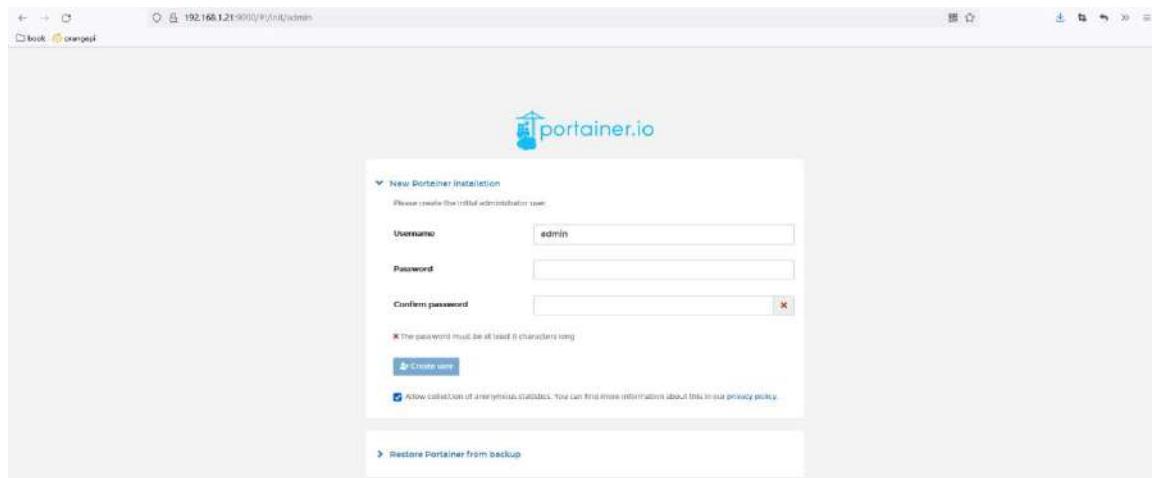
- b. After the installation of Portainer is completed, the display is as shown in the figure below, and then click the close button to **close** it.



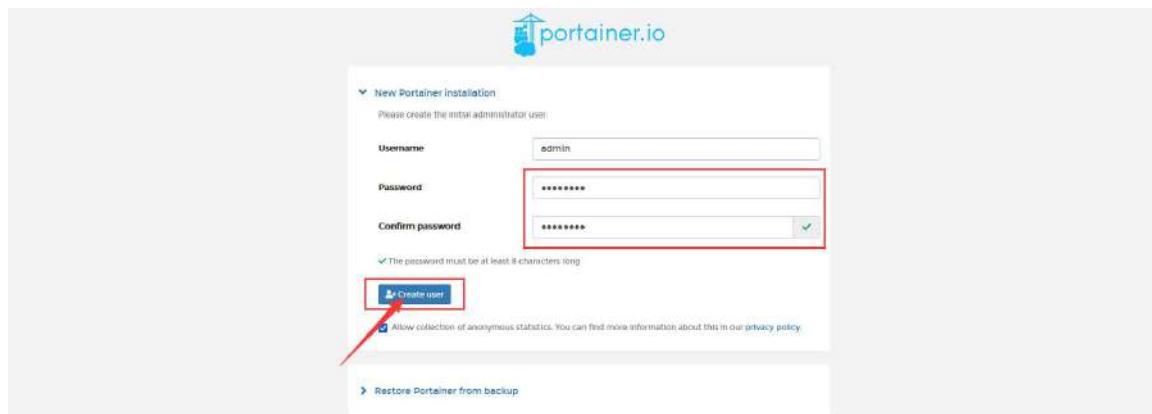
- c. Then open Portainer's control interface, and then click to **open the web** page to open Portainer's web control interface



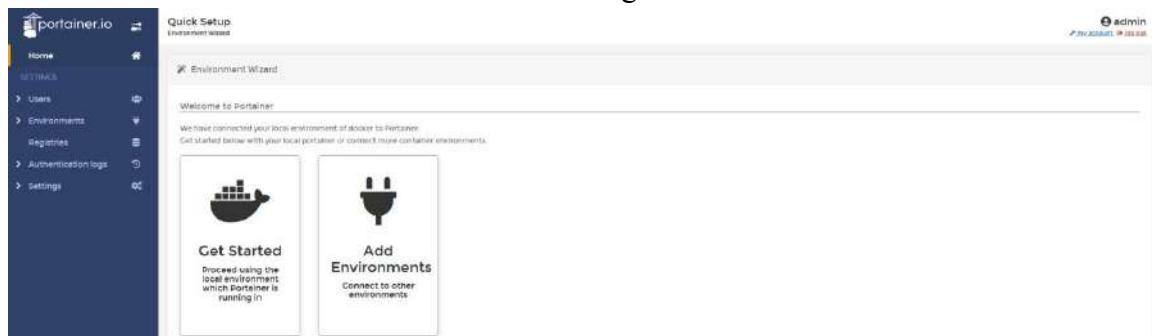
- d. The display of Portainer's web control interface after opening is as follows



- e. Then set the password of Portainer, and then click **Create user** to enter the web control interface of Portainer



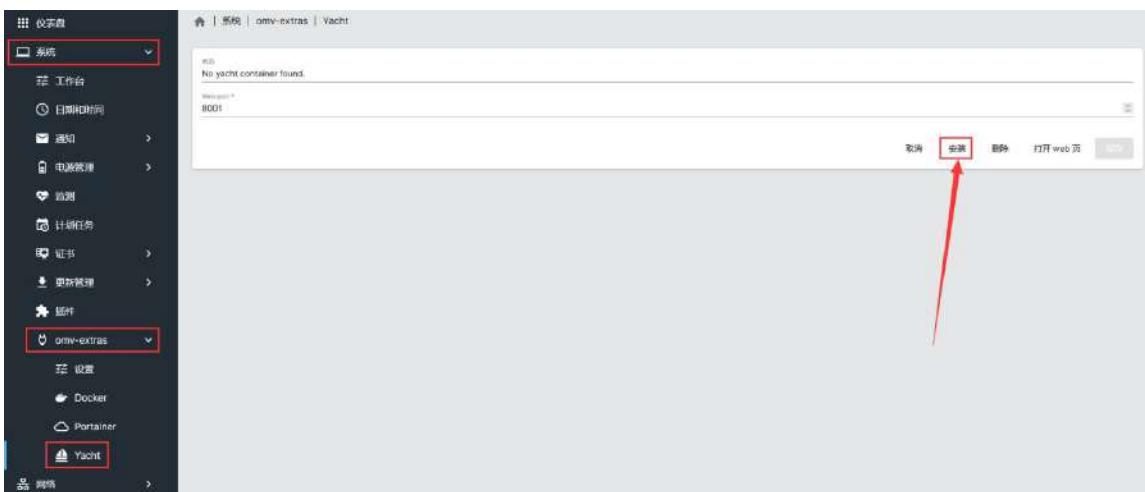
- f. The main interface of Portainer after login is shown as below



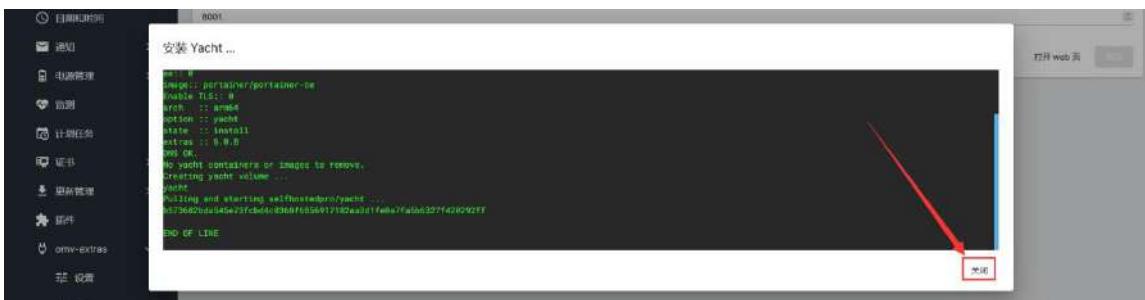
- g. Please study the usage of Portainer by yourself

19) **Yacht** and Portainer have similar functions. They are both docker visual management tools. The installation steps are as follows:

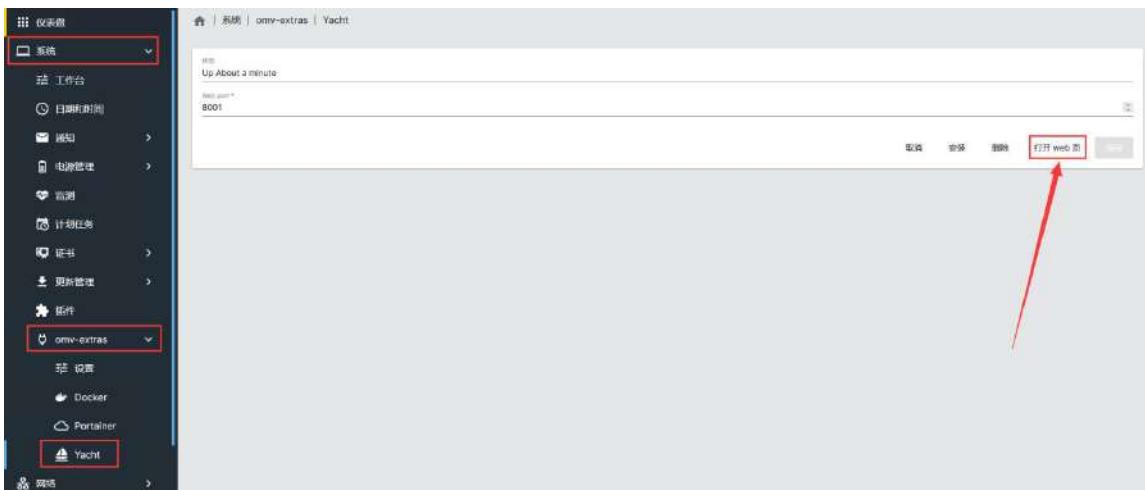
- a. First open the Yacht control interface in OMV, and then click the **install** button to start installing Yacht



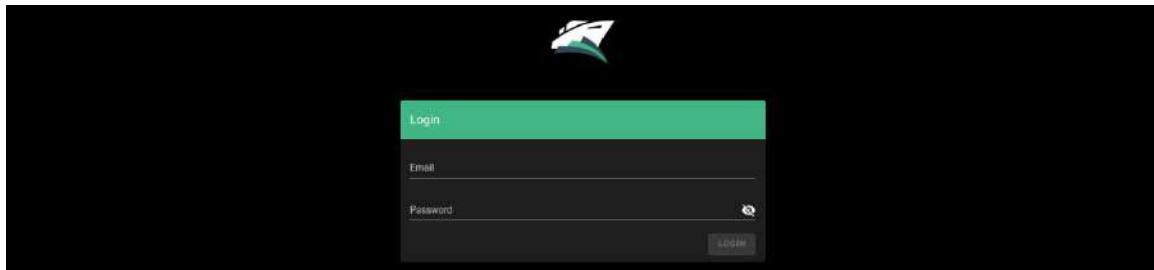
- b. The display after Yacht is installed is shown in the figure below, and then click the close button to **close** it



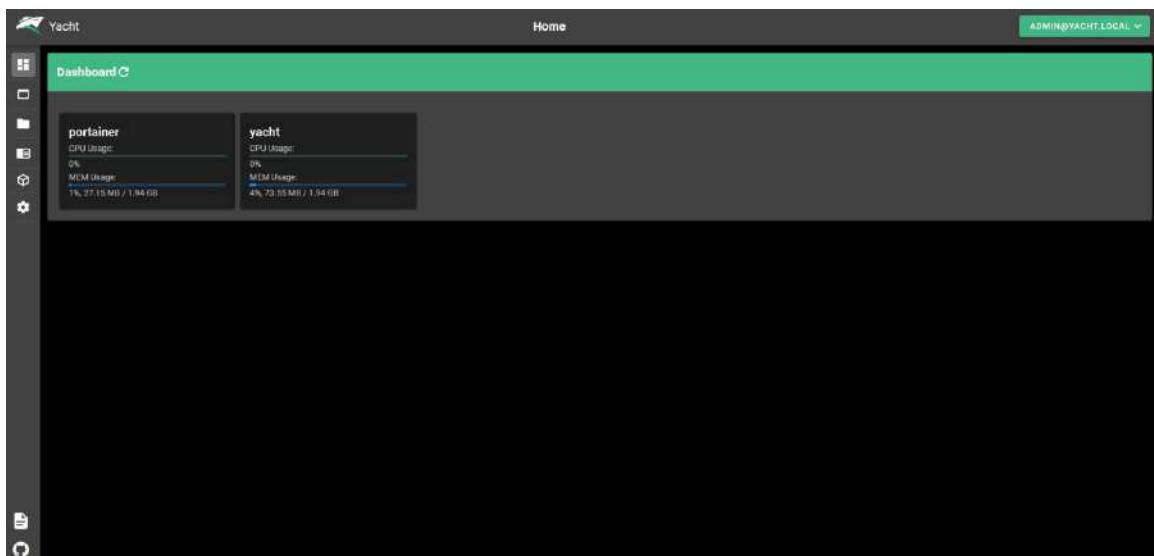
- c. Then open Yacht's control interface in OMV, and then click to **open the web** page to open Yacht's web control interface



- d. The display of Yacht's web control interface after opening is as follows



- e. Then enter Yacht's default account **admin@yacht.local** in the **Email** column, and enter the default password **pass** in the Password column, and then click **LOGIN** to enter Yacht's web control interface
- f. The main interface after Yacht login is shown as below



- g. Please do your own research on how to use Yacht

### 3. 39. Installation method of Pi-hole

Note that this section only provides the installation method of Pi-hole, please refer to the official documentation of Pi-hole for the usage method:

<https://pi-hole.net>

<https://docs.pi-hole.net>

Note that Pi-hole does not support **Ubuntu22.04** yet, so please do not use this system to install Pi-hole.



1) First download Pi-hole's installation script, then run

- a. Use Orange Pi to provide pi-hole script installation (**recommended**)

**Note that the pi-hole installation script provided by Orange Pi does not modify any pi-hole functions, but only solves the problem of download and installation failure caused by inability to access Github normally.**

- a) The first method: first download the repository of pi-hole, and then run the installation script

```
orangeypi@orangeypi:~$ git clone --depth 1 https://gitee.com/leeboby/pi-hole.git \
```

**Pi-hole**

```
orangeypi@orangeypi:~$ cd Pi-hole/automated\ install/
```

```
orangeypi@orangeypi:~$ sudo bash basic-install.sh
```

- b) The second method: first download the installation script, then run

```
orangeypi@orangeypi:~$ wget -O basic-install.sh \
```

```
https://gitee.com/leeboby/pi-hole/raw/master/automated%20install/basic-install.sh
```

```
orangeypi@orangeypi:~$ sudo bash basic-install.sh
```

- b. Install using pi-hole official script (**not recommended if it will not solve network problems**)

- a) The first method: use the following command to download the installation script of pi-hole and run it directly

```
orangeypi@orangeypi:~$ curl -sSL https://install.pi-hole.net | bash
```

- b) The second method: first download the repository of pi-hole, and then run the installation script

```
orangeypi@orangeypi:~$ git clone --depth 1 https://github.com/pi-hole/pi-hole.git \
```

**Pi-hole**

```
orangeypi@orangeypi:~$ cd Pi-hole/automated\ install/
```

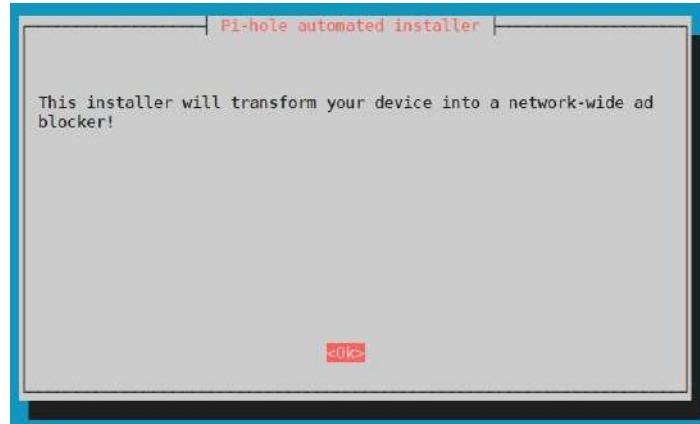
```
orangeypi@orangeypi:~$ sudo bash basic-install.sh
```

- c) The third method: first download the installation script, then run

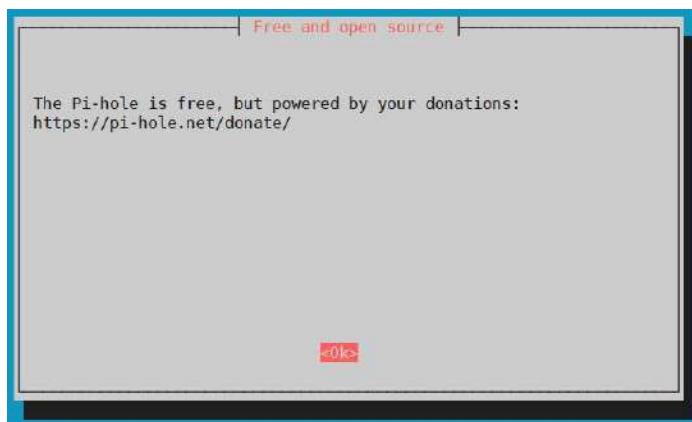
```
orangeypi@orangeypi:~$ wget -O basic-install.sh https://install.pi-hole.net
```

```
orangeypi@orangeypi:~$ sudo bash basic-install.sh
```

2) Then press enter

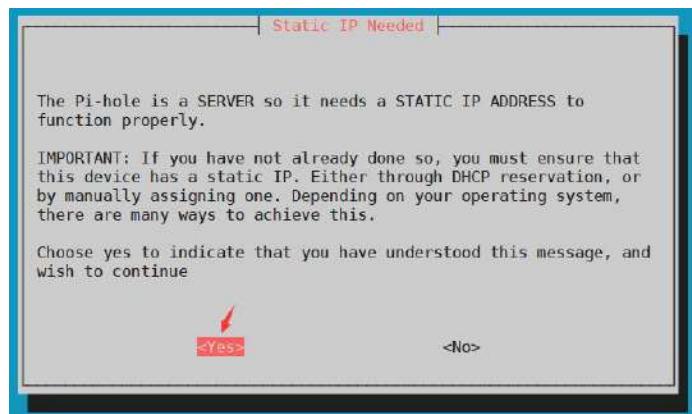


3) Continue to press Enter



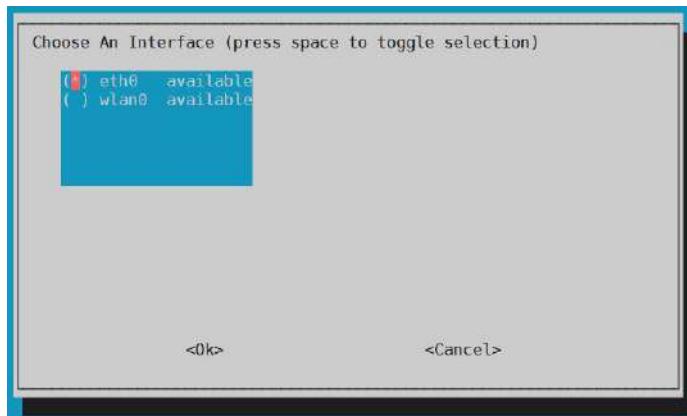
4) Then Pi-hole will prompt you to set a static IP address, please select **Yes** and press Enter

**Here, you can not set the static IP address first, and then set it after the installation is completed. For the setting method of the static IP address, please refer to the description in the section on the method of setting the static IP address.**

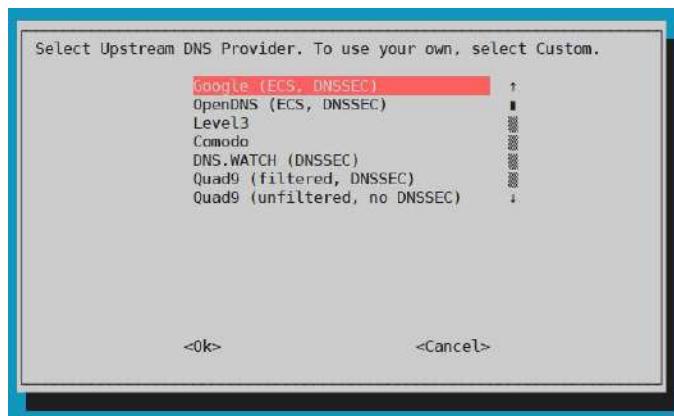




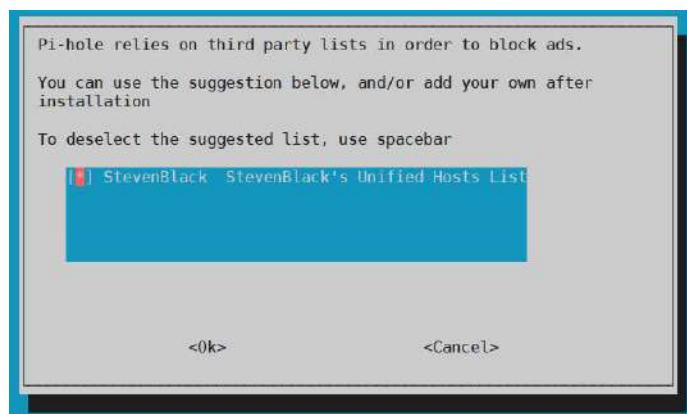
- 5) Then select the network interface. If you use a network cable, select eth0. If you want to use WIFI, select wlan0. After selecting, press Enter.



- 6) Then select the DNS provider, generally select Google.



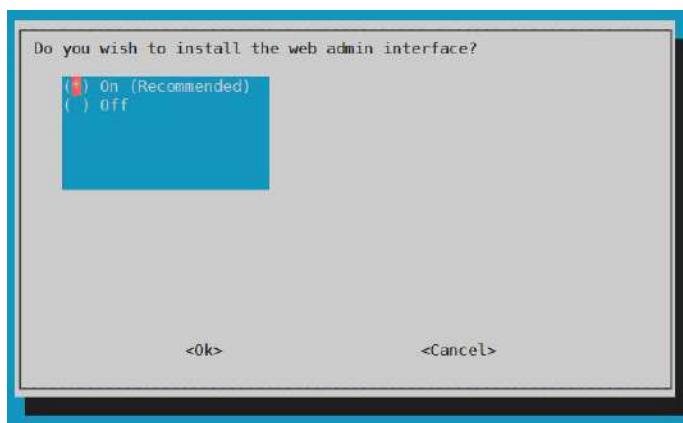
- 7) Then select the rule list and press Enter



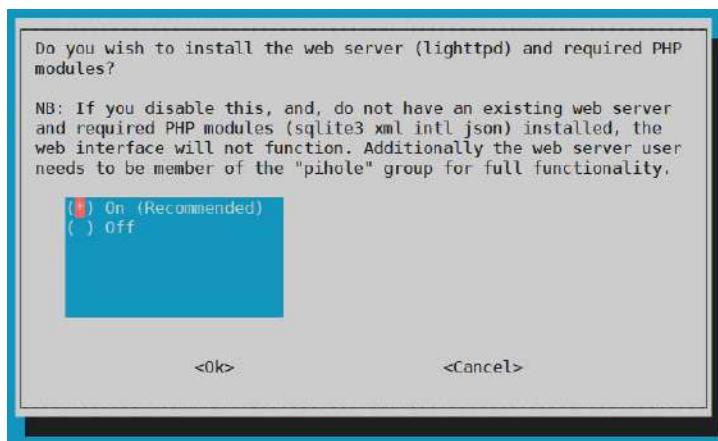
- 8) Then choose whether to install the web management interface, and press Enter to



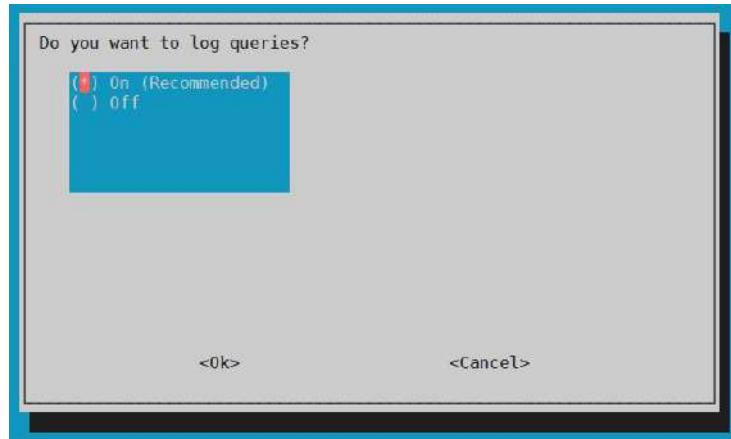
select the installation



9) Then press Enter to choose to install the web server and the required PHP modules



10) Then press Enter to select Open Log

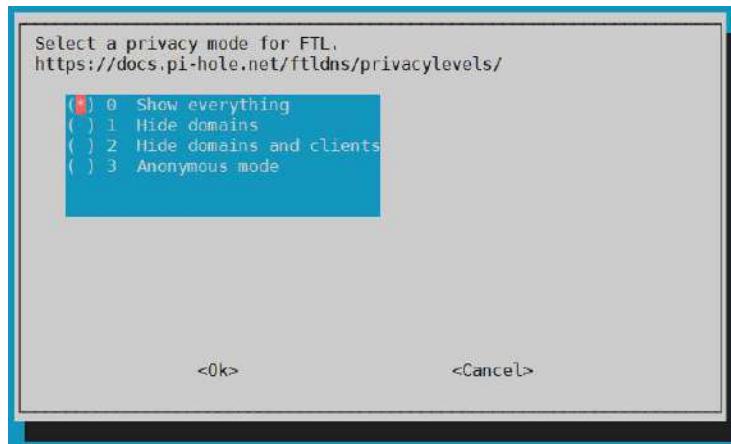


11) Then select the privacy mode, and select the default **0 Show everything** during



installation. Anyway, it can be modified after the installation is completed. Please check the following link for the difference between the different options of the privacy mode:

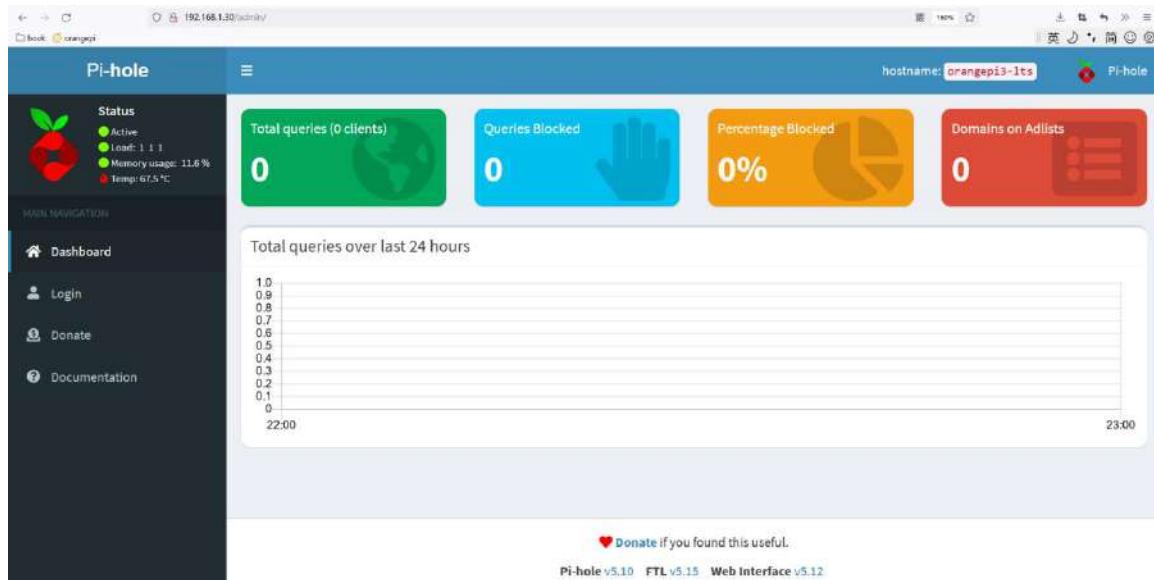
<https://docs.pi-hole.net/ftldns/privacylevels/>



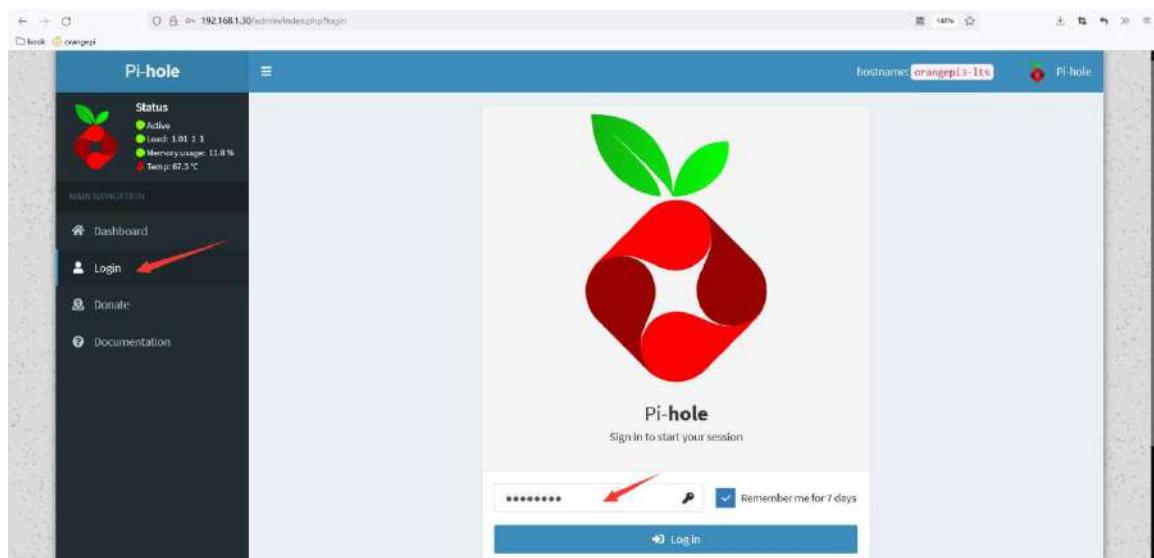
12) Then wait for the pi-hole installation to complete, and finally the following information will be prompted. The key point is to remember the web interface login address and login password



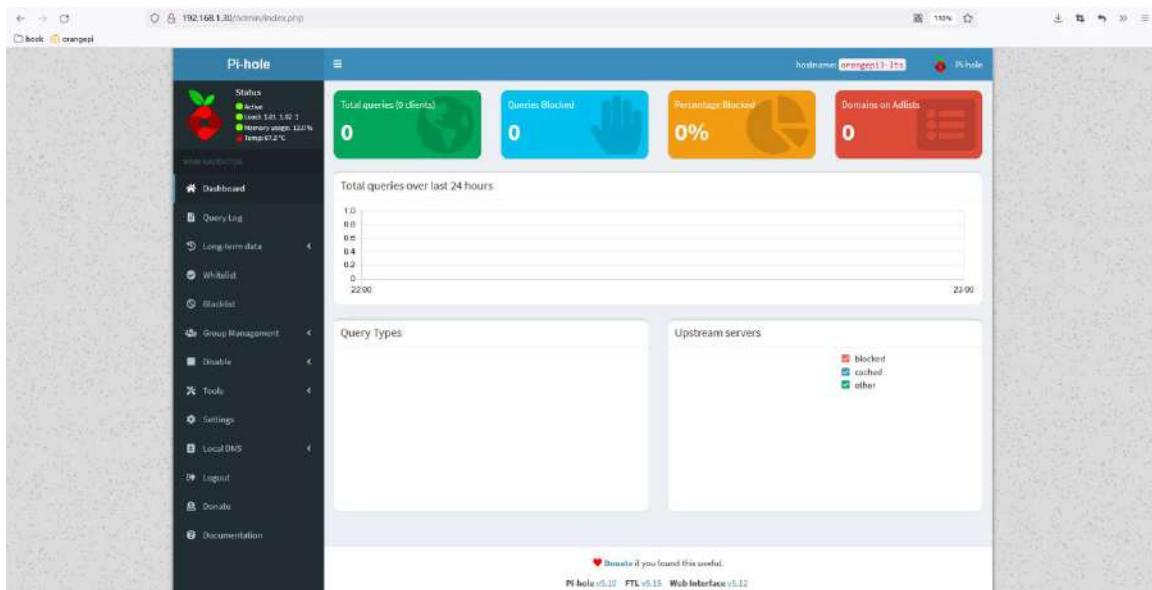
13) Then enter **the IP address /admin of the development board** in the browser to see the web management interface of pi-hole



- 14) Click **Login** on the left, then enter the password shown above and click **Login** below to log in to pi-hole



- 15) The interface after logging in is shown as follows, you can see that there are many more options than before logging in



### 3. 40. Introduction to the use of GotoHTTP

1) GotoHTTP can be used to remotely control the development board in the browser. The installation is relatively simple, and it supports devices of various platforms. As long as there is a browser to access the Internet, the development board can be remotely controlled. The link to the GotoHTTP official website documentation is as follows, please read it carefully before using it

<https://gotohttp.com/goto/help.12x>

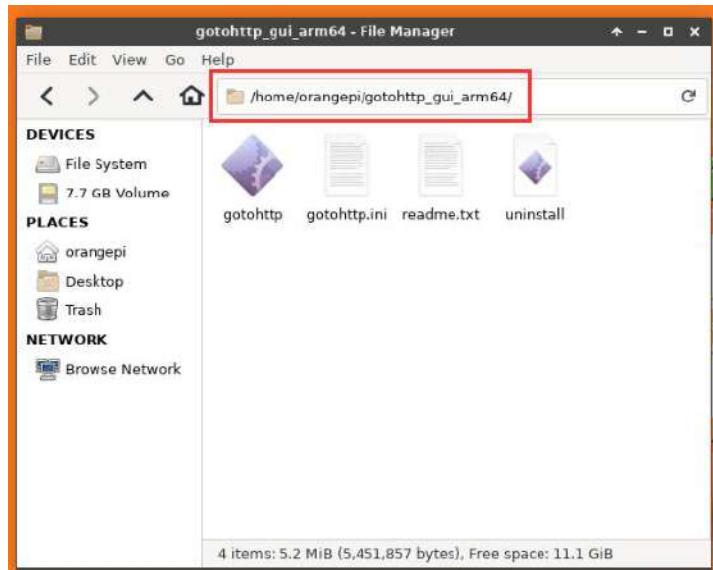
2) GotoHTTP has two versions of compressed packages that provide graphical interface and character interface. If you want to use the graphical interface, first make sure that the Linux system used by the development board is the desktop version system

3) The graphical interface version of the GotoHTTP download and run command is as follows

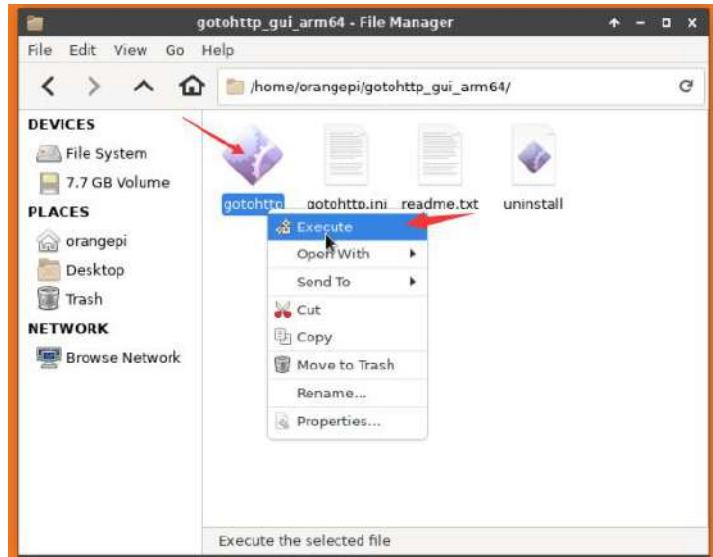
- First download and unzip `gotohttp_gui_arm64.tar.gz`

```
orangeipi@orangeipi:~$ wget http://gotohttp.com/gotohttp_gui_arm64.tar.gz
orangeipi@orangeipi:~$ tar -xvf gotohttp_gui_arm64.tar.gz
```

- Then open the folder where `gotohttp_gui_arm64` is located in the desktop



- c. Then select the gotohttp executable program, right-click, then select **Execute** to open gotohttp



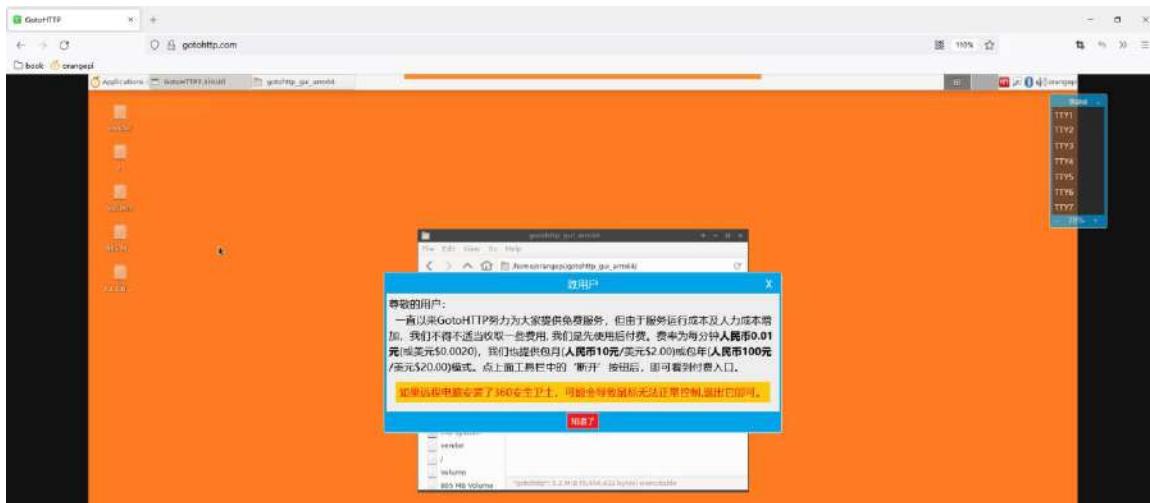
- d. The interface after gotohttp is opened is as follows, the key point is to remember **Computer ID** and **Access Code**



- e. Then open the browser of the computer or mobile phone, enter **http://gotohttp.com** in the address bar, then enter the Computer Id shown above in the computer ID column on the right, enter the Access Code shown above in the control code, and finally Click to start control



- f. Then you can see the desktop of the Linux system of the development board in the browser. After gotohttp is opened, a payment prompt will pop up. **After use, you need to pay**



4) The command to download and run the GotoHTTP version of the character interface version is as follows

- First download and unzip **gotohttp\_cli\_arm64.tar.gz**

```
orangepi@orangepi:~$ wget http://gotohttp.com/gotohttp_cli_arm64.tar.gz  
orangepi@orangepi:~$ tar -xvf gotohttp_cli_arm64.tar.gz
```

- Then use the following command to run GotoHTTP

```
orangepi@orangepi:~$ sudo ./gotohttp_cli_arm64/gotohttp_cli
```

- If you can see the following output after running GotoHTTP, it means the startup is normal. In addition, you can see that there are output Computer Id and Access Code below, please remember them, which will be used later

```
Starting GotoHTTP...  
orangepi@orangepi:~$ Registering to server.....  
Connecting to server.....  
Connected.(secure connection)
```

File Transfer: ON, Try Framebuffer: OFF

This computer is ready for controlling remotely.

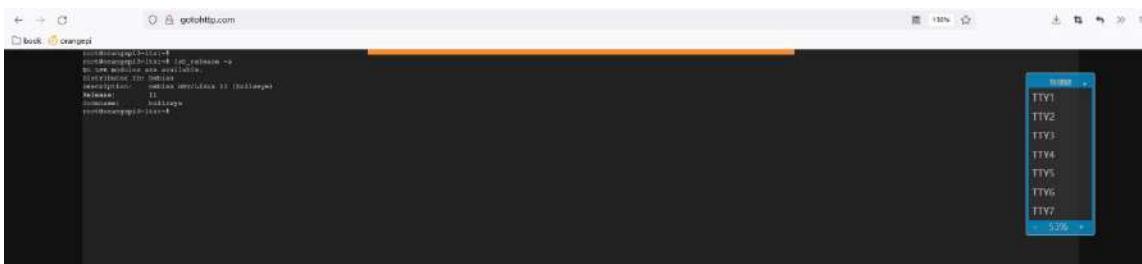
**To control it, please open the URL <http://gotohttp.com> in web browser, then input Computer Id: **137133946** and Access Code: **6103****

- Then open the browser of the computer or mobile phone, enter <http://gotohttp.com> in the address bar, then enter the Computer Id shown above in the computer ID column, enter the Access Code shown above in the control

code, and finally click start control



- e. Then you can see the command line input interface of the development board Linux system in the browser



### **3. 41. Installation method of QT**

### 3. 41. 1. How to install QT5

- 1) The command to install QT5 is as follows

- a. The installation commands for **Debian10**, **Ubuntu20.04** and **Ubuntu18.04** are:

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt-get -y install qt5-default \\\nqttools5-dev-tools qtbase5-doc-html qt5-assistant qt5-doc
```

- b The installation commands for **Debian11** and **Ubuntu22.04** are:

```
orangenpi@orangenpi:~$ sudo apt update
```

```
orangepi@orangepi:~$ sudo apt-get -y install qttools5-dev-tools qtbase5-doc-html \
qt5-assistant qt5-doc qt5-qmake qt5-qmake-bin
```

- 2) After installation, use the following command to view the version number of QT

- a. The default version of Ubuntu 18.04 is:

```
orangepi@orangepi:~$ qmake -v
```



QMake version 3.1

Using Qt version **5.9.5** in /usr/lib/aarch64-linux-gnu

b. The default version of Ubuntu 20.04 is:

```
orangeipi@orangeipi:~$ qmake -v
```

QMake version 3.1

Using Qt version **5.12.8** in /usr/lib/aarch64-linux-gnu

c. The default version of Ubuntu 22.04 is:

```
orangeipi@orangeipi:~$ qmake -v
```

QMake version 3.1

Using Qt version **5.15.3** in /usr/lib/aarch64-linux-gnu

d. The default version for Debian 10 is:

```
orangeipi@orangeipi:~$ qmake -v
```

QMake version 3.1

Using Qt version **5.11.3** in /usr/lib/aarch64-linux-gnu

e. The default version for Debian 11 is:

```
orangeipi@orangeipi:~$ qmake -v
```

QMake version 3.1

Using Qt version **5.15.2** in /usr/lib/aarch64-linux-gnu

3) The command to view the QT installation path is as follows (Ubuntu 20.04 as an example)

```
orangeipi@orangeipi:~$ qmake -qt5 -query
QT_SYSROOT:
QT_INSTALL_PREFIX:/usr
QT_INSTALL_ARCHDATA:/usr/lib/aarch64-linux-gnu/qt5
QT_INSTALL_DATA:/usr/share/qt5
QT_INSTALL_DOCS:/usr/share/qt5/doc
QT_INSTALL_HEADERS:/usr/include/aarch64-linux-gnu/qt5
QT_INSTALL_LIBS:/usr/lib/aarch64-linux-gnu
QT_INSTALL_LIBEXECS:/usr/lib/aarch64-linux-gnu/qt5/libexec
QT_INSTALL_BINS:/usr/lib/qt5/bin
QT_INSTALL_TESTS:/usr/tests
QT_INSTALL_PLUGINS:/usr/lib/aarch64-linux-gnu/qt5/plugins
QT_INSTALL_IMPORTS:/usr/lib/aarch64-linux-gnu/qt5imports
QT_INSTALL_QML:/usr/lib/aarch64-linux-gnu/qt5/qml
```

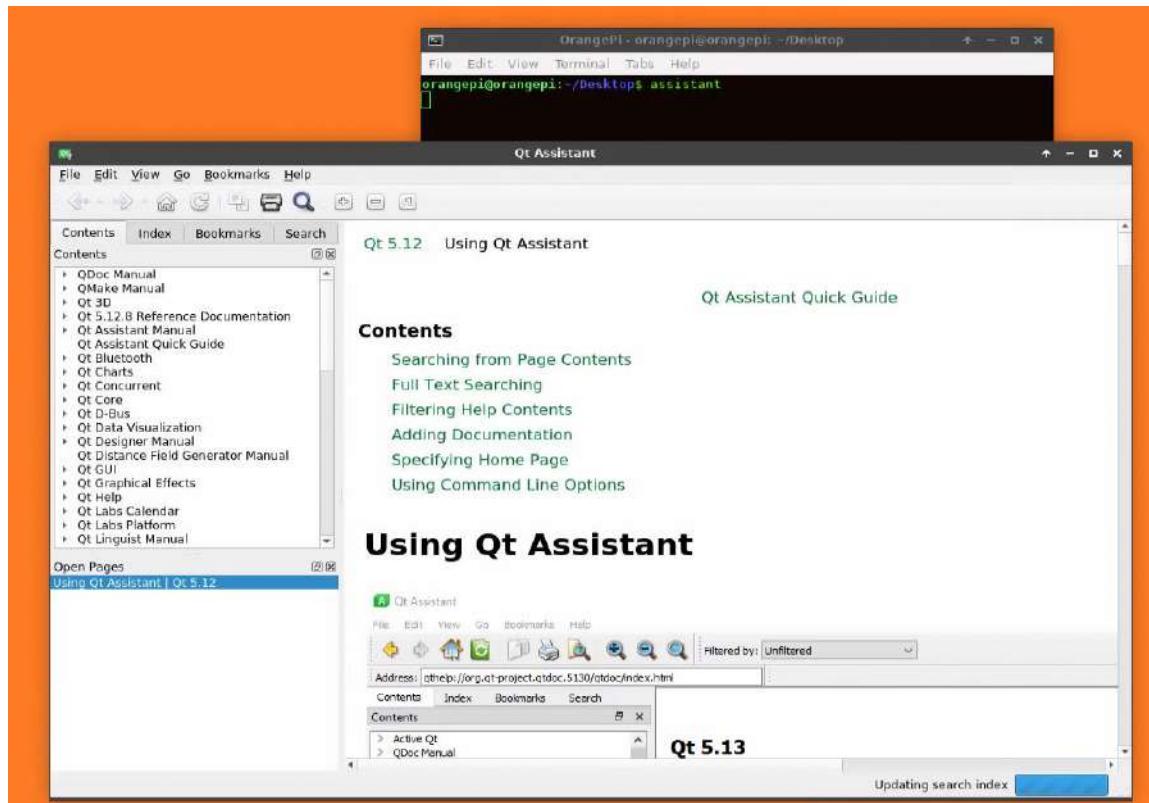


```
QT_INSTALL_TRANSLATIONS:/usr/share/qt5/translations
QT_INSTALL_CONFIGURATION:/etc/xdg
QT_INSTALL_EXAMPLES:/usr/lib/aarch64-linux-gnu/qt5/examples
QT_INSTALL_DEMOS:/usr/lib/aarch64-linux-gnu/qt5/examples
QT_HOST_PREFIX:/usr
QT_HOST_DATA:/usr/lib/aarch64-linux-gnu/qt5
QT_HOST_BINS:/usr/lib/qt5/bin
QT_HOST_LIBS:/usr/lib/aarch64-linux-gnu
QMAKE_SPEC:linux-g++
QMAKE_XSPEC:linux-g++
QMAKE_VERSION:3.1
QT_VERSION:5.12.8
```

#### 4) Use the following command to open QT assistant

```
orangeipi@orangeipi:~$ assistant
```

The interface after QT assistant is opened is as follows

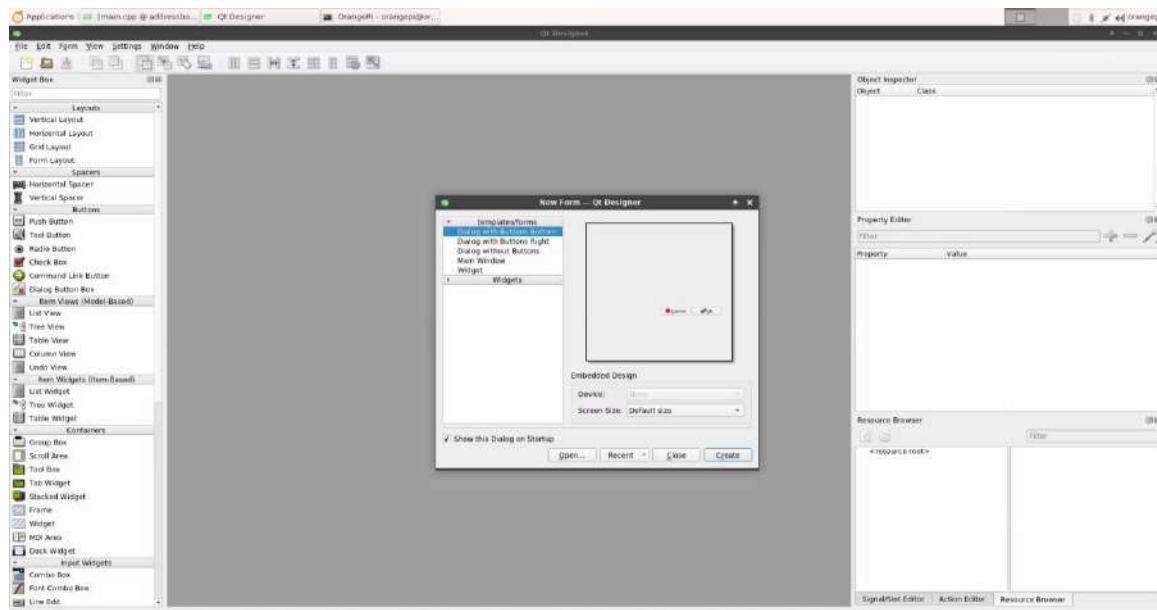


#### 5) Use the following command to open QT designer



```
orangeipi@orangeipi:~$ designer
```

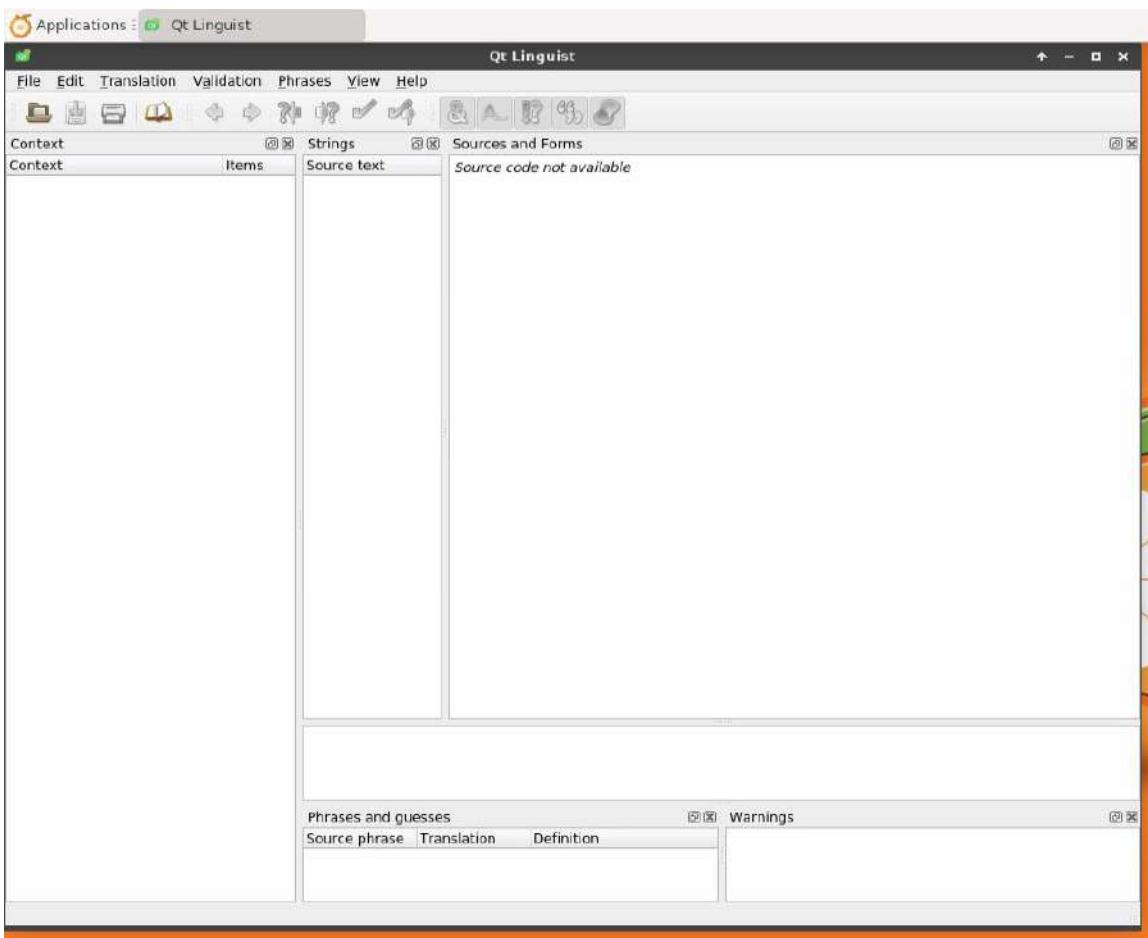
The interface after QT designer is opened is as follows



6) Use the following command to open QT linguist

```
orangeipi@orangeipi:~$ linguist
```

The interface after QT linguist is opened is as follows

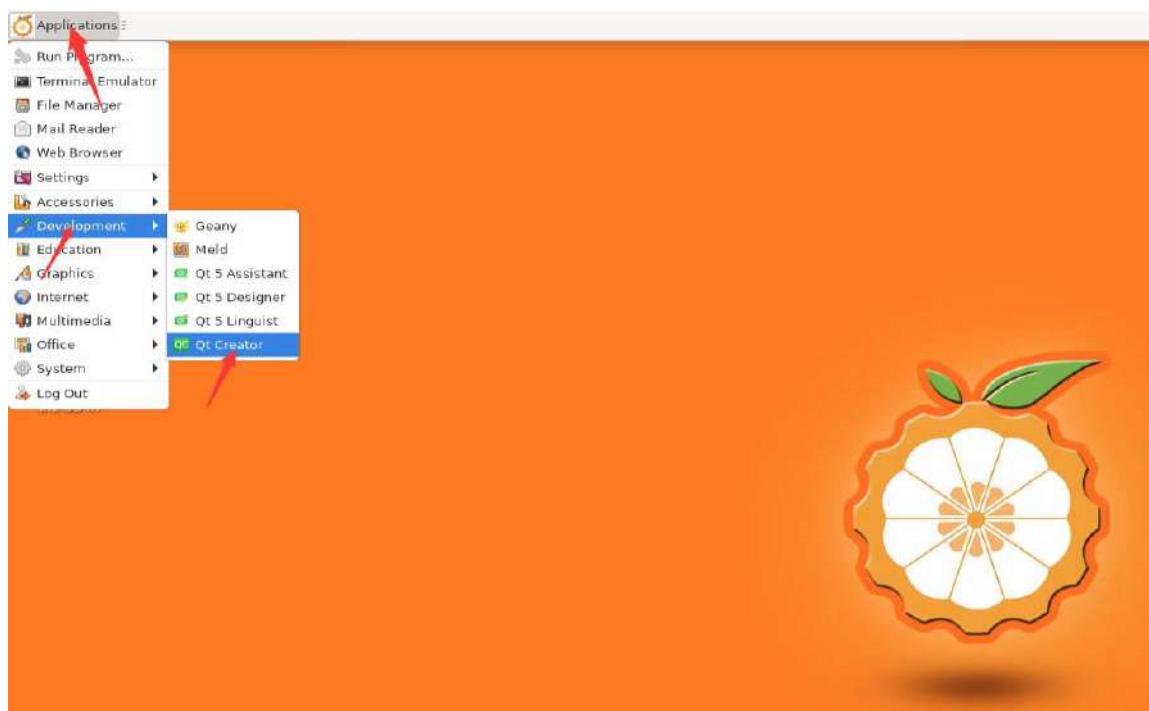


### 3. 41. 2. How to install QT Creator

- 1) The command to install QT Creator is as follows

```
orangepi@orangepi:~$ sudo apt update  
orangepi@orangepi:~$ sudo apt-get -y install qtcreator qmlscene gdb \\\nqtdeclarative5-dev qtbase5-examples
```

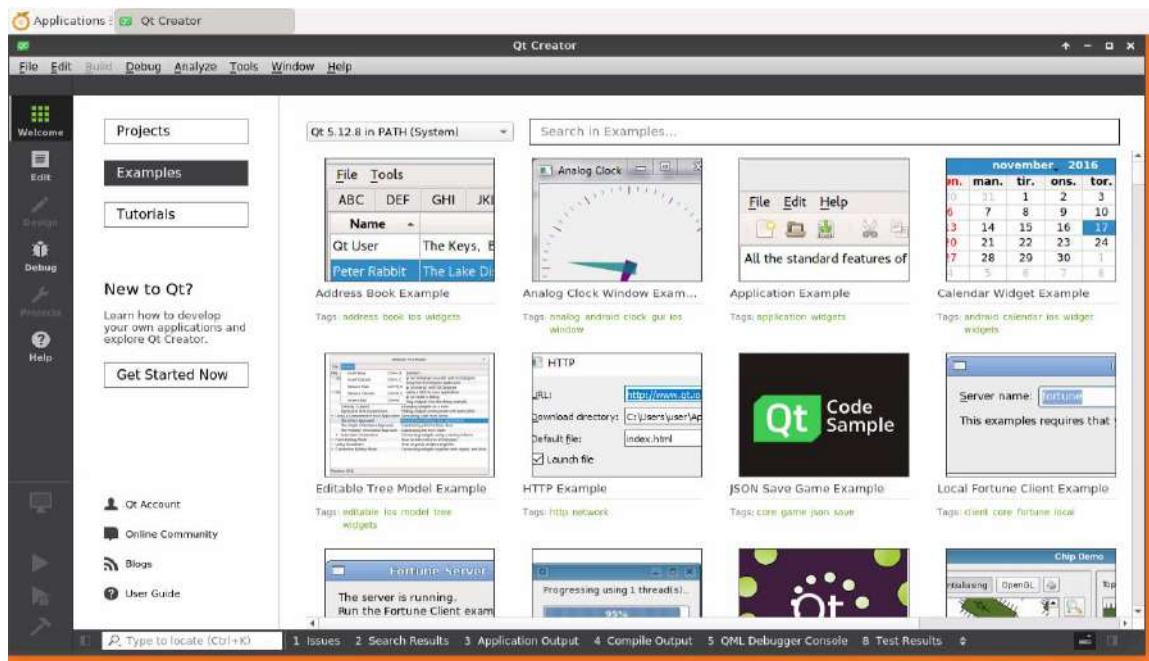
- 2) After QT Creator is installed, you can see the startup icon of QT Creator in **Applications**



You can also open QT Creator with the following command

```
orangeipi@orangeipi:~$ qtcreator
```

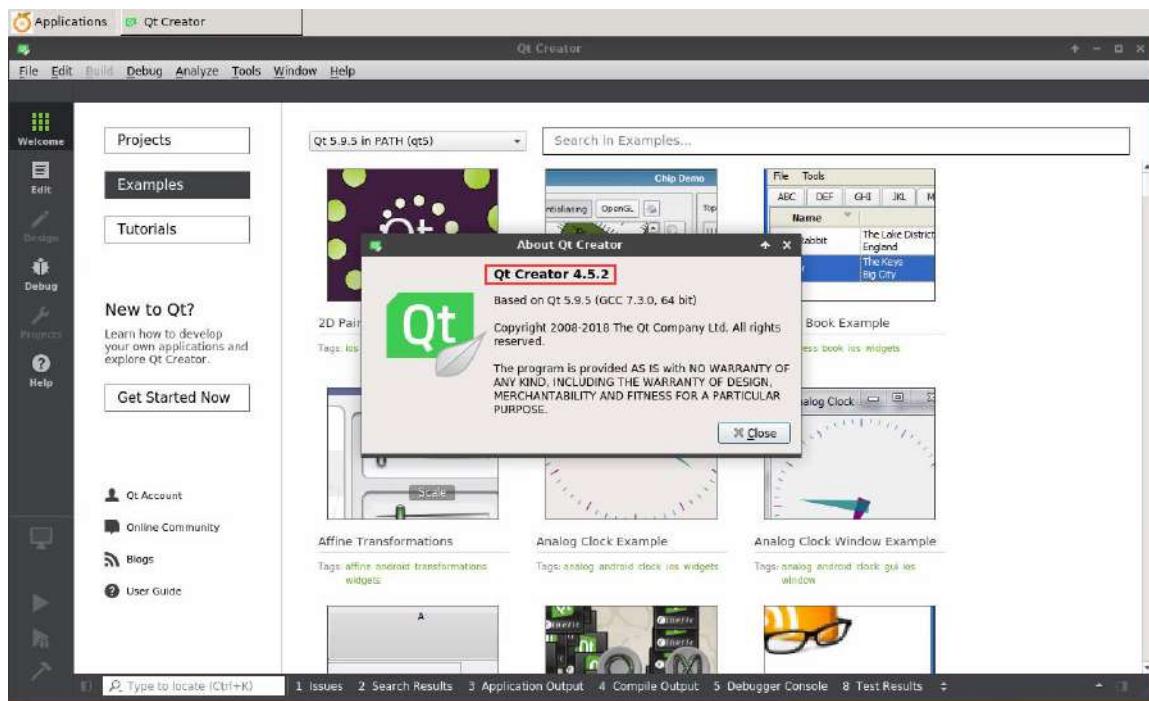
### 3) The interface after QT Creator is opened is as follows



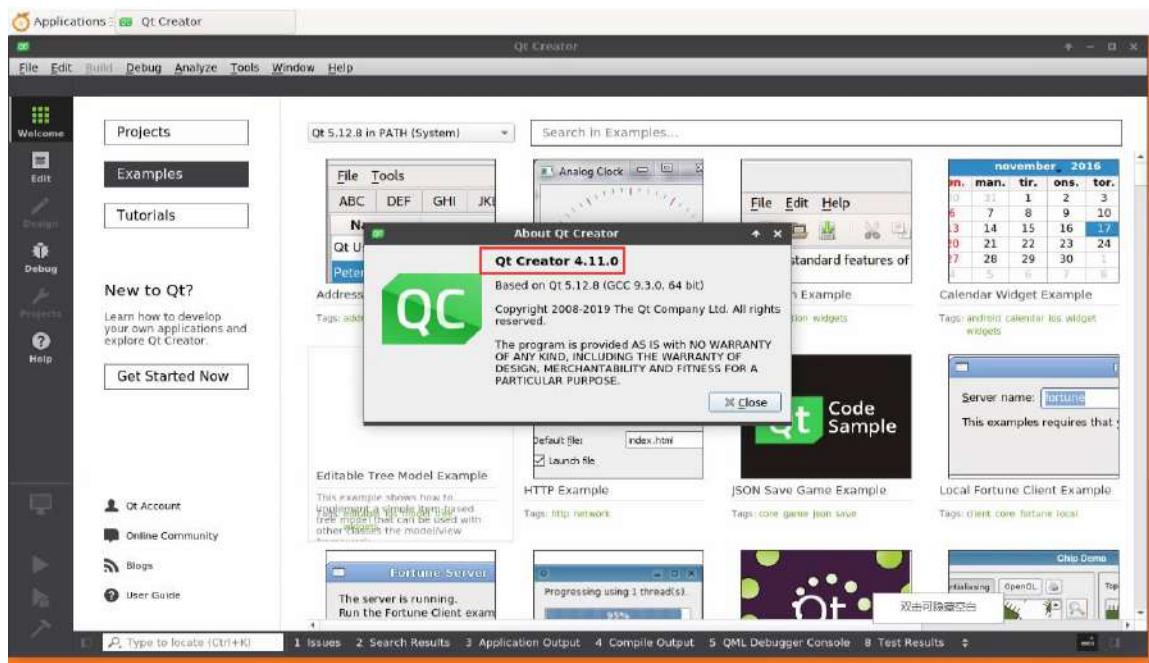
### 4) The version of QT Creator is as follows



a. The default version of QT Creator in **Ubuntu18.04** is as follows



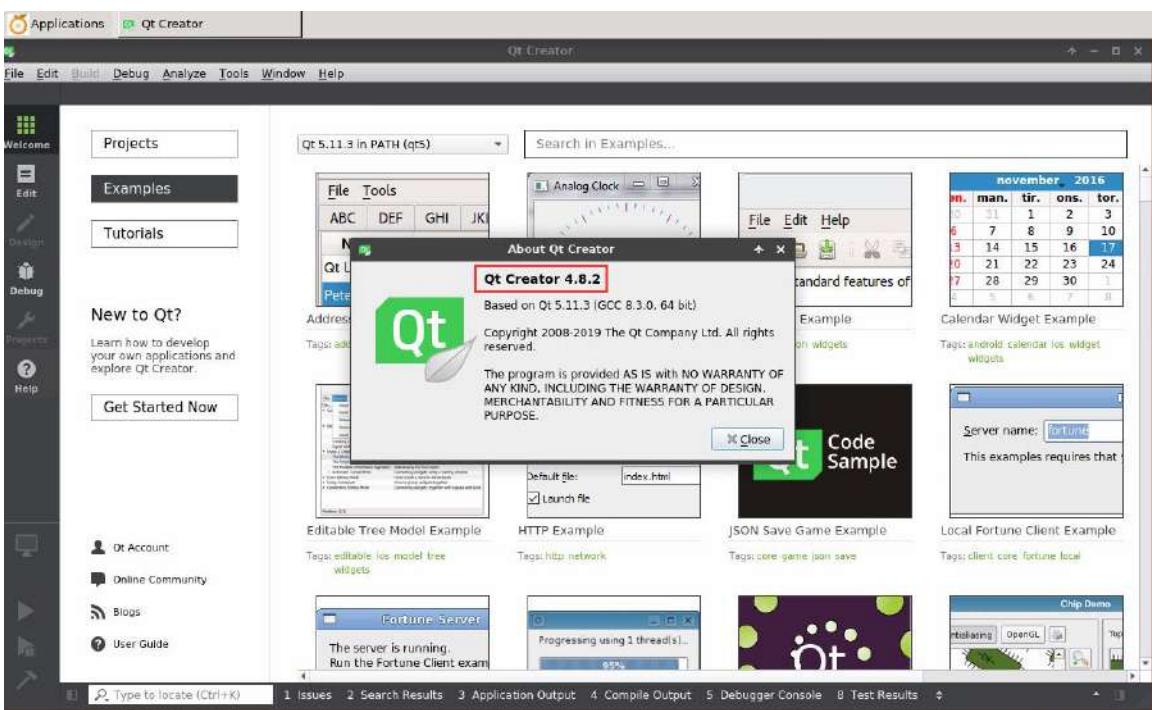
b. The default version of QT Creator in **Ubuntu20.04** is as follows



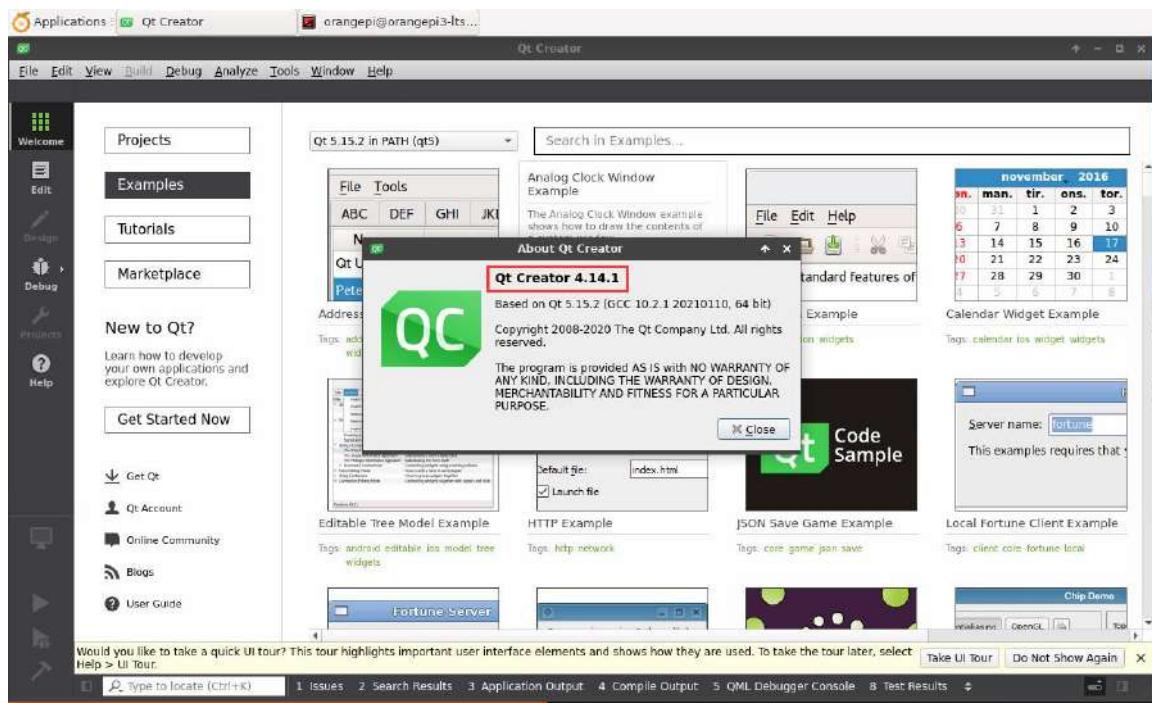
c. The default version of QT Creator in **Ubuntu22.04** is as follows



d. The default version of QT Creator in **Debian10** is as follows



e. The default version of QT Creator in **Debian11** looks like this

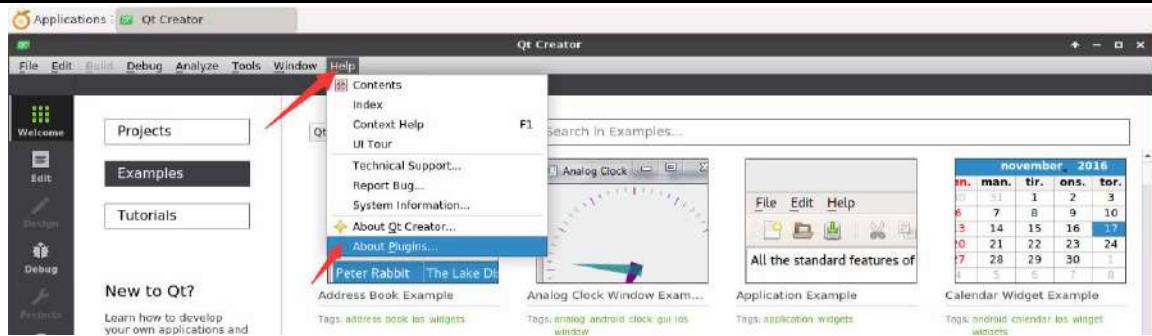


5) Let's test the sample code that comes with QT. Before the test, you need to modify the owner and group of the **examples** folder to the **orangeipi** user

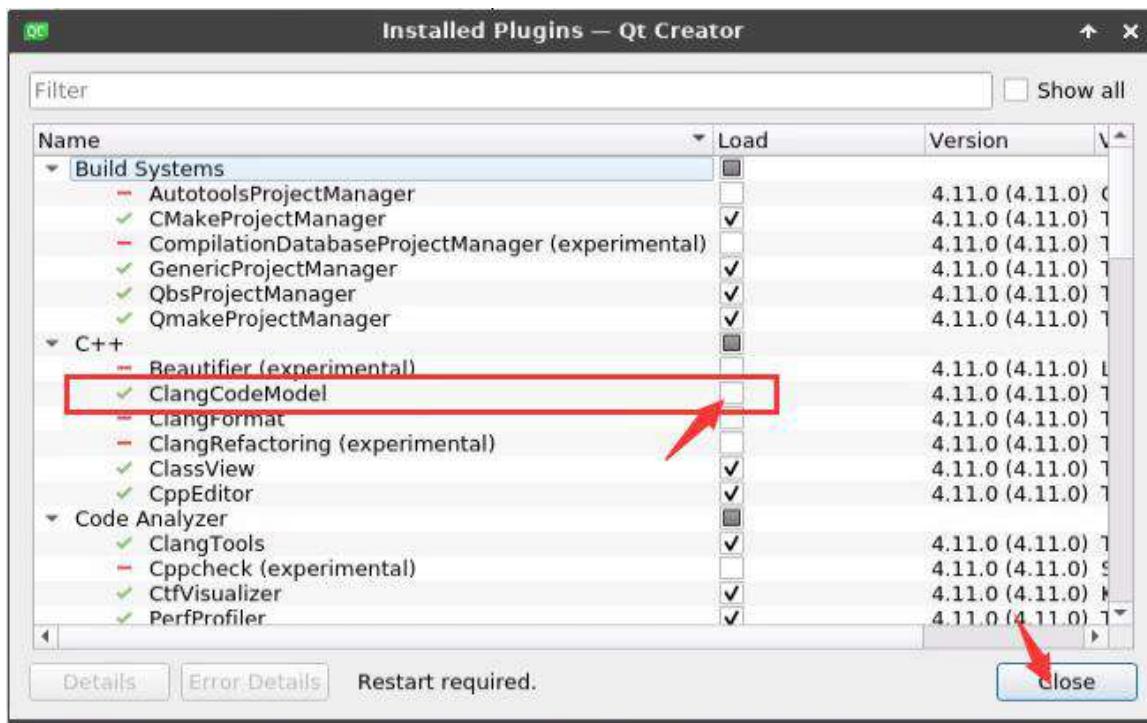
```
orangeipi@orangeipi:~$ sudo chown orangeipi:orangeipi \
/usr/lib/aarch64-linux-gnu/qt5/examples -R
```

You also need to open **Help->About Plugins...**

Note that **Ubuntu18.04** does not need to set this step.

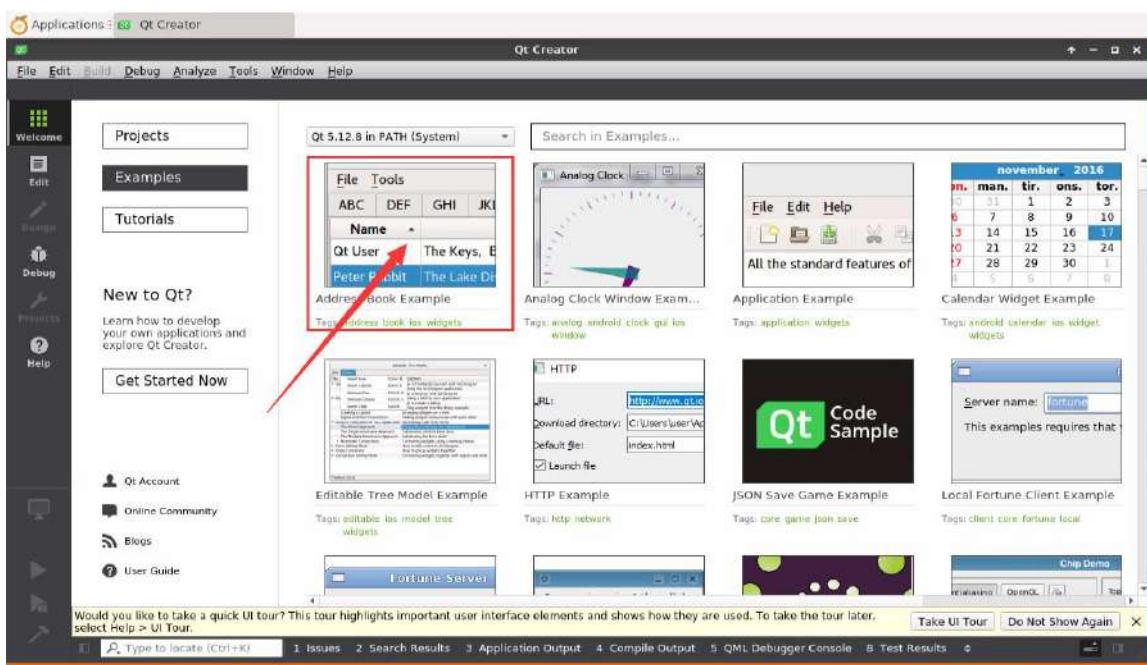


Then remove the tick of **ClangCodeModel**

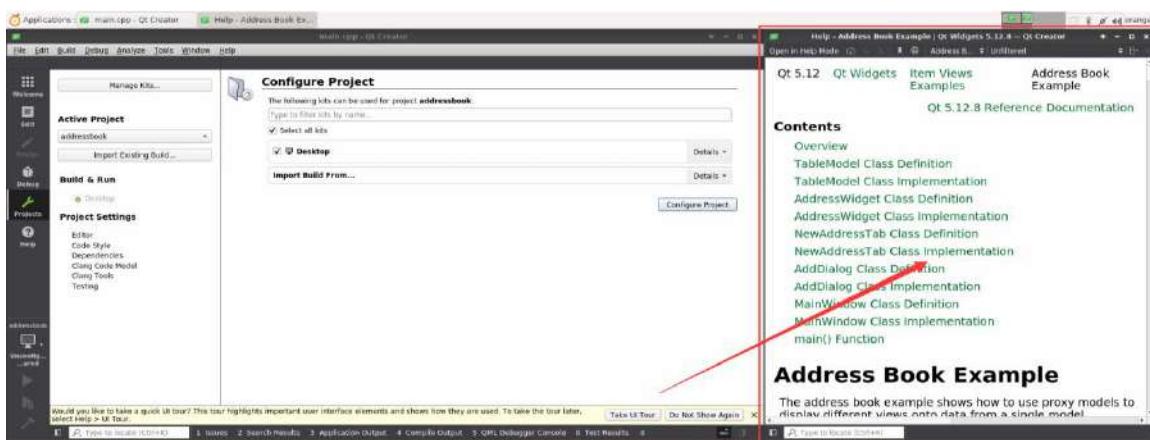


After setting, you need to restart QT Creator

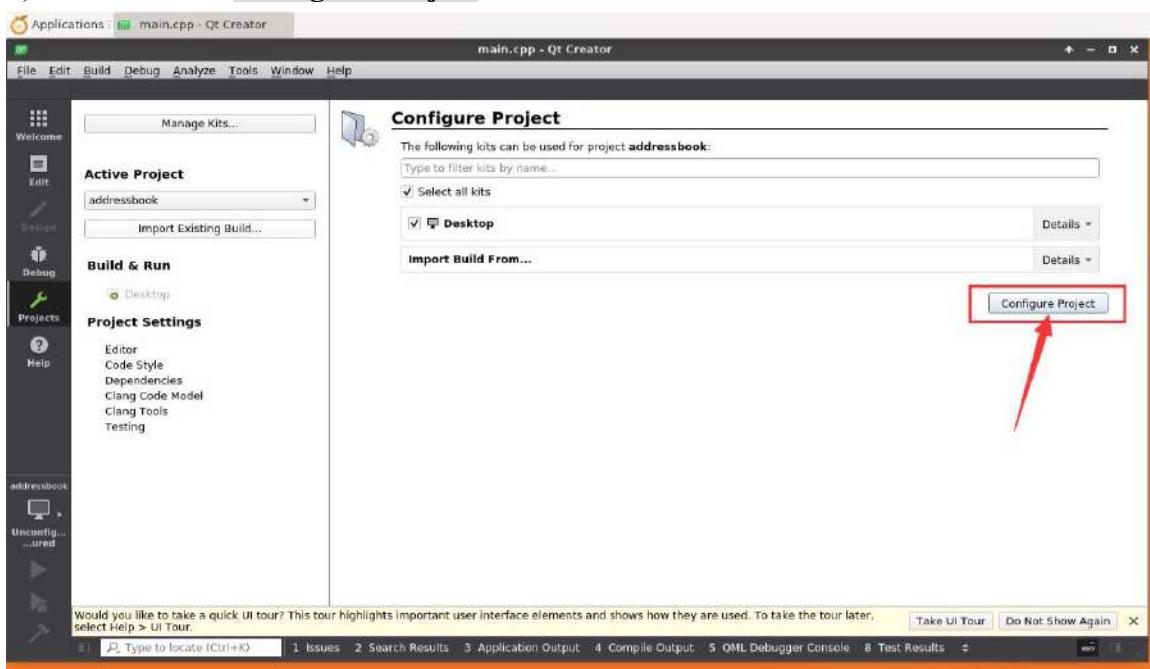
6) Then you can open a sample code



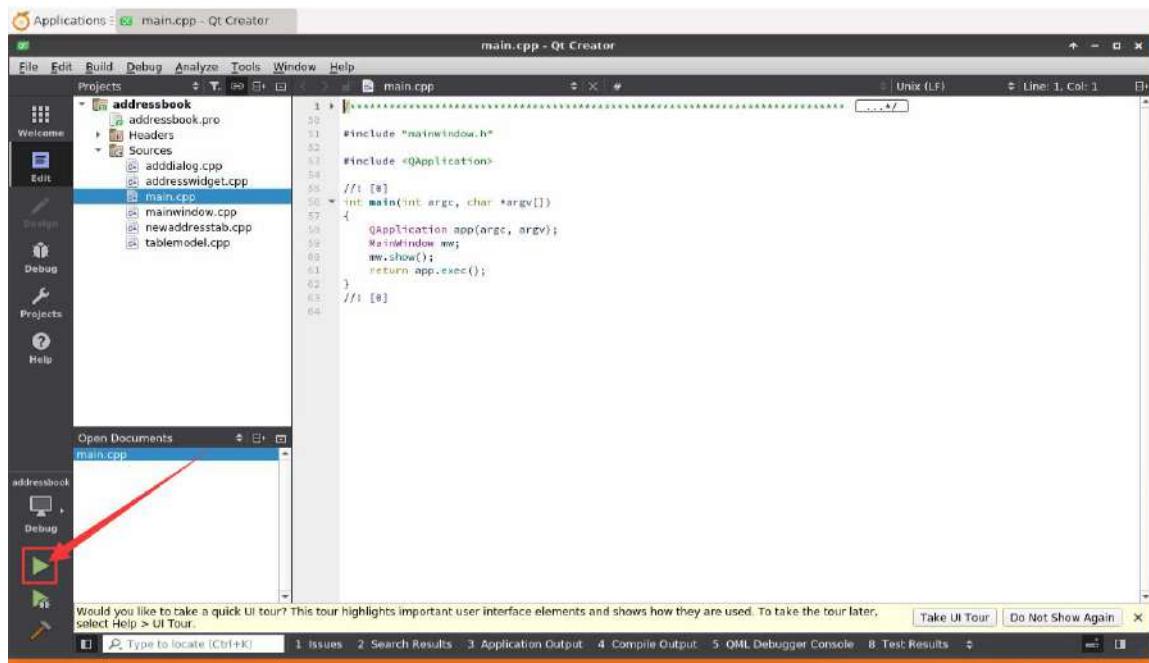
7) After clicking the sample code, the corresponding documentation will be automatically opened, and you can take a closer look at the usage instructions.



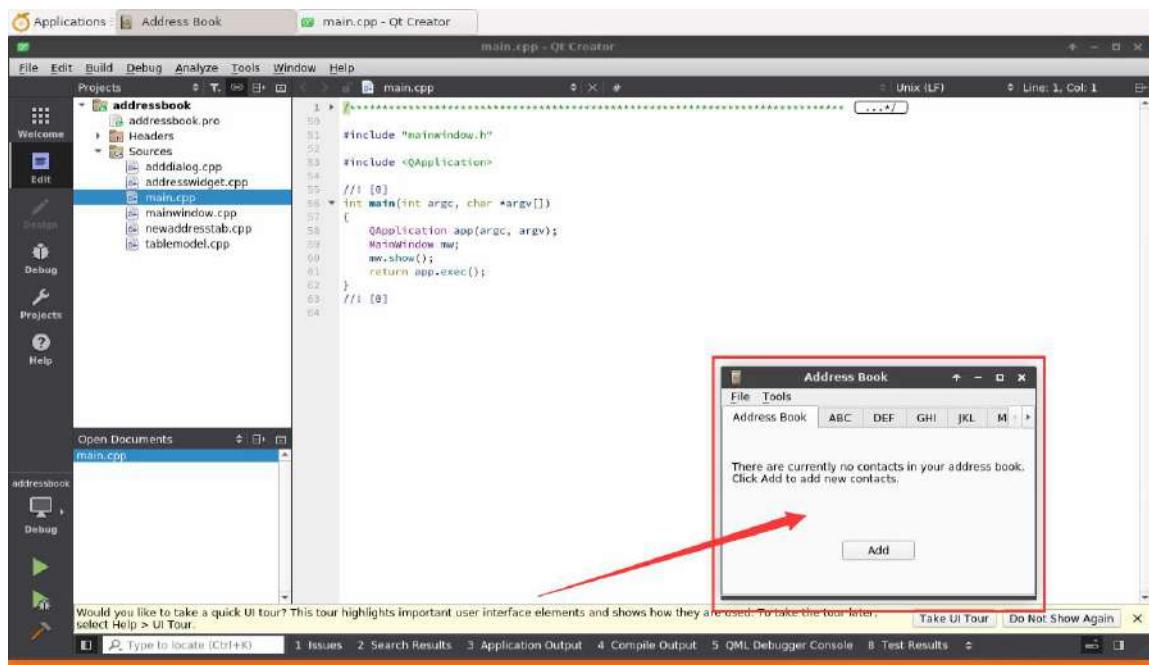
## 8) Then click on **Configure Project**



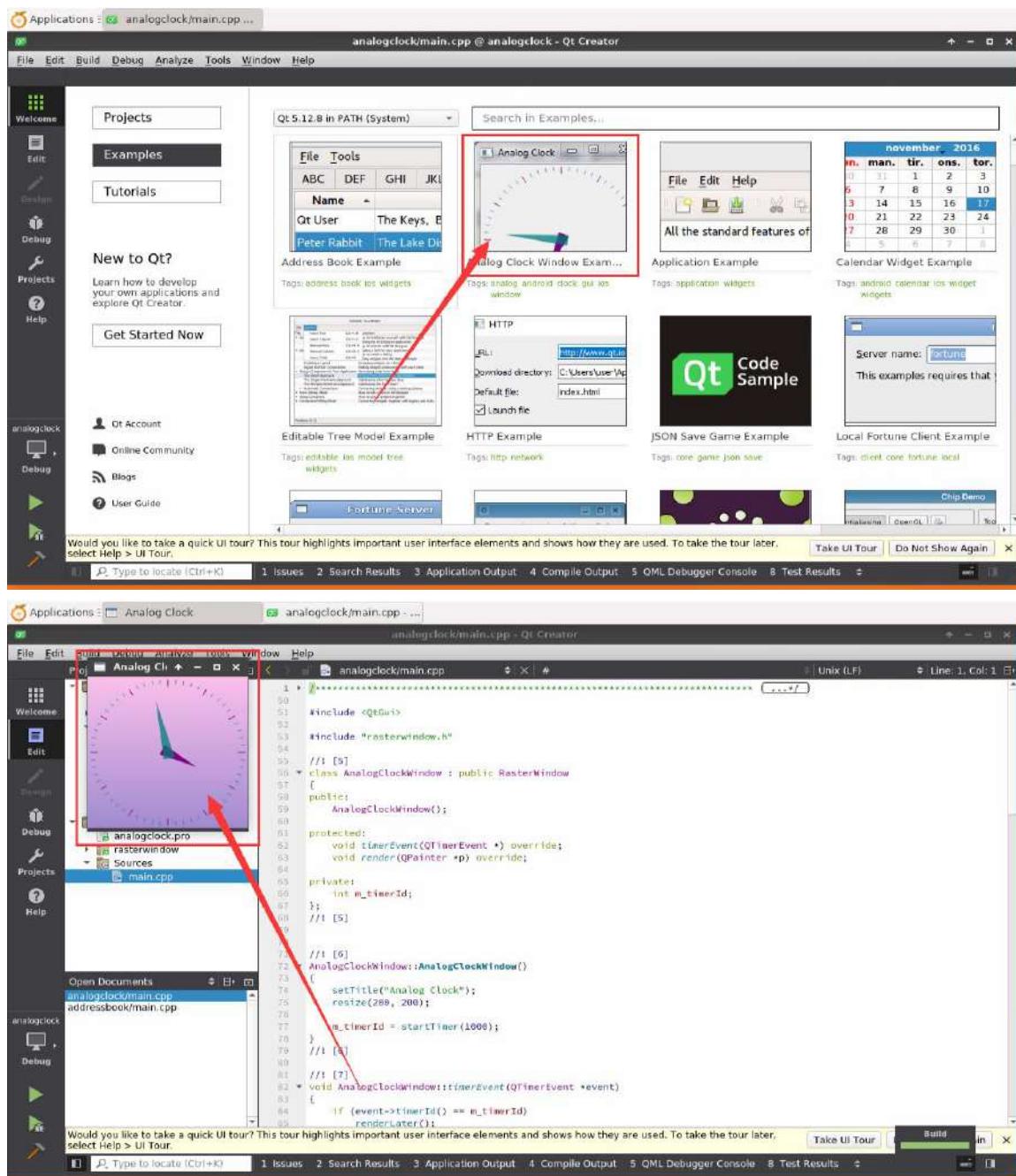
## 9) Then click the green triangle in the lower left corner to compile and run the sample code



- 10) After waiting for a period of time, the interface shown in the figure below will pop up, which means that QT can compile and run normally



- 11) The running result of the analogclock sample code is as follows



## 12) References

- [https://wiki.qt.io/Install\\_Qt\\_5\\_on\\_Ubuntu](https://wiki.qt.io/Install_Qt_5_on_Ubuntu)
- <https://download.qt.io/archive/qtcreator>
- <https://download.qt.io/archive/qt>



### 3. 42. GPU test description

Note that the following tests are all performed on the desktop version of the system, so please ensure that the system used by the development board is the desktop version of the system.

#### 3. 42. 1. Ubuntu 22.04 Linux5.16 system GPU test instructions

1) First, please connect the HDMI display. All the following commands are operated in the desktop displayed by HDMI, so please do not use ssh remote login or use serial port to log in to the Linux system. If you do not have an HDMI display, you can use NoMachine or VNC to remotely log in to the desktop of the Linux system.

2) After entering the desktop, first open a terminal, and then use the **glxinfo -B** command to see that the GPU driver used is **Mali-T720 (Panfrost)** instead of **llvmpipe**.

```
orangeipi@orangeipi:~$ glxinfo -B
```

```
orangeipi@orangeipi3-lts:~$ glxinfo -B
name of display: :0.0
display: :0 screen: 0
direct rendering: Yes
Extended renderer info (GLX_MESA_query_renderer):
  Vendor: Panfrost (0xffffffff)
  Device: Mali-T720 (Panfrost) (0xffffffff)
  Version: 22.0.1
  Accelerated: yes
  Video memory: 1984MB
  Unified memory: yes
  Preferred profile: compat (0x2)
  Max core profile version: 0.0
  Max compat profile version: 2.1
  Max GLES1 profile version: 1.1
  Max GLES[23] profile version: 2.0
OpenGL vendor string: Panfrost
OpenGL renderer string: Mali-T720 (Panfrost)
OpenGL version string: 2.1 Mesa 22.0.1
OpenGL shading language version string: 1.20

OpenGL ES profile version string: OpenGL ES 2.0 Mesa 22.0.1
OpenGL ES profile shading language version string: OpenGL ES GLSL ES 1.0.16

orangeipi@orangeipi3-lts:~$
```

3) Using the **screenfetch** command, you can also see that the GPU driver is using **Mali-T720**

```
orangeipi@orangeipi:~$ sudo apt install -y screenfetch
orangeipi@orangeipi:~$ screenfetch
```



```
orangepi@orangepi3-lts:~$ screenfetch
orangepi@orangepi3-lts
OS: Ubuntu 22.04 jammy
Kernel: aarch64 Linux 5.16.17-sun50iw6
Uptime: 10m
Packages: 1399
Shell: bash 5.1.16
Resolution: 1920x1080
DE: Xfce
WM: Xfwm4
WM Theme: Numix
GTK Theme: Materia [GTK2]
Icon Theme: LoginIcons
Font: Sans 10
Disk: 3.2G / 15G (23%)
CPU: ARM Cortex-A53 @ 4x 1.704GHz
GPU: Mali-T720 (Panfrost)
RAM: 1109MiB / 1984MiB
orangepi@orangepi3-lts:~$
```

#### 4) The **Glx-Gears** test is shown below

```
orangepi@orangepi:~$ glxgears
```

```
orangepi@orangepi3-lts:~$ glxgears
Running synchronized to the vertical refresh. The framerate should be
approximately the same as the monitor refresh rate.
303 frames in 5.0 seconds = 60.540 FPS
301 frames in 5.0 seconds = 60.001 FPS
300 frames in 5.0 seconds = 59.992 FPS
300 frames in 5.0 seconds = 59.999 FPS
301 frames in 5.0 seconds = 60.009 FPS
300 frames in 5.0 seconds = 59.991 FPS
300 frames in 5.0 seconds = 60.000 FPS
300 frames in 5.0 seconds = 60.000 FPS
301 frames in 5.0 seconds = 60.008 FPS
300 frames in 5.0 seconds = 59.994 FPS
301 frames in 5.0 seconds = 60.007 FPS
300 frames in 5.0 seconds = 59.999 FPS
300 frames in 5.0 seconds = 60.000 FPS
```

#### 5) **glmark2-es2** is a benchmark tool for OpenGL (ES) 2.0. Using glmark2 can test the performance of GPU OpenGL ES 2.0

```
orangepi@orangepi:~$ sudo apt install -y glmark2-es2
```



```
orangeipi@orangeipi:~$ glmark2-es2
```

```
orangeipi@orangeipi:~$ glmark2-es2 --off-screen
```

a. The test scores for the **glmark2-es2** command are shown below

```
Applications : orangeipi@orangeipi3-lts:~
```

```
orangeipi@orangeipi3-lts:~$ glmark2-es2
```

```
glmark2 2021.02
```

```
OpenGL Information
GL_VENDOR: Panfrost
GL_RENDERER: Mali-T720 (Panfrost)
GL_VERSION: OpenGL ES 2.0 Mesa 22.0.1
```

```
[build] use-vbo=false: FPS: 125 FrameTime: 8.000 ms
[build] use-vbo=true: FPS: 133 FrameTime: 7.519 ms
[texture] texture-filter=nearest: FPS: 137 FrameTime: 7.299 ms
[texture] texture-filter=linear: FPS: 144 FrameTime: 6.944 ms
[texture] texture-filter=mipmap: FPS: 142 FrameTime: 7.042 ms
[shading] shading=gouraud: FPS: 127 FrameTime: 7.874 ms
[shading] shading=blinn-phong-inf: FPS: 124 FrameTime: 8.065 ms
[shading] shading=blinn-phong: FPS: 119 FrameTime: 8.696 ms
[shading] shading=cel: FPS: 113 FrameTime: 8.850 ms
[shading] shading=cg: FPS: 113 FrameTime: 8.850 ms
[bump] bump-render=high-poly: FPS: 94 FrameTime: 10.630 ms
[bump] bump-render=normals: FPS: 138 FrameTime: 7.246 ms
[bump] bump-render=height: FPS: 137 FrameTime: 7.299 ms
[effect2d] kernel=0,1,0;1,-4,1;0,1,0: FPS: 107 FrameTime: 9.346 ms
[effect2d] kernel=1,1,1,1,1,1,1,1,1,1,1,1: FPS: 53 FrameTime: 18.868 ms
[pulsar] light=false;quads=5;texture=false;windows=4: FPS: 135 FrameTime: 7.467 ms
(desktop) blur-radius=5;effect=blur;passes=1;separable=true;windows=4: FPS: 45 FrameTime: 22.222 ms
(buffer) columns=200;interleave=false;update-dispersion=0.9;update-fraction=0.5;update-method=map: FPS: 60 FrameTime: 16.667 ms
(buffer) columns=200;interleave=false;update-dispersion=0.9;update-fraction=0.5;update-method=subdata: FPS: 60 FrameTime: 16.667 ms
(buffer) columns=200;interleave=true;update-dispersion=0.9;update-fraction=0.5;update-method=nop: FPS: 56 FrameTime: 20.000 ms
(ideas) speed-duration: FPS: 99 FrameTime: 10.101 ms
(jellyfish) <default>: FPS: 75 FrameTime: 13.333 ms
(terrain) <default>: FPS: 8 FrameTime: 125.000 ms
(shadow) <default>: FPS: 94 FrameTime: 10.638 ms
(refract) <default>: FPS: 33 FrameTime: 30.303 ms
(conditional) fragment-steps=0;vertex-steps=0: FPS: 141 FrameTime: 7.092 ms
(conditional) fragment-steps=5;vertex-steps=0: FPS: 109 FrameTime: 9.174 ms
(conditional) fragment-steps=0;vertex-steps=5: FPS: 139 FrameTime: 7.194 ms
(function) fragment-complexity=low;fragment-steps=5: FPS: 129 FrameTime: 7.752 ms
(function) fragment-complexity=medium;fragment-steps=5: FPS: 107 FrameTime: 9.346 ms
(loop) fragment-loop=false;fragment-steps=5;vertex-steps=5: FPS: 127 FrameTime: 7.874 ms
(loop) fragment-steps=5;fragment-uniform=false;vertex-steps=5: FPS: 127 FrameTime: 7.874 ms
(loop) fragment-steps=5;fragment-uniform=true;vertex-steps=5: FPS: 94 FrameTime: 10.638 ms
```

```
glmark2 Score: 104
```

```
orangeipi@orangeipi3-lts:~$
```

b. The test scores for the **glmark2-es2 --off-screen** command are shown below

```
Applications : orangeipi@orangeipi3-lts:~
```

```
orangeipi@orangeipi3-lts:~$ glmark2-es2 --off-screen
```

```
glmark2 2021.02
```

```
OpenGL Information
GL_VENDOR: Panfrost
GL_RENDERER: Mali-T720 (Panfrost)
GL_VERSION: OpenGL ES 2.0 Mesa 22.0.1
```

```
[build] use-vbo=false: FPS: 337 FrameTime: 2.967 ms
[build] use-vbo=true: FPS: 425 FrameTime: 2.369 ms
[texture] texture-filter=nearest: FPS: 475 FrameTime: 2.105 ms
[texture] texture-filter=linear: FPS: 401 FrameTime: 2.879 ms
[texture] texture-filter=mipmap: FPS: 448 FrameTime: 2.232 ms
[shading] shading=gouraud: FPS: 341 FrameTime: 2.933 ms
[shading] shading=blinn-phong-inf: FPS: 317 FrameTime: 3.155 ms
[shading] shading=cel: FPS: 268 FrameTime: 3.731 ms
[shading] shading=cg: FPS: 249 FrameTime: 4.016 ms
[bump] bump-render=high-poly: FPS: 177 FrameTime: 5.650 ms
[bump] bump-render=normals: FPS: 427 FrameTime: 2.342 ms
[bump] bump-render=height: FPS: 409 FrameTime: 2.445 ms
[effect2d] kernel=0,1,0;1,-4,1;0,1,0: FPS: 230 FrameTime: 4.255 ms
[effect2d] kernel=1,1,1,1,1,1,1,1,1,1,1,1: FPS: 91 FrameTime: 10.989 ms
(desktop) blur-radius=5;effect=blur;passes=1;separable=true;windows=4: FPS: 56 FrameTime: 17.857 ms
(buffer) columns=200;interleave=false;update-dispersion=0.9;update-fraction=0.5;update-method=map: FPS: 81 FrameTime: 12.346 ms
(buffer) columns=200;interleave=false;update-dispersion=0.9;update-fraction=0.5;update-method=subdata: FPS: 81 FrameTime: 12.346 ms
(buffer) columns=200;interleave=true;update-dispersion=0.9;update-fraction=0.5;update-method=map: FPS: 57 FrameTime: 17.544 ms
(ideas) speed-duration: FPS: 165 FrameTime: 0.861 ms
(jellyfish) <default>: FPS: 118 FrameTime: 0.475 ms
(terrain) <default>: FPS: 111 FrameTime: 0.475 ms
(shadow) <default>: FPS: 156 FrameTime: 6.410 ms
(refract) <default>: FPS: 37 FrameTime: 27.027 ms
(conditional) fragment-steps=0;vertex-steps=0: FPS: 468 FrameTime: 2.137 ms
(conditional) fragment-steps=5;vertex-steps=0: FPS: 240 FrameTime: 4.167 ms
(conditional) fragment-steps=0;vertex-steps=5: FPS: 452 FrameTime: 2.212 ms
(function) fragment-complexity=low;fragment-steps=5: FPS: 354 FrameTime: 2.825 ms
(function) fragment-complexity=medium;fragment-steps=5: FPS: 230 FrameTime: 4.348 ms
(loop) fragment-loop=false;fragment-steps=5;vertex-steps=5: FPS: 347 FrameTime: 2.882 ms
(loop) fragment-steps=5;fragment-uniform=false;vertex-steps=5: FPS: 344 FrameTime: 3.007 ms
(loop) fragment-steps=5;fragment-uniform=true;vertex-steps=5: FPS: 179 FrameTime: 5.587 ms
```

```
glmark2 Score: 264
```

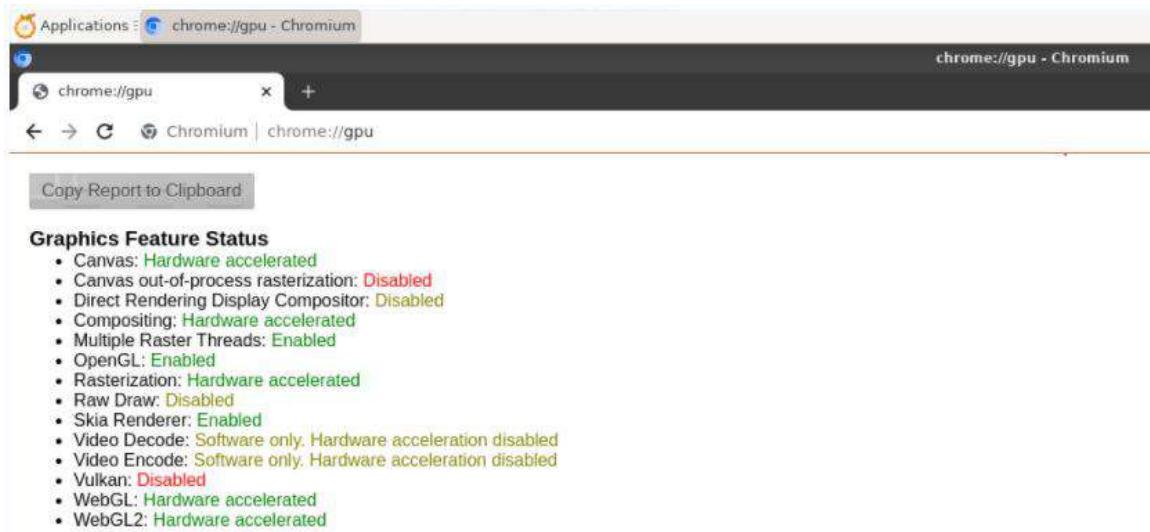
```
orangeipi@orangeipi3-lts:~$
```



6) Then you can install the Chromium browser through the snap command,

```
orangepi@orangepi:~$ sudo snap install chromium
```

7) Then open the Chromium browser and enter **chrome://gpu** in the address bar to view the GPU support in the Chromium browser



### 3. 43. Ubuntu22.04 method of installing browser

1) The installation command of Firefox browser is:

```
orangepi@orangepi:~$ sudo apt install firefox
```

2) The installation command for Chromium browser is:

```
orangepi@orangepi:~$ sudo snap install chromium
```

### 3. 44. Partial programming language test supported by Linux system

#### 3. 44. 1. Debian Bullseye System

1) Debian Bullseye is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is shown below

```
orangepi@orangepi:~$ gcc --version
```



```
gcc (Debian 10.2.1-6) 10.2.1 20210110
```

```
Copyright (C) 2020 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

- b. Write the **hello\_world.c** program in C language

```
orangeipi@orangeipi:~$ vim hello_world.c
```

```
#include <stdio.h>
```

```
int main(void)
{
    printf("Hello World!\n");

    return 0;
}
```

- c. Then compile and run **hello\_world.c**

```
orangeipi@orangeipi:~$ gcc -o hello_world hello_world.c
```

```
orangeipi@orangeipi:~$ ./hello_world
```

```
Hello World!
```

- 2) Debian Bullseye has Python3 installed by default

- a. The specific version of Python is as follows

```
orangeipi@orangeipi:~$ python3
```

```
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
```

```
[GCC 10.2.1 20210110] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>>
```

- b. Write the **hello\_world.py** program in the Python language

```
orangeipi@orangeipi:~$ vim hello_world.py
```

```
print('Hello World!')
```

- c. The result of running **hello\_world.py** is as follows

```
orangeipi@orangeipi:~$ python3 hello_world.py
```

```
Hello World!
```

- 3) Debian Bullseye does not install Java compilation tools and runtime environment by



default

- a. You can install openjdk with the following command, the latest version in Debian Bullseye is openjdk-17

```
orangeipi@orangeipi:~$ sudo apt install -y openjdk-17-jdk
```

- b. After installation, you can check the version of Java

```
orangeipi@orangeipi:~$ java --version
```

- c. Write the Java version of **hello\_world.java**

```
orangeipi@orangeipi:~$ vim hello_world.java
```

```
public class hello_world
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

- d. Then compile and run **hello\_world.java**

```
orangeipi@orangeipi:~$ javac hello_world.java
```

```
orangeipi@orangeipi:~$ java hello_world
```

```
Hello World!
```

### 3.44.2. Debian Buster System

4) Debian Buster is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

- a. The version of gcc is shown below

```
orangeipi@orangeipi:~$ gcc --version
```

```
gcc (Debian 8.3.0-6) 8.3.0
```

```
Copyright (C) 2018 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

- b. Write the **hello\_world.c** program in C language

```
orangeipi@orangeipi:~$ vim hello_world.c
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```



```
    printf("Hello World!\n");

    return 0;
}
```

c. Then compile and run **hello\_world.c**

```
orangeipi@orangeipi:~$ gcc -o hello_world hello_world.c
orangeipi@orangeipi:~$ ./hello_world
Hello World!
```

5) Debian Buster is installed with Python2 and Python3 by default

a. The specific version of Python is as follows

```
orangeipi@orangeipi:~$ python
Python 2.7.16 (default, Oct 10 2019, 22:02:15)
[GCC 8.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.

>>>

orangeipi@orangeipi:~$ python3
Python 3.7.3 (default, Jan 22 2021, 20:04:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.

>>>
```

b. Write the **hello\_world.py** program in the Python language

```
orangeipi@orangeipi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello\_world.py** is as follows

```
orangeipi@orangeipi:~$ python hello_world.py
Hello World!
orangeipi@orangeipi:~$ python3 hello_world.py
Hello World!
```

6) Debian Buster does not install Java compilation tools and runtime environment by default

a. You can install openjdk with the following command, the default version in Debian Buster is openjdk-11

```
orangeipi@orangeipi:~$ sudo apt install -y openjdk-11-jdk
```



- b. After installation, you can check the version of Java

```
orangeipi@orangeipi:~$ java --version
openjdk 11.0.13 2021-10-19
OpenJDK Runtime Environment (build 11.0.13+8-post-Debian-1deb10u1)
OpenJDK 64-Bit Server VM (build 11.0.13+8-post-Debian-1deb10u1, mixed mode)
```

- c. Write the Java version of **hello\_world.java**

```
orangeipi@orangeipi:~$ vim hello_world.java
public class hello_world
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

- d. Then compile and run **hello\_world.java**

```
orangeipi@orangeipi:~$ javac hello_world.java
orangeipi@orangeipi:~$ java hello_world
Hello World!
```

### 3. 44. 3. Ubuntu Jammy system

- 1) Ubuntu Jammy is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

- a. The version of gcc is shown below

```
orangeipi@orangeipi:~$ gcc --version
gcc (Ubuntu 11.2.0-19ubuntu1) 11.2.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

- b. Write the **hello\_world.c** program in C language

```
orangeipi@orangeipi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");
```



```
    return 0;  
}
```

c. Then compile and run **hello\_world.c**

```
orangeipi@orangeipi:~$ gcc -o hello_world hello_world.c
```

```
orangeipi@orangeipi:~$ ./hello_world
```

```
Hello World!
```

2) Ubuntu Jammy has Python3 installed by default

a. The specific version of Python3 is as follows

```
orangeipi@orangeipi:~$ python3
```

```
Python 3.10.4 (main, Apr 2 2022, 09:04:19) [GCC 11.2.0] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>>
```

b. Write the **hello\_world.py** program in the Python language

```
orangeipi@orangeipi:~$ vim hello_world.py
```

```
print('Hello World!')
```

c. The result of running **hello\_world.py** is as follows

```
orangeipi@orangeipi:~$ python3 hello_world.py
```

```
Hello World!
```

3) Ubuntu Jammy does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk-18

```
orangeipi@orangeipi:~$ sudo apt install -y openjdk-18-jdk
```

b. After installation, you can check the version of Java

```
orangeipi@orangeipi:~$ java --version
```

```
openjdk 18-ea 2022-03-22
```

```
OpenJDK Runtime Environment (build 18-ea+36-Ubuntu-1)
```

```
OpenJDK 64-Bit Server VM (build 18-ea+36-Ubuntu-1, mixed mode, sharing)
```

c. Write the Java version of **hello\_world.java**

```
orangeipi@orangeipi:~$ vim hello_world.java
```

```
public class hello_world
```

```
{
```

```
    public static void main(String[] args)
```



```
{  
    System.out.println("Hello World!");  
}  
}
```

d. Then compile and run **hello\_world.java**

```
orangeipi@orangeipi:~$ javac hello_world.java
```

```
orangeipi@orangeipi:~$ java hello_world
```

```
Hello World!
```

### 3.44.4. Ubuntu Focal system

4) Ubuntu Focal is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is shown below

```
orangeipi@orangeipi:~$ gcc --version
```

```
gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
```

```
Copyright (C) 2019 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR  
PURPOSE.
```

b. Write the **hello\_world.c** program in C language

```
orangeipi@orangeipi:~$ vim hello_world.c
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    printf("Hello World!\n");
```

```
    return 0;
```

```
}
```

c. Then compile and run **hello\_world.c**

```
orangeipi@orangeipi:~$ gcc -o hello_world hello_world.c
```

```
orangeipi@orangeipi:~$ ./hello_world
```

```
Hello World!
```

5) Ubuntu Focal has Python3 installed by default

a. The specific version of Python3 is as follows



```
orangeipi@orangeipi:~$ python3
Python 3.8.10 (default, Sep 28 2021, 16:10:42)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- b. Write the **hello\_world.py** program in the Python language

```
orangeipi@orangeipi:~$ vim hello_world.py
print('Hello World!')
```

- c. The result of running **hello\_world.py** is as follows

```
orangeipi@orangeipi:~$ python3 hello_world.py
Hello World!
```

6) Ubuntu Focal does not install Java compilation tools and runtime environment by default

- a. You can install openjdk-17 using the following command

```
orangeipi@orangeipi:~$ sudo apt install -y openjdk-17-jdk
```

- b. After installation, you can check the version of Java

```
orangeipi@orangeipi:~$ java --version
openjdk 17.0.2 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-120.04, mixed mode, sharing)
```

- c. Write the Java version of **hello\_world.java**

```
orangeipi@orangeipi:~$ vim hello_world.java
public class hello_world {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

- d. Then compile and run **hello\_world.java**

```
orangeipi@orangeipi:~$ javac hello_world.java
orangeipi@orangeipi:~$ java hello_world
Hello World!
```



### 3. 44. 5. Ubuntu Bionic system

1) Ubuntu Bionic is installed with the gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board.

a. The version of gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu/Linaro 7.5.0-3ubuntu1~18.04) 7.5.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

b. Write the **hello\_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");

    return 0;
}
```

c. Then compile and run **hello\_world.c**

```
root@orangepi:~# gcc -o hello_world hello_world.c
root@orangepi:~# ./hello_world
Hello World!
```

2) Ubuntu Bionic has Python3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.6.9 (default, Oct  8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello\_world.py** program in the Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```



- c. The result of running **hello\_world.py** is as follows

```
orangeipi@orangeipi:~$ python3 hello_world.py  
Hello World!
```

3) Ubuntu Bionic does not install Java compilation tools and runtime environment by default

- a. You can install openjdk with the following command, the default version in Debian Buster is openjdk-17

```
orangeipi@orangeipi:~$ sudo apt install -y openjdk-17-jdk
```

- b. After installation, you can check the version of Java

```
orangeipi@orangeipi:~$ java --version  
openjdk 17.0.2 2022-01-18  
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-118.04)  
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-118.04, mixed mode, sharing)
```

- c. Write the Java version of **hello\_world.java**

```
orangeipi@orangeipi:~$ vim hello_world.java  
public class hello_world  
{  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

- d. Then compile and run **hello\_world.java**

```
orangeipi@orangeipi:~$ javac hello_world.java  
orangeipi@orangeipi:~$ java hello_world  
Hello World!
```

### 3. 45. How to install kernel header files

1) There are two ways to obtain kernel header files:

- a. Method 1: Download the compiled header file deb package from Google cloud disk
- a) Google cloud disk download link is as follows:



<https://drive.google.com/drive/folders/1fie2GcF-6zfWMqBDufnLUV3B9qMnbUkO?usp=sharing>

- b) After opening the above link, first find the folder named **linux-headers**



- c) Then enter the linux-headers folder and download the deb package of the header file corresponding to the kernel

**The version number of the deb package of the kernel header file may be updated, depending on what is actually seen.**

| kernel version | The deb package name corresponding to the kernel header file   |
|----------------|----------------------------------------------------------------|
| Linux4.9       | <a href="#">linux-headers-legacy-sun50iw6_2.2.x_arm64.deb</a>  |
| Linux5.10.75   | <a href="#">linux-headers-current-sun50iw6_2.2.x_arm64.deb</a> |
| Linux5.10.76   | <a href="#">linux-headers-current-sun50iw6_3.0.x_arm64.deb</a> |
| Linux5.16.17   | <a href="#">linux-headers-next-sun50iw6_3.0.x_arm64.deb</a>    |

- b. Method 2: Using orangepi-build to compile the kernel source code will automatically generate the deb package of the kernel header file. For the specific method, please refer to the instructions in the two subsections [5.4 Compiling the Linux Kernel](#) and [6.4 Compiling the Linux Kernel](#)
- 2) Then upload the kernel header file deb package to the Linux system of the development board
- 3) Then use the following command to install the kernel header file deb package

**The name of the deb package of the kernel header file needs to be replaced with the actual name, please do not copy it.**

**If there is a warning message printed during the installation process, please ignore it directly, it will not affect the use.**

```
root@orangepi:~# dpkg -i linux-headers-next-sun50iw6_3.0.4_arm64.deb
```

- 4) After installation, you can see the folder where the kernel header files are located under **/usr/src**

```
root@orangepi:~# ls /usr/src
linux-headers-5.16.17-sun50iw6
```



5) Then you can write a hello kernel module to test the kernel header file

a. First write the code of the hello kernel module as follows:

```
root@orangepi:~# vim hello.c
#include <linux/init.h>
#include <linux/module.h>

static int hello_init(void)
{
    printk("Hello Orange Pi -- init\n");

    return 0;
}
static void hello_exit(void)
{
    printk("Hello Orange Pi -- exit\n");

    return;
}

module_init(hello_init);
module_exit(hello_exit);

MODULE_LICENSE("GPL");
```

b. Then write the Makefile for compiling the hello kernel module, as follows:

```
root@orangepi:~# vim Makefile
ifneq ($(KERNELRELEASE),)
obj-m:=hello.o
else
KDIR :=/lib/modules/$(shell uname -r)/build
PWD   :=$(shell pwd)
all:
    make -C $(KDIR) M=$(PWD) modules
clean:
    rm -f *.ko *.o *.mod.o *.mod *.symvers *.cmd *.mod.c *.order
```



```
endif
```

- c. Then use the **make** command to compile the hello kernel module. The output of the compilation process is as follows:

```
root@orangeipi:~# make
make -C /lib/modules/5.16.17-sun50iw6/build M=/root modules
make[1]: Entering directory '/usr/src/linux-headers-5.16.17-sun50iw6'
  CC [M]  /root/hello.o
  MODPOST /root/Module.symvers
  CC [M]  /root/hello.mod.o
  LD [M]  /root/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.16.17-sun50iw6'
```

- d. After compiling, the **hello.ko** kernel module will be generated

```
root@orangeipi:~# ls *.ko
hello.ko
```

- e. Use the **insmod** command to insert the **hello.ko** kernel module into the kernel

```
root@orangeipi:~# insmod hello.ko
```

- f. Then use the **dmesg** command to view the output of the **hello.ko** kernel module. If you can see the following output, the **hello.ko** kernel module is loaded correctly

```
root@orangeipi:~# dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
```

- g. Use the **rmmmod** command to uninstall the **hello.ko** kernel module

```
root@orangeipi:~# rmmmod hello
root@orangeipi:~# dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
[ 3173.800892] Hello Orange Pi -- exit
```

### 3. 46. Shutdown and restart methods

- 1) During the operation of the Linux system, if the power is directly unplugged, some data may be lost in the file system. It is recommended to use the **poweroff** command to shut down the Linux system of the development board before powering off, and then unplug the power.

```
orangeipi@orangeipi:~$ sudo poweroff
```



- 2) In addition, the development board has a power button, and you can also short press the power button on the development board to shut down



- 3) After shutdown, press and hold the power button on the development board for 2 seconds to turn it on



- 4) The command to restart the linux system is

```
orangeipi@orangeipi:~$ sudo reboot
```



## 4. Instructions for use of Android TV system

### 4. 1. Supported Android TV Versions

| Android version               | kernel version  |
|-------------------------------|-----------------|
| Android 9.0 <b>TV</b> version | <b>linux4.9</b> |

### 4. 2. Android 9.0 TV function adaptation

| Function           | State |
|--------------------|-------|
| HDMI video         | OK    |
| HDMI audio         | OK    |
| USB2.0             | OK    |
| USB3.0             | OK    |
| TF card boot       | OK    |
| eMMC starts        | OK    |
| network card       | OK    |
| Infrared           | OK    |
| WIFI               | OK    |
| WIFI hotspot       | OK    |
| Bluetooth          | OK    |
| Headphone Audio    | OK    |
| TV-OUT             | OK    |
| USB camera         | OK    |
| LED lights         | OK    |
| Temperature Sensor | OK    |
| Mali GPU           | OK    |
| Video codec        | OK    |
| power button       | OK    |
| ADB debugging      | OK    |

#### **4. 3. On-board LED light display description**

|                                  | yellow light | red light |
|----------------------------------|--------------|-----------|
| u-boot startup phase             | off          | on        |
| The kernel boots into the system | on           | off       |
| GPIO port                        | PL7          | PL4       |

#### **4.4. How to return to the previous interface on Android**

- 1) We generally use the mouse and keyboard to control the Android system of the development board. When entering some interfaces and need to return to the previous interface or desktop, you can only return by pressing **the right mouse button**, and the keyboard cannot be returned.
  - 2) If you buy the infrared remote control (other remote control is not available) and expansion board supporting the development board, after inserting the expansion board into the development board, you can return to the upper menu through the back button in the remote control, and the position of the back button is shown below



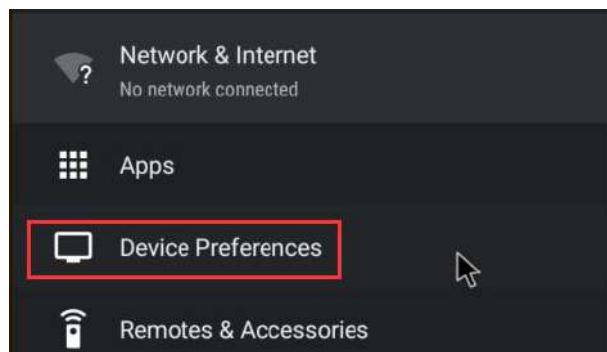
#### 4.5. How to use ADB

#### **4. 5. 1. Enable USB debugging option**

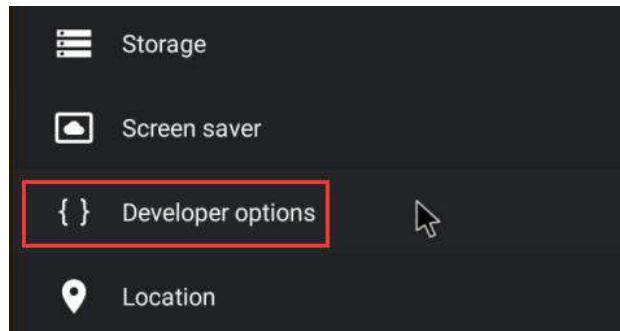
- 1) First select **Settings**



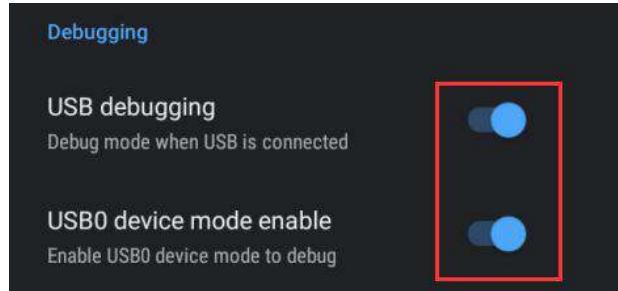
2) Then select **Device Preferences**



3) Then select **Developer options**



4) Then turn on **USB debugging** and **USB device mode enable**



#### 4. 5. 2. Using network connection adb debugging

1) Use network adb to connect the computer and the development board without using the data cable of the USB interface, but to communicate through the network, so first make sure that the wired or wireless network of the development board has been connected, and then obtain the IP address of the development board, will be used later

2) Make sure the **USB debugging option** is turned on

3) Make sure that the **service.adb.tcp.port** of the Android system is set to the port number 5555 (already set by default)

```
petrel-p1:/ # getprop | grep "adb.tcp"  
[service.adb.tcp.port]: [5555]
```

5) If **service.adb.tcp.port** is not set, you can use the following command to set the port number of network adb

```
petrel-p1:/ # setprop service.adb.tcp.port 5555  
petrel-p1:/ # stop adbd  
petrel-p1:/ # start adbd
```

6) Install adb tool on Ubuntu PC

```
test@test:~$ sudo apt update  
test@test:~$ sudo apt install -y adb
```

7) Then connect network adb on Ubuntu PC

```
test@test:~$ adb connect 192.168.1.xxx:5555  (The IP address needs to be changed  
to the IP address of the development board)  
* daemon not running; starting now at tcp:5037
```



```
* daemon started successfully  
connected to 192.168.1.xxx:5555
```

```
test@test:~$ adb devices  
List of devices attached  
192.168.1.xxx:5555      device
```

8) Then you can log in to the android system through adb shell on the Ubuntu PC

```
test@test:~$ adb shell  
petrel-p1:/ #
```

#### 4. 5. 3. Use the data cable to connect adb debugging

1) First make sure the **USB debugging** option is turned on

2) Then you need to connect the development board to the USB interface of the computer using a data cable with a double-ended USB interface. The interface to be connected to the development board is shown in the right figure below.



3) Install adb tool on Ubuntu PC

```
test@test:~$ sudo apt update  
test@test:~$ sudo apt install -y adb
```

4) Check to identify the ADB device

```
test@test:~$ adb devices  
List of devices attached  
* daemon not running; starting now at tcp:5037  
* daemon started successfully  
cc001421c7028941ed1  device
```

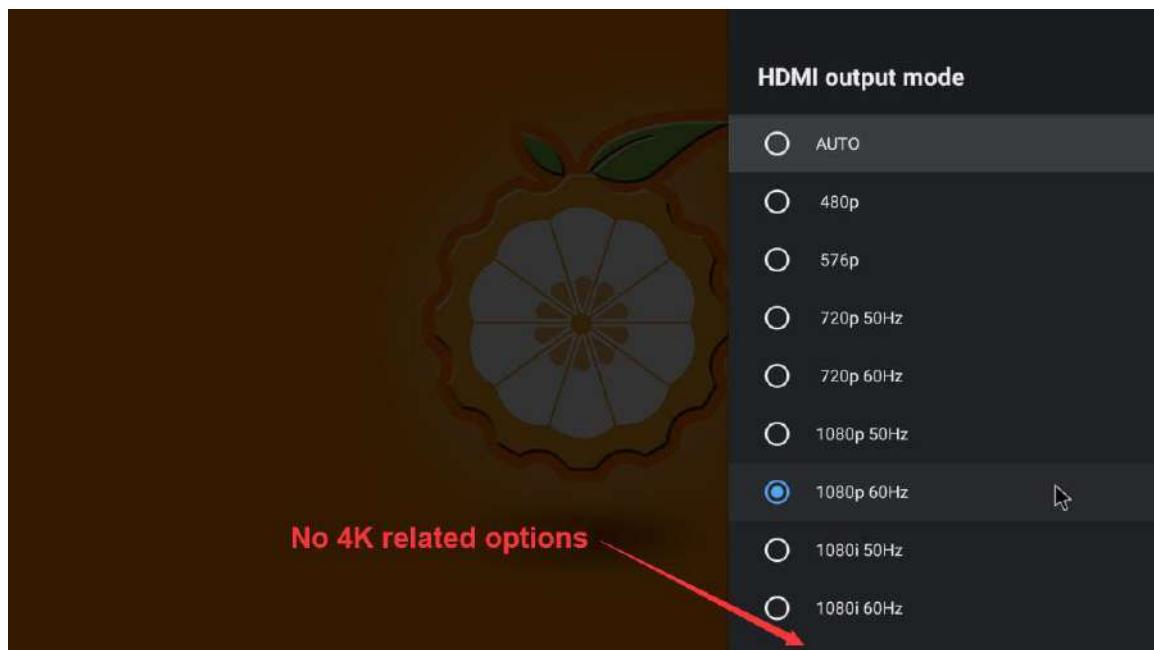


- 5) Then you can log in to the android system through adb shell on the Ubuntu PC

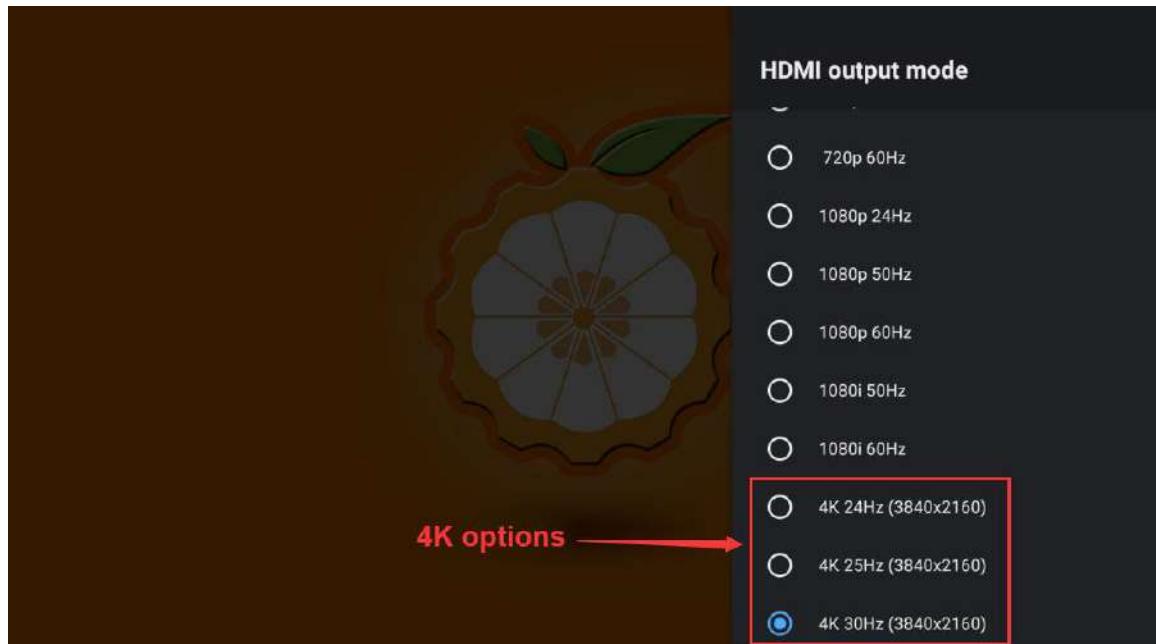
```
test@test:~$ adb shell  
petrel-p1:/ #
```

#### 4. 6. HDMI 4K Display Instructions

- 1) If you connect the HDMI of Orange Pi 3 LTS to a TV or monitor that does not support 4K, you will not see 4K related options when you check the resolutions supported by HDMI in the settings



- 2) Only when the HDMI of Orange Pi 3 LTS is connected to a TV or monitor that supports 4K, the 4K-related options can be seen in the resolutions supported by HDMI



#### 4. 7. HDMI to VGA display test

- 3) First you need to prepare the following accessories
  - a. HDMI to VGA converter

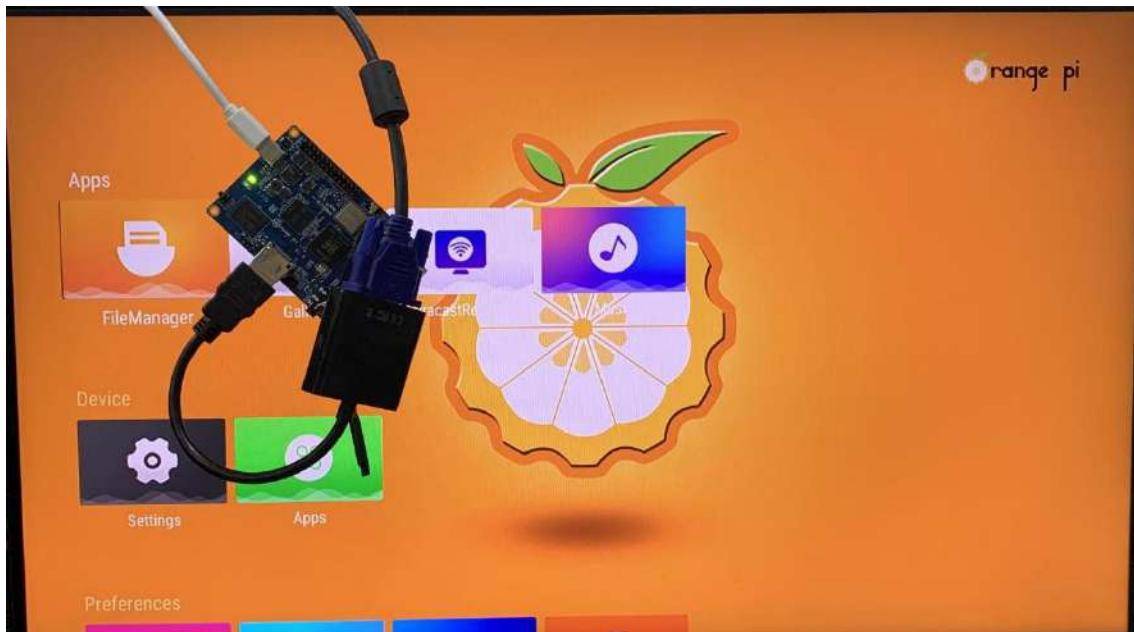


- b. A VGA cable



- c. A monitor or TV that supports VGA interface

- 4) HDMI to VGA display test as shown below



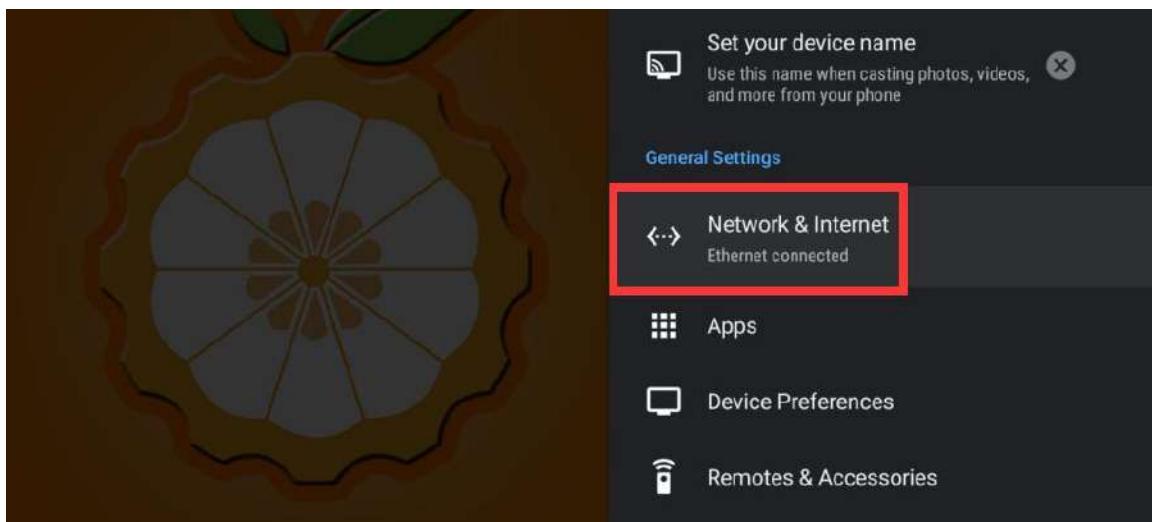
When using HDMI to VGA display, the development board and the Android system of the development board do not need to do any settings, as long as the Micro HDMI interface of the development board can display normally. So if there is a problem with the test, please check the HDMI to VGA converter, VGA cable and monitor if there is any problem

#### 4. 8. Wi-Fi connection method

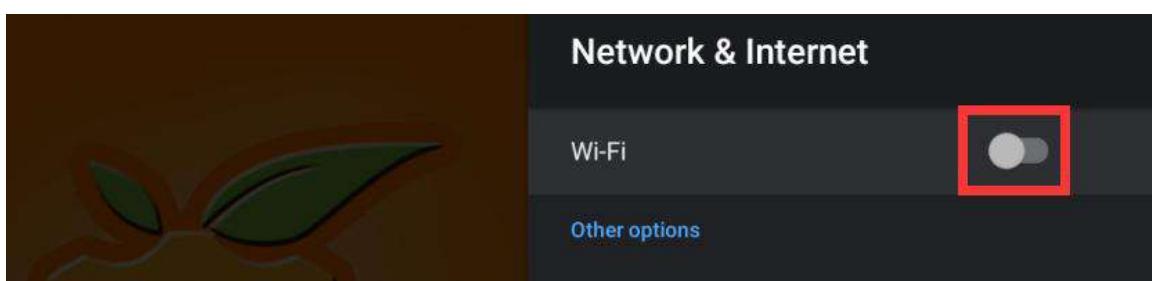
- 1) First select **Settings**



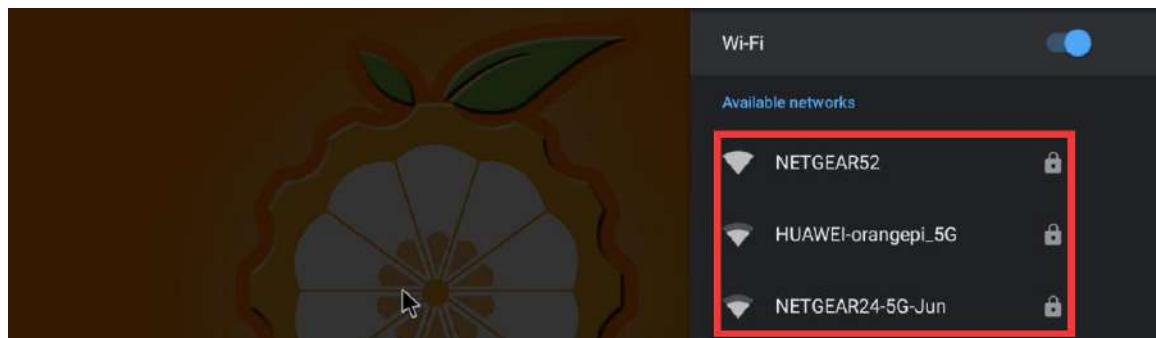
2) Then select **Network & Internet**



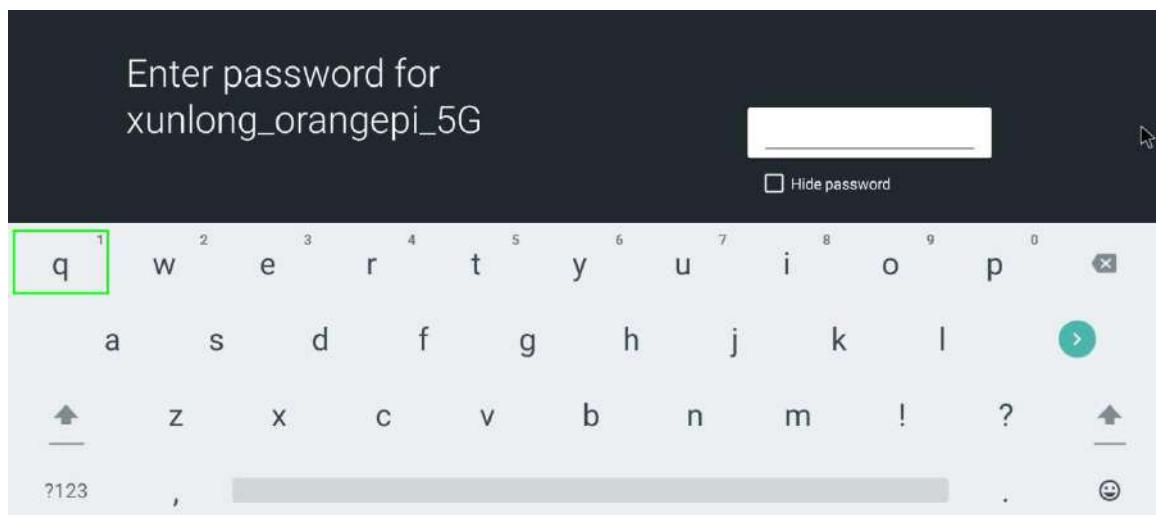
3) Then turn on WI-FI



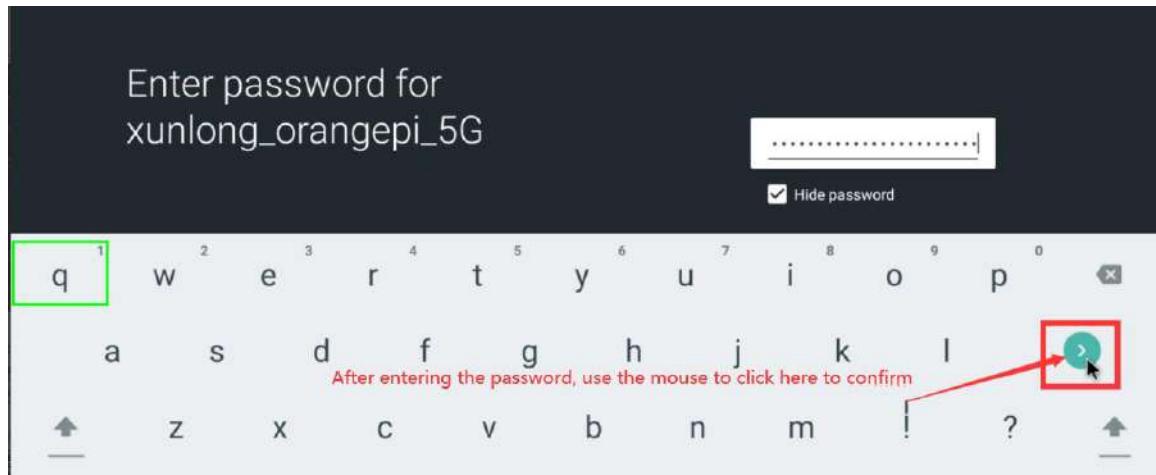
4) After turning on WI-FI, you can see the searched signal under **Available networks**



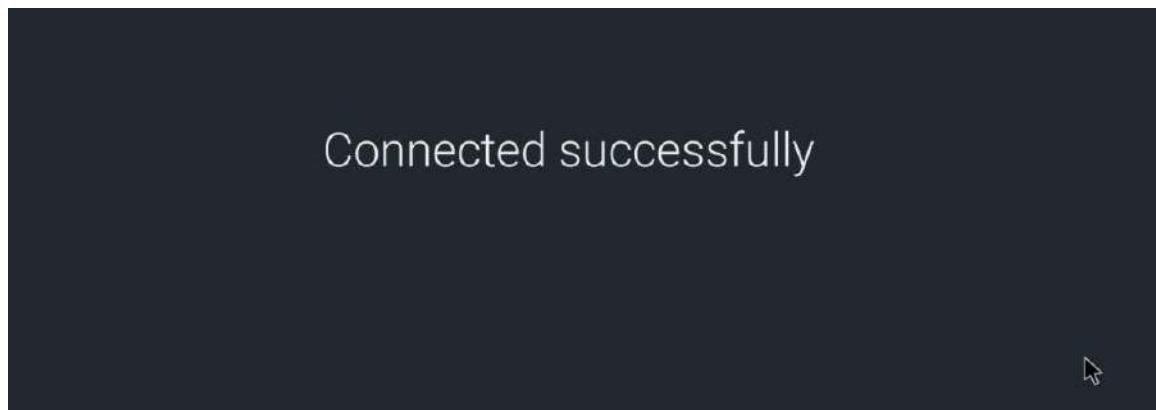
- 5) After selecting the WI-FI you want to connect to, the password input interface shown in the figure below will pop up



- 6) Then use the keyboard to enter the password corresponding to WI-FI, and then use the mouse to click the Enter button in the virtual keyboard to start connecting to WI-FI



- 7) The display after the WI-FI connection is successful is shown in the figure below



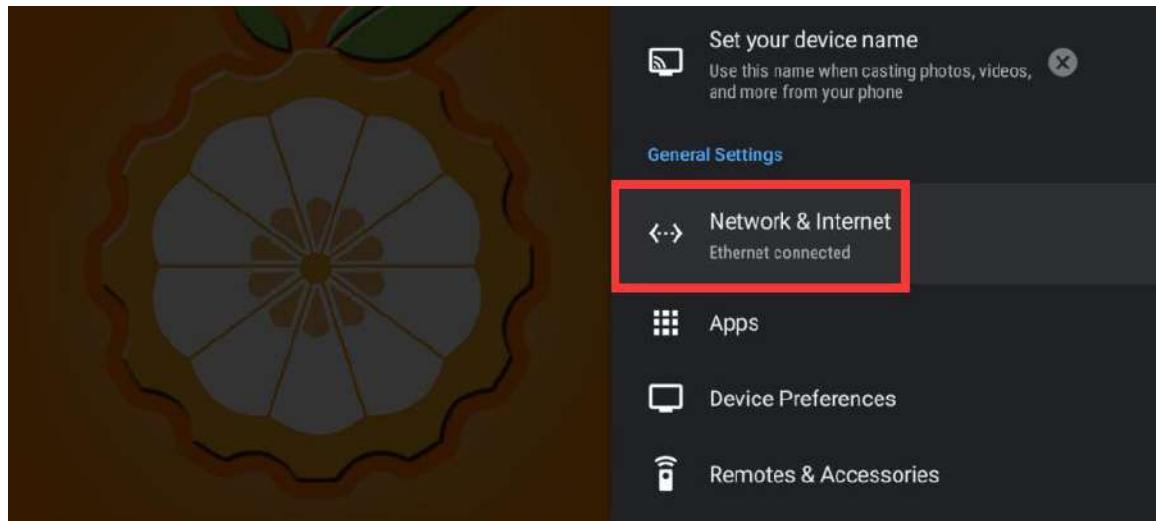
#### 4. 9. How to use WI-FI hotspot

1) First, please make sure that the Ethernet port is connected to the network cable and can access the Internet normally

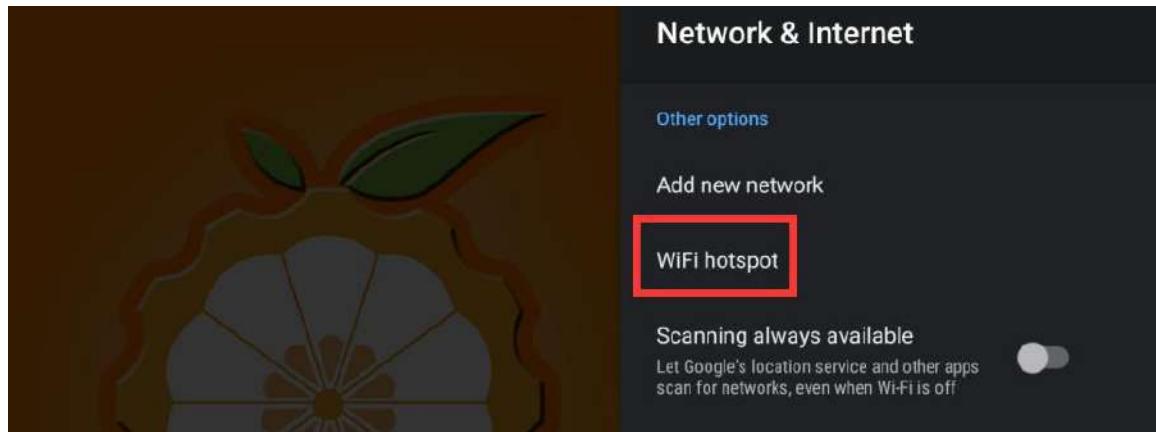
2) Then select **Settings**



3) Then select **Network & Internet**



4) Then select **WIFI hotspot**



5) Then open **Hotspot Enable**, you can also see the name and password of the generated hotspot in the figure below, remember them and use them when connecting to the hotspot (if you need to modify the name and password of the hotspot, you need to close Hotspot Enable first, then modify it)



- 6) At this point, you can take out your mobile phone. If everything is normal, you can find the WIFI hotspot with the same name (**AndroidAP\_7284 here**) displayed under the **Hotspot name** in the above picture in the WI-FI list searched by the mobile phone. Then you can click on **AndroidAP\_7284** to connect to the hotspot. The password can be seen under the **Hotspot password** in the picture above.



- 7) After the connection is successful, it will display as shown in the figure below (different mobile phone interfaces will be different, the specific interface is subject to your mobile phone display). At this point, you can open a web page on your mobile phone to see if you can access the Internet. If the web page can be opened normally, it means that the **WI-FI Hotspot** of the development board can be used normally.



#### 4. 10. Bluetooth connection method

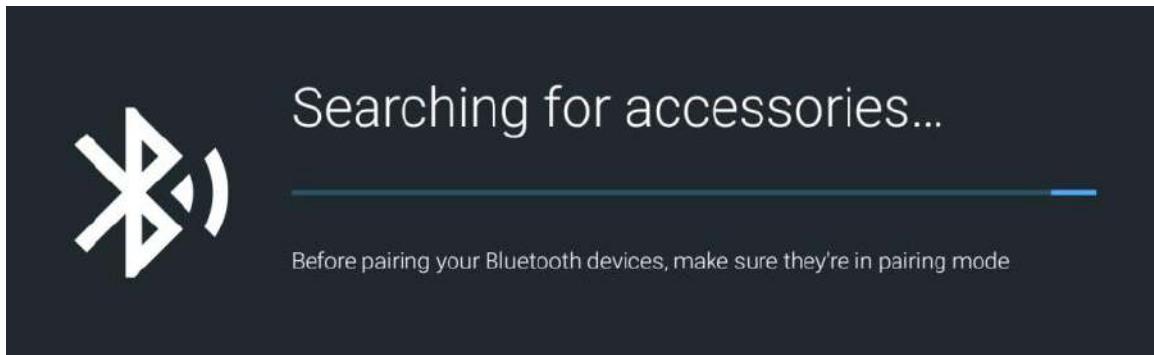
1) First select **Settings**



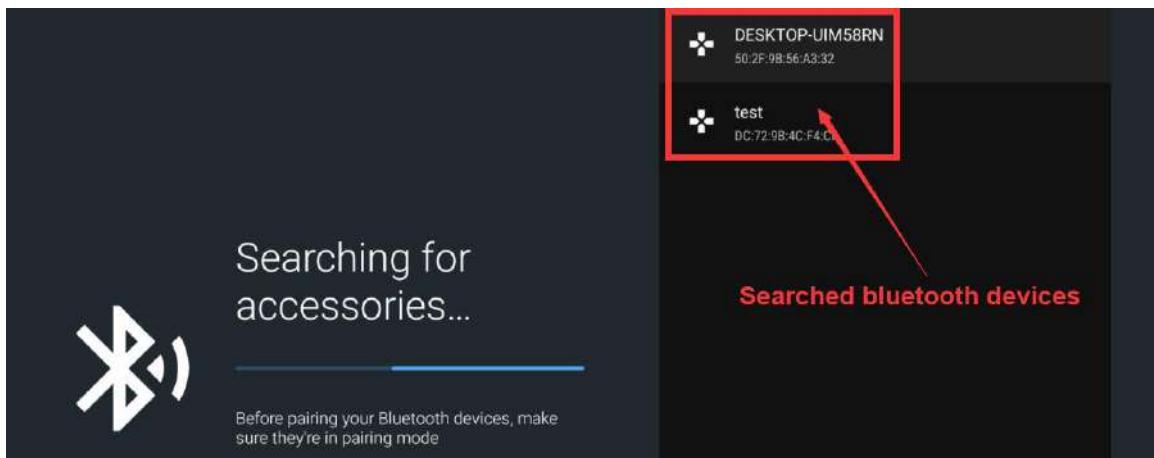
2) Then select **Remotes & Accessories**



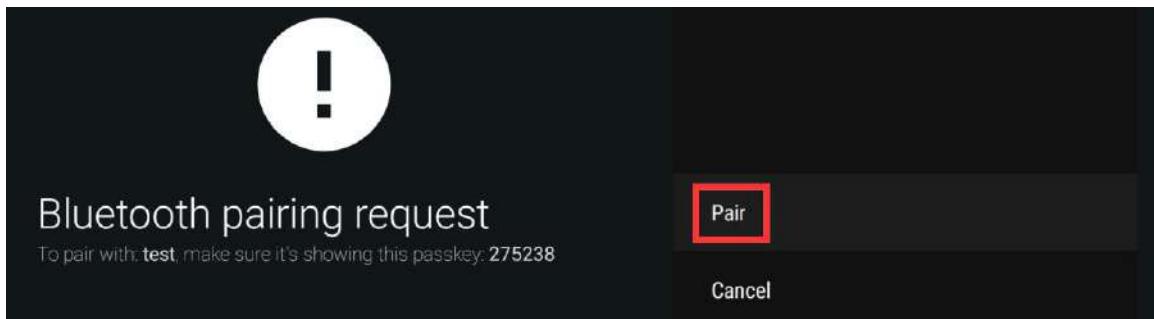
3) Then the system will start to search for the surrounding bluetooth devices



- 4) After searching for a Bluetooth device, the following interface will be displayed, and the list on the right is the searched Bluetooth device



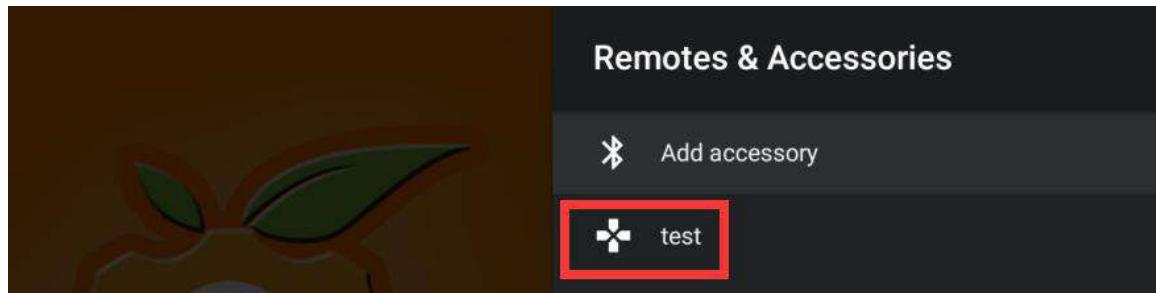
- 5) Then click on the Bluetooth device you want to connect to start pairing. When the following interface pops up, please use the mouse to select the **Pair** option



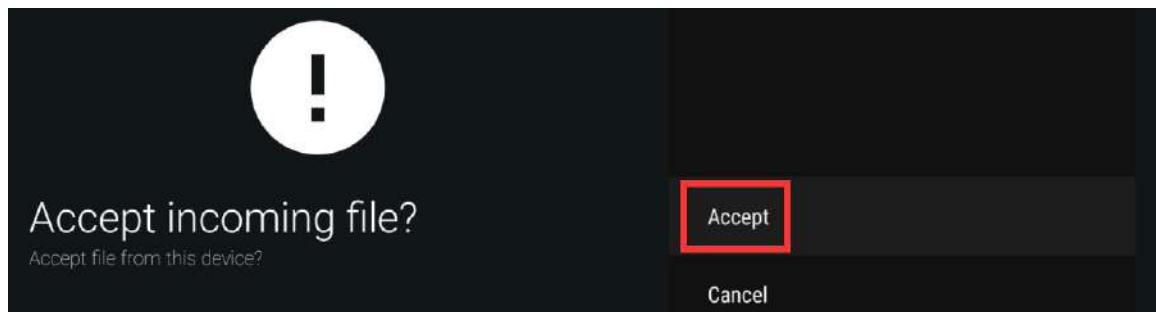
- 6) The test here is the Bluetooth configuration process of the development board and the **Android mobile phone**. At this time, the following confirmation interface will pop up on the mobile phone. After clicking the pairing button on the mobile phone, the pairing process will start.



7) After the pairing is completed, open **Remotes & Accessories** to see the paired Bluetooth devices

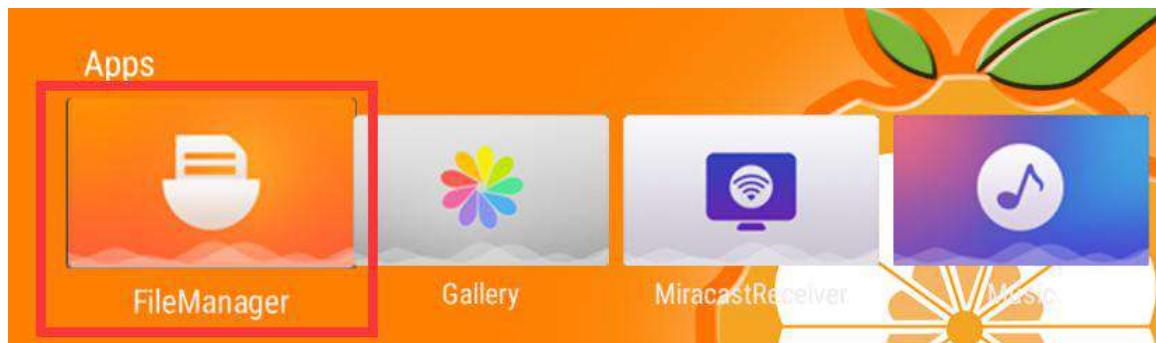


8) At this time, you can use the mobile phone Bluetooth to send a picture to the development board. After sending, you can see the following confirmation interface in the Android system of the development board, and then click **Accept** to start receiving the picture sent by the mobile phone.



9) The pictures received by the development board Android system Bluetooth can be seen in the following folder

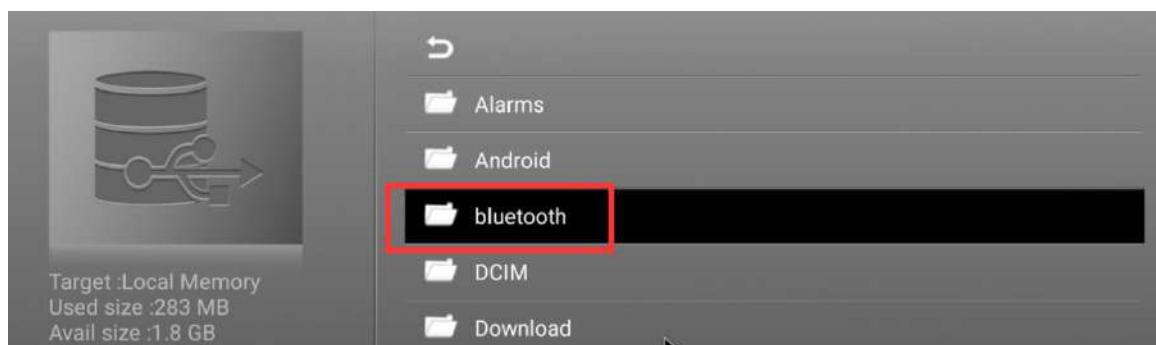
a. First select **FileManager**



b. Then select **Local Memory**



c. Then you can see a folder named bluetooth. The pictures received by the development board **bluetooth** are saved in this folder. You can see this folder when you open this folder.



#### 4.11. How to use the USB camera

1) Insert a USB camera into the USB interface of the development board, and then confirm that the kernel module related to the USB camera has been loaded normally

```
petrel-p1:/ # lsmod
Module           Size  Used by
sprdw1_ng       381753  0
sprdbt_tty      29622   2
uwe5622_bsp_sdio 260394  2 sprdw1_ng,sprdbt_tty
```



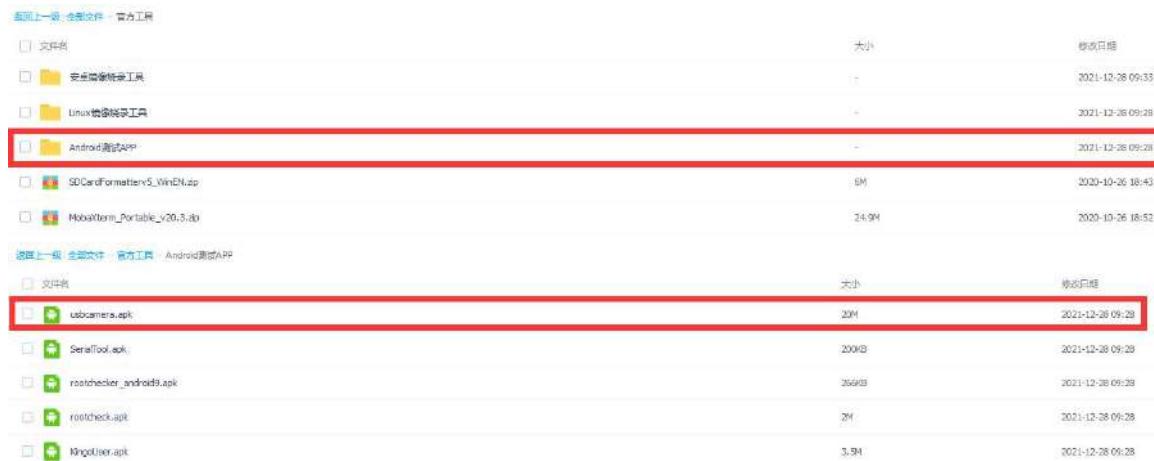
|                          |               |                                  |
|--------------------------|---------------|----------------------------------|
| <b>uvcvideo</b>          | <b>86387</b>  | <b>0</b>                         |
| <b>videobuf2_v4l2</b>    | <b>19657</b>  | <b>1 uvcvideo</b>                |
| <b>videobuf2_vmalloc</b> | <b>7119</b>   | <b>1 uvcvideo</b>                |
| <b>videobuf2_memops</b>  | <b>2671</b>   | <b>1 videobuf2_vmalloc</b>       |
| <b>videobuf2_core</b>    | <b>38644</b>  | <b>2 uvcvideo,videobuf2_v4l2</b> |
| <b>sunxi_ir_rx</b>       | <b>11332</b>  | <b>0</b>                         |
| <b>mali_kbase</b>        | <b>392286</b> | <b>22</b>                        |

- 2) If the USB camera is recognized normally, the corresponding video device node will be generated under **/dev**

```
petrel-pi:/ # ls /dev/video*
/dev/video0
petrel-pi:/ # ls /sys/class/video4linux/ -lh
total 0
lrwxrwxrwx 1 root root 0 2020-12-08 15:58 video0 -> ../../devices/soc/5311000.ehci3-controller/usb2/2-1/2-1:1.0/video4linux/video0
petrel-pi:/ #
```

- 3) Then make sure the adb connection between the Ubuntu PC and the development board is normal

- 4) Download the USB camera test APP in the official tool on the data download page of Orange Pi 3 LTS



- 5) Then use the adb command to install the USB camera test APP to the Android system, of course, you can also use the U disk copy method to install

```
test@test:~$ adb install usbcamera.apk
```

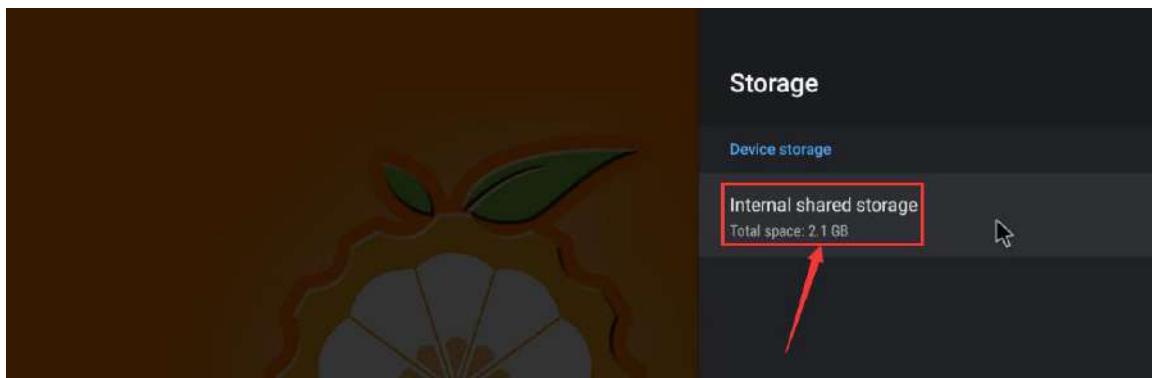
- 6) After installation, you can see the startup icon of the USB camera in the Android desktop



- 7) Then double-click to open the USB camera APP to see the output video of the USB camera

#### 4. 12. eMMC storage space description

- 1) The capacity of the onboard eMMC of the development board is 8GB, but the space that can be used to store other data in the Android system is only about 2GB. The rest of the space is occupied by the Android system itself. Please pay special attention to this.



#### 4. 13. Test method of serial port in 26pin interface

- 1) It can be seen from the schematic diagram of the 26pin interface that the uart available for the development board is uart3



- 2) After entering the Android system, please confirm whether there is a uart3 device node under **/dev**

a. The command to view with adb is as follows

```
test@test:~$ adb connect 192.168.1.82
connected to 192.168.1.82:5555
test@test:~$ adb shell ls /dev/ttyS3
/dev/ttyS3
```

b. The command to view using the debug serial port is as follows

```
petrel-p1:/ # ls /dev/ttyS3
/dev/ttyS3
```

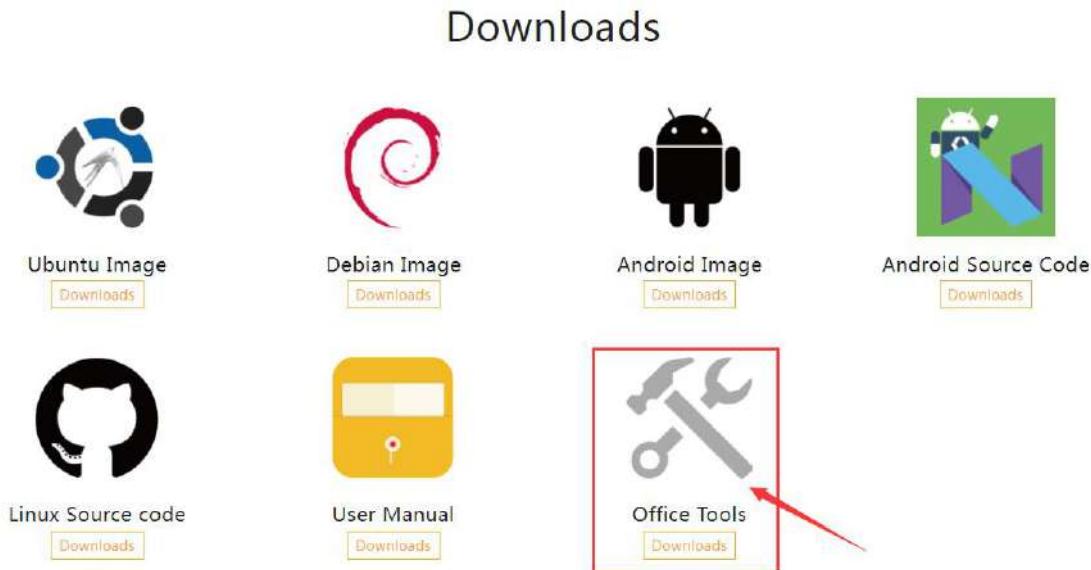
- 3) Then start to test the uart3 interface, first use a 2.54mm DuPont cable to short the rx and tx of the uart3 interface to be tested

|        |                       |
|--------|-----------------------|
|        | uart3                 |
| tx pin | Corresponds to pin 13 |
| rx pin | Corresponds to pin 11 |





- 4) Then download the serial port debugging assistant APP - **SerialTestTool.apk**, this APP can be downloaded in the official tool of Orange Pi 3 LTS



| Name                         | Owner |
|------------------------------|-------|
| Android Tool                 | me    |
| Linux Tool                   | me    |
| Android test APP             | me    |
| SDCardFormatterv5_WinEN.zip  | me    |
| MobaXterm_Portable_v20.3.zip | me    |
| rootchecker_android9.apk     | me    |
|                              |       |
| usbcamera.apk                | 20M   |
| SerialTool.apk               | 200KB |
| SerialTestTool.apk           | 5.5M  |
| rootchecker_android9.apk     | 266KB |

- 5) Then install **SerialTestTool.apk** The command to install using adb is as follows. If you don't use adb, you can use U disk to copy and install

```
test@test:~$ adb install SerialTestTool.apk
```

- 6) Then open the serial port debugging assistant APP, the installed position of the serial port debugging assistant is shown in the figure below



- 7) Then set the serial port debugging assistant
- Select the serial port number **/dev/ttyS3**
  - Baud rate selection **115200**
  - Finally, remember to click **the close button** in the upper right corner to open the serial port



- 8) Then you can send a string to test the loopback function of the serial port to see if the serial port can receive the sent string



- 9) As shown in the figure below, if the serial port debugging assistant can receive the sent string, it means that the serial port can be used normally



#### 4. 14. Android system ROOT description

**The Android 9.0 system released by Orange Pi has been ROOT, you can use the following method to test**

- 1) Download **rootchecker\_android9.apk** in the official tool on the Orange Pi 3 LTS data download page

| 返回上一级 全部文件 > 官方工具                   |                              | 大小    | 修改日期             |
|-------------------------------------|------------------------------|-------|------------------|
| <input type="checkbox"/>            | 文件夹                          |       |                  |
| <input type="checkbox"/>            | 安卓系统转录工具                     |       | 2021-12-28 09:33 |
| <input type="checkbox"/>            | Linux命令转录工具                  |       | 2021-12-28 09:28 |
| <input checked="" type="checkbox"/> | Android测试APP                 |       | 2021-12-28 09:28 |
| <input type="checkbox"/>            | SD CardFormatterv5_WinEN.apk | 6M    | 2020-10-26 18:43 |
| <input type="checkbox"/>            | MobiTerm_Portable_v20.3.8b   | 24.9M | 2020-10-26 18:52 |
| 返回上一级 全部文件 > 官方工具 > Android测试APP    |                              |       |                  |
| <input type="checkbox"/>            | 文件夹                          | 大小    | 修改日期             |
| <input type="checkbox"/>            | usbcamera.apk                | 29M   | 2021-12-28 09:28 |
| <input type="checkbox"/>            | SerialTool.apk               | 209KB | 2021-12-28 09:28 |
| <input checked="" type="checkbox"/> | rootchecker_android9.apk     | 266KB | 2021-12-28 09:28 |
| <input type="checkbox"/>            | rootcheck.apk                | 2M    | 2021-12-28 09:28 |
| <input type="checkbox"/>            | KingUser.apk                 | 3.9M  | 2021-12-28 09:28 |

- 2) Then make sure the adb connection between the Ubuntu PC and the development board is normal
- 3) Then use the adb command to install rootcheck\_android9.apk to the Android system, of course, you can also use the U disk copy method to install

```
test@test:~$ adb install rootcheck_android9.apk
```

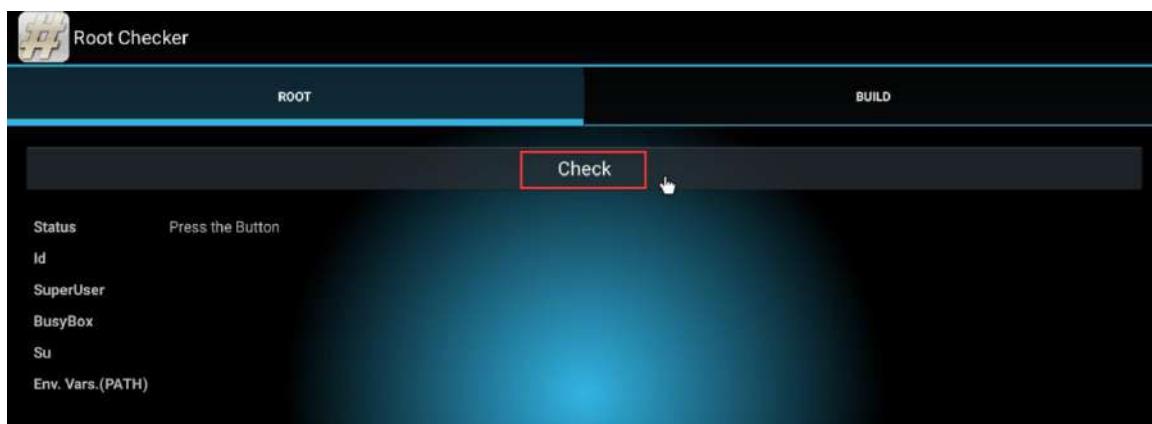
- 4) After installation, you can see the startup icon of the ROOT test tool on the Android



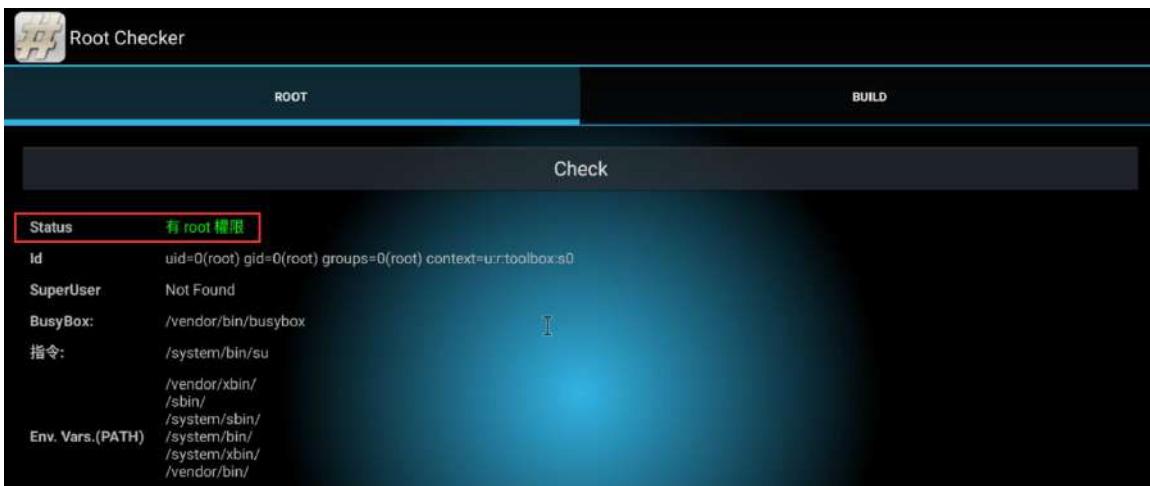
desktop



- 5) After opening the **ROOT test tool**, the display interface is shown in the figure below, click Check to start testing whether the system has ROOT permission



- 6) The display after clicking Check is as follows, you can see that the Android system has obtained ROOT permission



## 4. 15. Some Android APP installation instructions

The Allwinner H6 chip is mainly used in TV boxes, so the Android provided by Allwinner is also the TV version of the Android system. If you need to install an APP, you can first install an application market such as Dangbei, which is specially created for smart TVs, and then install the required APP through Dangbei.

In addition, it should be noted that many mobile APPs used on mobile phones cannot be installed normally in the Android TV system. Although some can be installed normally, there are many problems in use. Before installing the software, it is best to check whether the manufacturer of the software provides an installation package for the TV version. For example, Tencent Video or iQiyi both provide an installation package for the TV version of the TV. Compared with the mobile APP, the TV version is much smaller in size, consumes much less resources, and is smoother to use.

### 4. 15. 1. Browser Installation Instructions

- 1) The first thing to note is that the Android system provided by Allwinner is the TV version of the Android system, so the APP installed on the mobile terminal of Firefox or Chrome will be very stuck. If you need to install a browser, please select the TV version of the installation package, but there are very few TV version browsers on the market. At present, the TV version of the browser previously developed by Firefox can be found better, although it has stopped updating (open A warning message will be displayed at the top after the APP), but there is no problem in using it and it is relatively smooth.
- 2) Firefox TV version APP can be downloaded from [the official tool](#)



The screenshot shows a file manager interface with the following items:

- Android源码 (更新: 2020-11-04) - Download Now
- Linux 源码 (更新: 2020-11-04) - Download Now
- Ubuntu镜像 (更新: 2020-11-04) - Download Now
- Debian镜像 (更新: 2020-11-04) - Download Now
- 用户手册和原理图 (更新: 2020-11-04) - Download Now
- 官方工具 (更新: 2020-11-04) - Download Now

Below these are several APK files listed in a table:

| 文件名                        | 大小    | 更新时间             |
|----------------------------|-------|------------------|
| Android电视端工具               |       | 2022-03-09 15:21 |
| AndroidAPP                 |       | 2022-04-01 10:00 |
| vcredist_x64.exe           | 4.0M  | 2021-04-25 21:25 |
| REFile.apk                 | 4.4M  | 2020-11-04 13:48 |
| org.mozilla.tv.firefox.apk | 11.3M | 2022-04-01 10:00 |
| bledemo.apk                | 4.1M  | 2020-11-04 13:48 |

3) The source code location of the Firefox TV version browser is as follows

<https://github.com/mozilla-mobile/firefox-tv/releases>

#### 4. 15. 2. Tencent Video Installation Instructions

1) If you need to install Tencent Video to watch TV series and movies, please install the TV version of Tencent Video TV. The Android version of the mobile terminal cannot be used normally. The download address of the TV version of Tencent Video TV is as follows

<http://v.qq.com/download.html>

#### 4. 15. 3. Youku Video Installation Instructions

1) If you need to install Youku to watch TV series and movies, please install the Kumiao TV version of the TV terminal provided by Youku. The Android version of the mobile terminal cannot be used normally. The download address of the Kumiao TV version is as follows

[https://youku.com/product/index?spm=a2ha1.14919748\\_WEBHOME\\_GRAY.uerCenter.5!5~5~5~A](https://youku.com/product/index?spm=a2ha1.14919748_WEBHOME_GRAY.uerCenter.5!5~5~5~A)

#### 4. 15. 4. iQIYI Video Installation Instructions

2) If you need to install iQIYI to watch TV series and movies, please use iQIYI's Kiwi TV version, the download address is as follows

<http://app.iqiyi.com/tv/player/>



#### 4. 15. 5. Installation Instructions

1) The download address is as follows

<https://www.lebo.cn/Download.jsp>

2) In the Android TV system of the development board, you need to install the TV version of LeBao.

乐播投屏正式版



3) Then install LeBao screencasting in the mobile phone, you can cast the screen of the mobile phone to the HDMI display connected to the development board

## 5. Linux SDK - Instructions for using the old version of orangepi-build

### 5. 1. Compilation system requirements

1) The Linux SDK, namely **orangepi-build**, only supports running on a computer with **Ubuntu18.04** installed, so before downloading **orangepi-build**, please make sure that the Ubuntu version installed on your computer is Ubuntu18.04. The command to check the Ubuntu version installed on the computer is as follows. If the Release field



does not display **18.04**, it means that the current Ubuntu version does not meet the requirements. Please change the system before performing the following operations.

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.5 LTS
Release:        18.04
Codename:      bionic
test@test:~$
```

2) If the computer is installed with Windows system and there is no computer with Ubuntu18.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu18.04 virtual machine in the Windows system. But please note, do not compile orangepi-build on the WSL virtual machine, because orangepi-build has not been tested in the WSL virtual machine, so it cannot be guaranteed that orangepi-build can be used normally in WSL, and please do not use the Linux system of the development board using orangepi-build in

3) The download address of the installation image of the Ubuntu18.04 amd64 version is as follows

<https://repo.huaweicloud.com/ubuntu-releases/18.04.6/ubuntu-18.04.6-desktop-amd64.iso>

4) After installing Ubuntu18.04 in the computer or virtual machine, please set the software source of Ubuntu18.04 to Tsinghua source first, otherwise it is easy to make mistakes due to network reasons when installing the software later

a. For the method of replacing Tsinghua source, please refer to the description of this webpage.

<https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/>

b. Note that the Ubuntu version needs to be switched to 18.04



## Ubuntu 镜像使用帮助

Ubuntu 的软件源配置文件是 `/etc/apt/sources.list`。将系统自带的该文件做个备份，将该文件替换为下面内容，即可使用 TUNA 的软件源镜像。

选择你的ubuntu版本: 18.04 LTS

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
```

本镜像仅包含 32/64 位 x86 架构处理器的软件包，在 ARM(arm64, armhf)、PowerPC(ppc64el)、RISC-V(riscv64) 和 S390x 等架构的设备上（对应官方源为ports.ubuntu.com）请使用 [ubuntu-ports 镜像](#)。

c. The content of the `/etc/apt/sources.list` file that needs to be replaced is

```
test@test:~$ sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak
test@test:~$ sudo vim /etc/apt/sources.list

# The source image is commented by default to improve the speed of apt update, you can
uncomment it yourself if necessary
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe
multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted
universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted
universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted
universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted
universe multiverse

# Pre-release software sources, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted
```



```
universe multiverse
```

```
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted  
universe multiverse
```

- d. After the replacement, you need to update the package information and ensure that no errors are reported

```
test@test:~$ sudo apt update
```

- e. In addition, since the source code such as the kernel and U-boot are stored on GitHub, it is very important to ensure that the computer can download the code from GitHub normally when compiling the image.

## 5. 2. Get the source code of linux sdk

### 5. 2. 1. Download orangepi-build from github

- 1) The linux sdk actually refers to the code of orangepi-build. Using orangepi-build, multiple versions of linux images can be compiled. First download the code of orangepi-build, currently H6 series development boards already support legacy branch and current branch

```
test@test:~$ sudo apt update
```

```
test@test:~$ sudo apt install -y git
```

```
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git
```

**Downloading the code of orangepi-build through the git clone command does not require entering the username and password of the github account (the same is true for downloading other codes in this manual). If the Ubuntu PC prompts the user who needs to enter the github account after entering the git clone command. The name and password are usually the wrong address of the orangepi-build repository behind git clone. Please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account here.**

- 2) The legacy branch generally uses the BSP version of u-boot and the kernel, the current branch generally uses the u-boot and kernel close to the mainline version, and the u-boot and linux kernel currently used by the Orange Pi 3 LTS development board are as follows

| branch | u-boot version | linux kernel version |
|--------|----------------|----------------------|
| legacy | u-boot 2014.07 | linux4.9             |



|         |                |           |
|---------|----------------|-----------|
| current | u-boot 2020.04 | linux5.10 |
|---------|----------------|-----------|

- 3) Orangepi-build will contain the following files and folders after downloading
- build.sh**: Compile startup script
  - external**: Contains configuration files, specific scripts and source code of some programs needed to compile the image, etc.
  - LICENSE**: GPL 2 license file
  - README.md**: orangepi-build documentation
  - scripts**: Generic script for compiling linux images

```
test@test:~/orangepi-build$ ls  
build.sh  external  LICENSE  README.md  scripts
```

If you download the code of orangepi-build from github, you may find that orangepi-build does not contain the source code of u-boot and linux kernel after downloading, and the cross-compilation toolchain is not required to compile u-boot and linux kernel. , which is normal, because these things are stored in other separate github repositories or some servers (the addresses will be detailed below). orangepi-build will specify the address of u-boot, linux kernel and cross-compilation toolchain in the script and configuration file. When running orangepi-build, when it finds that these things are not available locally, it will automatically go to the corresponding place to download.

### 5. 2. 2. Download the cross-compilation toolchain

- 1) When orangepi-build runs for the first time, it will automatically download the cross-compilation **toolchain** and put it in the **toolchains** folder. After running the build.sh script of orangepi-build, it will check whether the cross-compilation toolchain in toolchains exists. , if it does not exist, it will restart the download, if it exists, it will be used directly, and the download will not be repeated



```
[ o.k. ] Checking for external GCC compilers
[ o.k. ] downloading using https(s) network [ gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz ]
#8d7829 16MiB/24MiB (65%) CN:1 DL:7.9MiB ETA:1s
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz: 24.9MiB [4.4MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-arm-none-eabi-4.8-2014.04_linux.tar.xz ]
#e30ec 17MiB/33MiB (50%) CN:1 DL:10MiB ETA:1s
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-linaro-arm-none-eabi-4.8-2014.04_linux.tar.xz: 33.9MiB [9.66MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz ]
#041c24 48MiB/48MiB (99%) CN:1 DL:2.7MiB
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz: 48.8MiB [13.0MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz ]
#3dee3e 72MiB/76MiB (93%) CN:1 DL:3.7MiB ETA:1s
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz: 77.0MiB [14.2MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz ]
#42e728 104MiB/104MiB (99%) CN:1 DL:2.8MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz: 104MiB [13.9MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz ]
#2c065e 108MiB/111MiB (97%) CN:1 DL:3.9MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz: 111MiB [13.4MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz ]
#d323ee 250MiB/251MiB (99%) CN:1 DL:2.8MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz: 251MiB [13.7MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu.tar.xz ]
#88b441 268MiB/269MiB (99%) CN:1 DL:9.9MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
```

- 2) The mirror website of the cross-compilation tool chain in China is the open source software mirror site of Tsinghua University

[https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/\\_toolchain/](https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/)

- 3) After the toolchains is downloaded, it will contain multiple versions of the cross-compilation **toolchains**

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

- 4) The cross-compilation toolchain used to compile the H6 linux kernel source code is
- linux4.9

gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu

- linux5.10

gcc-arm-9.2-2019.12-x86\_64-aarch64-none-linux-gnu



- 5) The cross-compilation toolchain used to compile the H6 u-boot source code is

- a. u-boot 2014.07

```
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
```

- b. u-boot 2020.04

```
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
```

### 5. 2. 3. Orangepi-build complete directory structure description

1) After the orangepi-build repository is downloaded, it does not contain the source code of the linux kernel, u-boot and the cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository

- a. The git repository where the Linux kernel source code is stored is as follows, where sun50iw6 is the code name of the H6 SOC chip

- a) linux4.9

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-4.9-sun50iw6
```

- b) linux5.10

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10
```

- b. The git repository where the source code of u-boot is stored is as follows, where sun50iw6 is the code name of the H6 SOC chip

- a) u-boot 2014.07

```
https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2014.07-sun50iw6-linux4.9
```

- b) u-boot 2020.04

```
https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2020.04
```

If you are not familiar with orangepi-build and do not know the detailed process of compiling the linux kernel and u-boot, please do not download and use the above linux kernel and u-boot source code for compilation operation, because the compilation script and configuration file of orangepi-build Some adjustments and optimizations will be made to u-boot and linux. If you do not use orangepi-build to compile u-boot and linux, you may encounter problems of compilation failure or failure to start.

2) When orangepi-build runs for the first time, it will download the cross-compilation toolchain, u-boot and linux kernel source code. After successfully compiling a linux image, the files and folders that can be seen in orangepi-build are:

- a. **build.sh**: Compile startup script



- b. **external**: Contains configuration files, scripts for specific functions and source code of some programs needed to compile the image. The rootfs compressed package cached during the process of compiling the image is also stored in external
- c. **kernel**: The source code of the Linux kernel is stored. The folder named **orange-pi-4.9-sun50iw6** stores the kernel source code of the legacy branch of the H6 development board. The folder named **orange-pi-5.10** stores the current branch of the H6 development board. (if only the linux image of the legacy branch is compiled, then only the kernel source code of the legacy branch can be seen; if only the linux image of the current branch is compiled, then only the kernel source code of the current branch can be seen), the kernel Please do not manually modify the name of the source code folder. If the compilation system is modified, the kernel source code will be downloaded again.
- d. **LICENSE**: GPL 2 license file
- e. **README.md**: orangepi-build documentation
- f. **output**: Store the compiled u-boot, linux and other deb packages, compilation logs, and compiled images and other files
- g. **scripts**: Generic script for compiling linux images
- h. **toolchains**: Store the cross-compilation toolchain
- i. **u-boot**: Store the source code of u-boot. The folder named **v2014.07-sun50iw6-linux4.9** stores the u-boot source code of the legacy branch of the H6 development board. The folder named **v2020.04** stores the H6 development board. The u-boot source code of the current branch of the board (if only the linux image of the legacy branch is compiled, then only the u-boot source code of the legacy branch can be seen; if only the linux image of the current branch is compiled, then only the current branch can be seen The u-boot source code of the branch), please do not modify the name of the folder of the u-boot source code manually, if the u-boot source code is re-downloaded when the compilation system runs
- j. **userpatches**: Store the configuration files needed to compile the script

```
test@test:~/orangepi-build$ ls
build.sh    external    kernel    LICENSE    output    README.md    scripts
toolchains  u-boot     userpatches
```

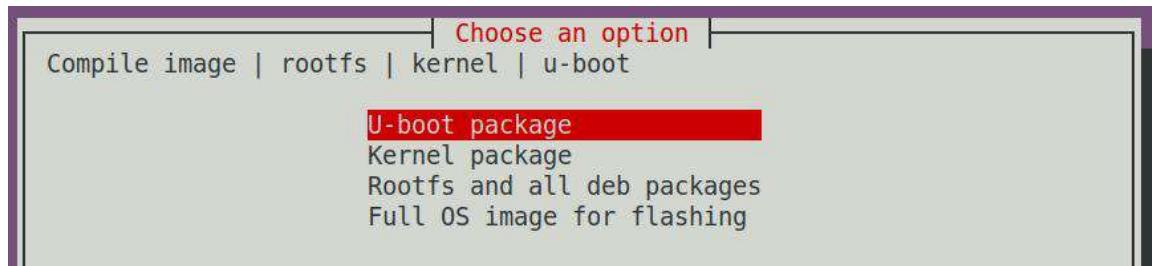


### 5. 3. Compile u-boot

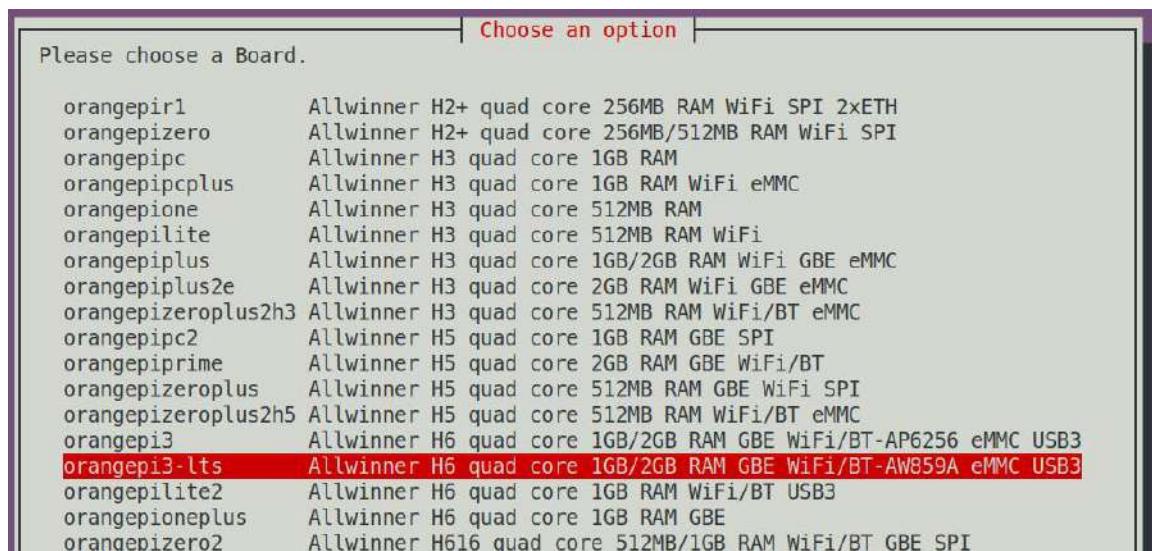
- 1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

- 2) Select **U-boot package** and press Enter

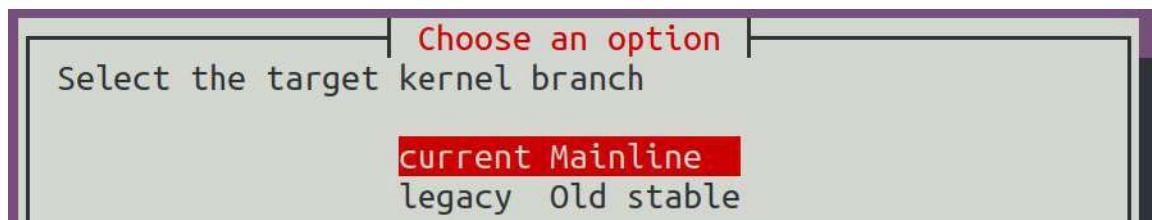


- 3) Then select the model of the development board



- 4) Then select branch

- a. current will compile u-boot v2020.04
- b. legacy will compile u-boot v2014.07





5) Then it will start compiling u-boot, some of the information prompted when compiling is explained as follows (take the legacy branch as an example)

a. u-boot source code version

[ o.k. ] Compiling u-boot [ **v2014.07** ]

b. The version of the cross-compile toolchain

[ o.k. ] Compiler version [ **arm-linux-gnueabi-gcc 4.9.4** ]

c. The path to the generated u-boot deb package

[ o.k. ] Target directory [ **output/debs/u-boot** ]

d. The package name of the u-boot deb package generated by compilation

[ o.k. ] File name [ **linux-u-boot-legacy-orangepi3-lts\_2.1.8\_arm64.deb** ]

e. Compilation time used

[ o.k. ] Runtime [ **1 min** ]

f. Repeat the command to compile u-boot, use the following command to start compiling u-boot directly without selecting through the graphical interface

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangepi3-lts** ]

**BRANCH=legacy BUILD\_OPT=u-boot KERNEL\_CONFIGURE=no** ]

6) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/  
linux-u-boot-legacy-orangepi3-lts_2.1.8_arm64.deb
```

7) The files contained in the generated u-boot deb package are as follows

a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot  
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \  
linux-u-boot-legacy-orangepi3-lts_2.1.8_arm64.deb . (Note that there is a "." at  
the end of the command)
```

```
test@test:~/orangepi_build/output/debs/u-boot$ ls  
linux-u-boot-current-orangepi3-lts_2.1.8_armhf.deb usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/u-boot$ tree usr  
usr  
└── lib  
    └── linux-u-boot-legacy-orangepi3-lts_2.1.8_arm64
```



```
|   └── boot0_sdcard.fex      //binary for boot0
|   └── boot_package.fex     //u-boot binaries
└── u-boot
    ├── LICENSE
    ├── orangepi3-lts-u-boot.dts
    └── platform_install.sh
```

3 directories, 5 files

- 8) The orangepi-build compilation system will first synchronize the u-boot source code with the u-boot source code of the github server when compiling the u-boot source code, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code (**You need to compile u-boot once before closing this function, otherwise you will be prompted that the source code of u-boot cannot be found**), otherwise the modifications will be restored. The method is as follows:

Set the IGNORE\_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

- 9) When debugging the u-boot code, you can use the following method to update the u-boot in the linux image for testing

- Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-legacy-orangepi3-lts_2.1.8_arm64.deb root@192.168.1.xxx:/root
```

- Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepi3-lts-legacy
```

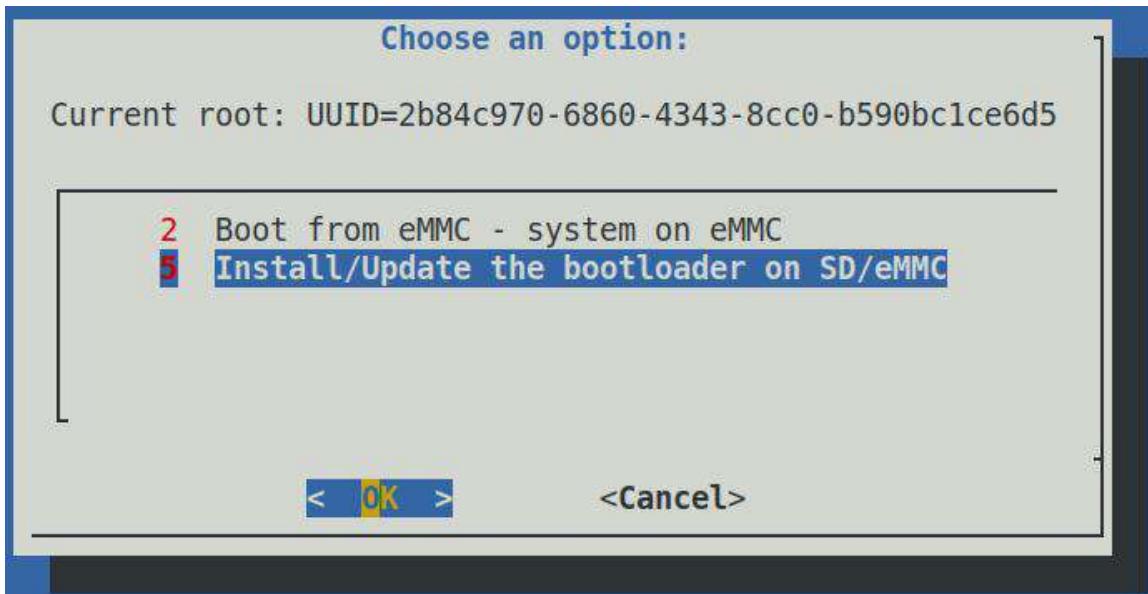
- Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-legacy-orangepi3-lts_2.1.8_arm64.deb
```

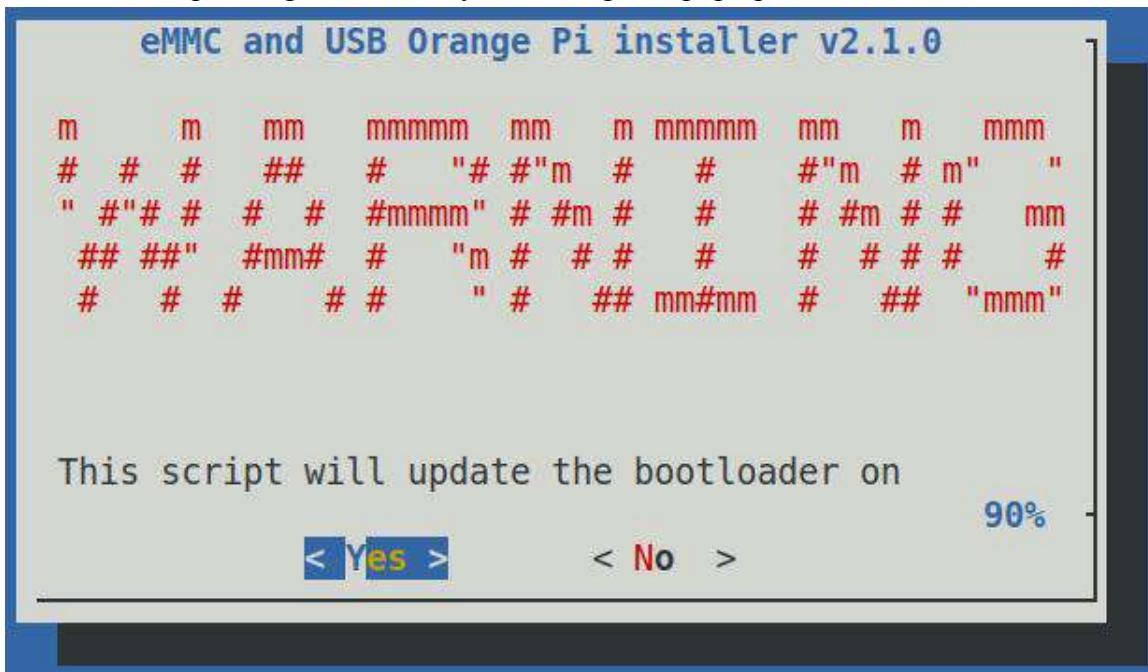
- Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

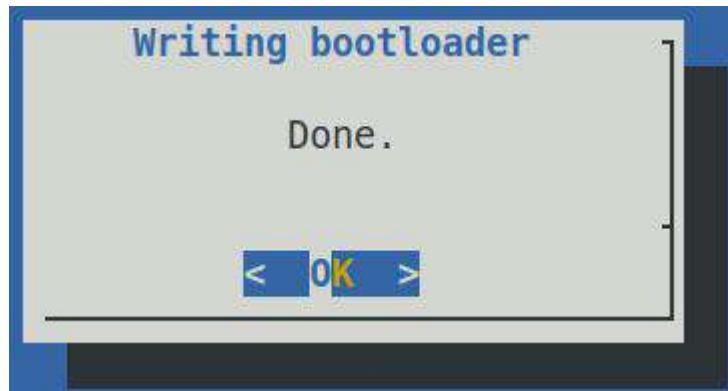
- then select **5 Install/Update the bootloader on SD/eMMC**



f. After pressing the Enter key, a Warring will pop up first



g. Press the Enter key again to start updating u-boot. After the update, the following information will be displayed



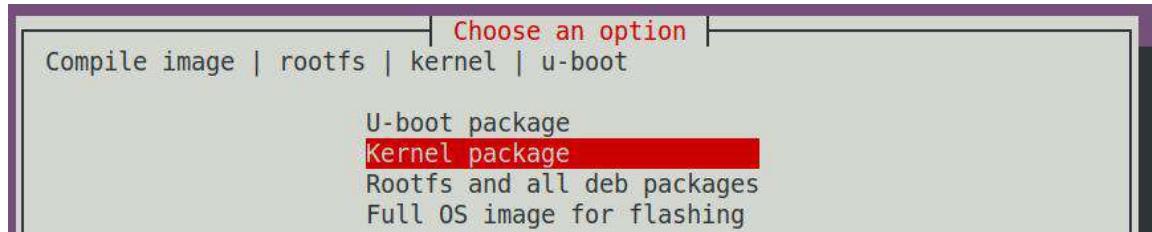
- h. Then you can restart the development board to test whether the modification of u-boot takes effect

## 5. 4. Compile the linux kernel

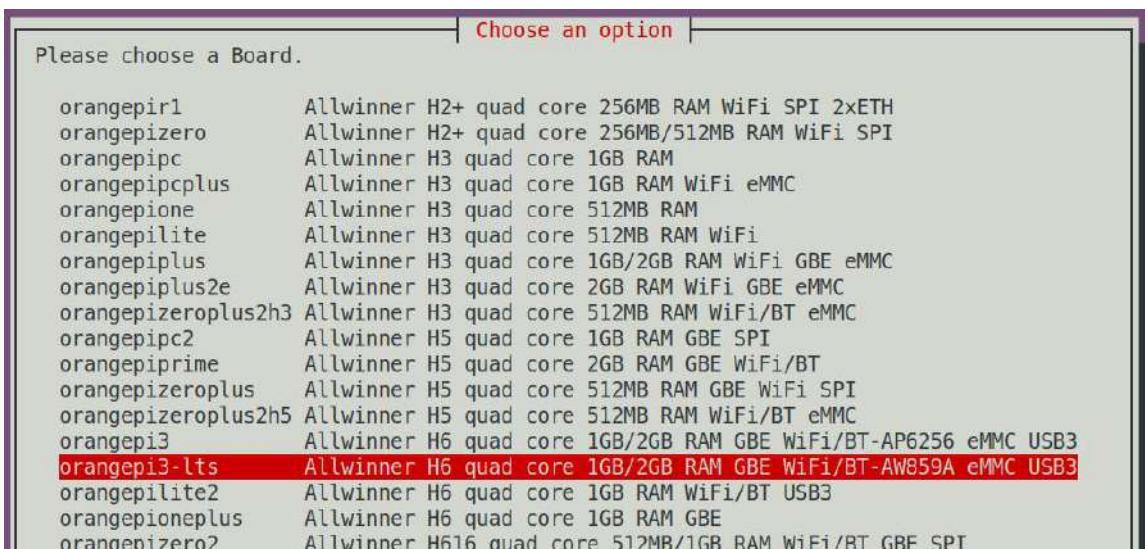
- 1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangeipi-build$ sudo ./build.sh
```

- 2) Select **Kernel package**, then press Enter

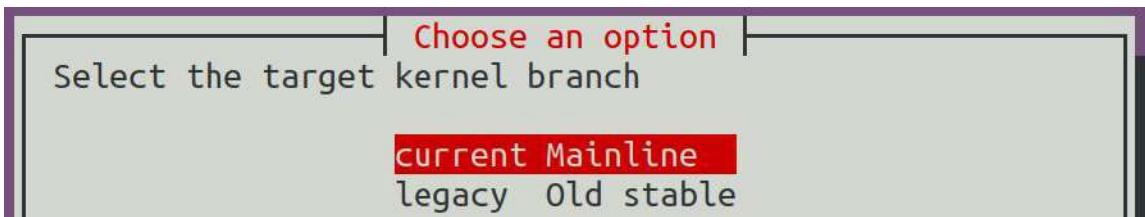


- 3) Then select the model of the development board

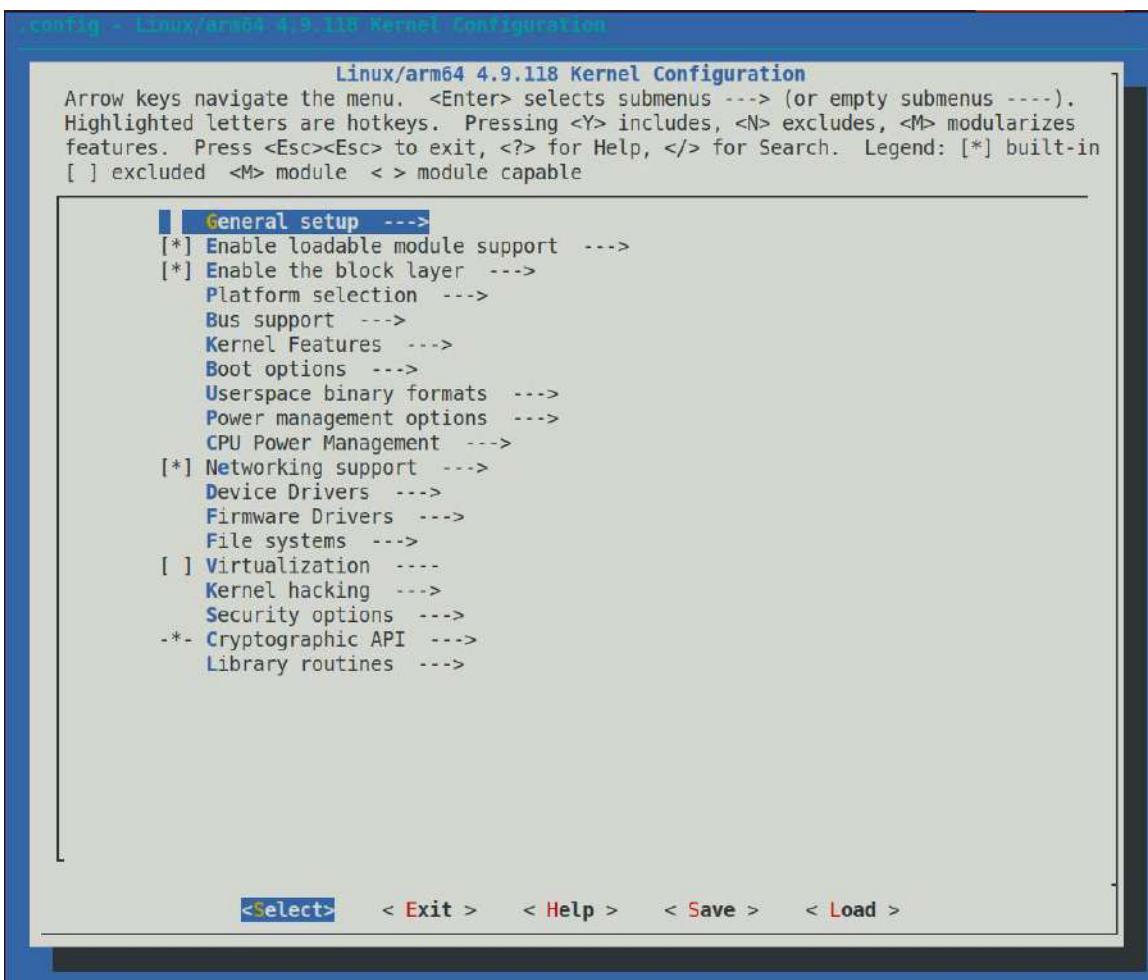


4) Then select branch

- current will compile linux 5.10
- legacy will compile linux4.9



5) Then the kernel configuration interface opened by **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration. If you do not need to modify the kernel configuration, you can simply exit. After exiting, the kernel source code will be compiled.



- a. If you do not need to modify the configuration options of the kernel, when running the build.sh script, pass in **KERNEL\_CONFIGURE=no** to temporarily shield the configuration interface of the pop-up kernel

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

- b. You can also set **KERNEL\_CONFIGURE=no** in the `orangeipi-build/userpatches/config-default.conf` configuration file to permanently disable this feature

- c. If the following error is displayed when compiling the kernel, this is because the terminal interface of Ubuntu PC is too small, so the interface of `make menuconfig` cannot be displayed. Please adjust the terminal of Ubuntu PC to the maximum, and then re-run the build.sh script



```
HOSTCC scripts/kconfig/mconf.o
HOSTCC scripts/kconfig/lxdialog/checklist.o
HOSTCC scripts/kconfig/lxdialog/util.o
HOSTCC scripts/kconfig/lxdialog/inputbox.o
HOSTCC scripts/kconfig/lxdialog/textbox.o
HOSTCC scripts/kconfig/lxdialog/yesno.o
HOSTCC scripts/kconfig/lxdialog/menubox.o
HOSTLD scripts/kconfig/mconf
scripts/kconfig/mconf_Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:560: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
[ error ] ERROR in function compile_kernel [ compilation.sh:376 ]
[ error ] Error kernel menuconfig failed
[ o.k. ] Process terminated
```

6) Part of the information prompted when compiling the kernel source code is explained as follows (take the legacy branch as an example)

a. The version of the linux kernel source code

[ o.k. ] Compiling legacy kernel [ **4.9.118** ]

b. The version of the cross-compilation toolchain used

[ o.k. ] Compiler version [ **aarch64-none-linux-gnu-gcc 9.2.1** ]

c. The configuration file used by the kernel by default and the path where it is stored

[ o.k. ] Using kernel config file [ **config/kernel/linux-sun50iw6-legacy.config** ]

d. **KERNEL\_CONFIGURE=yes** is set, the final configuration file **.config** used by the kernel will be copied to **output/config**. If the kernel configuration is not modified, the final configuration file is the same as the default configuration file

[ o.k. ] Exporting new kernel config [ **output/config/linux-sun50iw6-legacy.config** ]

e. The path to the generated kernel-related deb package

[ o.k. ] Target directory [ **output/debs/** ]

f. The package name of the kernel image deb package generated by compilation

[ o.k. ] File name [ **linux-image-legacy-sun50iw6\_2.1.8\_arm64.deb** ]

g. Compilation time used

[ o.k. ] Runtime [ **5 min** ]

h. Finally, the compilation command to repeat the compilation of the last selected kernel will be displayed. Use the following command to directly start compiling the kernel source code without selecting through the graphical interface.

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangepi3-lts** ]

**BRANCH=legacy BUILD\_OPT=kernel KERNEL\_CONFIGURE=yes** ]



7) The path of the kernel configuration file used by default when orangepi-build compiles the kernel is as follows

- a. linux4.9——legacy branch

```
orangepi-build/external/config/kernel/linux-sun50iw6-legacy.config
```

- b. linux5.10 - current branch

```
orangepi-build/external/config/kernel/linux-5.10-sunxi64-current.config
```

8) View the compiled and generated kernel-related deb packages

- a. **linux-dtb-legacy-sun50iw6\_2.1.8\_arm64.deb** Contains dtb files used by the kernel
- b. **linux-headers-legacy-sun50iw6\_2.1.8\_arm64.deb** Include kernel header files
- c. **linux-image-legacy-sun50iw6\_2.1.8\_arm64.deb** Contains kernel images and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-legacy-sun50iw6_2.1.8_arm64.deb
output/debs/linux-headers-legacy-sun50iw6_2.1.8_arm64.deb
output/debs/linux-image-legacy-sun50iw6_2.1.8_arm64.deb
```

9) The files contained in the generated linux-image deb package are as follows

- a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-legacy-sun50iw6_2.1.8_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-legacy-sun50iw6_2.1.8_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-legacy-sun50iw6_2.1.8_arm64.deb  usr
```

- b. The decompressed file is as follows

```
test@test:~/orangepi_build/output/debs/test$ tree -L 2
.
├── boot
└── config-4.9.118-sun50iw6      //The configuration file used to compile the
```



kernel source code

```
|   └── System.map-4.9.118-sun50iw6
|   └── vmlinuz-4.9.118-sun50iw6      //Compile the generated kernel image file
└── etc
    └── kernel
└── lib
    └── modules                      //Compile the generated kernel module
└── linux-image-legacy-sun50iw6_2.1.8_arm64.deb
└── usr
    └── lib
        └── share
```

8 directories, 4 files

- 10) The orangepi-build compilation system will first synchronize the linux kernel source code with the linux kernel source code of the github server when compiling the linux kernel source code, so if you want to modify the linux kernel source code, you first need to turn off the update function of the source code (**you need to compile it once This function can only be turned off after the linux kernel source code, otherwise it will prompt that the source code of the linux kernel cannot be found**), otherwise the modification will be restored, the method is as follows:

Set the IGNORE\_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

- 11) If the kernel is modified, the following methods can be used to update the kernel and kernel modules of the development board linux system

- Upload the compiled deb package of the linux kernel to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-legacy-sun50iw6_2.1.8_arm64.deb root@192.168.1.207:/root
```

- Then log in to the development board and uninstall the deb package of the installed linux kernel

```
root@orangepi:~# apt purge -y linux-image-legacy-sun50iw6
```



- c. Install the deb package of the new linux kernel just uploaded

```
root@orangepi:~# dpkg -i linux-image-legacy-sun50iw6_2.1.8_arm64.deb
```

- d. Then restart the development board, and then check whether the kernel-related modifications have taken effect

- 12) The method of installing the kernel header files into the linux system is as follows

- a. Upload the deb package of the compiled linux header file to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs  
test@test:~/orangepi-build/output/debs$ scp \  
linux-headers-legacy-sun50iw6_2.1.8_arm64.deb root@192.168.1.207:/root
```

- b. Then log in to the development board and install the deb package of the linux header file just uploaded

```
root@orangepi:~# dpkg -i linux-headers-legacy-sun50iw6_2.1.8_arm64.deb
```

- c. After installation, you can see the contents of the kernel header files just installed in /usr/src

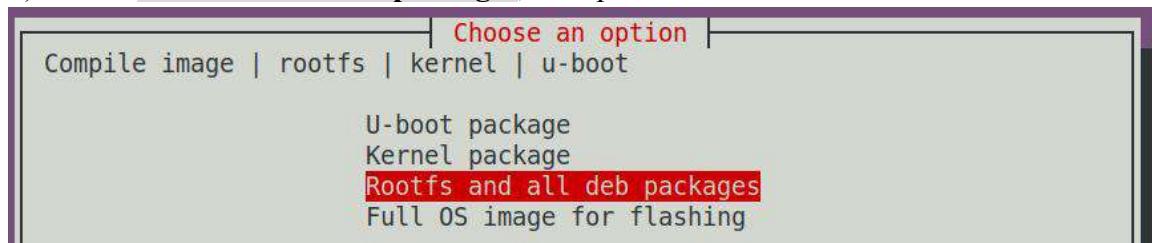
```
root@orangepi:~# ls /usr/src  
linux-headers-4.9.118-sun50iw6  
root@orangepi:~# ls /usr/src/linux-headers-4.9.118-sun50iw6  
Documentation Module.symvers certs firmware init lib net security usr  
Kconfig arch crypto fs ipc mm samples sound virt Makefile block  
drivers include kernel modules scripts tools
```

## 5. 5. Compile rootfs

- 1) Run the build.sh script, remember to add sudo permissions

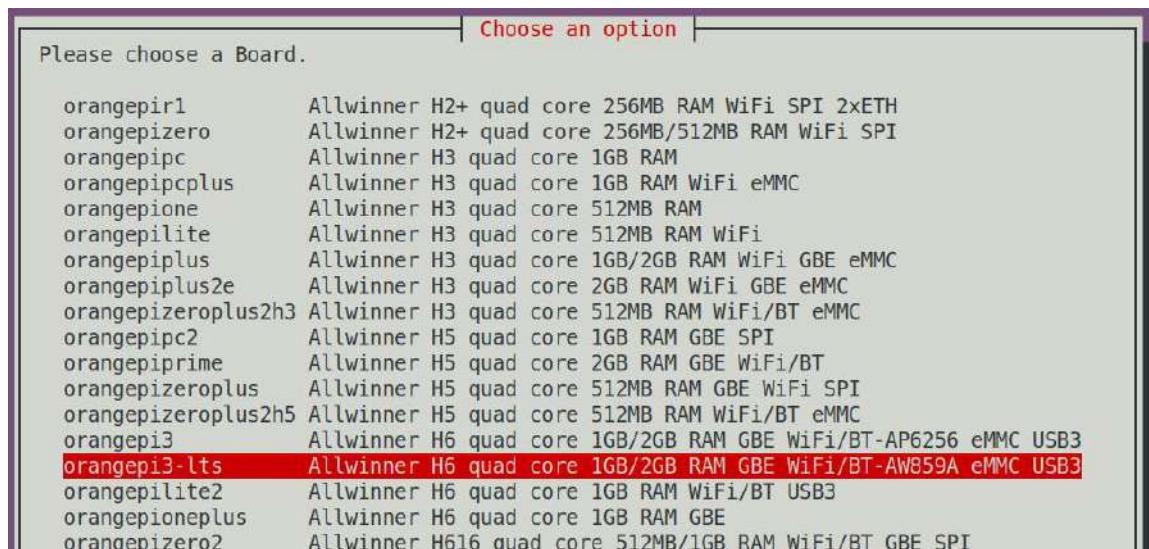
```
test@test:~/orangepi-build$ sudo ./build.sh
```

- 2) Select **Rootfs and all deb packages**, then press Enter

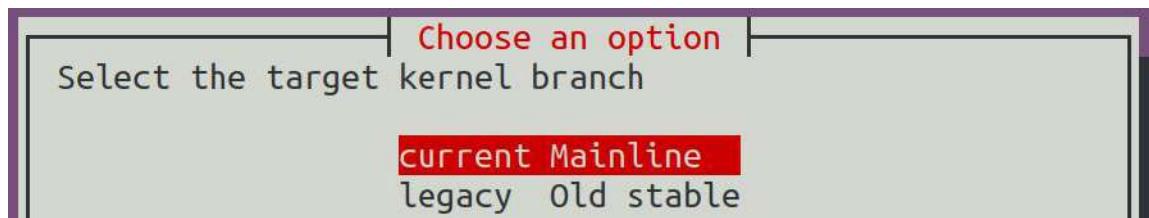




## 3) Then select the model of the development board



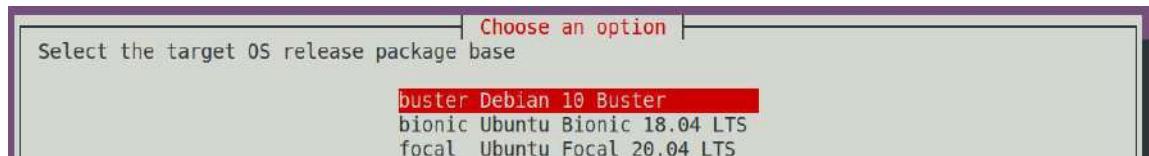
## 4) Then select branch



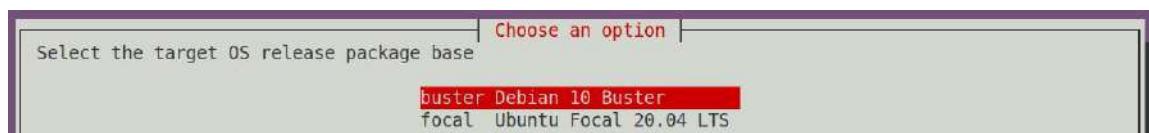
## 5) Then select the type of rootfs

|        |              |
|--------|--------------|
| buster | Debian 10    |
| bionic | Ubuntu 18.04 |
| focal  | Ubuntu 20.04 |

a. The linux distributions supported by linux4.9 are as follows



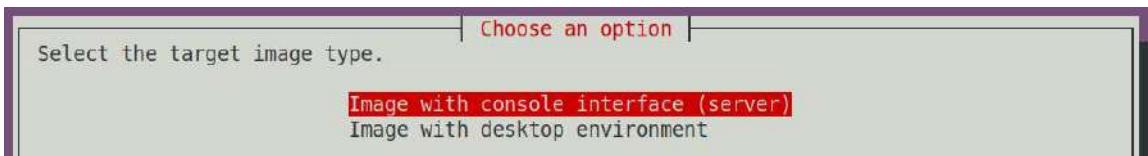
b. The linux distributions supported by linux5.10 are as follows



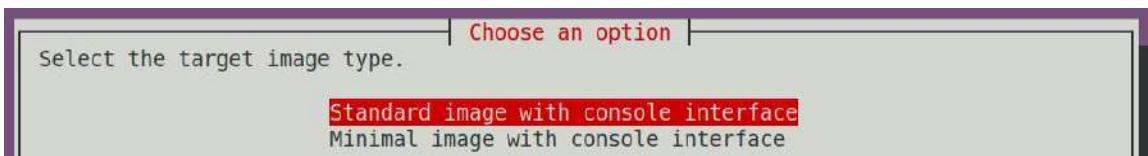
## 6) Then select the type of image



- a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size
- b. **Image with desktop environment** Indicates a image image with a desktop, which is relatively large in size



7) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.



8) After selecting the type of image, the rootfs will be compiled, and some of the information prompted during compilation are as follows

- a. type of rootfs

[ o.k. ] local not found [ Creating new rootfs cache for **focal** ]

- b. The storage path of the rootfs compressed package generated by compilation

[ o.k. ] Target directory [ **external/cache/rootfs** ]

- c. The name of the rootfs compressed package generated by compilation

[ o.k. ] File name [ **focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4** ]

- d. Compilation time

[ o.k. ] Runtime [ **13 min** ]

- e. Repeat the command to compile rootfs, use the following command to start compiling rootfs directly without selecting through the graphical interface

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangepi3-lts** ]

**BRANCH=legacy BUILD\_OPT=rootfs RELEASE=focal BUILD\_MINIMAL=no  
BUILD\_DESKTOP=no KERNEL\_CONFIGURE=yes ]**

9) View the rootfs compressed package generated by compilation

- a. **focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4** is the compressed



package of rootfs, the meaning of each field of the name is

- a) **focal** indicates the type of linux distribution of rootfs
  - b) **cli** indicates that rootfs is the type of the server version, and if it is **dektop**, it indicates the type of the desktop version
  - c) **arm64** represents the architecture type of rootfs
  - d) **153618961f14c28107ca023429aa0eb9** is the MD5 hash value generated by the package names of all packages installed by rootfs. As long as the list of packages installed by rootfs is not modified, this value will not change, and the compilation script will use this MD5 hash value to Determine if you need to recompile rootfs
- b. **focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list** lists the package names of all packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/  
focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4  
focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list
```

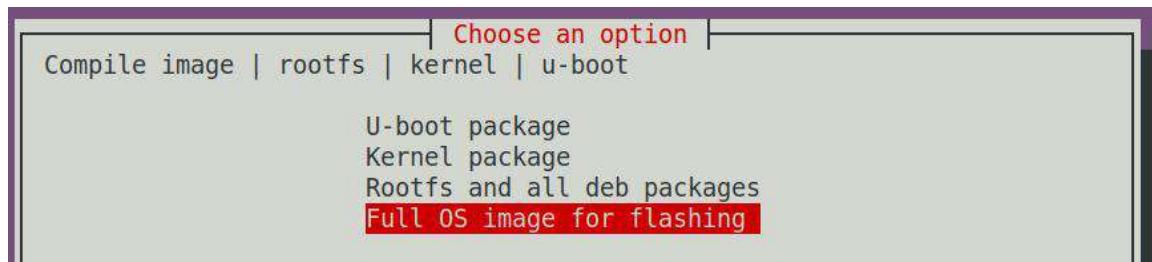
10) If the required rootfs already exists under **external/cache/rootfs**, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it has There is a rootfs available for cache, if there is one, use it directly, which can save a lot of download and compilation time

## 5. 6. Compile the linux image

1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Full OS image for flashing** and press Enter





## 3) Then select the model of the development board

Please choose a Board.

| Choose an option    |                                                                 |
|---------------------|-----------------------------------------------------------------|
| orangepir1          | Allwinner H2+ quad core 256MB RAM WiFi SPI 2xETH                |
| orangepizero        | Allwinner H2+ quad core 256MB/512MB RAM WiFi SPI                |
| orangepic           | Allwinner H3 quad core 1GB RAM                                  |
| orangepicplus       | Allwinner H3 quad core 1GB RAM WiFi eMMC                        |
| orangepine          | Allwinner H3 quad core 512MB RAM                                |
| orangepilete        | Allwinner H3 quad core 512MB RAM WiFi                           |
| orangepiplus        | Allwinner H3 quad core 1GB/2GB RAM WiFi GBE eMMC                |
| orangepiplus2e      | Allwinner H3 quad core 2GB RAM WiFi GBE eMMC                    |
| orangepizeroplus2h3 | Allwinner H3 quad core 512MB RAM WiFi/BT eMMC                   |
| orangepicp2         | Allwinner H5 quad core 1GB RAM GBE SPI                          |
| orangepiprime       | Allwinner H5 quad core 2GB RAM GBE WiFi/BT                      |
| orangepizeroplus    | Allwinner H5 quad core 512MB RAM GBE WiFi SPI                   |
| orangepizeroplus2h5 | Allwinner H5 quad core 512MB RAM WiFi/BT eMMC                   |
| orangepi3           | Allwinner H6 quad core 1GB/2GB RAM GBE WiFi/BT-AP6256 eMMC USB3 |
| orangepi3-lts       | Allwinner H6 quad core 1GB/2GB RAM GBE WiFi/BT-AW859A eMMC USB3 |
| orangepilete2       | Allwinner H6 quad core 1GB RAM WiFi/BT USB3                     |
| orangepineplus      | Allwinner H6 quad core 1GB RAM GBE                              |
| orangepizero2       | Allwinner H616 quad core 512MB/1GB RAM WiFi/BT GBE SPI          |

## 4) Then select branch

- d. current will compile linux 5.10
- e. legacy will compile linux4.9

Select the target kernel branch

| Choose an option |            |
|------------------|------------|
| current          | Mainline   |
| legacy           | Old stable |

## 5) Then select the type of rootfs

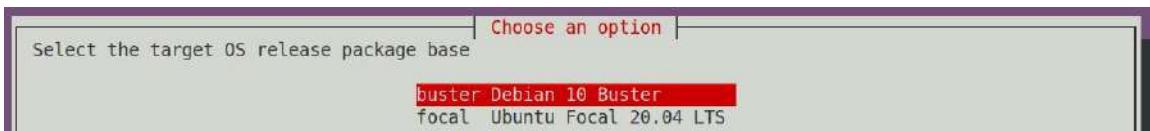
|        |              |
|--------|--------------|
| buster | Debian 10    |
| bionic | Ubuntu 18.04 |
| focal  | Ubuntu 20.04 |

- a. The linux distributions supported by linux4.9 are as follows

Select the target OS release package base

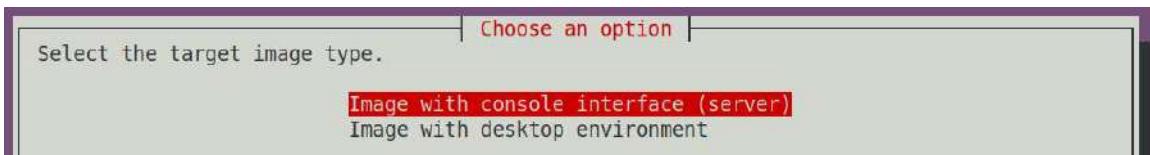
| Choose an option |                         |
|------------------|-------------------------|
| buster           | Debian 10 Buster        |
| bionic           | Ubuntu Bionic 18.04 LTS |
| focal            | Ubuntu Focal 20.04 LTS  |

- b. The linux distributions supported by linux5.10 are as follows

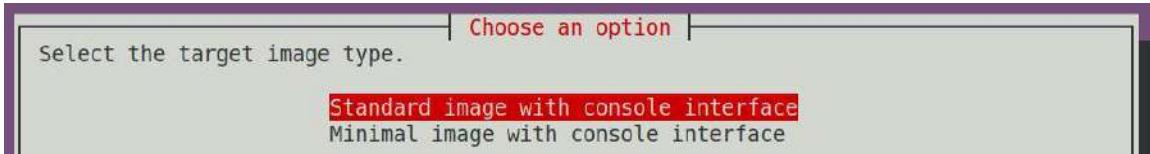


6) Then select the type of image

- a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size
- b. **Image with desktop environment** Indicates a image image with a desktop, which is relatively large in size



7) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.



8) After selecting the type of image, the linux image will be compiled. The general process of compilation is as follows

- a. Initialize the compilation environment of the Ubuntu PC and install the software packages required for the compilation process
- b. Download the source code of u-boot and linux kernel (if cached, only update the code)
- c. Compile u-boot source code and generate u-boot deb package
- d. Compile the linux source code to generate linux-related deb packages
- e. Make a deb package of linux firmware
- f. Make the deb package of the orangepi-config tool
- g. Make board-level supported deb packages
- h. If you are compiling the desktop version of the image, you will also create a desktop-related deb package
- i. Check whether the rootfs has been cached, if there is no cache, then recreate the



- rootfs, if it has been cached, directly decompress and use
- j. Install the deb package generated earlier into rootfs
  - k. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configuration, etc.
  - l. Then make an image file and format the partition, the default type is ext4
  - m. Copy the configured rootfs to the imageed partition
  - n. then update initramfs
  - o. Finally, write the bin file of u-boot into the image through the dd command

9) After compiling the image, the following information will be prompted (take legacy branch as an example)

- a. The storage path of the compiled image

[ o.k. ] Done building

[ **output/images/OrangePi3-lts\_2.1.8\_ubuntu\_focal\_server\_linux4.9.118/OrangePi3-lts\_2.1.8\_ubuntu\_focal\_server\_linux4.9.118.img** ]

- b. Compilation time used

[ **o.k. ] Runtime [ 19 min ]** ]

- c. Repeat the command to compile the image, use the following command to start compiling the image directly without selecting through the graphical interface

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangePi3-lts** ]

**BRANCH=legacy BUILD\_OPT=image RELEASE=focal BUILD\_MINIMAL=no  
BUILD\_DESKTOP=no KERNEL\_CONFIGURE=yes** ]



## 6. Linux SDK - Instructions for using the new version of orangepi-build

### 1. What is the difference between the compilation host of the new version and the old version?

The old version of orangepi-build was run on an x64 computer or virtual machine with Ubuntu 18.04.

The new version of orangepi-build is run on an x64 computer or virtual machine of **Ubuntu21.04**.

If you want to use the new version of orangepi-build described in this chapter to compile Linux images, you first need to prepare a computer or virtual machine with **Ubuntu 21.04** installed.

### 2. What is the difference between the source code storage address of the new version and the old version?

The old version of the orangepi-build source code is stored in the main branch of the orangepi-build repository:

<https://github.com/orangepi-xunlong/orangepi-build/tree/main>

The source code of the new version of orangepi-build is stored in the **next** branch of the orangepi-build repository:

<https://github.com/orangepi-xunlong/orangepi-build/tree/next>

### 3. What is the difference between the new version and the old version supporting the kernel?

The old version of orangepi-build mainly supports Linux4.9 and Linux5.10.

The new version of orangepi-build currently supports Linux5.10-media and Linux5.16.

Linux5.10-media kernel+**Debian11**+Kodi can support hard decoding and playback of videos in H264, H265 and other formats. For the specific test method,



please refer to the content of the [Debian11 Kodi Image Instructions](#) section.

**4. What is the difference between the types of Linux distributions supported by the new version and the old version?**

The old version of **orangeipi-build** mainly supports **Debian10**, **Ubuntu18.04**, **Ubuntu20.04**.

The new version of **orangeipi-build** mainly supports **Debian11**, **Ubuntu20.04**, **Ubuntu22.04**.

**Ubuntu 22.04** will be released after the test is stable.

**5. What is the difference between the image names generated by the new version and the old version?**

The version of the image compiled by the old version of **orangeipi-build** is named **2.xx**.

The image version generated by the new version of **orangeipi-build** is **3.xx**. Desktop images also add fields for desktop types, such as **xfce**.

## 6. 1. Compilation system requirements

1) The Linux SDK, namely **orangeipi-build**, only supports running on a computer with **Ubuntu 21.04** installed, so before downloading **orangeipi-build**, please make sure that the Ubuntu version installed on your computer is Ubuntu 21.04. The command to check the Ubuntu version installed on the computer is as follows. If the Release field does not display **21.04**, it means that the current Ubuntu version does not meet the requirements. Please change the system before performing the following operations.

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 21.04
Release:        21.04
Codename:       hirsute
test@test:~$
```



2) If the computer is installed with Windows system and there is no computer with Ubuntu 21.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu 21.04 virtual machine in the Windows system. But please note, do not compile orangepi-build on the WSL virtual machine, because orangepi-build has not been tested in the WSL virtual machine, so it cannot be guaranteed that orangepi-build can be used normally in WSL, and please do not use the Linux system of the development board. using orangepi-build in

3) The installation image download address of Ubuntu 21.04 **amd64** version is:

<https://repo.huaweicloud.com/ubuntu-releases/21.04/ubuntu-21.04-desktop-amd64.iso>

4) After installing Ubuntu 21.04 in the computer or virtual machine, please set the software source of Ubuntu 21.04 to Tsinghua source, otherwise it is easy to make mistakes due to network reasons when installing the software later

a. For the method of replacing Tsinghua source, please refer to the description of this webpage.

<https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/>

b. Note that the Ubuntu version needs to be switched to 21.04

## Ubuntu 镜像使用帮助

Ubuntu 的软件源配置文件是 `/etc/apt/sources.list`。将系统自带的该文件做个备份，将该文件替换为下面内容，即可使用 TUNA 的软件源镜像。

选择你的ubuntu版本: 21.04

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted universe multiverse
```

本镜像仅包含 32/64 位 x86 架构处理器的软件包，在 ARM(arm64, armhf)、PowerPC(ppc64el)、RISC-V(riscv64) 和 S390x 等架构的设备上（对应官方源为ports.ubuntu.com）请使用 [ubuntu-ports 镜像](#)。

c. The content of the `/etc/apt/sources.list` file that needs to be replaced is

test@test:~\$ sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak

test@test:~\$ sudo vim /etc/apt/sources.list

# The source image is commented by default to improve the speed of apt update, you can



```
uncomment it yourself if necessary
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe
multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted
universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted
universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted
universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted universe
multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted
universe multiverse

# Pre-release software sources, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted
universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted
universe multiverse
```

- d. After the replacement, you need to update the package information and ensure that no errors are reported

```
test@test:~$ sudo apt update
```

- e. In addition, since the source code such as the kernel and U-boot are stored on GitHub, it is very important to ensure that the computer can download the code from GitHub normally when compiling the image.

## 6. 2. Get the source code of linux sdk

### 6. 2. 1. Download orangepi-build from github

- 1) The linux sdk actually refers to the code of orangepi-build. Orangepi-build is modified



based on the armbian build compilation system. Using orangepi-build, multiple versions of linux images can be compiled. The command to download the orangepi-build code looks like this:

```
test@test:~$ sudo apt update  
test@test:~$ sudo apt install git  
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git -b next
```

**Note that the instructions in this section are for the source code of the next branch of orangepi-build. The above git clone command needs to specify the branch of the source code of orangepi-build as next.**

The screenshot shows a GitHub repository page for 'orangepi-xunlong/orangepi-build'. At the top, there are navigation links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below these, a dropdown menu shows 'next' (highlighted with a red box) and 'main'. A red arrow points from the text in the previous paragraph to this dropdown. A message below the dropdown says 'This branch is 3 commits ahead of main.' To the right, there is a 'Contribute' button. The main area displays a list of files: external, scripts, .gitignore, LICENSE, README.md, and build.sh. A note at the bottom of the list says 'When viewing the code of orangepi-build, you need to switch to the next branch'. On the right side of the list, there is a commit count of '141 commits'.

**Downloading the code of orangepi-build through the git clone command does not require entering the username and password of the github account (the same is true for downloading other codes in this manual). If the Ubuntu PC prompts the user who needs to enter the github account after entering the git clone command The name and password are usually the wrong address of the orangepi-build repository behind git clone. Please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account here.**

- 2) The u-boot and linux kernel versions currently used by the H6 series development boards are as follows



| branch  | u-boot version | linux kernel version |
|---------|----------------|----------------------|
| current | u-boot 2020.04 | linux5.10            |
| next    | u-boot 2021.10 | linux5.16            |

Note that the next branch mentioned here is used to distinguish u-boot and kernel versions. It is not the same thing as the next branch of the orangepi-build source code mentioned above. Please don't get confused.

- 3) OrangePi-build will contain the following files and folders after downloading
  - f. **build.sh**: Compile startup script
  - g. **external**: Contains configuration files, specific scripts and source code of some programs needed to **compile the image**, etc.
  - h. **LICENSE**: GPL 2 license file
  - i. **README.md**: orangepi-build documentation
  - j. **scripts**: Generic script for compiling linux images

```
test@test:~/orangepi-build$ ls  
build.sh  external  LICENSE  README.md  scripts
```

If you download the code of orangepi-build from github, you may find that orangepi-build does not contain the source code of u-boot and linux kernel after downloading, nor does it require cross-compilation tools to compile u-boot and linux kernel chain, this is normal, because these things are stored in other separate github repositories or some servers (the addresses will be detailed below). orangepi-build will specify the address of u-boot, linux kernel and cross-compilation toolchain in the script and configuration file. When running orangepi-build, when it finds that these things are not available locally, it will automatically go to the corresponding place to download.

### 6. 2. 2. Download the cross-compilation toolchain

- 1) When orangepi-build runs for the first time, it will automatically download the cross-compilation **toolchains** and put it in the **toolchains** folder. After running the build.sh script of orangepi-build, it will check whether the cross-compilation toolchain in toolchains exists. , if it does not exist, it will restart the download, if it exists, it will be used directly, and the download will not be repeated



```
[ o.k. ] Checking for external GCC compilers
[ o.k. ] downloading using https(s) network [ gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz ]
#8d7829 16MiB/24MiB (65%) CN:1 DL:7.9MiB ETA:1s
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz: 24.9MiB [4.4MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-arm-none-eabi-4.8-2014.04_linux.tar.xz ]
#e30ec 17MiB/33MiB (50%) CN:1 DL:10MiB ETA:1s
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-arm-arm-none-eabi-4.8-2014.04_linux.tar.xz: 33.9MiB [9.66MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz ]
#041c24 48MiB/48MiB (99%) CN:1 DL:2.7MiB
[ o.k. ] Verified [ PGP ]
[ ... ] decompressing
[ ... ] gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz: 48.8MiB [13.0MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz ]
#3dee3e 72MiB/76MiB (93%) CN:1 DL:3.7MiB ETA:1s
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz: 77.0MiB [14.2MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz ]
#42e728 104MiB/104MiB (99%) CN:1 DL:2.8MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz: 104MiB [13.9MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz ]
#2c065e 108MiB/111MiB (97%) CN:1 DL:3.9MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz: 111MiB [13.4MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz ]
#d323ee 250MiB/251MiB (99%) CN:1 DL:2.8MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
[ ... ] gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz: 251MiB [13.7MiB/s] [=====>] 100%
[ ... ] downloading using https(s) network [ gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu.tar.xz ]
#88b441 268MiB/269MiB (99%) CN:1 DL:9.9MiB
[ o.k. ] Verified [ MD5 ]
[ ... ] decompressing
```

- 2) The mirror website of the cross-compilation tool chain in China is the open source software mirror site of Tsinghua University

[https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/\\_toolchain/](https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/)

- 3) After the **toolchains** is downloaded, it will contain multiple versions of the cross-compilation toolchain

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

- 4) The cross-compilation toolchain used to compile the H6 Linux kernel source code is

[gcc-arm-9.2-2019.12-x86\\_64-aarch64-none-linux-gnu](https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/)

- 5) The cross-compilation toolchain used to compile the H6 u-boot source code is

[gcc-arm-9.2-2019.12-x86\\_64-aarch64-none-linux-gnu](https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/)



### 6. 2. 3. Orangepi-build complete directory structure description

1) After the orangepi-build repository is downloaded, it does not contain the source code of the linux kernel, u-boot and the cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository

- a. The git repository where the linux kernel source code is stored is as follows, pay attention to switch the branch of the linux-orangepi repository to
  - a) Linux5.10

<https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10-media>

- b) Linux5.16

<https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.16-sunxi64>

- b. The git repository where the u-boot source code is stored is as follows. Note that the branch of the u-boot-orangepi repository is switched to
  - a) u-boot v2020.04

<https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2020.04>

- b) u-boot v2021.10

<https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2021.10-sunxi>

If you are not familiar with orangepi-build and do not know the detailed process of compiling the linux kernel and u-boot, please do not download and use the above linux kernel and u-boot source code for compilation operation, because the compilation script and configuration file of orangepi-build Some adjustments and optimizations will be made to u-boot and linux. If you do not use orangepi-build to compile u-boot and linux, you may encounter problems of compilation failure or failure to start.

2) When orangepi-build runs for the first time, it will download the cross-compilation toolchain, u-boot and linux kernel source code. After successfully compiling a linux image, the files and folders that can be seen in orangepi-build are:

- a. **build.sh**: Compile startup script
- b. **external**: Contains configuration files, scripts for specific functions and source code of some programs needed to compile the image. The rootfs compressed package cached during the process of compiling the image is also stored in external
- c. **kernel**: Stores the source code of the linux kernel. The folder named **orange-pi-5.10-media** stores the kernel source code of the current branch of the



H6 development board, and the folder named **orange-pi-5.16-sunxi64** stores the kernel source code of the next branch of the H6 development board ( If only the linux image of the current branch is compiled, then only the kernel source code of the current branch can be seen; if only the linux image of the next branch is compiled, then only the kernel source code of the next branch can be seen), the folder of the kernel source code Please do not modify the name manually. If modified, the kernel source code will be re-downloaded when the compilation system runs.

- d. **LICENSE**: GPL 2 license file
- e. **README.md**: orangeipi-build documentation
- f. **output**: Store the compiled u-boot, linux and other deb packages, compilation logs, and compiled images and other files
- g. **scripts**: Generic script for compiling linux images
- h. **toolchains**: Store the cross-compilation toolchain
- i. **u-boot**: Store the source code of u-boot. The folder named **v2020.04** stores the u-boot source code of the current branch of the H6 development board, and the folder named **v2021.10-sunxi** stores the u-boot source code of the next branch of the H6 development board (if only If you compile the linux image of the current branch, you can only see the source code of the current branch; if you only compile the linux image of the next branch, you can only see the source code of the next branch), the name of the folder of the u-boot source code Please do not modify it manually. If modified, the u-boot source code will be downloaded again when the compilation system is running.
- j. **userpatches**: Store the configuration files needed to compile the script

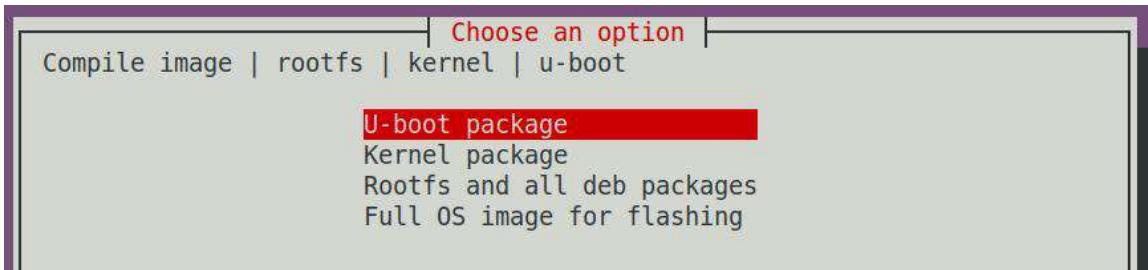
```
test@test:~/orangeipi-build$ ls
build.sh    external    kernel    LICENSE    output    README.md    scripts
toolchains  u-boot    userpatches
```

### 6. 3. Compile u-boot

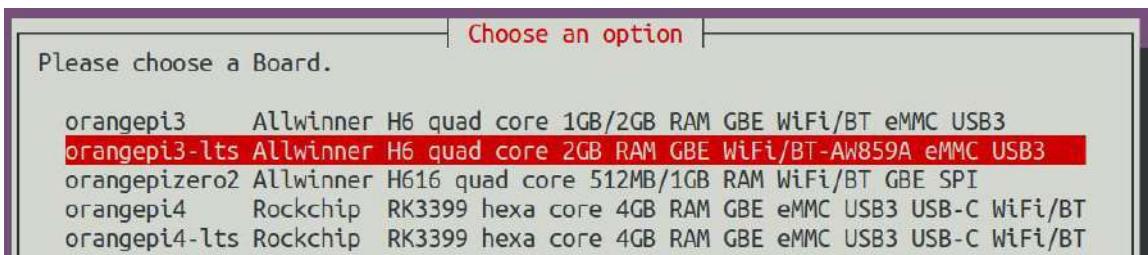
- 1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangeipi-build$ sudo ./build.sh
```

- 2) Select **U-boot package** and press Enter



3) Then select the model of the development board



4) Then it will start to compile u-boot, and some of the information prompted during compilation are as follows

a. u-boot source code version

[ o.k. ] Compiling u-boot [ **v2021.10** ]

b. The version of the cross-compile toolchain

[ o.k. ] Compiler version [ **aarch64-none-linux-gnu-gcc 9.2.1** ]

c. The path to the generated u-boot deb package

[ o.k. ] Target directory [ **orangepi-build/output/debs/u-boot** ]

d. The package name of the u-boot deb package generated by compilation

[ o.k. ] File name [ **linux-u-boot-current-orangepi3-lts\_3.0.0\_arm64.deb** ]

e. Compilation time used

[ o.k. ] Runtime [ **1 min** ]

f. Repeat the command to compile u-boot, use the following command to start compiling u-boot directly without selecting through the graphical interface

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangepi3-lts BRANCH=current BUILD\_OPT=u-boot KERNEL\_CONFIGURE=yes** ]

5) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/  
linux-u-boot-current-orangepi3-lts_3.0.0_arm64.deb
```



6) The files contained in the generated u-boot deb package are as follows

a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \
linux-u-boot-current-orangepi3-lts_3.0.0_arm64.deb .  (Note that there is a "." at
the end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-current-orangepi3-lts_3.0.0_arm64.deb  usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-next-orangepi3-lts_3.0.0_arm64
    │   └── u-boot-sunxi-with-spl.bin
    └── u-boot
        ├── LICENSE
        ├── orangepi_3_lts_defconfig
        └── platform_install.sh
3 directories, 4 files
```

7) The orangepi-build compilation system will first synchronize the u-boot source code with the u-boot source code of the github server when compiling the u-boot source code, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code (**You need to compile u-boot once before closing this function, otherwise it will prompt that the source code of u-boot cannot be found. If it is a source code compressed package downloaded from Google cloud disk, there is no such problem, because the source code of u-boot have been cached**), otherwise the modifications will be restored, as follows:

Set the IGNORE\_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

8) When debugging the u-boot code, you can use the following method to update the



u-boot in the linux image for testing

- a. Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-current-orangepi3-lts_3.0.0_arm64.deb root@192.168.1.xxx:/root
```

- b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepi3-lts-current
```

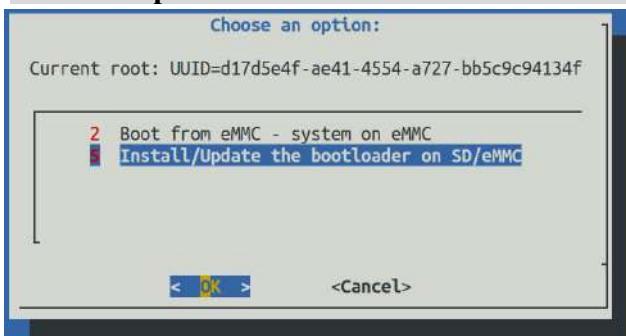
- c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-current-orangepi3-lts_3.0.0_arm64.deb
```

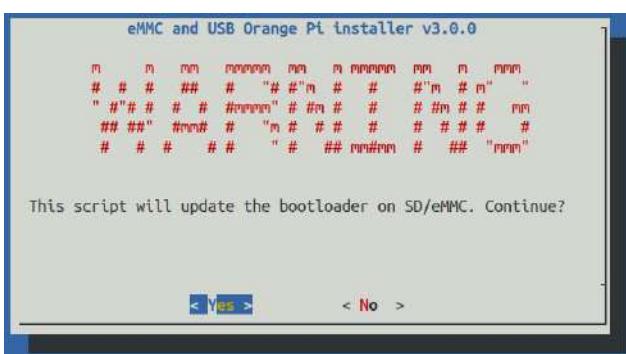
- d. Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

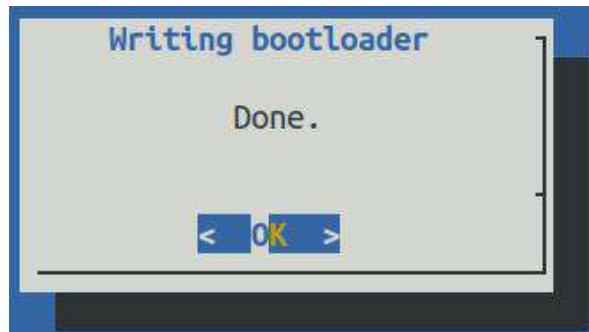
- e. Then select **5 Install/Update the bootloader on SD/eMMC**



- f. After pressing the Enter key, a Warring will pop up first



- g. Press the Enter key again to start updating u-boot, after the update, the following information will be displayed



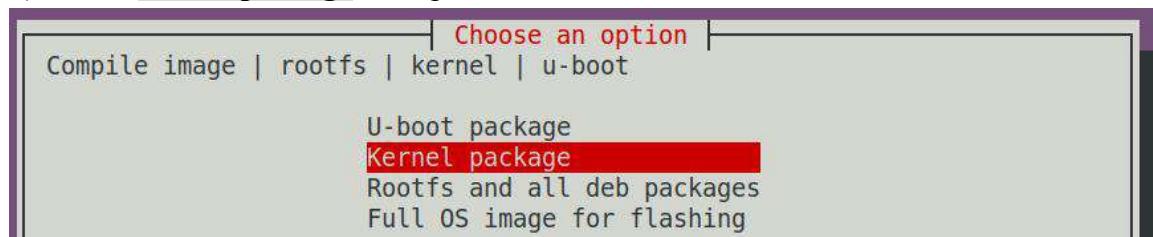
- h. Then you can restart the development board to test whether the modification of u-boot takes effect

## 6.4. Compile the linux kernel

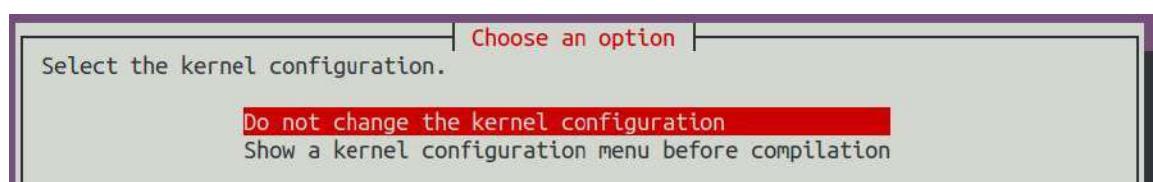
- 1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

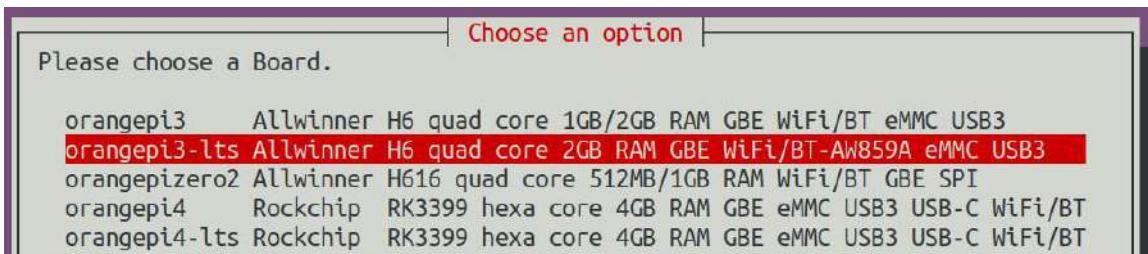
- 2) Select **Kernel package**, then press Enter



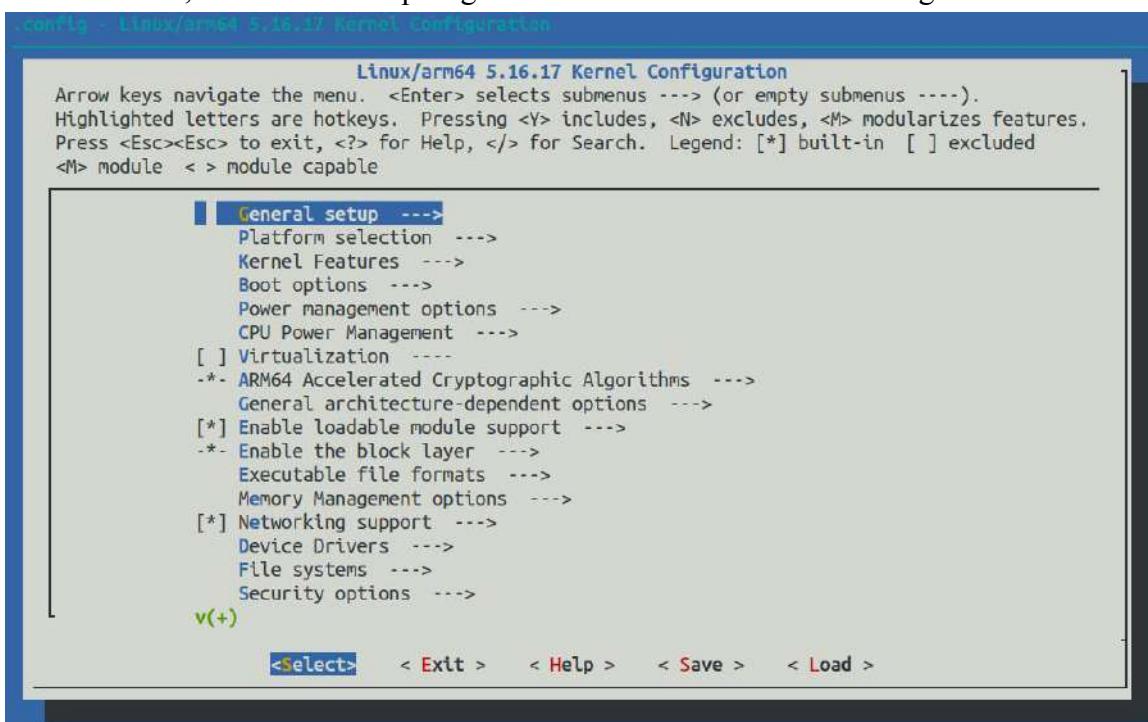
- 3) Then you will be prompted whether you need to display the kernel configuration interface. If you do not need to modify the kernel configuration, you can choose the first one. If you need to modify the kernel configuration, choose the second one.



- 4) Then select the model of the development board



- 5) If you choose to display the kernel configuration menu (the second option) in the second step, the kernel configuration interface opened by **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration, save and exit after modification. , it will start compiling the kernel source code after exiting



- If you do not need to modify the configuration options of the kernel, when running the build.sh script, pass in **KERNEL\_CONFIGURE=no** to temporarily shield the configuration interface of the pop-up kernel

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

- You can also set **KERNEL\_CONFIGURE=no** in the orangepi-build/userpatches/config-default.conf configuration file to permanently disable this feature
- If the following error is displayed when compiling the kernel, this is because the terminal interface of Ubuntu PC is too small, so the interface of **make menuconfig** cannot be displayed. Please adjust the terminal of Ubuntu PC to the maximum, and



then re-run the build.sh script

```
HOSTCC scripts/kconfig/mconf.o
HOSTCC scripts/kconfig/lxdialog/checklist.o
HOSTCC scripts/kconfig/lxdialog/util.o
HOSTCC scripts/kconfig/lxdialog/inputbox.o
HOSTCC scripts/kconfig/lxdialog/textbox.o
HOSTCC scripts/kconfig/lxdialog/yesno.o
HOSTCC scripts/kconfig/lxdialog/menubox.o
HOSTLD scripts/kconfig/mconf
scripts/kconfig/mconf Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:560: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
[ error ] ERROR in function compile_kernel [ compilation.sh:376 ]
[ error ] Error kernel menuconfig failed
[ o.k. ] Process terminated
```

6) Part of the information prompted when compiling the kernel source code is explained as follows

a. The version of the linux kernel source code

[ o.k. ] Compiling current kernel [ **5.16.17** ]

b. The version of the cross-compilation toolchain used

[ o.k. ] Compiler version [ **aarch64-none-linux-gnu-gcc 9.2.1** ]

c. The configuration file used by the kernel by default and the path where it is stored

[ o.k. ] Using kernel config file [ **config/linux-5.16-sunxi64-current.config** ]

d. The path to the generated kernel-related deb package

[ o.k. ] Target directory [ **output/debs/** ]

e. Package name of the kernel image deb package generated by compilation

[ o.k. ] File name [ **linux-image-current-sun50iw6\_3.0.0\_arm64.deb** ]

f. Compilation time used

[ o.k. ] Runtime [ **5 min** ]

g. Finally, the compilation command to repeat the compilation of the last selected kernel will be displayed. Use the following command to directly start compiling the kernel source code without selecting through the graphical interface.

[ o.k. ] Repeat Build Options [ **sudo ./build.sh BOARD=orangepi3-lts** ]

**BRANCH=current BUILD\_OPT=kernel KERNEL\_CONFIGURE=yes** ]

7) View the compiled and generated kernel-related deb packages

a. **linux-dtb-current-sun50iw6\_3.0.0\_arm64.deb** Contains dtb files used by the



- kernel
- b. **linux-headers-current-sun50iw6\_3.0.0\_arm64.deb** Include kernel header files
  - c. **linux-image-current-sun50iw6\_3.0.0\_arm64.deb** Contains kernel images and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-current-sun50iw6_3.0.0_arm64.deb
output/debs/linux-image-current-sun50iw6_3.0.0_arm64.deb
output/debs/linux-headers-current-sun50iw6_3.0.0_arm64.deb
```

8) The files contained in the generated linux-image deb package are as follows

- a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-current-sun50iw6_3.0.0_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-current-sun50iw6_3.0.0_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-current-sun50iw6_3.0.0_arm64.deb  usr
```

- b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/test$ tree -L 2
.
├── boot
│   ├── config-5.16.17-sun50iw6
│   ├── System.map-5.16.17-sun50iw6
│   └── vmlinuz-5.16.17-sun50iw6
├── etc
│   └── kernel
├── lib
│   └── modules
└── linux-image-current-sun50iw6_3.0.0_arm64.deb
└── usr
    ├── lib
    └── share
```



8 directories, 4 files

9) The orangepi-Bulid compiler will first synchronize the Linux kernel source code with github server Linux kernel source code, so if you want to modify the Linux kernel source code, First of all need to be closed source update feature (**need complete compiled a Linux kernel source code to close the function, otherwise it will prompt can not find the Linux kernel source code, if from Google cloud disk to download the source code package, there is no this problem, because Linux is the source of all cached**), otherwise the changes will be restored, The method is as follows:

Set the IGNORE\_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf  
IGNORE_UPDATES="yes"
```

10) If the kernel is modified, the following methods can be used to update the kernel and kernel modules of the development board linux system

a. Upload the compiled linux kernel deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs  
test@test:~/orangepi-build/output/debs$ scp \  
linux-image-current-sun50iw6_3.0.0_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the deb package of the installed linux kernel

```
root@orangepi:~# apt purge -y linux-image-current-sun50iw6
```

c. Install the deb package of the new linux kernel just uploaded

```
root@orangepi:~# dpkg -i linux-image-current-sun50iw6_3.0.0_arm64.deb
```

d. Then restart the development board, and then check whether the kernel-related modifications have taken effect

```
root@orangepi:~# reboot
```

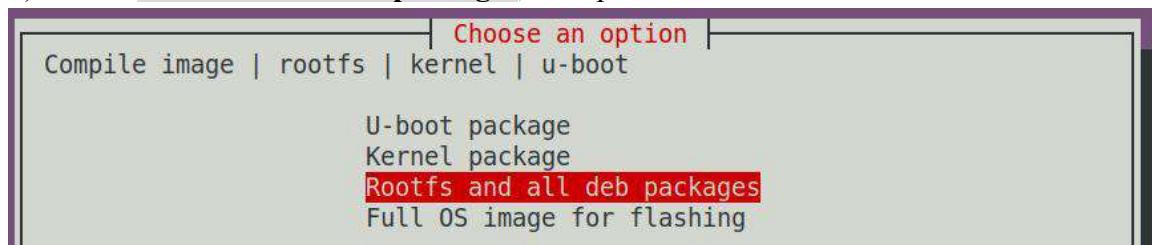
## 6. 5. Compile rootfs

1) Run the build.sh script, remember to add sudo permissions

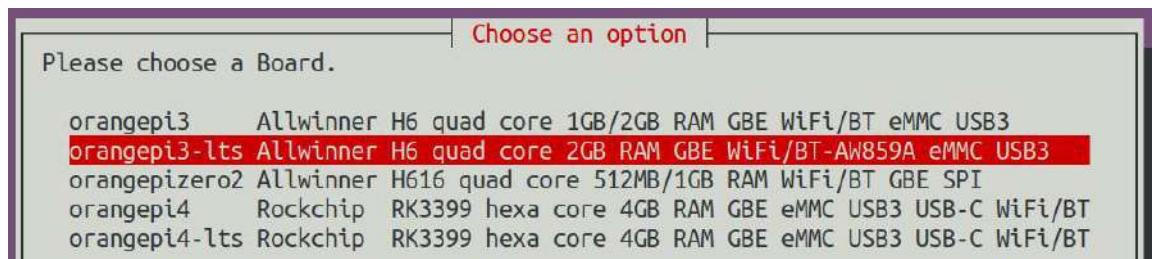
```
test@test:~/orangepi-build$ sudo ./build.sh
```



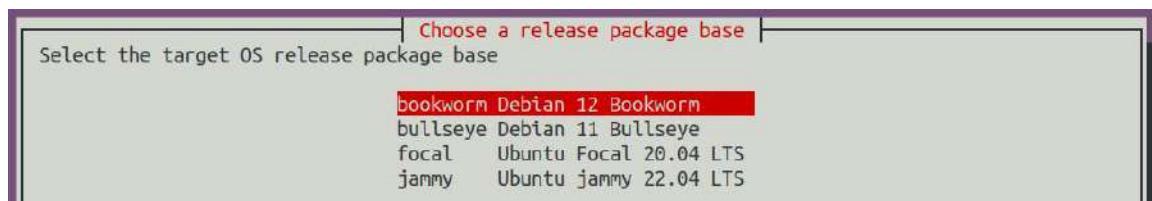
2) Select **Rootfs and all deb packages**, then press Enter



3) Then select the model of the development board

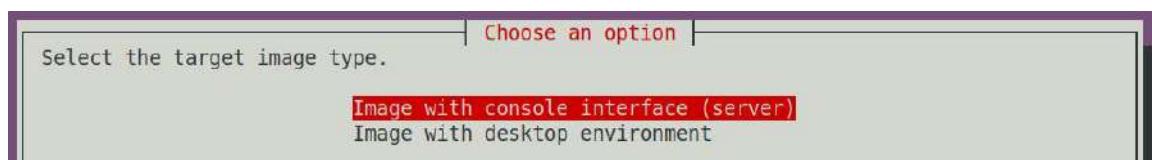


4) Then select the type of rootfs

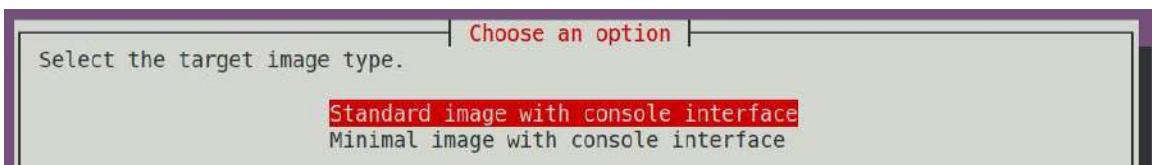


5) Then select the type of image

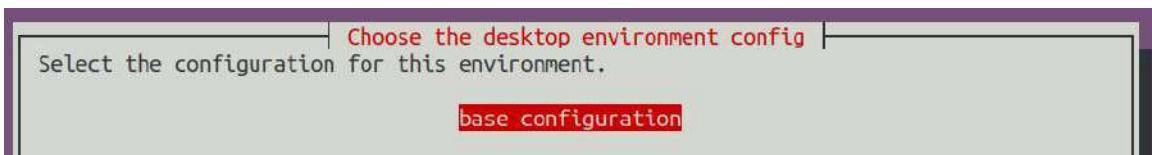
- a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size
- b. **Image with desktop environment** Indicates a image with a desktop, which is relatively large in size



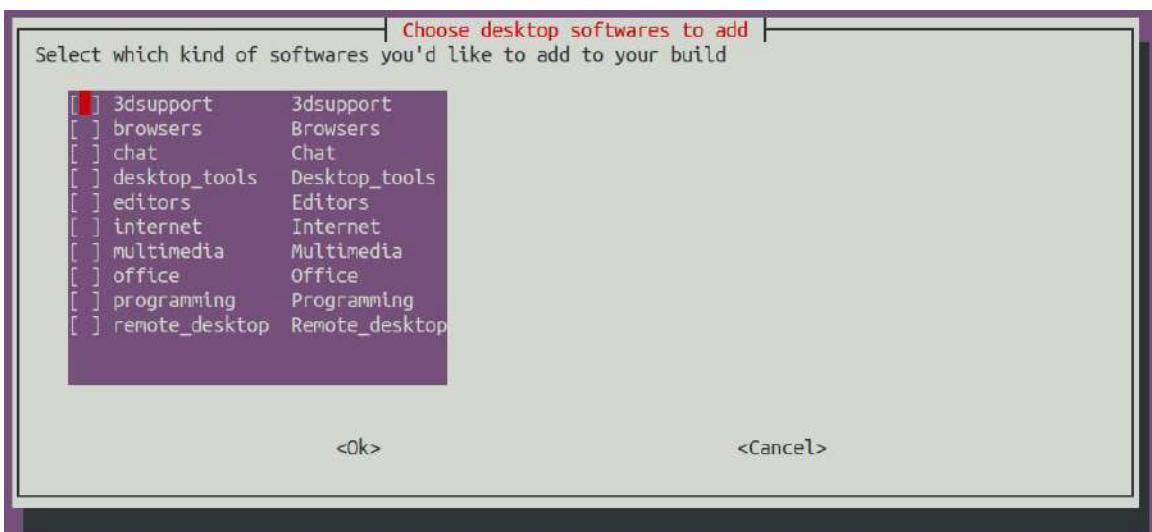
6) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.



7) If you are compiling the desktop version of the image, you also need to select the type of desktop environment, but only XFCE is currently supported, so just press Enter.



You can then select additional packages that need to be installed. For example, if you need to install a browser, you can choose **browsers**.



Which packages are included in each selection can be seen in the code of orangepi-build. You can also modify these configuration files to add the packages you want to install:

- a. First enter the **external/config/desktop** directory to see the desktop configuration folders of different linux distributions. Note that not all the Orange



Pi development boards that can be seen in the code are supported and tested.

```
test@test:~/orangepi-build$ cd external/config/desktop
test@test:~/orangepi-build/external/config/desktop$ ls
bionic  bookworm  bullseye  buster  focal  jammy  README.md  sid
```

- b. Then select the type of distribution you want to view or modify, and enter the corresponding directory, such as **bullseye**

```
test@test:~/orangepi-build/external/config/desktop$ cd bullseye
test@test:~/orangepi-build/external/config/desktop/bullseye$ ls
appgroups  environments
```

- c. Then enter the **appgroups** directory to see all app groups

```
test@test:~/orangepi-build/external/config/desktop/bullseye$ cd appgroups
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ ls
3dsupport  browsers  chat  desktop_tools  editors  internet  multimedia
office  programming  remote_desktop
```

- d. Open the packages file under different groups to view the software contained in the group

```
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat \
programming/packages
build-essential
clang
meld
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat \
office/packages
libreoffice
```

8) Then it will start to compile rootfs, and some of the information prompted during compilation are as follows

- a. type of rootfs

```
[ o.k. ] local not found [ Creating new rootfs cache for bullseye ]
```

- b. The storage path of the rootfs compressed package generated by compilation

```
[ o.k. ] Target directory [ external/cache/rootfs ]
```

- c. The name of the rootfs compressed package generated by compilation

```
[ o.k. ] File name
```

```
[ bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4 ]
```

- d. Compilation time



## [ o.k. ] Runtime [ 13 min ]

- e. Repeat the command to compile rootfs, use the following command to start compiling rootfs directly without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh BOARD=orangepi3-lts
```

```
BRANCH=current BUILD_OPT=rootfs RELEASE=bullseye
```

```
BUILD_MINIMAL=no BUILD_DESKTOP=no
```

```
KERNEL_CONFIGURE=yes ]
```

## 9) View the rootfs compressed package generated by compilation

- a. `bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4` is the compressed package of rootfs, the meaning of each field of the name is
- a) **Bullseye** indicates the type of linux distribution of rootfs
  - b) **xfce** indicates that the rootfs is of the desktop version, and if it is cli, it indicates the server version
  - c) **arm64** represents the architecture type of rootfs
  - d) **25250ec7002de9e81a41de169f1f89721** is the MD5 hash value generated by the package names of all packages installed by rootfs. As long as the list of packages installed by rootfs is not modified, this value will not change, and the compilation script will use this MD5 hash value to Determine if you need to recompile rootfs
- b. `bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list` lists the package names of all packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
```

```
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4
```

```
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.current
```

```
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list
```

10) If the required rootfs already exists under **external/cache/rootfs**, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it has There is a rootfs available for cache, if there is one, use it directly, which can save a lot of download and compilation time

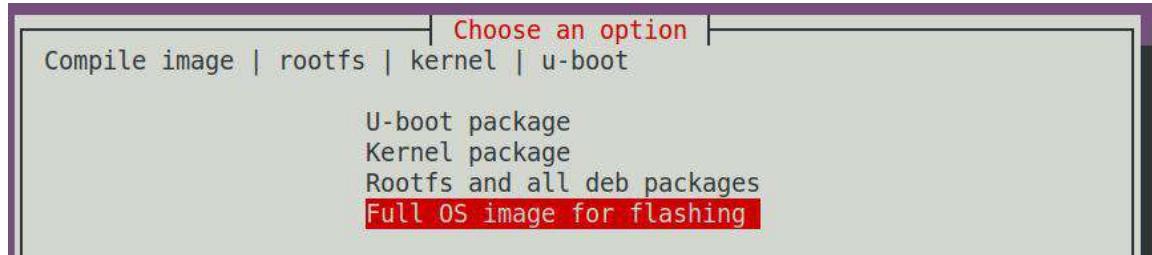


## 6. 6. Compile the linux image

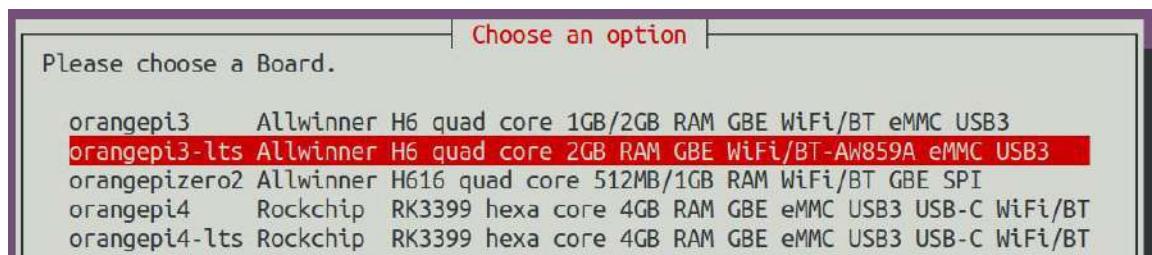
- 1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

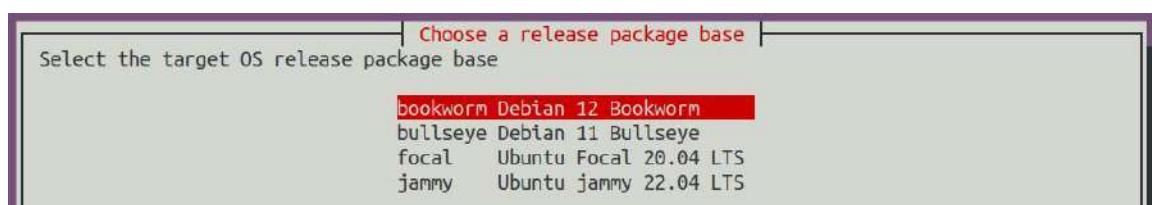
- 2) Select **Full OS image for flashing** and press Enter



- 3) Then select the model of the development board

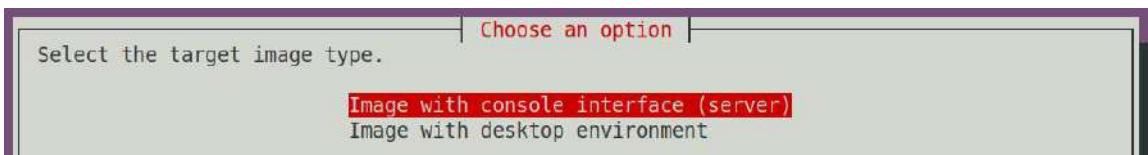


- 4) Then select the type of rootfs

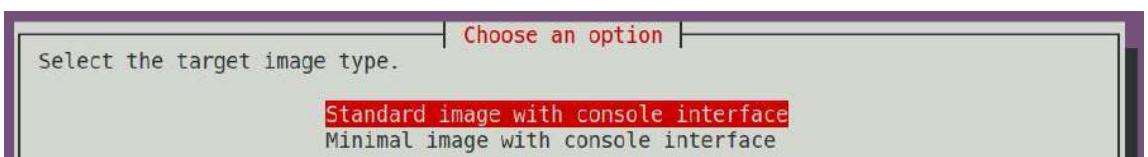


- 5) Then select the type of image

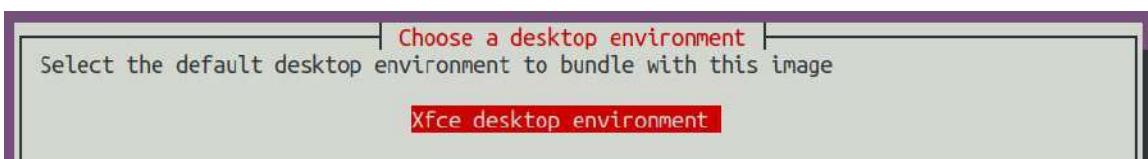
- a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size
- b. **Image with desktop environment** Indicates a image with a desktop, which is relatively large in size



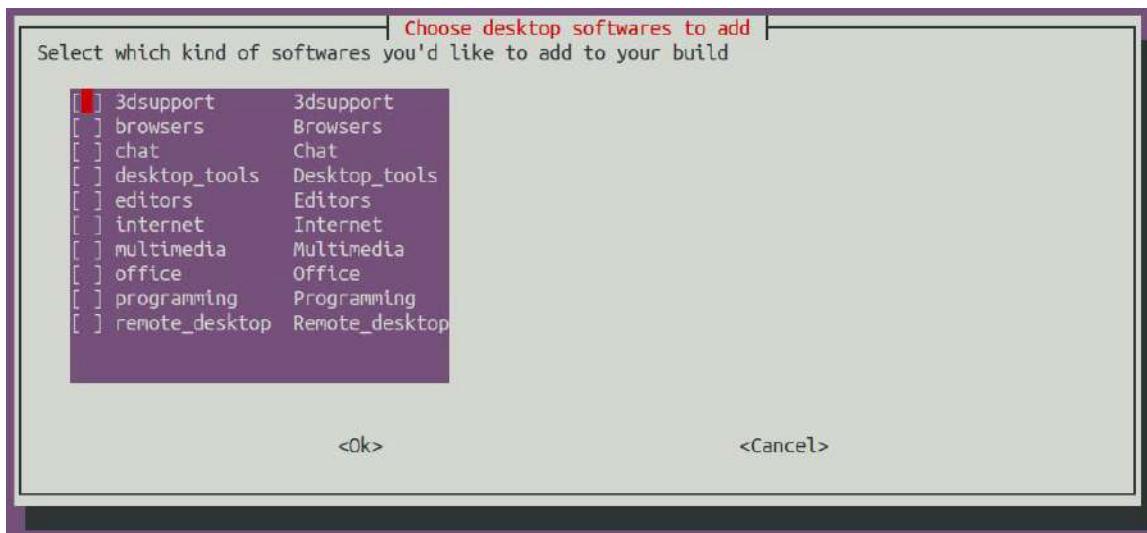
- 6) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.



- 7) If you are compiling the desktop version of the image, you also need to select the type of desktop environment, but only XFCE is currently supported, so just press Enter.



You can then select additional packages that need to be installed. For example, if you need to install a browser, you can choose **browsers**.



Which packages are included in each selection can be seen in the code of orangepi-build. You can also modify these configuration files to add the packages you want to install:

- a. First enter the **external/config/desktop** directory to see the desktop configuration folders of different linux distributions. Note that not all the Orange Pi development boards that can be seen in the code are supported and tested.

```
test@test:~/orangepi-build$ cd external/config/desktop
test@test:~/orangepi-build/external/config/desktop$ ls
bionic bookworm bullseye buster focal jammy README.md sid
```

- b. Then select the type of distribution you want to view or modify, and enter the corresponding directory, such as **bullseye**

```
test@test:~/orangepi-build/external/config/desktop$ cd bullseye
test@test:~/orangepi-build/external/config/desktop/bullseye$ ls
appgroups environments
```

- c. Then enter the **appgroups** directory to see all app groups

```
test@test:~/orangepi-build/external/config/desktop/bullseye$ cd appgroups
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ ls
3dsupport browsers chat desktop_tools editors internet multimedia
office programming remote_desktop
```

- d. Open the packages file under different groups to view the software contained in the group

```
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat \
```

**programming/packages****build-essential****clang****meld**

test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups\$ cat \

**office/packages****libreoffice**

8) Then it will start to compile the linux image. The general process of compilation is as follows

- a. Initialize the compilation environment of the Ubuntu PC and install the software packages required for the compilation process
- b. Download the source code of u-boot and linux kernel (if cached, only update the code)
- c. Compile u-boot source code and generate u-boot deb package
- d. Compile the linux source code to generate linux-related deb packages
- e. Make a deb package of linux firmware
- f. Make the deb package of the orangepi-config tool
- g. Make board-level supported deb packages
- h. If you are compiling the desktop version of the image, you will also create a desktop-related deb package
- i. Check whether the rootfs has been cached, if there is no cache, then recreate the rootfs, if it has been cached, directly decompress and use
- j. Install the deb package generated earlier into rootfs
- k. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configuration, etc.
- l. Then make an image file and format the partition, the default type is ext4
- m. Copy the configured rootfs to the imageed partition
- n. then update initramfs
- o. Finally, write the bin file of u-boot into the image through the dd command

9) After compiling the image, the following information will be prompted

- a. The storage path of the compiled image

[ o.k. ] Done building

[ output/images/orangepi3-lts\_3.0.0\_debian\_bullseye\_linux5.10.43\_xfce\_desktop/ora

**[ngepi3-lts\_3.0.0\_debian\_bullseye\_linux5.10.43\_xfce\_desktop.img ]**

- b. Compilation time used

**[ o.k. ] Runtime [ 19 min ]**

- c. Repeat the command to compile the image, use the following command to start compiling the image directly without selecting through the graphical interface

**[ o.k. ] Repeat Build Options [ sudo ./build.sh BOARD=orangepi3-lts****BRANCH=current BUILD\_OPT=image RELEASE=bullseye****BUILD\_MINIMAL=no BUILD\_DESKTOP=no KERNEL\_CONFIGURE=yes ]**

## 7. Android 9.0 SDK Instructions

The compilation of the Android SDK is carried out on a PC with Ubuntu 18.04 installed, and other versions of Ubuntu systems may have some differences. If you want to use other versions of Ubuntu to compile the Android source code, please test it yourself.

The H6 Android9.0 SDK provided by Orange Pi is the original SDK released by the chip manufacturer. If you want to use the Android image compiled with the Android9.0 SDK on the Orange Pi 3 LTS, you need to adapt the board to make the compiled Android image in the The Orange Pi 3 LTS can be started normally, but the code adapted from the original SDK of the Orange Pi is not provided.

### 7. 1. Download the source code of Android SDK

- 1) The Android9.0 source code folder of Orange Pi 3 LTS contains the following 4 files
  - a. **android.tar.gz:** android related source code
  - b. **android.tar.gz.md5sum:** MD5 checksum file of android.tar.gz
  - c. **lichee.tar.gz:** Contains u-boot, linux kernel and other source code
  - d. **lichee.tar.gz.md5sum:** MD5 checksum file of lichee.tar.gz

| 返回上一级 / 全部文件 - H6_Android_9.0                  |        |                  |  |
|------------------------------------------------|--------|------------------|--|
| <input type="checkbox"/> 文件名                   | 大小     | 修改日期             |  |
| <input type="checkbox"/> lichee.tar.gz.md5sum  | 488    | 2021-12-27 10:50 |  |
| <input type="checkbox"/> lichee.tar.gz         | 2.24G  | 2021-12-27 10:50 |  |
| <input type="checkbox"/> android.tar.gz.md5sum | 498    | 2021-12-27 10:50 |  |
| <input type="checkbox"/> android.tar.gz        | 17.94G | 2021-12-27 10:50 |  |



- 2) After downloading the Android source code, please check whether the MD5 checksum is correct. If it is not correct, please download the source code again.

```
test@test:~$ md5sum -c android.tar.gz.md5sum
```

**android.tar.gz: Sure**

```
test@test:~$ md5sum -c lichee.tar.gz.md5sum
```

**longan.tar.gz: Sure**

- 3) Then unzip the Android source code

a. android: Store android related source code

b. lichee: Store the source code of the linux kernel and u-boot, as well as other configuration files

```
test@test:~$ tar -zxf android.tar.gz
```

```
test@test:~$ tar -zxf lichee.tar.gz
```

```
test@test:~$ ls
```

android lichee

## 7. 2. Build Android Compilation Environment

- 1) Install JDK

```
test@test:~$ sudo add-apt-repository ppa:openjdk-r/ppa
```

```
test@test:~$ sudo apt-get update
```

```
test@test:~$ sudo apt-get install openjdk-8-jdk
```

- 2) Configure JAVA environment variables

a. First determine the installation path of java, generally

```
test@test:~$ ls /usr/lib/jvm/java-8-openjdk-amd64
```

ASSEMBLY\_EXCEPTION bin docs include jre lib man src.zip

THIRD\_PARTY\_README

b. Then use the following command to export java environment variables

```
test@test:~$ export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

```
test@test:~$ export PATH=$JAVA_HOME/bin:$PATH
```

```
test@test:~$ export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

- 3) Install platform support software

```
test@test:~$ sudo apt-get update
```



```
test@test:~$ sudo apt-get install -y 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 u-boot-tools
```

## 7.3. Compile Android image

### 7.3.1. Compile the kernel

1) The compilation method of lichee is as follows

- a. When compiling for the first time, you need to configure the compile options first

```
test@test:~$ cd ichee
test@test:~/lichee$ ./build.sh config
```

Welcome to mkscript setup progress

All available platform:

0. android
1. dragonboard
2. linux
3. camdroid

Choice [android]: **0**

All available chip:

0. sun3iw1p1
1. sun50iw1p1
2. sun50iw2p1
3. sun50iw3p1
4. sun50iw6p1
5. sun50iw8p1
6. sun8iw10p1
7. sun8iw11p1
8. sun8iw12p1
9. sun8iw15p1
10. sun8iw17p1
11. sun8iw1p1
12. sun8iw3p1



- 13. sun8iw5p1
- 14. sun8iw6p1
- 15. sun8iw7p1
- 16. sun8iw8p1
- 17. sun8iw9p1
- 18. sun9iw1p1

Choice [sun50iw6p1]: **4**

All available kern\_ver:

- 0. linux-4.9

Choice [linux-4.9]: **0**

All available board:

- 0. fpga
- 1. perf1\_v1\_0
- 2. perf2\_v1\_0
- 3. perf3\_v1\_0
- 4. petrel-cmcc-p1
- 5. petrel-h603-axpdummy
- 6. petrel-iptv-p1
- 7. petrel-p1-axp802
- 8. petrel-p1-axpdummy
- 9. petrel-p1
- 10. pro\_v1\_0
- 11. qc
- 12. sata

Choice [petrel-p1]: **9**

b. Then start compiling

```
test@test:~/lichee$ ./build.sh
```

2) After the compilation is successful, the output content is as follows

```
sun50iw6p1 compile Kernel successful
```

```
INFO: build kernel OK.
```

```
INFO: build rootfs ...
```

```
INFO: skip make rootfs for android
```



```
INFO: build rootfs OK.  
INFO: -----  
INFO: build lichee OK.  
INFO: -----
```

### 7. 3. 2. Compile Android source code

- 1) The command to compile Android is as follows

```
test@test:~$ cd android  
test@test:~/android$ source build/envsetup.sh  
test@test:~/android$ lunch petrel_fvd_p1-eng  
test@test:~/android$ extract-bsp  
test@test:~/android$ make -j8  
test@test:~/android$ pack
```

- 2) The pack command is used to package and generate Android firmware. If the compilation and packaging process passes smoothly, the following information will be prompted

```
Dragon execute image.cfg SUCCESS !  
-----image is at-----  
lichee/tools/pack/sun50iw6p1_android_petrel-p1_uart0.img  
  
pack finish
```

- 3) The path where the generated Android image is stored is

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