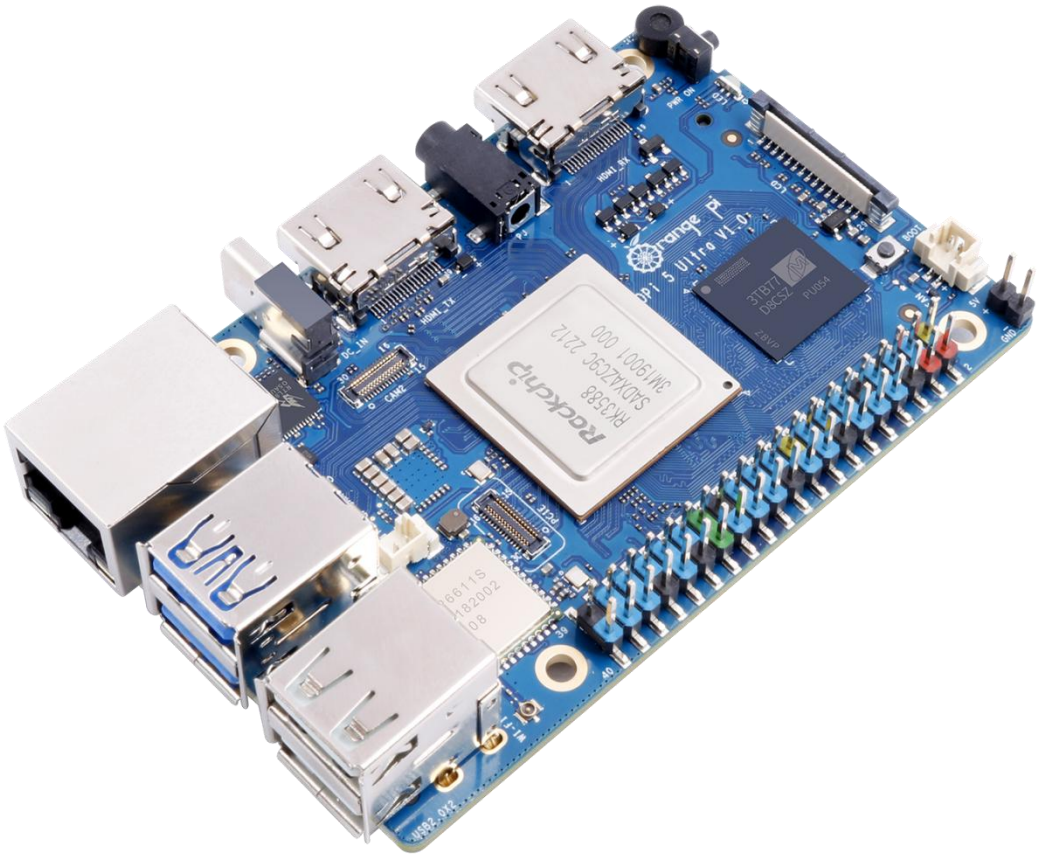
Orange Pi 5 Ultra

User Manual



Catalogue

1. Basic features of Orange Pi 5 Ultra 11
1.1. What is Orange Pi 5 Ultra 11
1.2. Purpose of Orange Pi 5 Ultra 11
1.3. Orange Pi 5 Ultra hardware features 12
1.4. Orange Pi 5 Ultra top and bottom views 14
1.5. Orange Pi 5 Ultra interface details 16
2. Introduction to the use of development board 18
2.1. Prepare the necessary accessories 18
2.2. Download the development board image and related information 22
2.3. How to burn Linux image to TF card based on Windows PC 25
2.3.1. How to burn Linux image using balenaEtcher 25
2.3.2. How to use RKDevTool to burn Linux image to TF card 29
2.4. Method of burning Linux image to TF card based on Ubuntu PC 37
2.5. Method for burning Linux images to eMMC 41
2.5.1. Method of burning Linux images to eMMC using RKDevTool 41
2.5.2. Method of burning Linux images to eMMC using the dd command 50
2.6. Method for burning Linux images to SPIFlash+NVMe SSD 53
2.6.1. Method of burning using RKDevTool 53
2.6.2. Method of burning with dd command 62
2.6.3. Method of burning using balenaEtcher software 67
2.7. Method for burning Linux images to SPIFlash+USB storage devices 83
2.8. Method of burning Android image to TF card 86
2.8.1. Method of burning using RKDevTool 86
2.9. Method for burning Android images to eMMC 92
2.9.1. Method of burning using RKDevTool 92
2.10. Method for burning Android images to SPIFlash+NVMe SSD 97

2. 11. Method for burning Orange Pi OS (Droid) image to TF card 102
2. 12. Burn Orange Pi OS (Droid) image to SPIFlash+NVMe SSD 105
2. 13. How to clear SPIFlash using RKDevTool 110
2. 14. Start the Orange Pi Development Board 117
2. 15. How to use the debug serial port 118
2. 15. 1. Connection Instructions for Debug Serial Port 118
2. 15. 2. How to use the debug serial port on Ubuntu platform 119
2. 15. 3. How to use the debug serial port on Windows platform 122
2. 16. Power supply instructions using the 5V pin in the 40pin interface of the development board 126
3. Instructions for using Ubuntu/Debian Server and Xfce desktop system 127
3. 1. Supported Linux image types and kernel versions 127
3. 2. 3.2. Linux 5.10 system compatibility 128
3. 3. Linux 6.1 system compatibility 129
3. 4. Linux command format description in this manual 130
3. 5. Linux system login instructions 132
3. 5. 1. 3.5.1.Linux system default login account and password 132
3. 5. 2. 3.5.2. How to set up automatic login for Linux system terminal 132
3. 5. 3. Linux desktop system automatic login instructions 133
3. 5. 4. How to set up automatic login for root user in Linux desktop system 134
3. 5. 5. How to disable the desktop in Linux desktop system 135
3. 6. Onboard LED light test instructions 135
3. 7. Network connection test 138
3. 7. 1. Ethernet port test 138
3. 7. 2. WIFI connection test 140
3. 7. 3. How to set a static IP address 148
3. 7. 4. How to create a WIFI hotspot through create_ap 155
3. 8. SSH remote login development board 162
3. 8. 1. SSH remote login to the development board under Ubuntu 162

3.8.2. SSH remote login development board under Windows 163
3.9. How to use ADB 165
3.9.1. How to use network adb 165
3.9.2. Use a USB male-to-male data cable to connect to adb 167
3.10. How to upload files to the Linux system of the development board 169
3.10.1. How to upload files from Ubuntu PC to the Linux system of the development board 169
3.10.2. How to upload files from Windows PC to the Linux system of the development board 173
3.11. HDMI test 178
3.11.1. HDMI display test 178
3.11.2. HDMI resolution setting method 179
3.11.3. Test method for HDMI RX 181
3.12. How to use Bluetooth 183
3.12.1. Testing methods for desktop images 183
3.13. USB interface test 186
3.13.1. Test by connecting USB mouse or keyboard 186
3.13.2. Test by connecting USB storage device 186
3.13.3. USB wireless network card test 187
3.13.4. USB camera test 195
3.14. Audio Test 197
3.14.1. Testing Audio Methods on Desktop Systems 197
3.14.2. How to play audio using commands 199
3.14.3. How to test recording using commands 201
3.15. Temperature sensor 201
3.16. 40 Pin Interface Pin Description 203
3.17. How to install wiringOP 205
3.18. 40pin interface GPIO, I2C, UART, SPI, CAN and PWM test 207
3.18.1. 40pin GPIO port test 207
3.18.2. 40 How to set pull-up and pull-down resistors on GPIO pin 208
3.18.3. 40pin SPI test 210

3.18.4. 40pin I2C test 213
3.18.5. 40pin UART test 217
3.18.6. How to test PWM using /sys/class/pwm 219
3.18.7. CAN test methods 224
3.19. How to use wiringOP hardware PWM 231
3.19.1. How to set PWM using wiringOP's gpio command 232
3.19.2. How to use the PWM test program 237
3.20. Installation and use of wiringOP-Python 238
3.20.1. Installation of wiringOP-Python 239
3.20.2. 40pin GPIO port test 241
3.20.3. 40pin SPI test 243
3.20.4. 4.40pin I2C test 247
3.20.5. 40pin UART test 250
3.21. Hardware watchdog test 253
3.22. Check the serial number of the RK3588 chip 254
3.23. How to install Docker 254
3.24. How to download and install the arm64 version of balenaEtcher 255
3.25. How to install Baota Linux Panel 257
3.26. Set up Chinese environment and install Chinese input method 262
3.26.1. Installation method of Debian system 262
3.26.2. How to install Ubuntu 20.04 system 268
3.26.3. Installation method for Ubuntu 22.04 system 272
3.27. How to remotely log in to the Linux system desktop 278
3.27.1. Remote login using NoMachine 278
3.27.2. Remote login using VNC 282
3.28. Test of some programming languages supported by Linux system 284
3.28.1. Debian Bullseye System 284
3.28.2. Debian Bookworm System 286
3.28.3. Ubuntu Focal system 287
3.28.4. Ubuntu Jammy System 289
3.29. How to install QT 291

3.30. ROS installation method 299
3.30.1. How to install ROS 1 Noetic on Ubuntu 20.04 299
3.30.2. How to install ROS 2 Galactic on Ubuntu 20.04 303
3.30.3. How to install ROS 2 Humble on Ubuntu 22.04 306
3.31. How to install kernel header files 308
3.32. How to use 10.1 inch MIPI LCD screen 311
3.32.1. 10.1 inch MIPI screen assembly method 311
3.32.2. How to open the 10.1-inch MIPI LCD screen configuration 313
3.32.3. How to rotate the display direction of the server version image 315
3.32.4. Desktop version mirroring rotation display and touch direction method 316
3.33. Instructions for using the power on/off logo 318
3.34. Test Methods for OV13850 and OV13855 MIPI Cameras 319
3.35. How to use ZFS file system 325
3.35.1. How to install ZFS 325
3.35.2. How to create a ZFS pool 326
3.35.3. Test ZFS data deduplication function 328
3.35.4. Test ZFS data compression function 329
3.36. How to install and use CasaOS 330
3.36.1. How to install CasaOS 330
3.36.2. How to use CasaOS 330
3.37. Methods of using NPU 339
3.37.1. Prepare tools 339
3.37.2. Install RKNN-Toolkit2 on Ubuntu PC 339
3.37.3. Model conversion and model inference using RKNN-Toolkit2 340
3.37.4. Call the C interface to deploy the RKNN model to the development board to run 346
3.38. How RK3588 uses Baidu Feijiang 349
3.38.1. Ubuntu PC environment setup 349
3.38.2. Board environment construction 353
3.38.3. Example of deploying a model using FastDeploy 357
3.39. RK3588 How to run the RKLLM large model 361

3.39.1. Introduction to RKLLM 361
3.39.2. Prepare tools 364
3.39.3. Detailed steps for model conversion and source code compilation on Ubuntu PC 364
3.39.4. Detailed steps for development board deployment and operation 376
3.39.5. Detailed steps for deploying and running the server on the development board 388
3.39.6. Performance test results of RK3588 running RKLLM large model . 399
3.40. How to shut down and restart the development board 403
4. Orange Pi OS Arch system usage instructions 405
4.1. Orange Pi OS Arch system compatibility 405
4.2. 10.1-inch MIPI LCD screen usage 406
4.2.1. 10.1 inch MIPI screen assembly method 406
4.2.2. How to open the 10.1-inch MIPI LCD screen configuration 409
4.2.3. Methods for rotating display and touch direction 410
4.3. Test methods for OV13850 and OV13855 MIPI cameras 413
4.4. How to set up the Chinese environment and install the Chinese input method 417
4.5. How to install wiringOP 425
4.6. 40pin interface GPIO, I2C, UART, SPI, CAN and PWM test 427
4.6.1. 40pin GPIO port test 427
4.6.2. 40Pin GPIO port pull-up and pull-down resistor settings 429
4.6.3. 40pin SPI test 430
4.6.4. 40pin I2C test 432
4.6.5. 40pin UART Test 434
4.6.6. PWM Testing Method 436
4.6.7. Testing Methods for CAN 439
5. Linux SDK - Orangepi build usage instructions 442
5.1. Compile System Requirements 442
5.1.1. Compile using the Ubuntu 22.04 system on the development board .. 442
5.1.2. Compile on Ubuntu 22.04 computer using x64 442
5.2. Obtain the source code of Linux SDK 445
5.2.1. Download Orangepi build from GitHub 445

5.2.2. Download the cross compilation toolchain 447
5.2.3. Explanation of the complete directory structure of orangepi build 448
5.3. Compiling u-boot 449
5.4. Compiling Linux Kernel 453
5.5. Compile rootfs 458
5.6. Compiling Linux Images 462
6. Instructions for using Android 13 system 466
6.1. Supported Android versions 466
6.2. Adaptation of Android Features 466
6.3. WIFI connection testing method 467
6.4. Bluetooth testing method 470
6.5. 10.1 inch MIPI screen usage method 473
6.6. Testing Method for OV13850 and OV13855 MIPI Cameras 474
6.7. 40pin interface GPIO, UART, SPI, and PWM testing 479
6.7.1. 40pin GPIO port test 479
6.7.2. 40pin UART Test 483
6.7.3. 40pin SPI test 486
6.7.4. 40 pin PWM test 488
6.8. Usage of ADB 491
6.8.1. Method for Switching USB OTG Mode 491
6.8.2. Using a data cable to connect adb for debugging 494
6.8.3. Debugging with network connection adb 495
6.9. Testing Method for HDMI RX 496
7. Compilation method of Android 13 source code 498
7.1. Download the source code of Android 13 498
7.2. Compile the source code for Android 13 499
8. OpenWRT System User Manual 500
8.1. OpenWRT version 500
8.2. OpenWRT adaptation situation 500

8.3. The first boot to expand rootfs 501
8.4. Method of logging into the system 501
8.4.1. Login via serial port 501
8.4.2. Login to the system via SSH 502
8.4.3. Login to LuCI Management Interface 503
8.4.4. Log in to the terminal through the LuCI management interface 504
8.4.5. Login to the terminal using IP address and port number 506
8.5. Method of modifying LAN port IP address through command line 507
8.6. Method for changing root password 509
8.6.1. Modify via Command Line 509
8.6.2. Modify through LuCI management interface 509
8.7. USB interface testing 511
8.7.1. Mounting USB storage devices at the command line 511
8.7.2. Mounting USB storage devices on the LuCI management interface .. 512
8.8. USB Wireless Network Card Test 515
8.8.1. Method of using a USB wireless network card to connect to a WIFI hotspot 516
8.8.2. Method for creating a WIFI hotspot using a USB wireless network card519
8.9. Installing software packages through the command line 524
8.9.1. Installing through OPkg on the terminal 524
8.10. OpenWRT management interface installation software package 525
8.10.1. View the list of available software packages in the system 525
8.10.2. Example of Installing Software Packages 526
8.10.3. Example of Removing Software Packages 528
8.11. Using Samba Network Sharing 530
8.12. Zerotier User Manual 533
9. Compilation method of OpenWRT source code 537
9.1. Download OpenWRT source code 537
9.2. Compile OpenWRT source code 537
10. Appendix 539

10. 1. User Manual Update History 539

10. 2. Image update history 539

1. Basic features of Orange Pi 5 Ultra

1. 1. What is Orange Pi 5 Ultra

Orange Pi 5 Ultra uses Rockchip RK3588, a new generation of octa-core 64-bit ARM processor, which includes quad-core A76 and quad-core A55. It adopts Samsung's 8nm LP process technology, with a large core frequency of up to 2.4GHz. It integrates ARM Mali-G610 MP4 GPU, embedded high-performance 3D and 2D image acceleration modules, built-in AI accelerator NPU with up to 6 Tops computing power, 4GB/8GB/16GB (LPDDR5) memory, and has up to 8K display processing capabilities.

Orange Pi 5 Ultra introduces a wide range of interfaces, including HDMI output, HDMI input, Wi-Fi6, M.2 M-key PCIe3.0x4, 2.5G network port, USB2.0, USB3.1 interface and 40pin expansion pin header, etc. It can be widely used in high-end tablets, edge computing, artificial intelligence, cloud computing, AR/VR, smart security, smart home and other fields, covering all industries of AIoT.

Orange Pi 5 Ultra supports Orange Pi OS, the official operating system developed by Orange Pi. It also supports operating systems such as Android 13, Debian11, Debian12, Ubuntu20.04 and Ubuntu22.04.

1. 2. Purpose of Orange Pi 5 Ultra

We can use it to achieve:

- A Linux desktop computer.
- A Linux network server.
- Android tablet.
- Android game consoles, etc

Of course, there are many more functions. Relying on a powerful ecosystem and a wide range of expansion accessories, Orange Pi can help users easily realize the delivery from creativity to prototype to mass production. It is an ideal creative platform for makers, dreamers and amateurs.

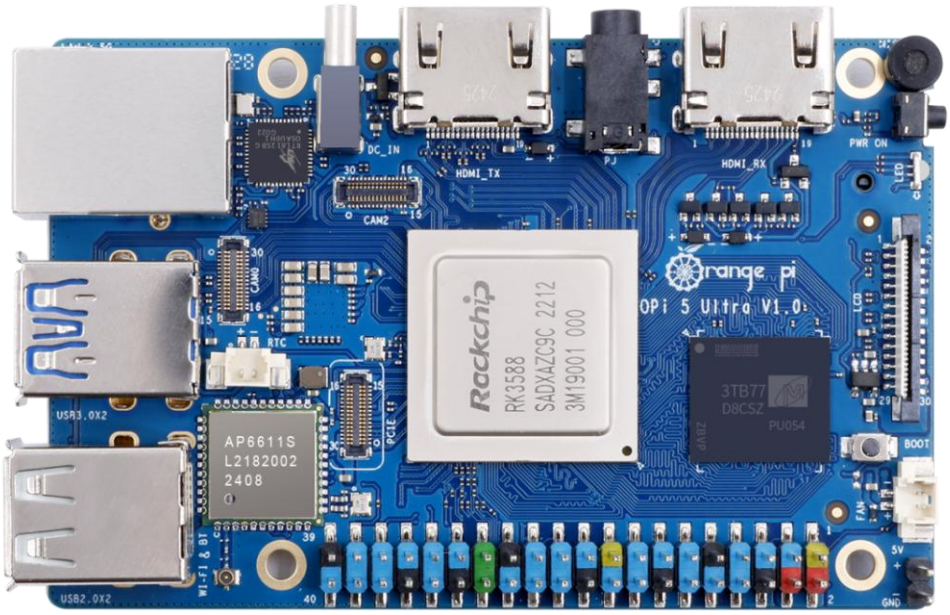
1. 3. Orange Pi 5 Ultra hardware features

Hardware Features	
CPU	• Rockchip RK3588 (8nm LP process) • 8-core 64-bit processor • Quad-core Cortex-A76 and quad-core Cortex-A55 big and small core architecture • The main frequency of the big core is up to 2.4GHz, and the main frequency of the small core is up to 1.8GHz
GPU	• Integrated ARM Mali-G610 • OpenGL ES1.1/2.0/3.2, OpenCL 2.2 and Vulkan 1.2
NPU	• Built-in AI accelerator NPU with up to 6 Tops computing power • Support INT4/INT8/INT16 mixed operations
video	• 1 * HDMI 2.1，Maximum support 8K @60Hz • 1 * HDMI Input, up to4K@60FPS • 1 * MIPI D-PHY TX 4Lane
Memory	4GB/8GB/16GB（LPDDR5）
Camera	• 2 * MIPI CSI 4Lane • 1 * MIPI D-PHY RX 4Lane
PMU	RK806-1
Onboard storage	• eMMC socket, can be connected to external eMMC module • 16MB QSPI Nor FLASH • MicroSD (TF) Card Slot • PCIe3.0x4 M.2 M-KEY (SSD) Slots
Ethernet	1 * PCIe 2.5G Ethernet port（RTL8125BG ）
WIFI+BT	• Onboard Wi-Fi 6E+BT 5.3/BLE Modules：AP6611 • Wi-Fi interface：SDIO3.0 • BT interface：UART/PCM
Audio	• 3.5mm Headphone jack Audio input/output • Onboard MIC input

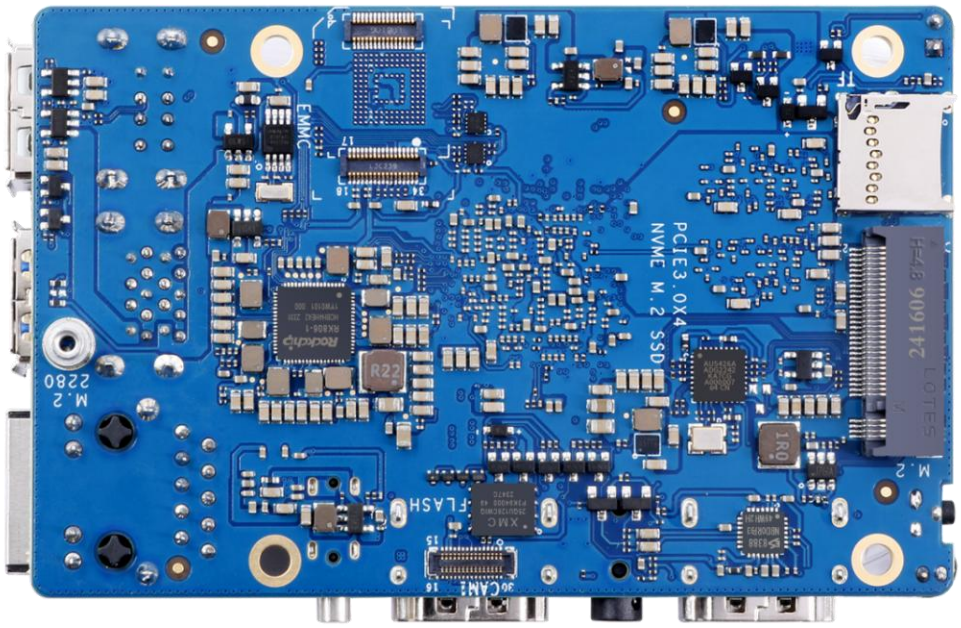
	• 2 * HDMI Output
PCIe M.2 M-KEY	• PCIe 3.0 x 4 lanes，For connecting NVMe SSD
USB interface	• 1 * USB3.0 support Device or HOST model • 1 * USB3.0 HOST • 2 * USB2.0 HOST
40pin Extension pin header	For expansion UART、PWM、I2C、SPI、CAN and GPIO interface
Debug serial port	Included in 40PIN expansion port
LED Light	RGB LED three-color indicator
button	1 * MaskROM Key，1 * Power button
powered by	Type-C Interface power supply 5V/5A
Supported operating systems	Orange Pi OS（Droid）、Orange Pi OS（Arch）、Android13、Debian11、Debian12、Ubuntu20.04 and Ubuntu22.04 Other operating systems
Appearance Specifications	
Product size	89mm*56mm
weight	58g
range Pi™ is a registered trademark of Shenzhen Xunlong Software Co., Ltd.	

1.4. Orange Pi 5 Ultra top and bottom views

Top View:

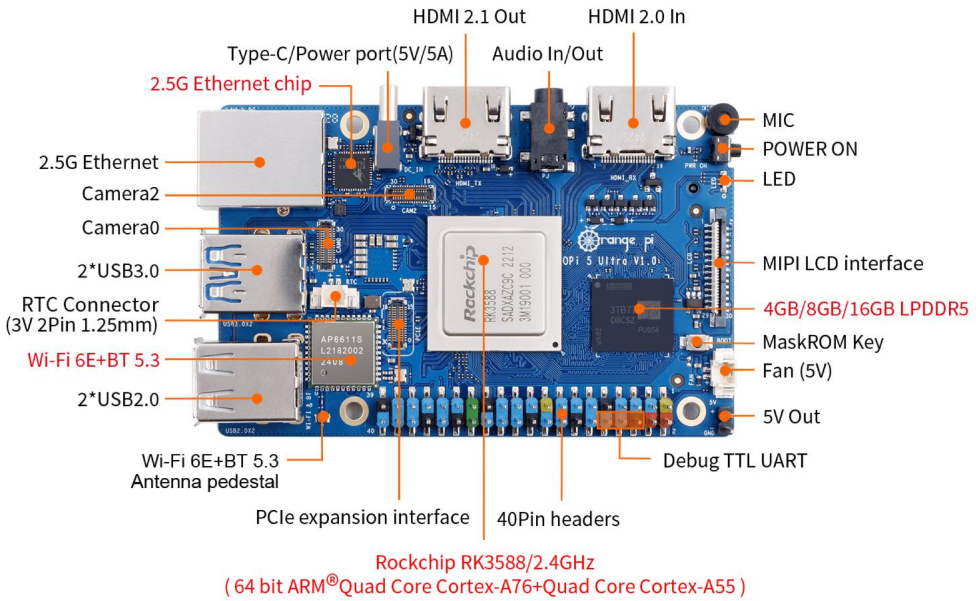


Bottom layer view:

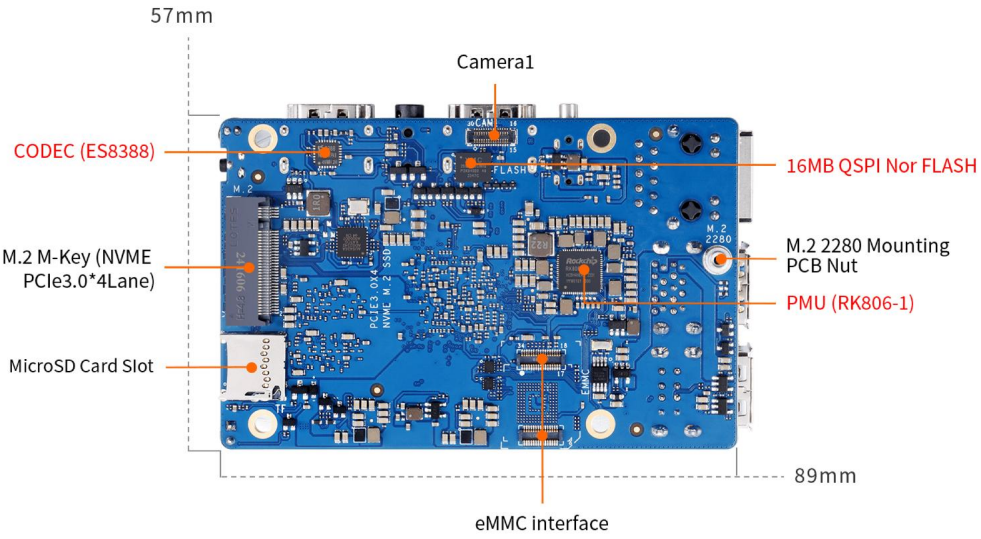

EMMC
PCIE3.0X4
NVME M.2 SSD
M.2
2280
FLASH
CAM1
R22
1R0

1. 5. Orange Pi 5 Ultra interface details

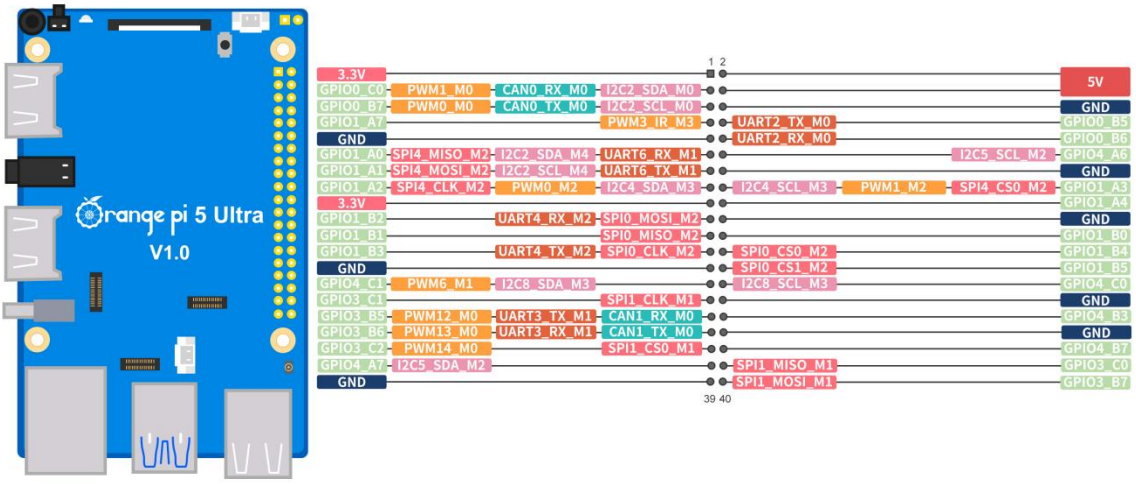
Product display



Top View



Bottom View



The diameter of the four positioning holes is 2.7mm

2. Introduction to the use of development board

2. 1. Prepare the necessary accessories

1) TF card, minimum 16GB capacity (32GB or above recommended), **class 10** or above high-speed SanDisk card.



2) TF card reader, used to burn the image into the TF card.



3) Display with HDMI interface.



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



Note that if you want to connect a 4K or 8K display, please make sure the HDMI cable supports 4K or 8K video output.

5) 10.1-inch MIPI screen, used to display the system interface of the development board (this screen is common to the adapter board and OPi5Plus/OPi5B/OPi5/OPi5Pro/OPi5Max/OPi5Ultra).



6) Power adapter: Orange Pi 5 Ultra recommends using a 5V/5A Type-C power adapter.



The Type-C power interface of the development board does not support the PD negotiation function and only supports a fixed 5V voltage input.

7) A USB mouse and keyboard. Any standard USB mouse and keyboard will do. The mouse and keyboard can be used to control the Orange Pi development board.

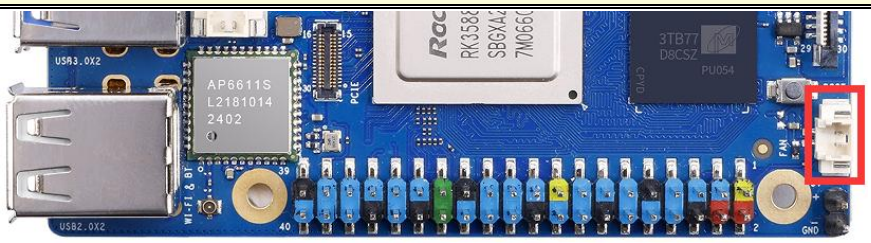


8) USB camera



9) 5V cooling fan. As shown in the figure below, there is an interface for connecting the cooling fan on the development board, and the interface specification is **2pin 1.25mm** spacing.

The fan on the development board can adjust the speed and switch through PWM.



10) 100M or 1000M Ethernet cable, used to connect the development board to the Internet.



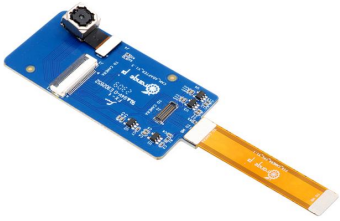
11) USB2.0 male-to-male data cable, used for burning images and using ADB and other functions.



12) 13MP OV13850 camera with MIPI interface.



13) OV13855 camera with 13MP MIPI interface.



14) **3.3V** USB to TTL module and DuPont cable. When using the serial port debugging function, a USB to TTL module and DuPont cable are required to connect the development board and the computer.





15) A personal computer with Ubuntu and Windows operating systems installed.

1	Ubuntu22.04 PC	Optional, used to compile Linux source code
2	Windows PC	Used to burn Android and Linux images

2.2. Download the development board image and related information

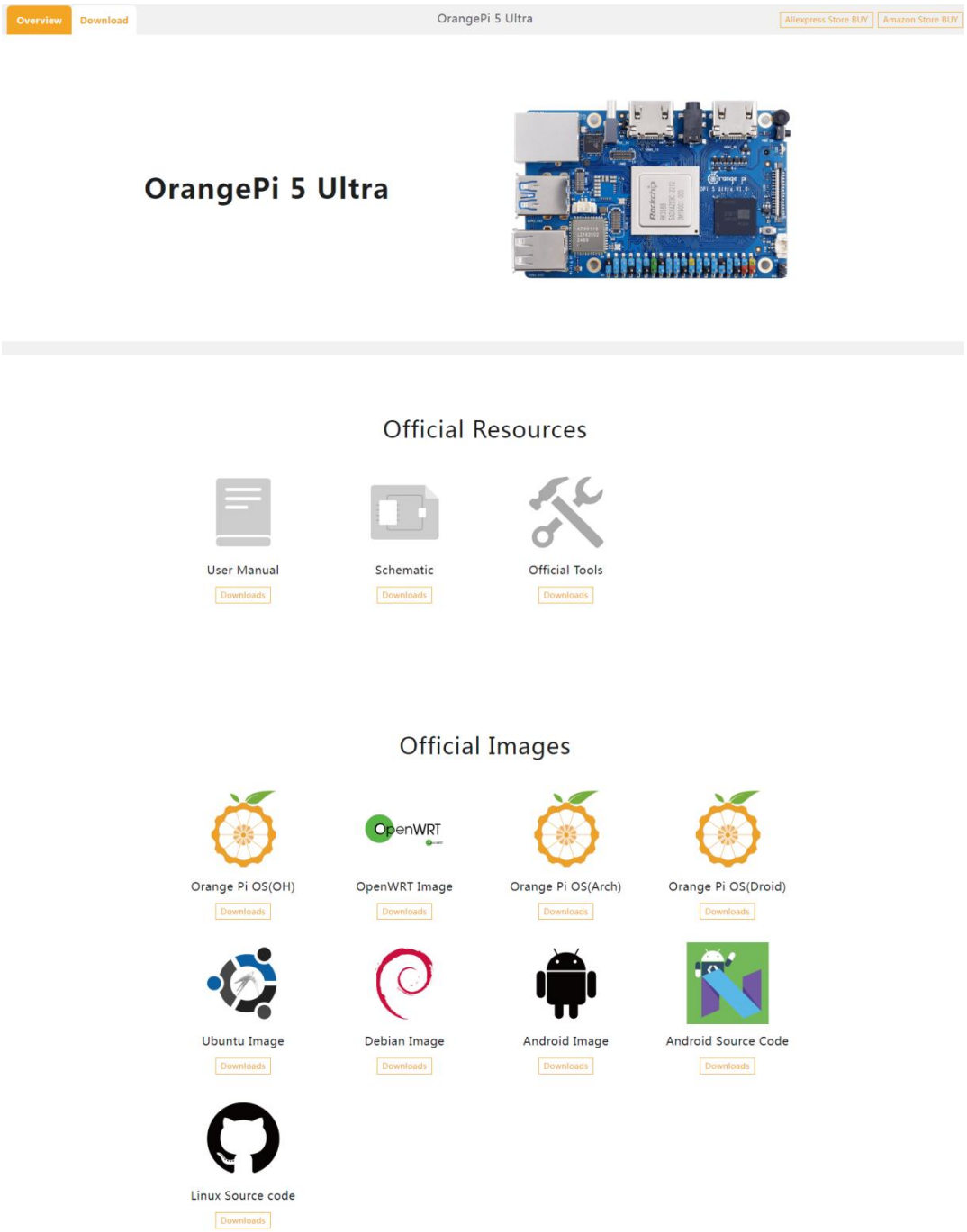
1) The download link for the Chinese version is:

http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-5-Ultra.html



2) The download link for the English version is:

http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-5-Ultra.html



3) The data mainly includes

a. **Android source code：** Save on Baidu Cloud and Google Drive.

b. **Linux source code：** Saved on Github.

c. **User Manual and Schematics：** Save on Baidu Cloud and Google Drive

d. **Official Tools：** It mainly includes the software needed during the use of the development board.

e. **Android Image：** Save on Baidu Cloud and Google Drive.

f. **Ubuntu Image：** Save on Baidu Cloud and Google Drive.
g. **Debian Image：** Save on Baidu Cloud and Google Drive.
h. **Orange Pi OS Image：** Save on Baidu Cloud and Google Drive.
i. **OpenWRT Image：** Save on Baidu Cloud and Google Drive.

2. 3. How to burn Linux image to TF card based on Windows PC

Note that the Linux image mentioned here specifically refers to the Linux distribution image such as Debian, Ubuntu, OpenWRT or OPi OS Arch downloaded from the Orange Pi download page.

2. 3. 1. How to burn Linux image using balenaEtcher

1) First, prepare a TF card with a capacity of 16GB or larger. 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.

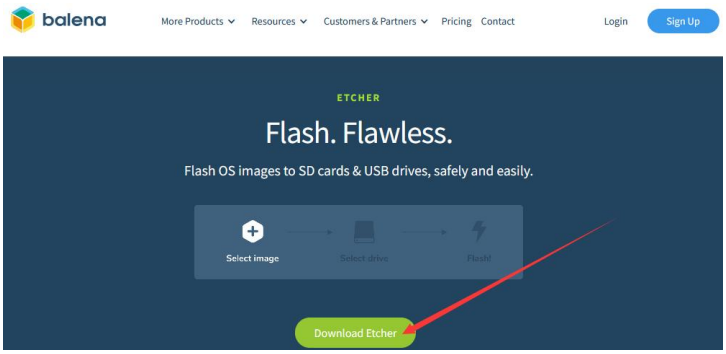
2) Then use the card reader to insert the TF card into the computer.

3) Download the compressed Linux operating system image file you want to burn from the **Orange Pi data download page**, and then use decompression software to decompress it. Among the decompressed files, the file ending with **".img"** is the operating system image file, which is generally more than 2GB in size.

4) Then download the burning software for the Linux image——**balenaEtcher**，Download address:

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

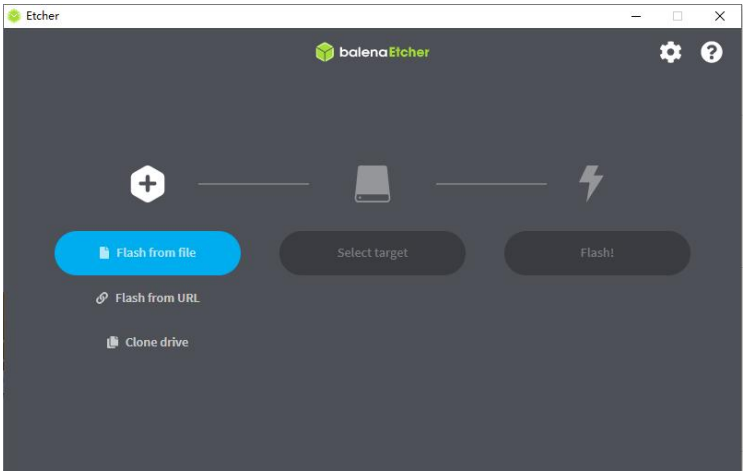
5) After entering the balenaEtcher download page, click the green download button to jump to the software download location.



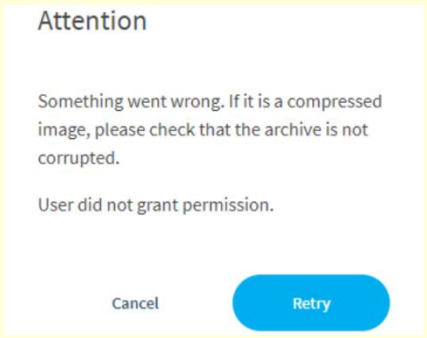
6) Then you can choose to download the Portable version of balenaEtcher. The Portable version does not need to be installed, just double-click to open it and you can use it.



7) If you downloaded the version of balenaEtcher that needs to be installed, please install it before using it. If you downloaded the portable version of balenaEtcher, just double-click to open it. The interface of balenaEtcher after opening is as shown below:



If the following error is prompted when opening balenaEtcher:



Select balenaEtcher, right-click it, and then choose Run as administrator.



8) The specific steps to use balenaEtcher to burn the Linux image are as follows:

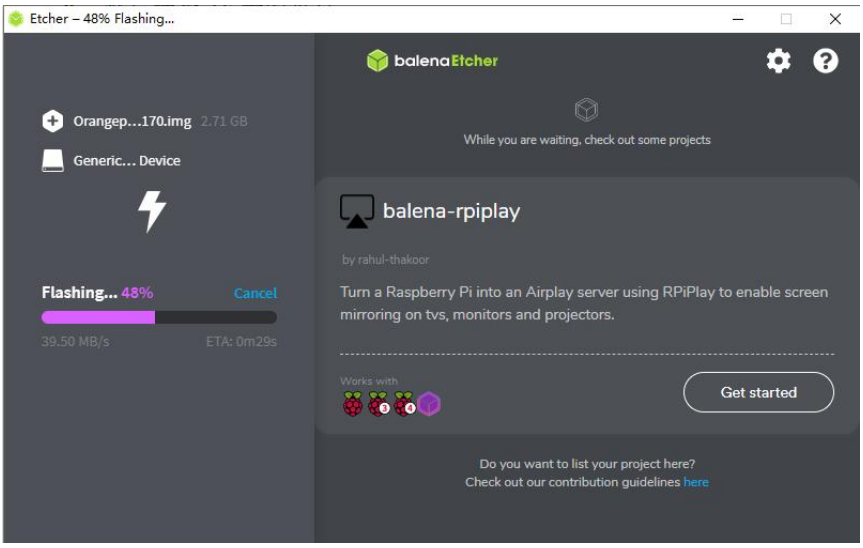
a. First select the path of the Linux image file to be burned.

b. Then select the drive letter of the TF card.

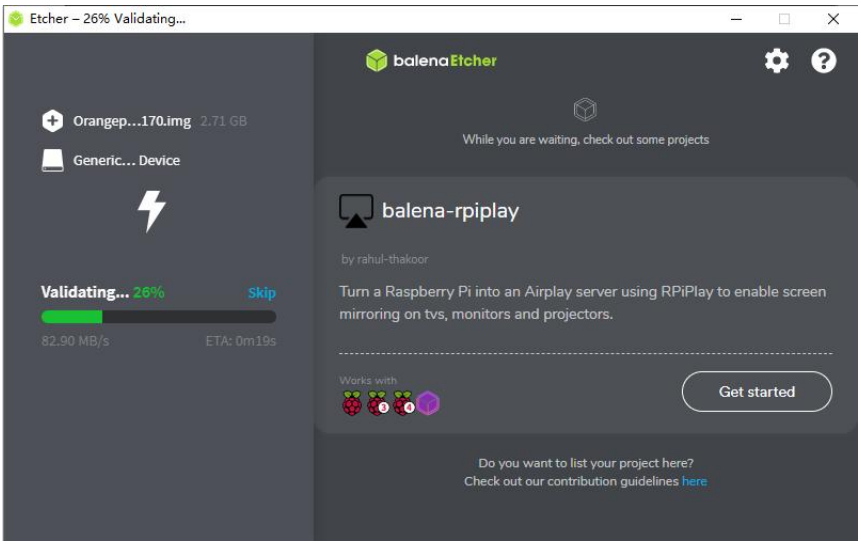
c. Finally, click Flash to start burning the Linux image to the TF card.



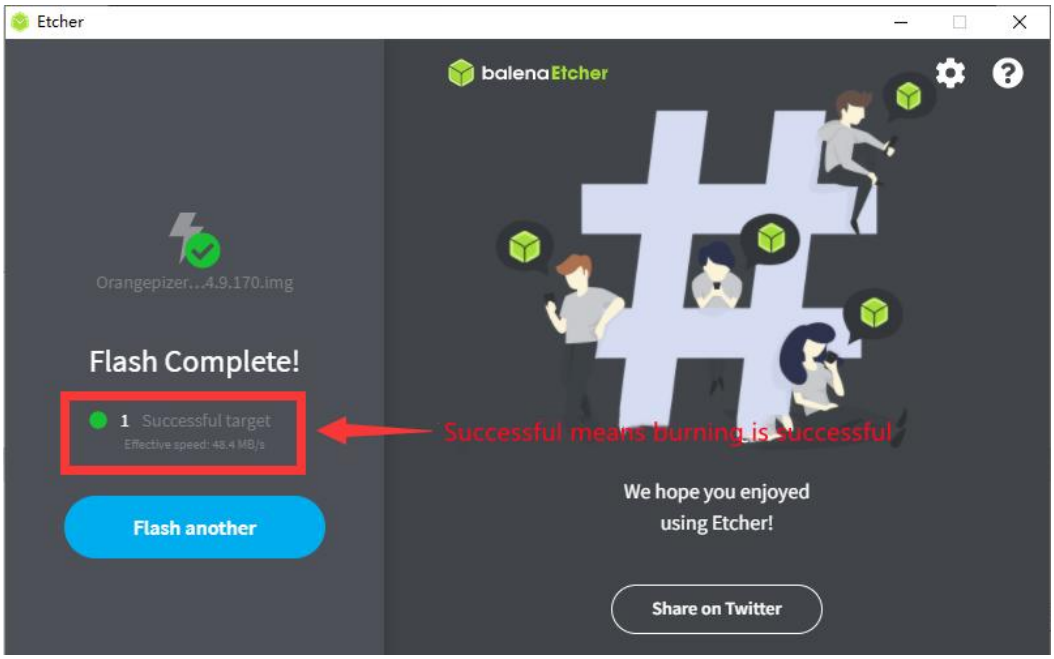
9) The interface displayed by balenaEtcher during burning Linux image is shown in the figure below. In addition, the progress bar is purple, indicating that the Linux image is being burned to the TF card.



10) After the Linux image is burned, balenaEtcher will verify the image burned to the TF card by default to ensure that there are no problems 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.



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



2. 3. 2. How to use RKDevTool to burn Linux image to TF card

1) First, you need to prepare a good quality USB2.0 male-to-male data cable.



2) You also need to prepare a TF card with a capacity of 16GB or larger. 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.

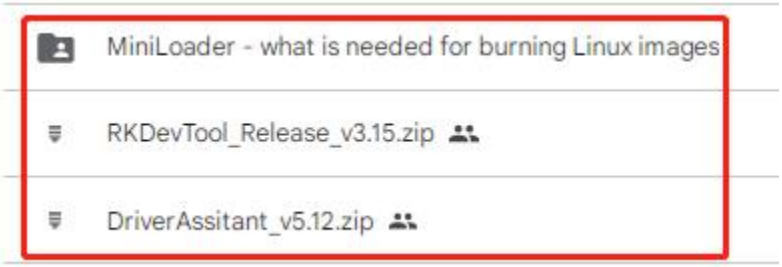
3) Then insert the TF card into the card slot of the development board.

4) Then download Rockchip driver **DriverAssitant_v5.12.zip** and **MiniLoader** and burning tool **RKDevTool_Release_v3.15.zip** from **Orange Pi's download page**.

a. On the Orange Pi download page, first select the **official tool**, then go to the folder below.



b. Then download all the files below.



Note that the "MiniLoader-things needed for burning Linux images" folder is referred to as the MiniLoader folder below.

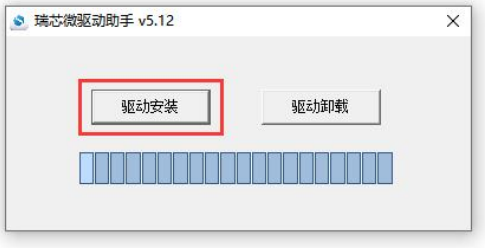
5) Then download the compressed file of the Linux operating system image file you want to burn from the **Orange Pi data download page**, and then use the decompression software to decompress it. Among the decompressed files, the file ending with **".img"** is the image file of the operating system, and the size is generally more than 2GB.

6) Then use the decompression software to decompress **DriverAssitant_v5.12.zip**, find the **DriverInstall.exe** executable file in the decompressed folder and open it.

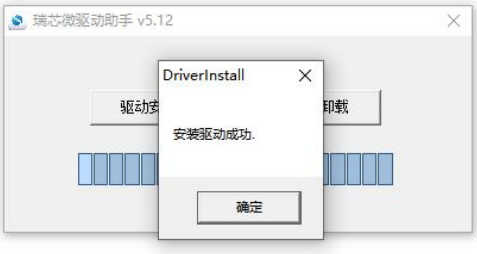


7) After opening **DriverInstall.exe**, the steps to install the Rockchip driver are as follows:

a. Click the "**Driver Installation**" button.



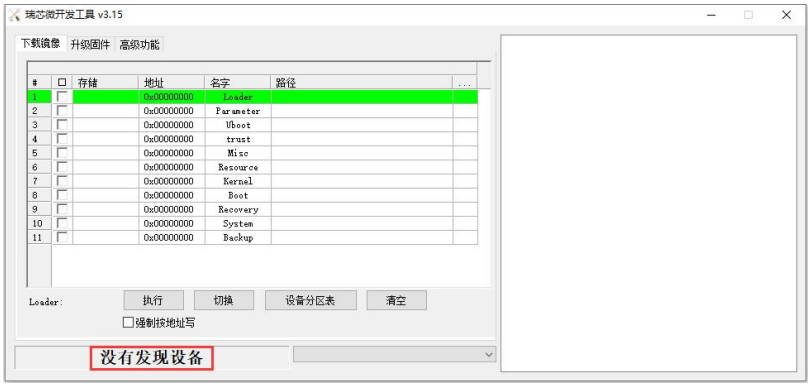
b. After waiting for a while, a window will pop up prompting "**Driver installation successful**", then click the "**OK**" button.



8) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed. Just find **RKDevTool** in the unzipped folder and open it.

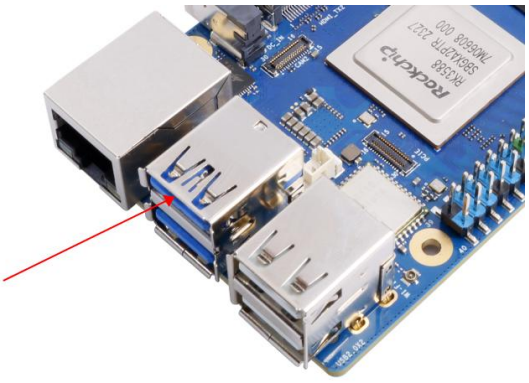


9) After opening the **RKDevTool** burning tool, because the computer has not yet connected to the development board via the USB2.0 male-to-male data cable, the lower left corner will prompt "**No device found**".



10) Then start burning the Linux image to the TF card.

a. First, connect the development board to the Windows computer via a USB male-to-male data cable. The location of the USB burning port on the development board is shown in the figure below:

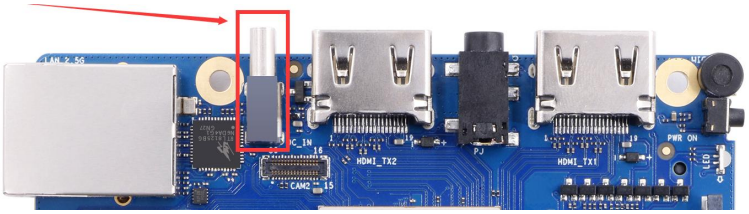


b. Then insert the TF card into the development board and make sure the development board is not connected to the power supply.

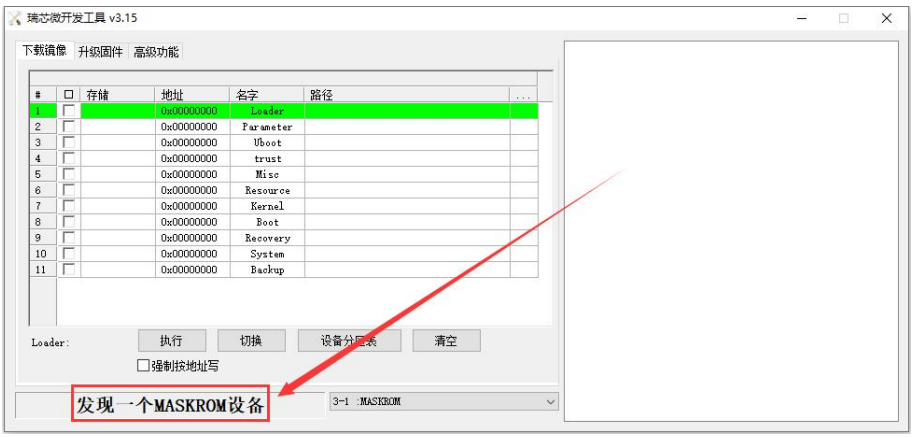
c. Then press and hold the MaskROM button on the development board. The location of the MaskROM button on the development board is shown in the figure below:



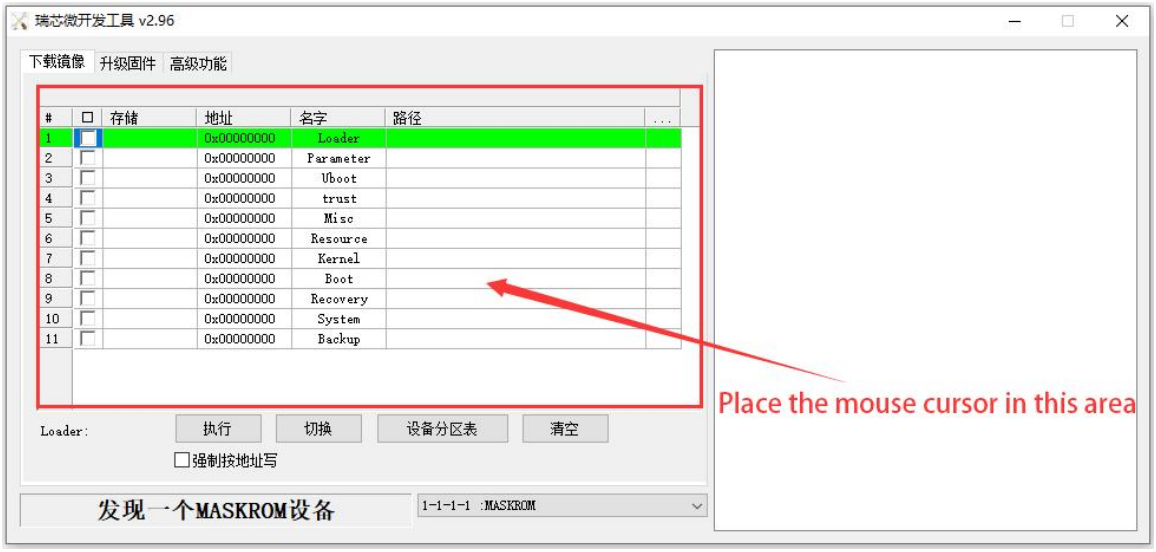
d. Then connect the development board to the power supply of the Type-C interface and power it on. Then you can release the MaskROM button.



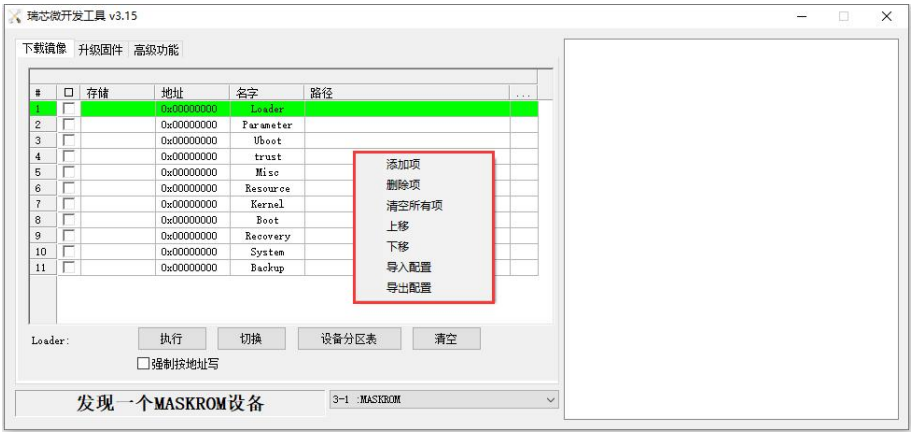
e. If the previous steps go smoothly, the development board will enter the **MASKROM** mode and the burning tool interface will prompt "**A MASKROM device is found**".



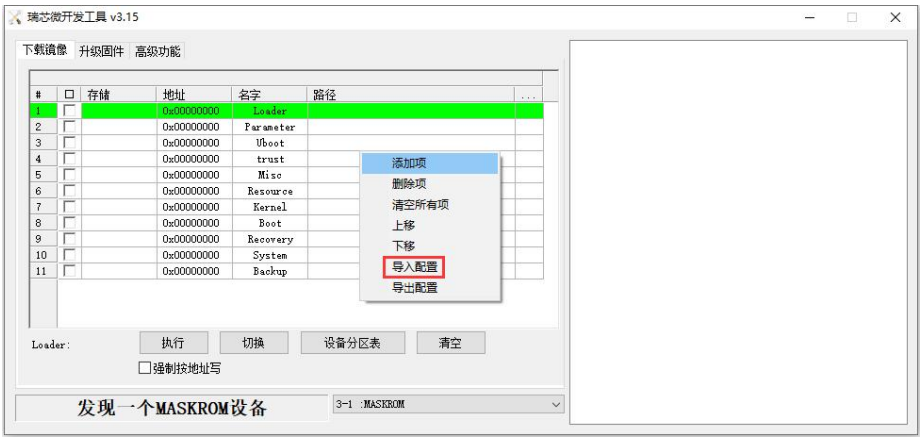
f. Then place the mouse cursor in the area below



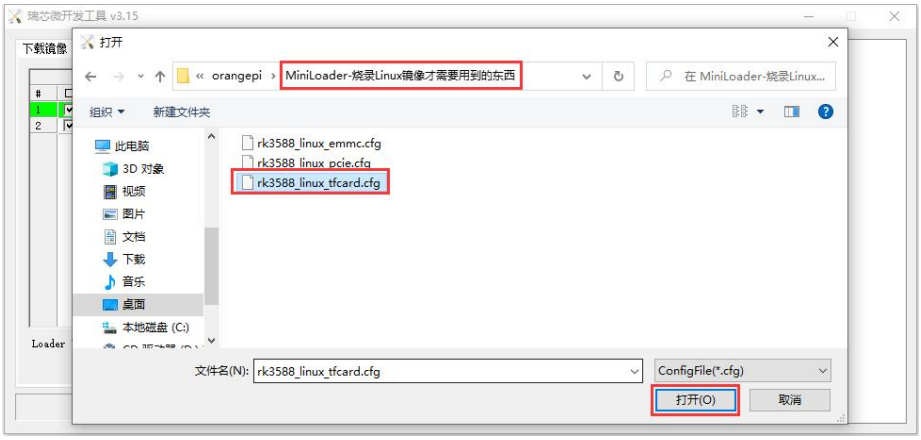
g. Then click the right button of the mouse to pop up the selection interface shown in the figure below.



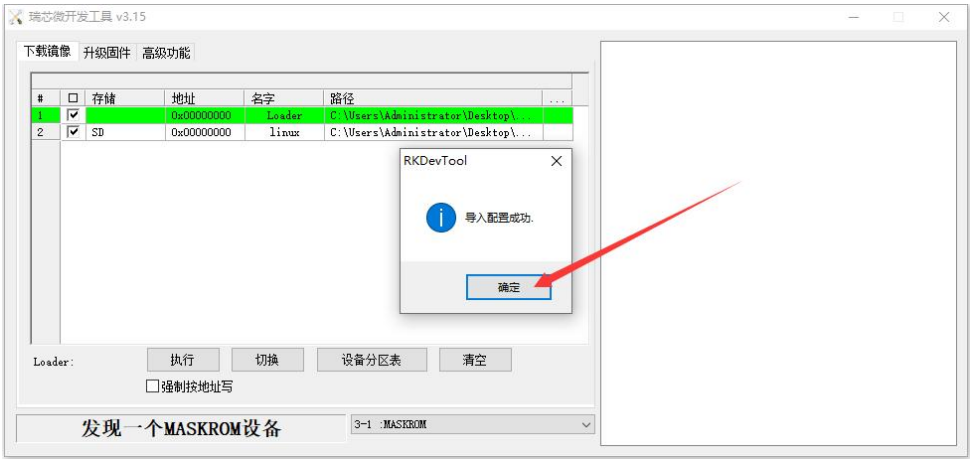
h. Then select the **Import Configuration** option.



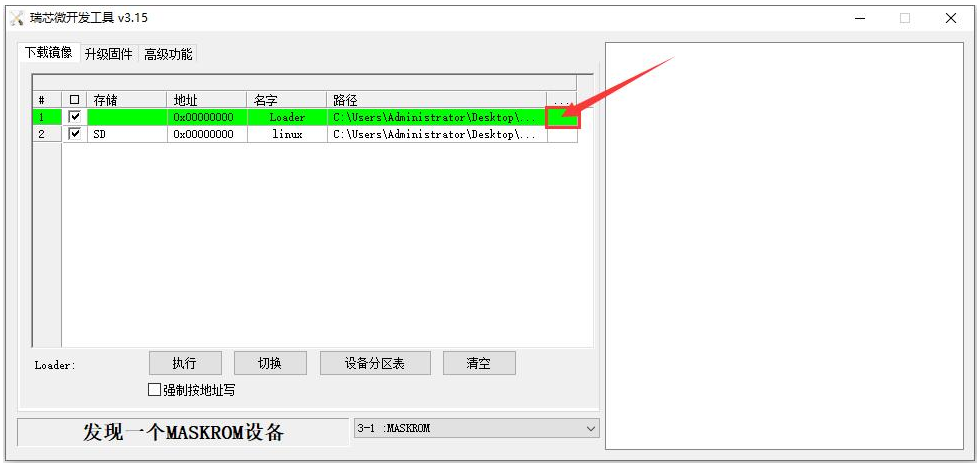
i. Then select the **rk3588_linux_tfcard.cfg** configuration file in the **MiniLoader** folder downloaded earlier, and then click **Open**.



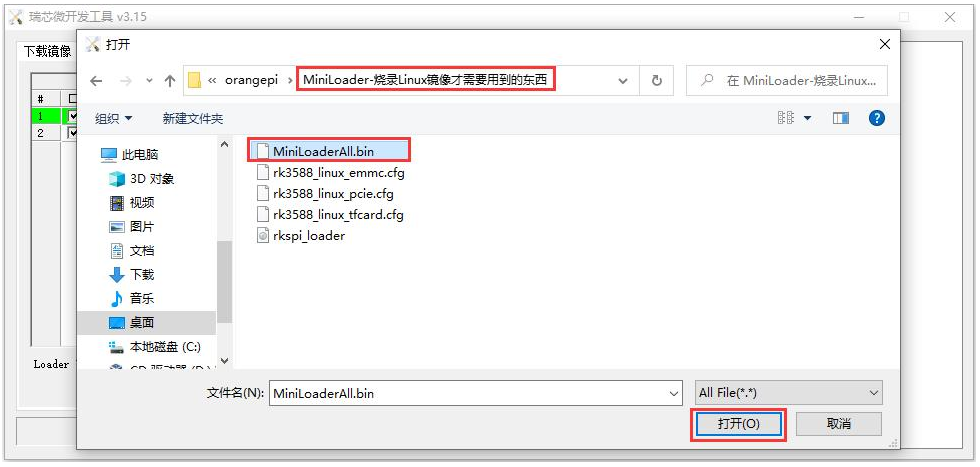
j. Then click **OK.**



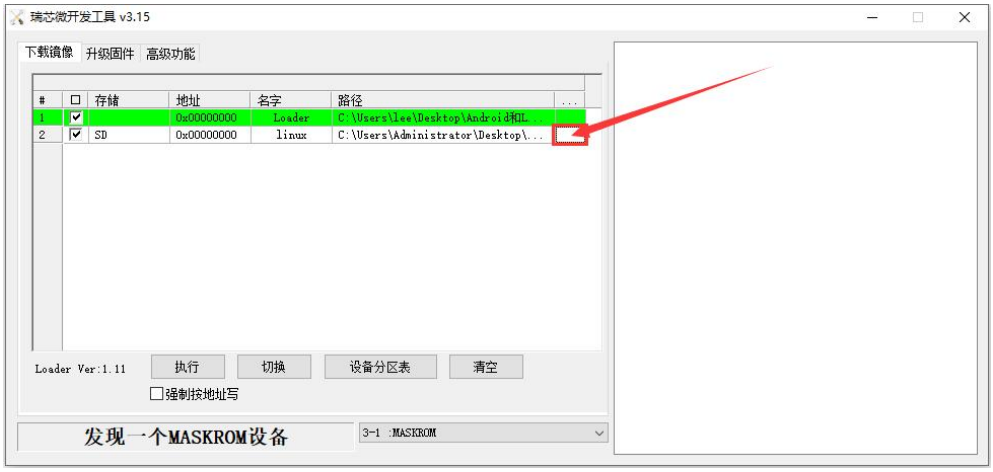
k. Then click the location shown in the picture below.



l. Select **MiniLoaderAll.bin** in the **MiniLoader** folder downloaded earlier, and then click **Open.**



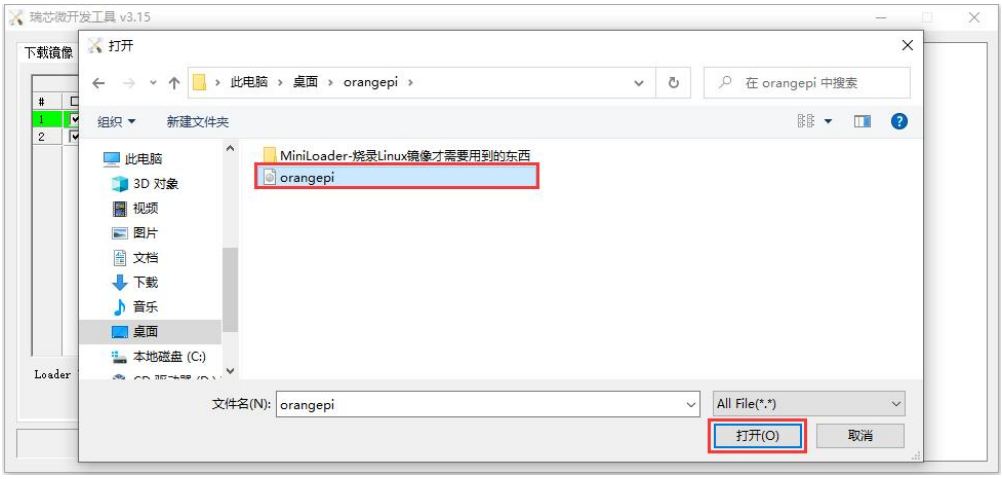
m. Then click the location shown in the picture below



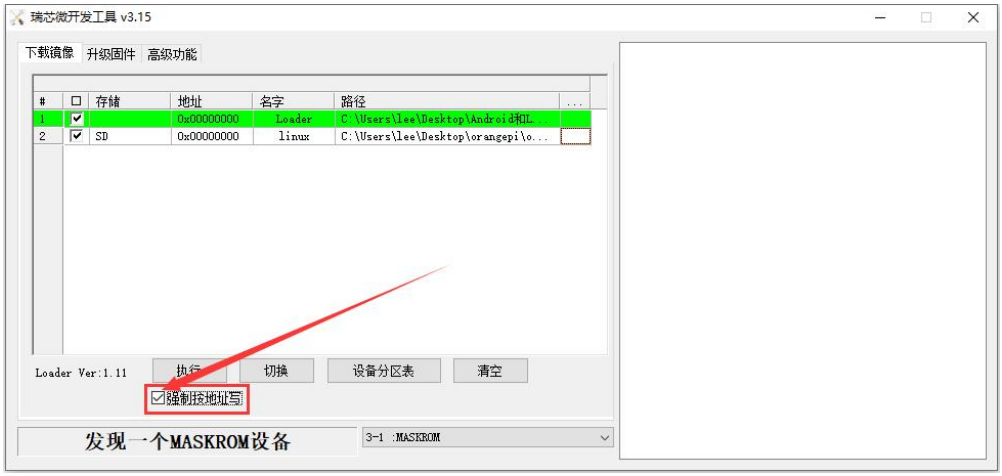
n. Then select the path of the Linux image you want to burn, and click **Open**.

Before burning the image, it is recommended to rename the Linux image to be

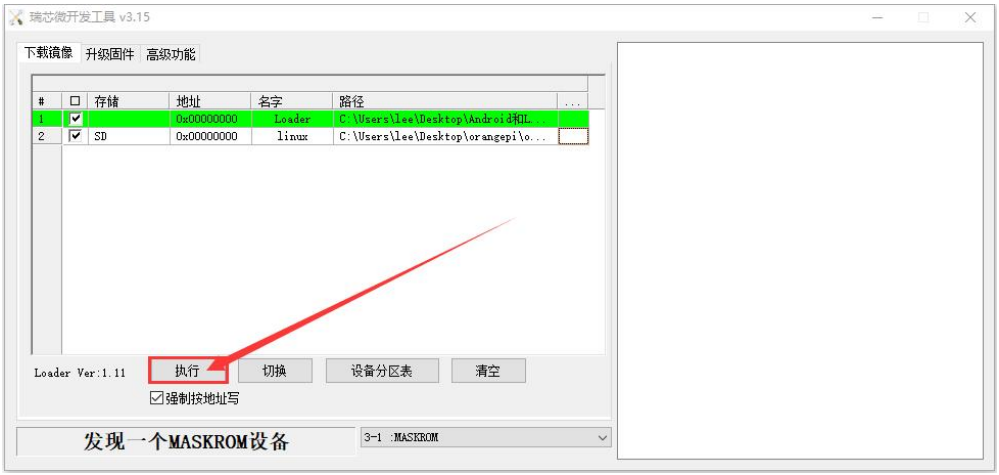
burned to orangepi.img or other shorter names, so that you can see the percentage value of the burning progress when burning the image.



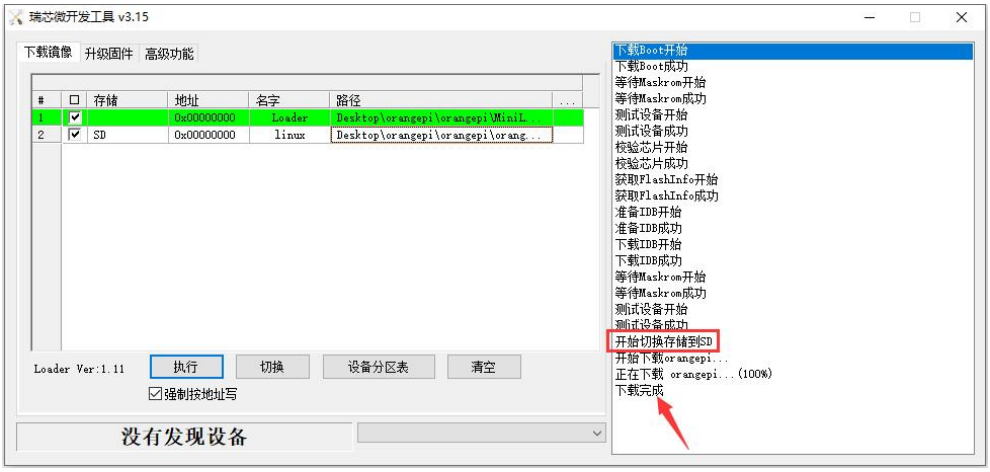
o. Then please check the option of **Force write by address.**



p. Click the execute button again to start burning the Linux image to the TF card of the development board.



q. The display log after the Linux image is burned is as shown below:



r. After burning the Linux image to the TF card, the Linux system will start automatically.

2.4. Method of burning Linux image to TF card based on Ubuntu PC

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

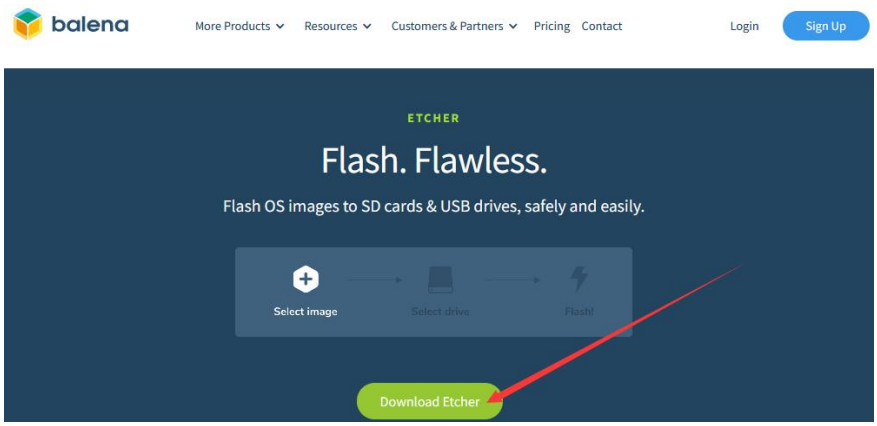
1) First, prepare a TF card with a capacity of 16GB or larger. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk.

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

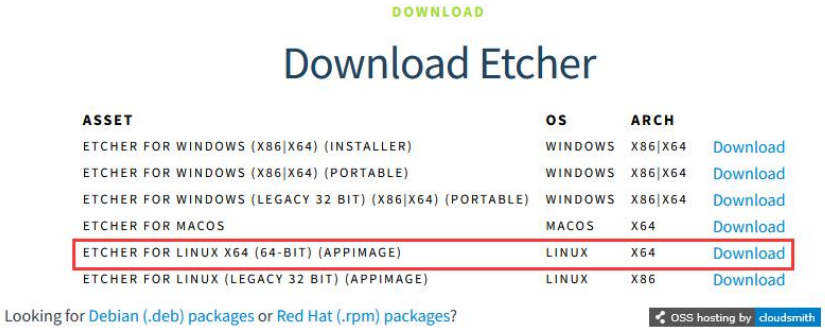
3) Download balenaEtcher software from:

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

4) After entering the balenaEtcher download page, click the green download button to jump to the software download location.

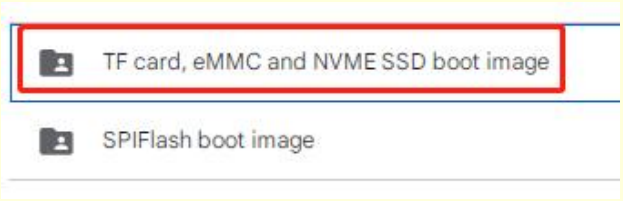


5) Then choose to download the Linux version of the software.



6) Download the compressed file of the Linux operating system image that you want to burn from the **Orange Pi's download page**, and then use decompression software to decompress it. In the decompressed file, the file ending with "**.img**" is the operating system image file, which is usually over 2GB in size.

Note that if you are downloading an OpenWRT image, you will see the following two types of images in the OpenWRT image download link. Please select the image files in the "TF card, eMMC, and NVME SSD boot images" folder.



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

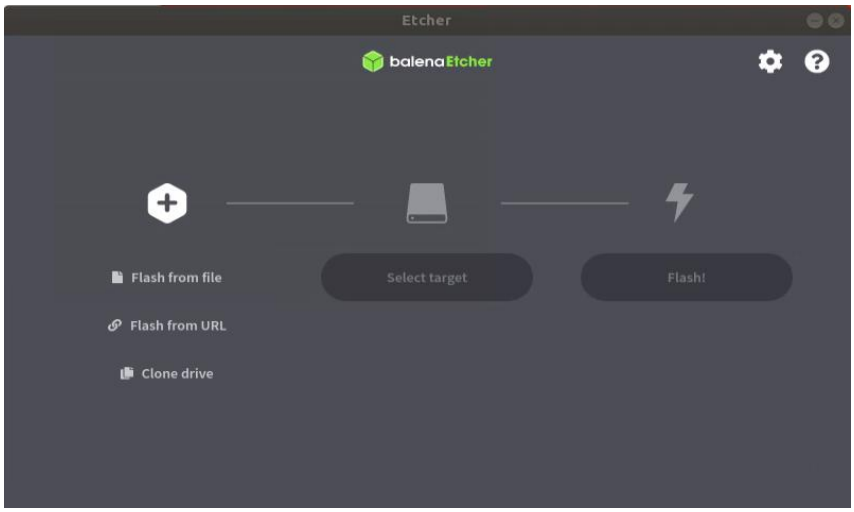
```
test@test:~$ 7z x orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.7z
```

```
test@test:~$ ls orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.*
orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.7z
orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.sha     #校验和文件
orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.img     #镜像文件
```

7) After decompressing the image, you can first use the **sha256sum -c *.sha** command to calculate if the checksum is correct. If the prompt is **successful**, it means that the downloaded image is correct and can be safely burned to the TF card. If the prompt is that the **checksum does not match**, it means that there is a problem with the downloaded image. Please try downloading it again.

```
test@test:~$ sha256sum -c *.sha
orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.img: OK
```

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

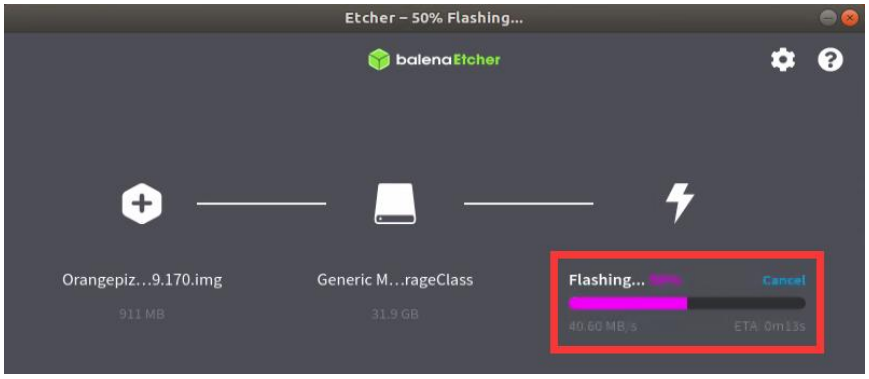


9) The specific steps for burning a Linux image using balenaEtcher are as follows:

a. Firstly, select the path of the Linux image file to be burned.
b. Then select the drive letter of the TF card.
c. Finally, clicking Flash will start burning the Linux image onto the TF card.



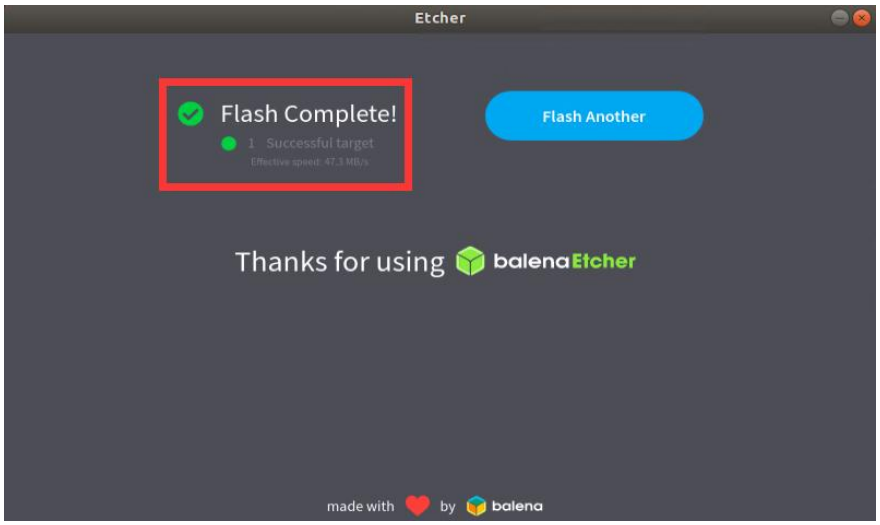
10) The interface displayed during the process of burning a Linux image by balenaEtcher is shown in the following figure. In addition, the progress bar displaying purple indicates that the Linux image is being burned to the TF card.



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



13) After successful burning, the display interface of balenaEtcher is shown in the following figure. If a green indicator icon is displayed, it indicates that the image burning is successful. At this time, you can exit balenaEtcher, then unplug the TF card and insert it into the TF card slot of the development board for use.



2. 5. Method for burning Linux images to eMMC

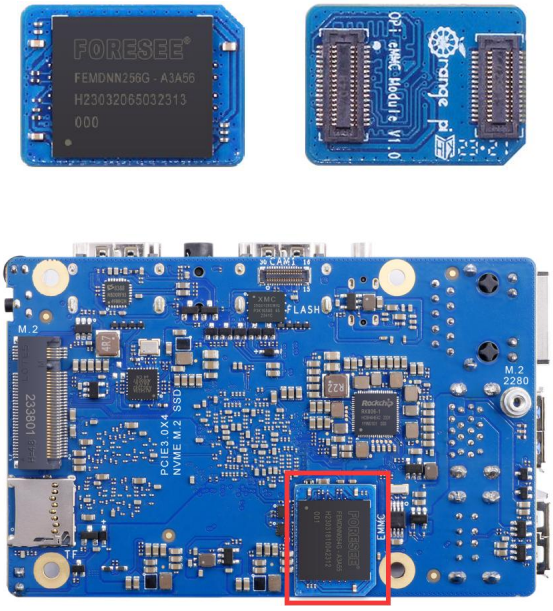
2. 5. 1. Method of burning Linux images to eMMC using RKDevTool

Note that all the following operations were performed on a Windows computer.

Note that the Linux image referred to here specifically refers to Linux distribution images such as Debian, Ubuntu, OpenWRT, or OPi OS Arch downloaded from the Orange Pi data download page.

1) The development board has reserved an expansion interface for the eMMC module.

Before burning the system to eMMC, it is necessary to purchase an eMMC module that matches the eMMC interface of the development board. Then install the eMMC module onto the development board. The method of inserting the eMMC module into the development board is as follows:



2) We also need to prepare a high-quality USB 2.0 male to male data cable.

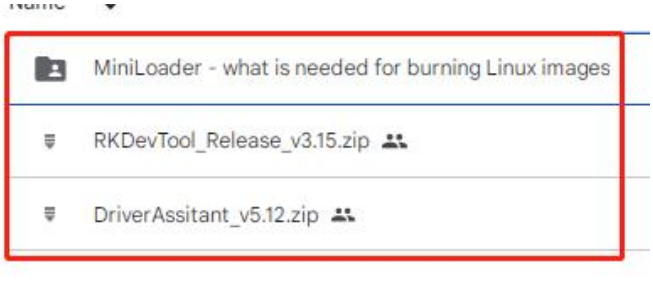


3) Then download the Rockchip micro driver **DriverAssitant_v5.12.zip** and **MiniLoader**, as well as the burning tool **RKDevTool_Release_v3.15.zip**, from the **Orange Pi's download page**.

a. On the Orange Pi download page, first select the official tool and then go to the folder below.



b. Then download all the files below.



Note that the "MiniLoader - What is needed to burn Linux images" folder is referred to as the MiniLoader folder in the following text.

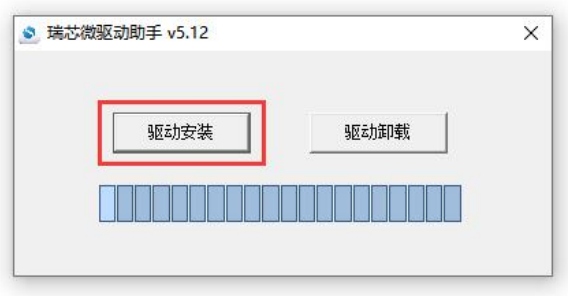
4) Then download the compressed file of the Linux operating system image that you want to burn from the **Orange Pi's download page**, and use decompression software to decompress it. In the decompressed file, the file ending with "**.img**" is the operating system image file, which is usually over 2GB in size.

5) Then use decompression software to extract the **DriverAssitant_v5.12.zip** file, and find the **DriverInstall.exe** executable file in the extracted folder and open it.

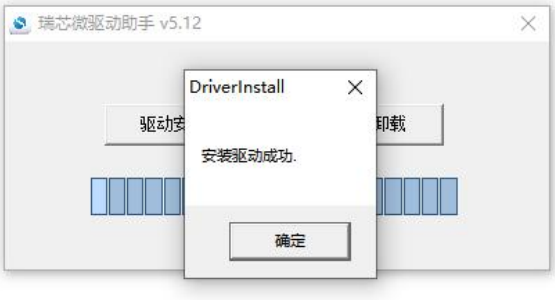


6) The steps to install the Rockchip driver after opening **DriverInstall.exe** are as follows:

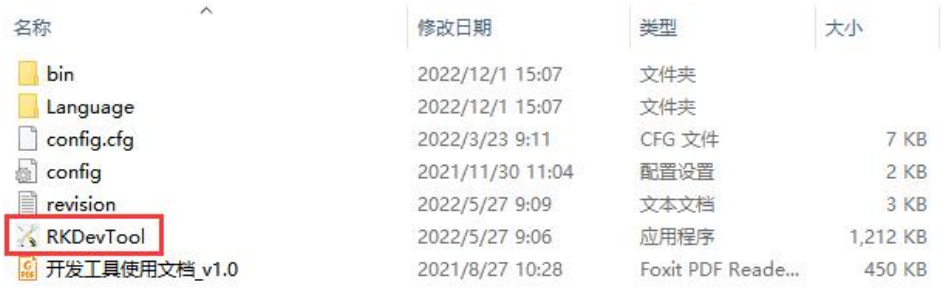
a. Click the '**Driver Installation**' button.



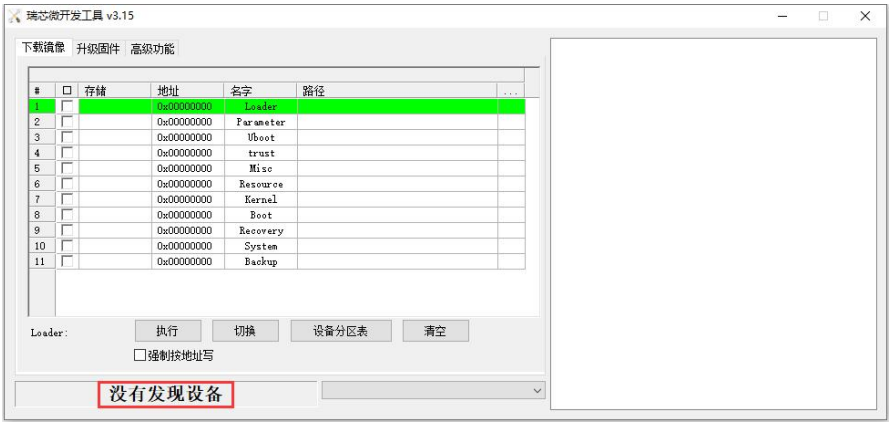
b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.



7) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.

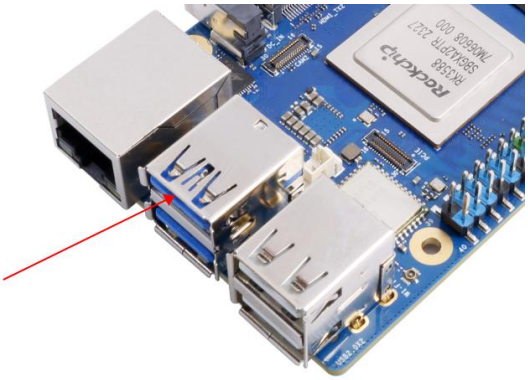


8) After opening the **RKDevTool** burning tool, the bottom left corner will prompt "**Device not found**" because the computer has not yet been connected to the development board through a USB 2.0 male to male data cable.



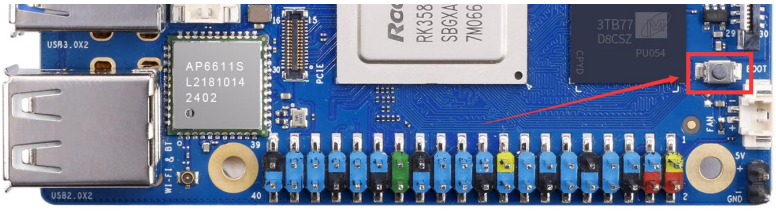
9) Then start burning the Linux image into eMMC.

a. Firstly, connect the development board to the Windows computer via a USB2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:

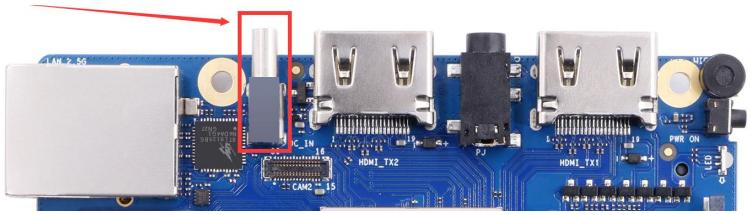


b. Ensure that the development board is not inserted with a TF card or connected to a power source.

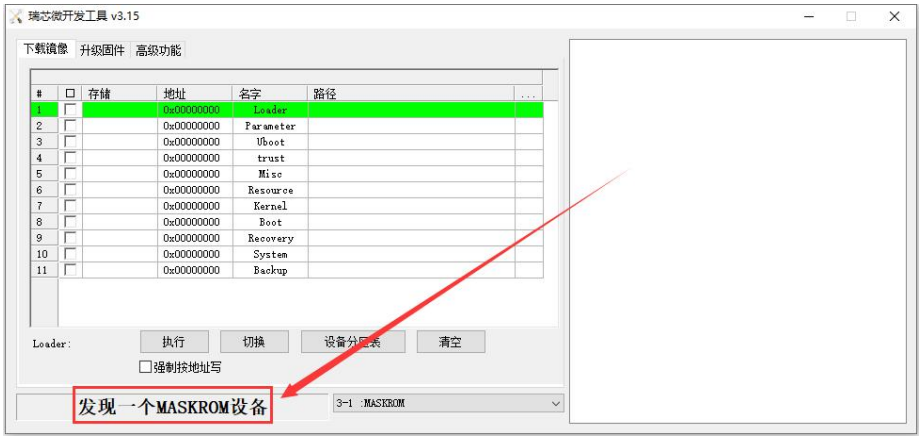
c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is shown in the following figure:



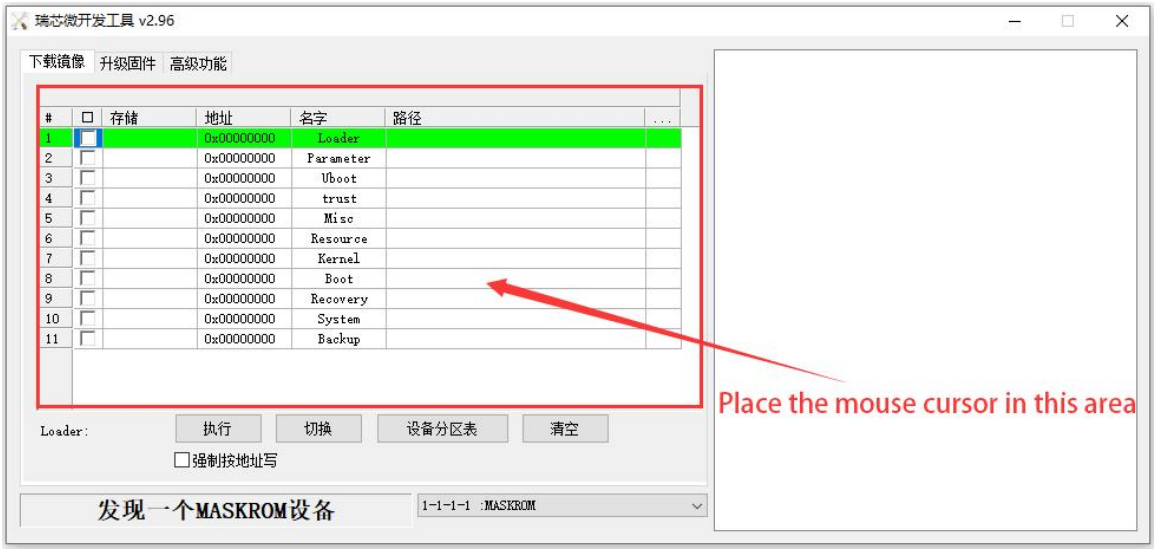
d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



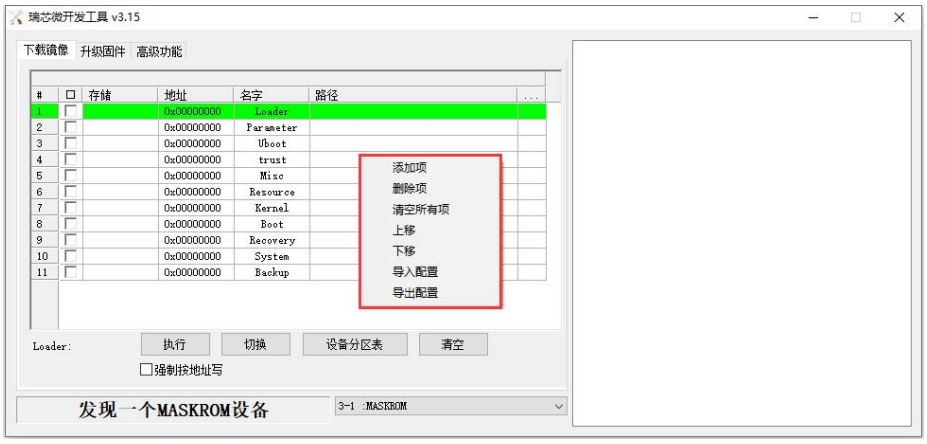
e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**".



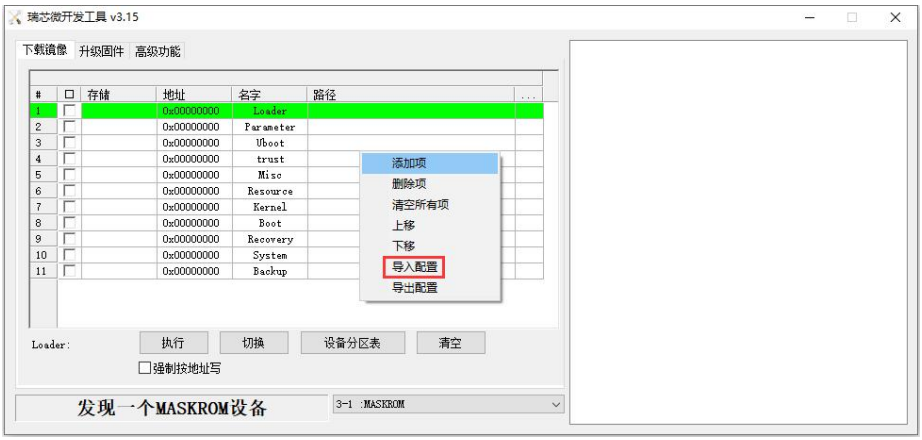
f. Then place the mouse cursor in the area below.



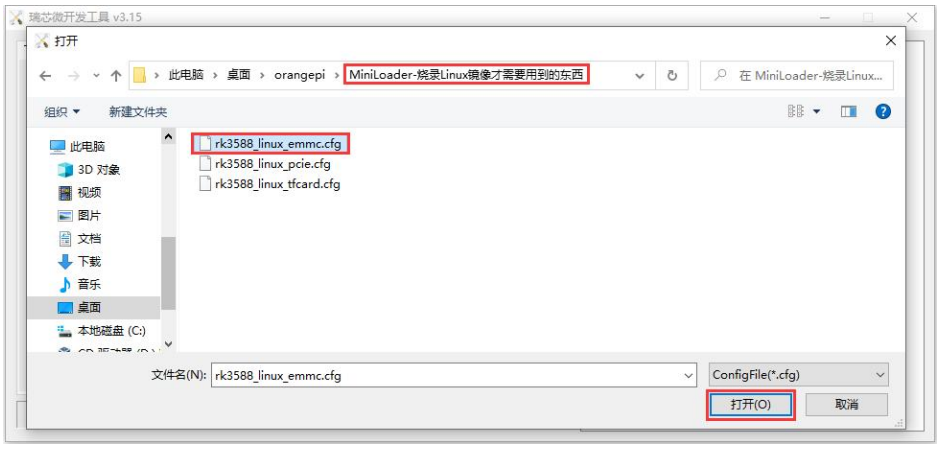
g. Then right-click and the selection interface shown in the following figure will pop up.



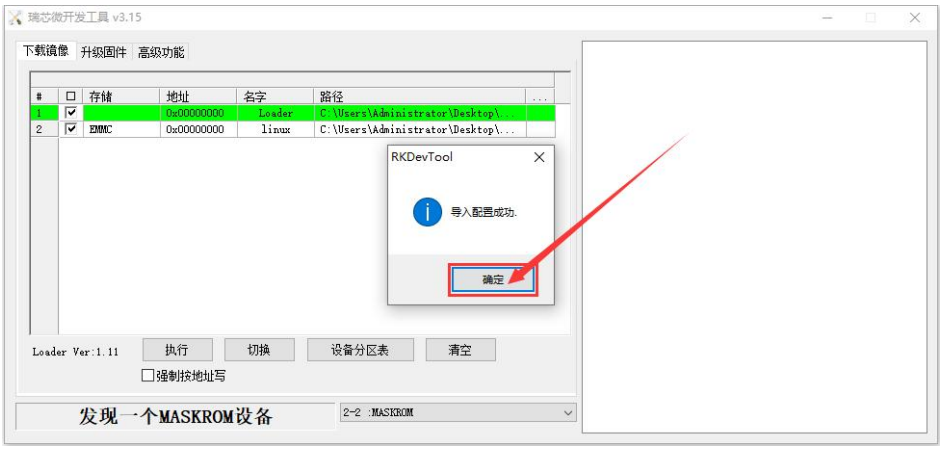
h. Then select the **import configuration** option.



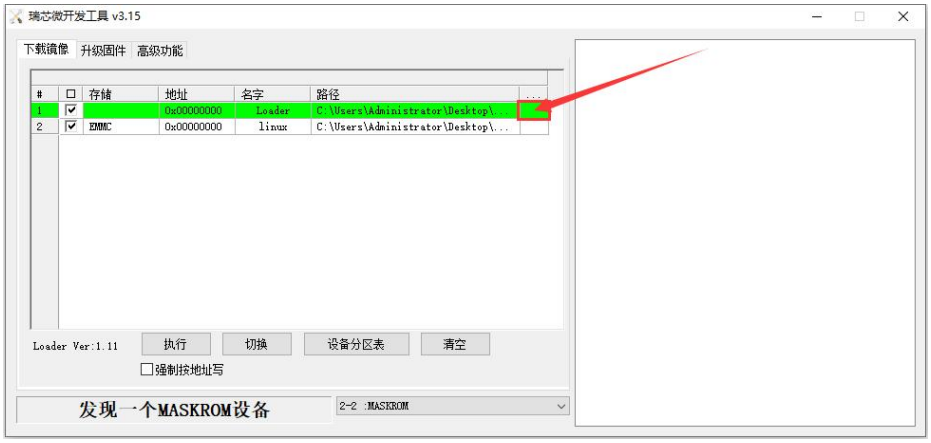
i. Then select the **rk3588_linux_emmc.cfg** configuration file from the **MiniLoader** folder downloaded earlier, and click to **open it**.



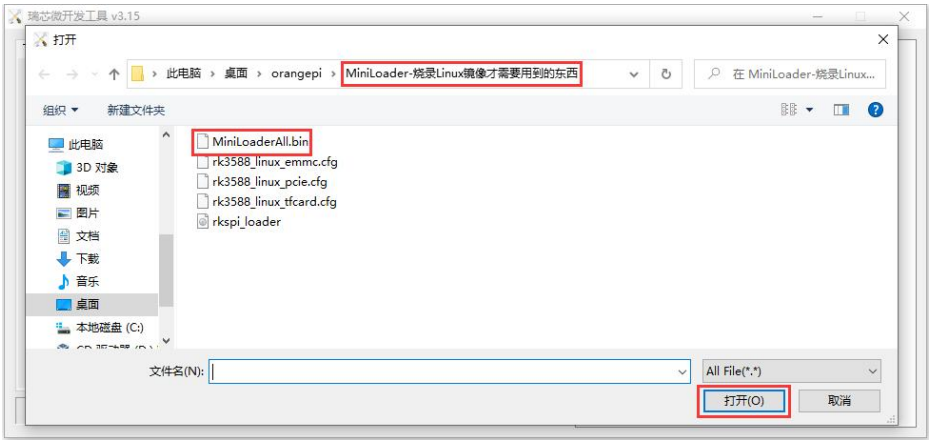
j. Then click **OK**.



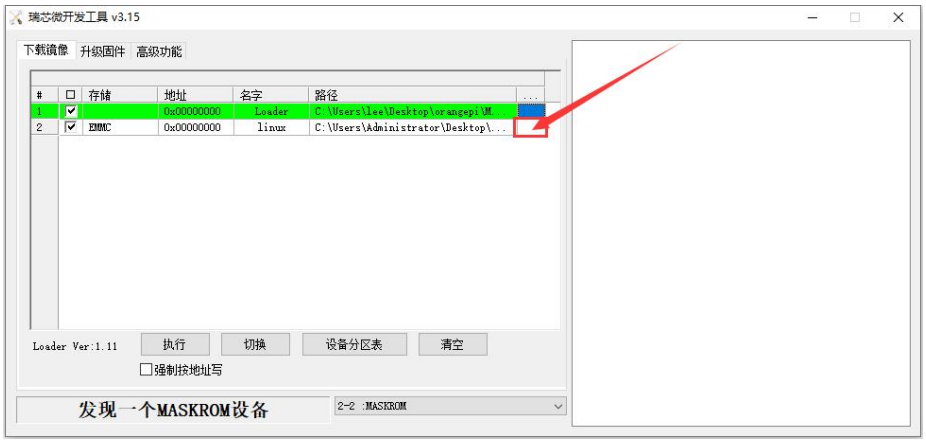
k. Then click on the position shown in the following image.



l. Select **MiniLoaderAll.bin** from the **MiniLoader** folder downloaded earlier, and then click **Open**.

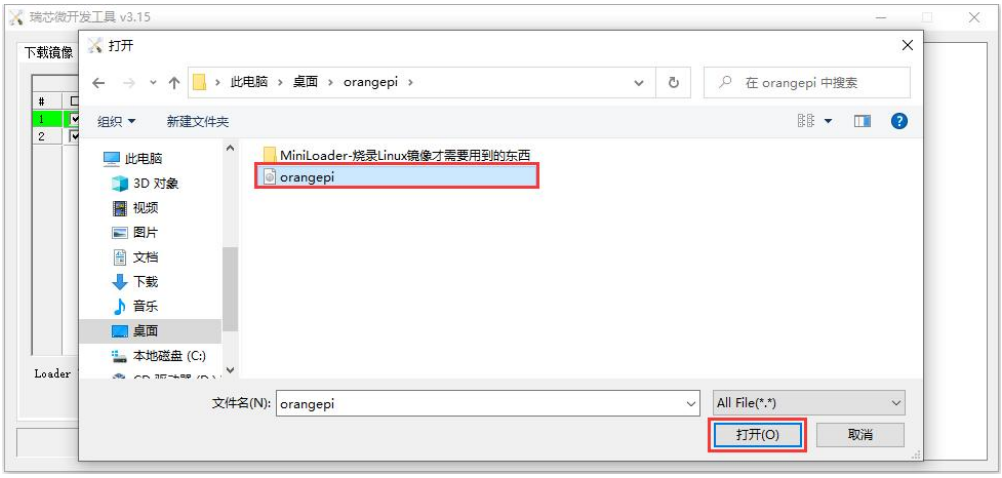


m. Then click on the position shown in the following image.

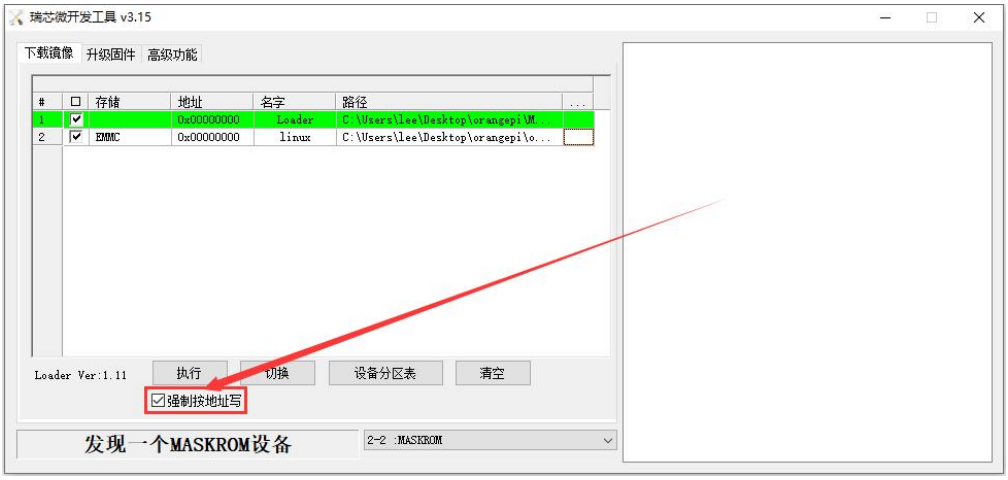


n. Then select the path of the Linux image you want to burn, and click **Open**.

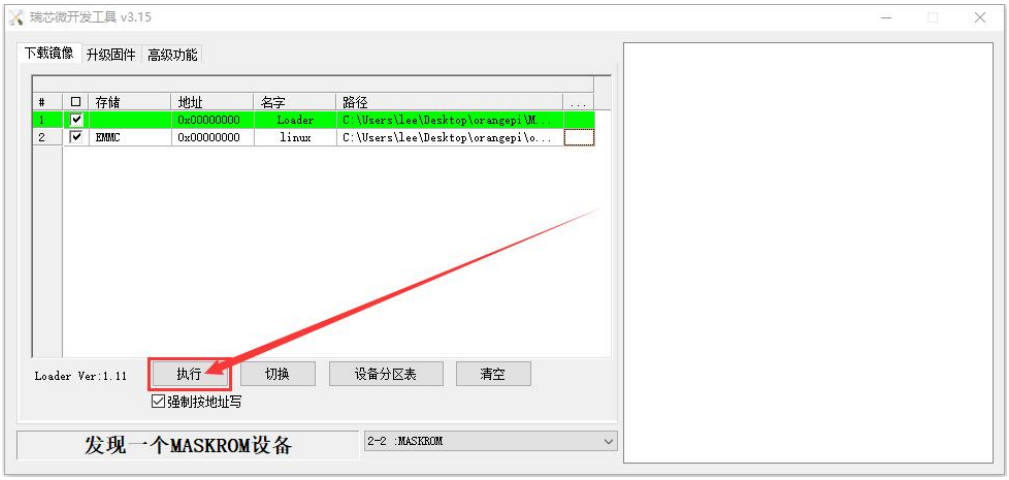
Before burning the image, it is recommended to rename the Linux image to orangepi.img or another shorter name, so that the percentage of burning progress can be seen when burning the image.



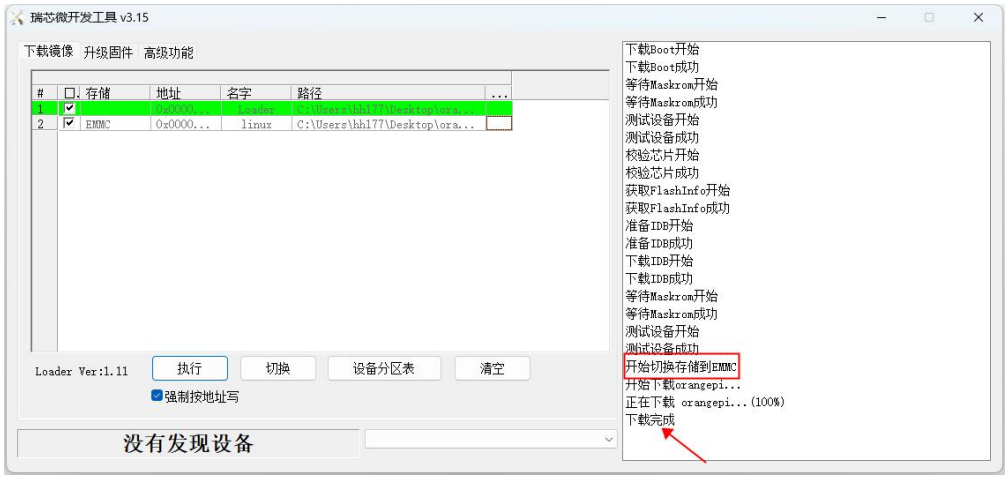
o. Then please check the option to **force writing by address**.



p. Clicking the execute button again will start burning the Linux image to the eMMC on the development board.



q. The display log after burning the linux image is shown in the following figure:



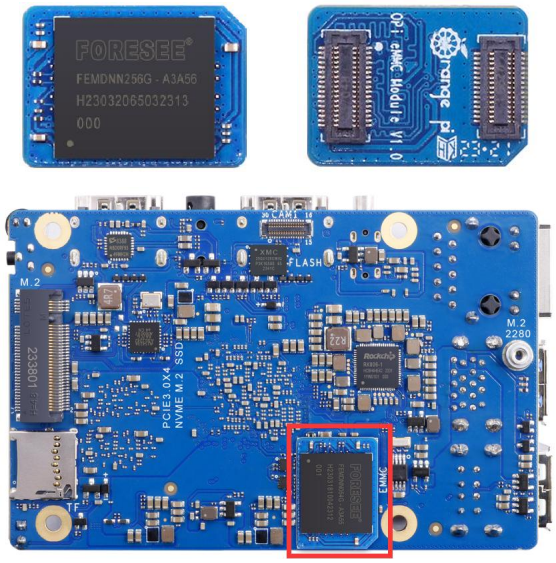
r. After burning the Linux image to eMMC, the linux system will automatically start.

2. 5. 2. Method of burning Linux images to eMMC using the dd command

Note that the Linux image referred to here specifically refers to Linux distribution images such as Debian, Ubuntu, OpenWRT, or OPi OS Arch downloaded from the Orange Pi data download page.

1) The development board has reserved an expansion interface for the eMMC module. Before burning the system to eMMC, it is necessary to purchase an eMMC module that matches the eMMC interface of the development board. Then install the eMMC module

onto the development board. The method of inserting the eMMC module into the development board is as follows:



2) Burning a Linux image to eMMC using the dd command requires the use of a TF card, so the first step is to burn the Linux image onto the TF card, and then use the TF card to start the development board and enter the Linux system. The method of burning Linux images to TF cards can be found in the two sections of **Burning Linux Images to TF Cards on Windows PC** and **Burning Linux Images to TF Cards on Ubuntu PC**.

3) After starting the Linux system with a TF card, we first upload the decompressed Linux image file (downloaded from the official website as a Debian, Ubuntu, or OPi Arch image) to the TF card. Please refer to the instructions in the section on **uploading Linux image files to the development board Linux system** for the method of uploading files to the development board.

4) After uploading the image to the Linux system of the development board, we can enter the storage path of the image file in the command line of the Linux system of the development board. For example, I stored the Linux image of the development board in the **/home/orangepi/Desktop** directory, and then enter the **/home/orangepi/Desktop** directory to see the uploaded image file.

```
orangepi@orangepi:~$ cd /home/orangepi/Desktop
orangepi@orangepi:~/Desktop$ ls
orangepi5ultra_x.x.x_debian_bullseye_desktop_xfce_linux5.10.160.img
```

How to enter the command line of the Linux system on the development board?

1. Please refer to the instructions in the Debugging Serial Port Usage section for the method of logging into the terminal using a serial port.

2. Please refer to the instructions in the SSH Remote Login Development Board section for remote login to Linux systems using SSH.

3. If HDMI, LCD or other display screens are connected, a command line terminal can be opened on the desktop.

5) Next, let's use the following command to confirm the device node of eMMC.

```
orangepi@orangepi:~/Desktop$ ls /dev/mmcblk*boot0 | cut -c1-12
/dev/mmcblk1
```

6) Then we can use the dd command to clear eMMC. Please note that after the **of=** parameter, please fill in the output of the command above.

```
orangepi@orangepi:~/Desktop$ sudo dd bs=1M if=/dev/zero of=/dev/mmcblk1 count=1000 status=progress
orangepi@orangepi:~/Desktop$ sudo sync
```

7) Then you can use the dd command to burn the Linux image of the development board to eMMC.

a. The **if=** parameter in the following command should be followed by the full path where the Linux image is stored and the name of the Linux image (such as **/home/orangepi/Desktop/Linux image name**). Because we have already entered the path of the Linux image, we only need to fill in the name of the Linux image.

b. Please do not copy the Linux image name in the following command, replace it with the actual image name (as the version number of the image may be updated).

```
sudo dd bs=1M if=orangepi5ultra_x.x.x_debian_bullseye_desktop_xfce_linux5.10.160.img of=/dev/mmcblk1 status=progress

sudo sync
```

Note that if you are uploading a compressed Linux image file ending in. 7z or. xz, please remember to decompress it before burning it with the dd command.

The detailed explanation and more usage of all parameters of the dd command

can be viewed by executing the man dd command in a Linux system.

8) After successfully burning the linux image of the development board to eMMC, you can use the **poweroff** command to shut down. Then please unplug the TF card and press the power button briefly to start the linux system in eMMC.

2. 6. Method for burning Linux images to SPIFlash+NVMe SSD

Note that the Linux image referred to here specifically refers to Linux distribution images such as Debian, Ubuntu, OpenWRT, or OPi OS Arch downloaded from the Orange Pi data download page.

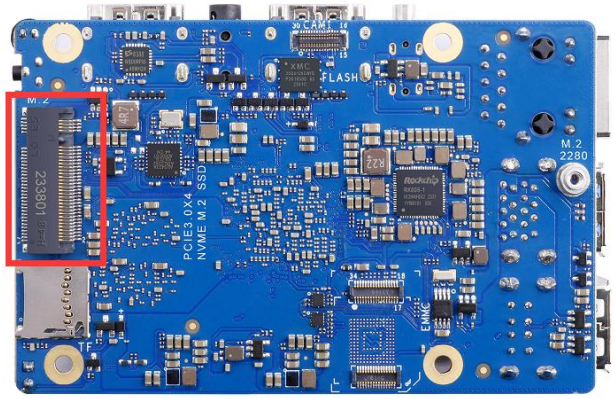
Note that all the following operations were performed on a Windows computer.

2. 6. 1. Method of burning using RKDevTool

1) Firstly, it is necessary to prepare an NVMe SSD solid state drive with a PCIe interface specification of PCIe 3.0x4 for the M.2 slot of the development board.

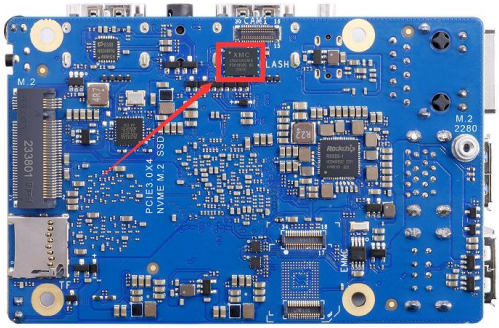


2) Then insert the NVMe SSD into the M.2 PCIe interface of the development board and secure it in place.



3) The position of SPI Flash on the development board is shown in the following figure,

and no other settings are required before starting to burn.



4) Then you need to prepare a high-quality USB 2.0 male to male data cable.

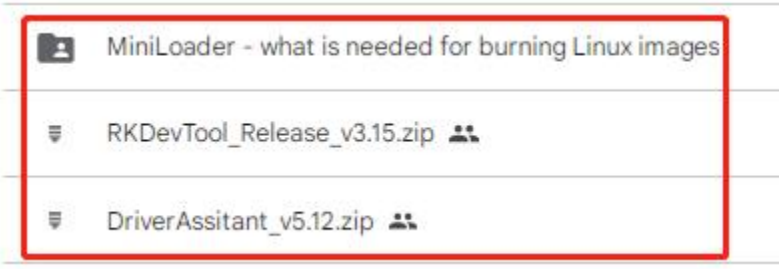


5) Then download the Rockchip micro driver **DriverAssitant_v5.12.zip** and **MiniLoader**, as well as the burning tool **RKDevTool_Release_v3.15.zip**, from the **Orange Pi's download page**.

a. On the Orange Pi download page, first select the **official tool** and then go to the folder below.



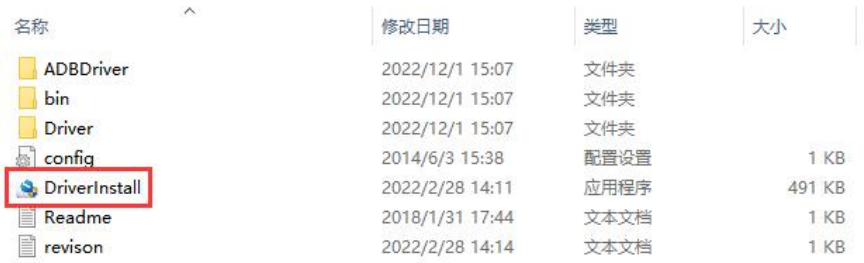
b. Then download all the files below.



Note that the "MiniLoader - What is needed to burn Linux images" folder is referred to as the MiniLoader folder in the following text.

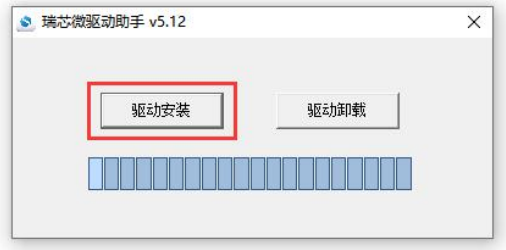
6) Then download the compressed file of the Linux operating system image that you want to burn from the **Orange Pi's download page**, and use decompression software to decompress it. In the decompressed file, the file ending with "**.img**" is the operating system image file, which is usually over 2GB in size.

7) Then use decompression software to extract the **DriverAssitant_v5.12.zip** file, and find the **DriverInstall.exe** executable file in the extracted folder and open it.

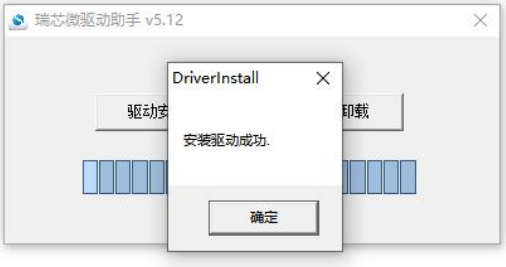


8) The steps to install the Rockchip driver after opening **DriverInstall.exe** are as follows:

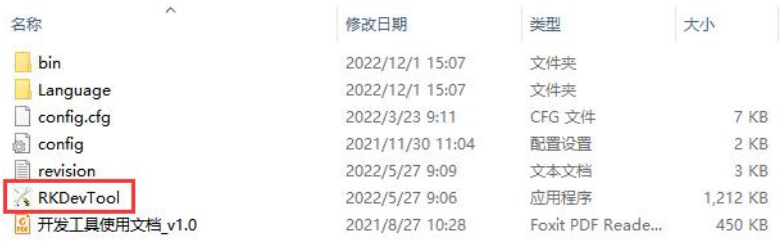
a. Click the "**Driver Installation**" button.



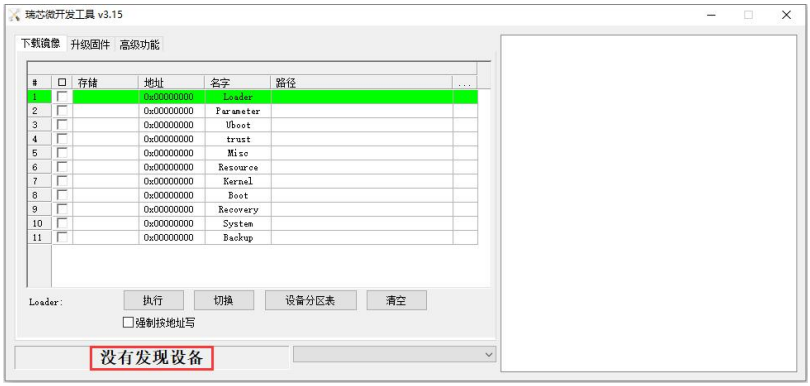
b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.



9) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.

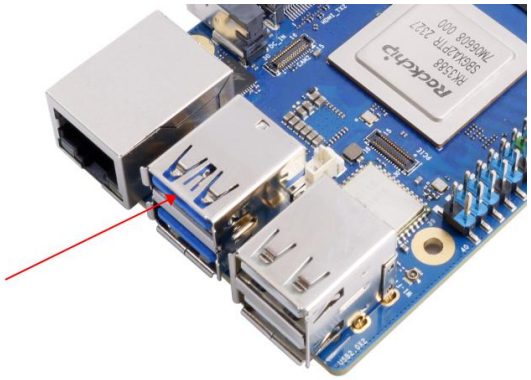


10) After opening the **RKDevTool** burning tool, the bottom left corner will prompt "**Device not found**" because the computer has not yet been connected to the development board through a USB2.0 male to male data cable.



11) Then start burning the Linux image onto the SSD.

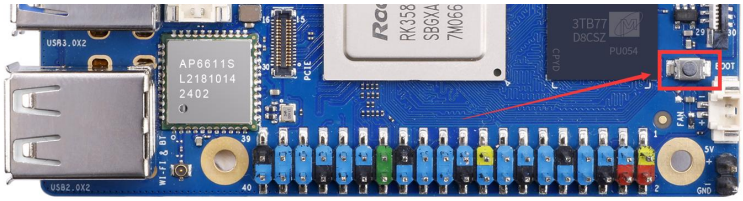
a. Firstly, connect the development board to the Windows computer via a USB2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:



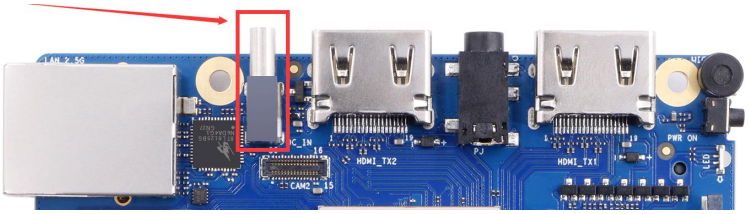
b. Ensure that the development board is not connected to a power source or inserted with a TF card.

c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is

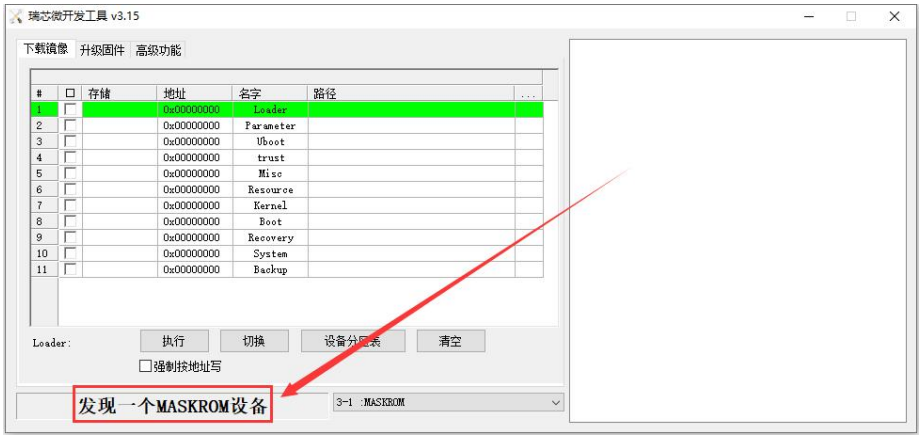
shown in the following figure:



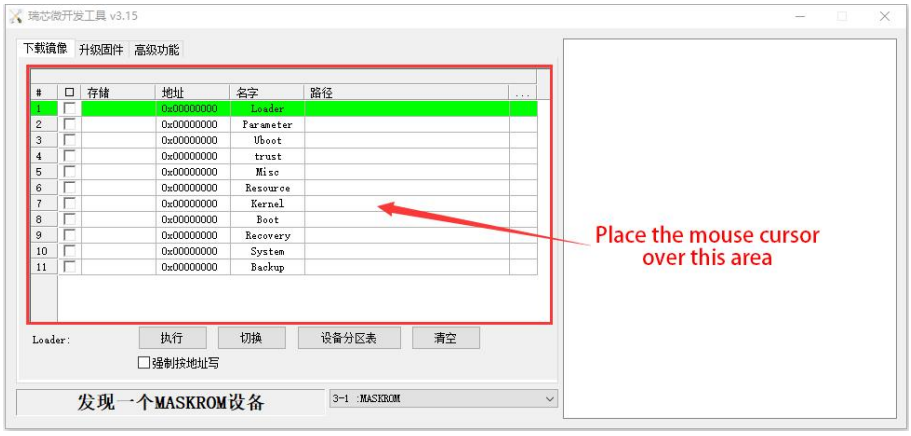
d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**".

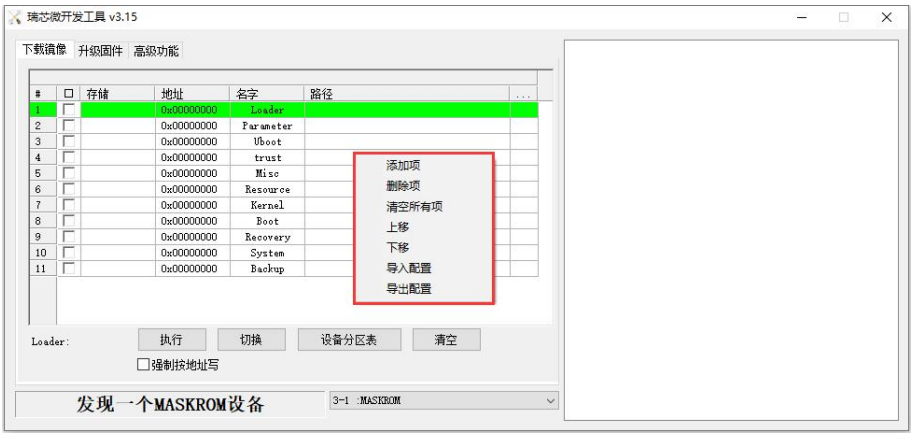


f. Then place the mouse cursor in the area below.

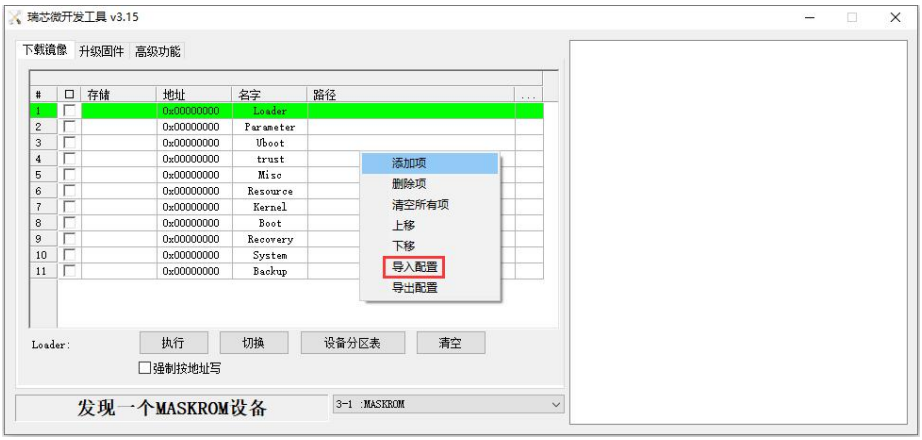


g. Then right-click and the selection interface shown in the following figure will

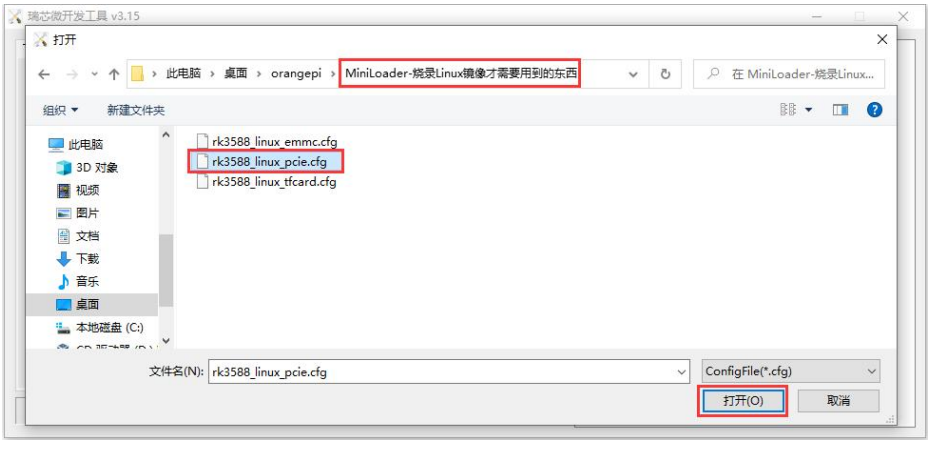
pop up.



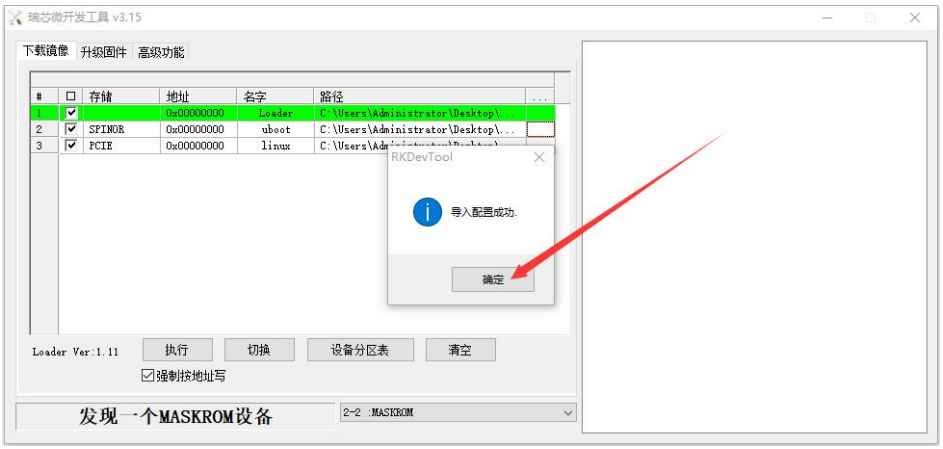
h. Then select the **import configuration** option.



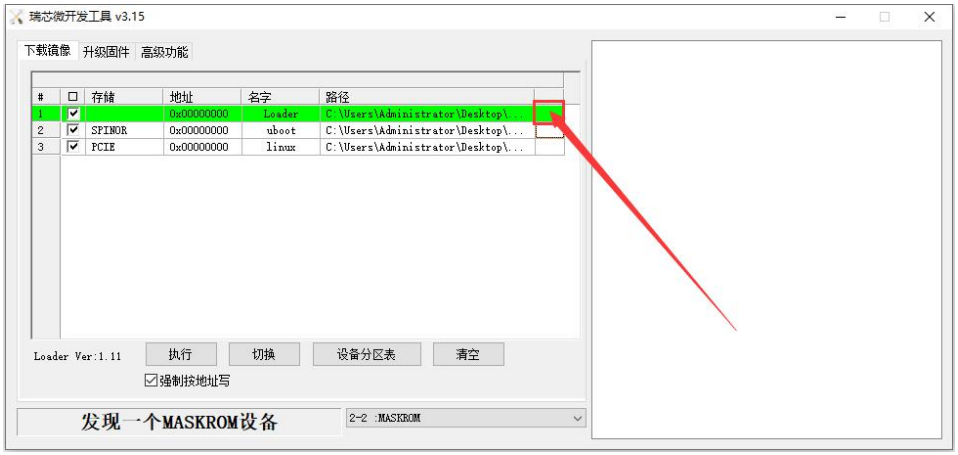
i. Then enter the **MiniLoader** folder downloaded earlier, select the **rk3588_linux_pcie.cfg** configuration file, and click **Open**.



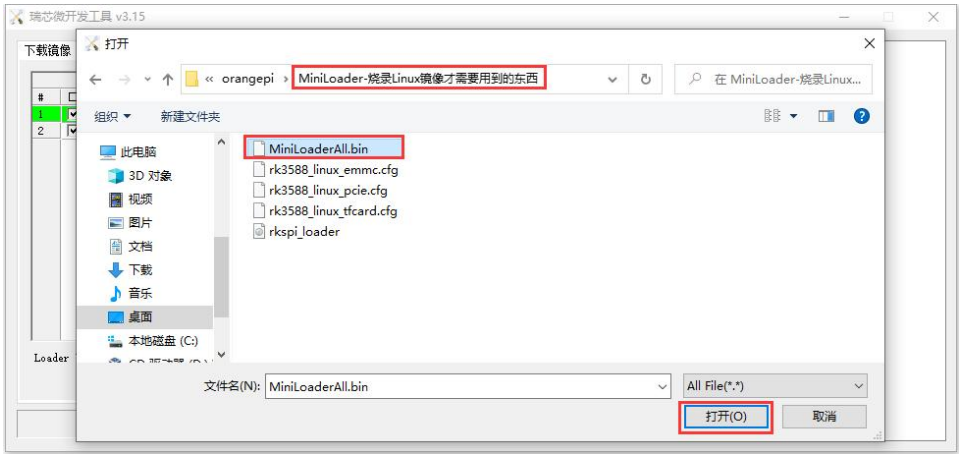
j. Then click **OK**.



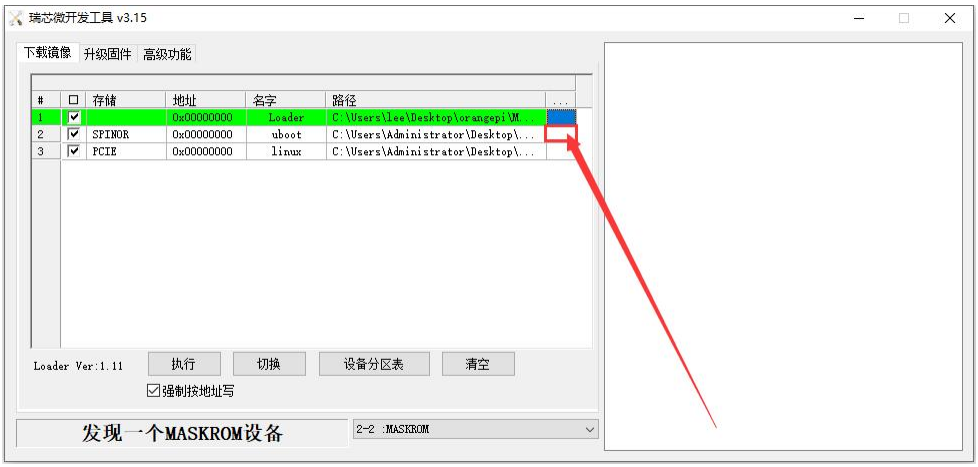
k. Then click on the position shown in the following image.



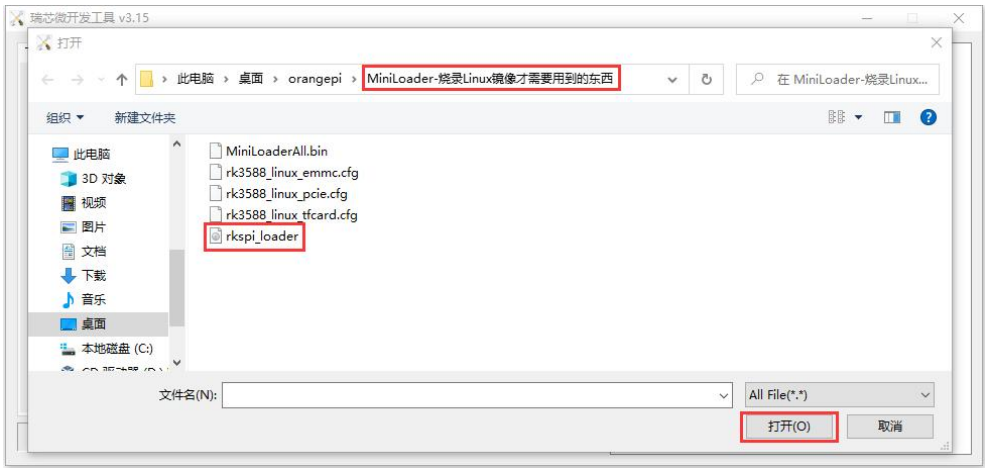
l. Select **MiniLoaderAll.bin** from the **MiniLoader** folder downloaded earlier, and then click **Open**.



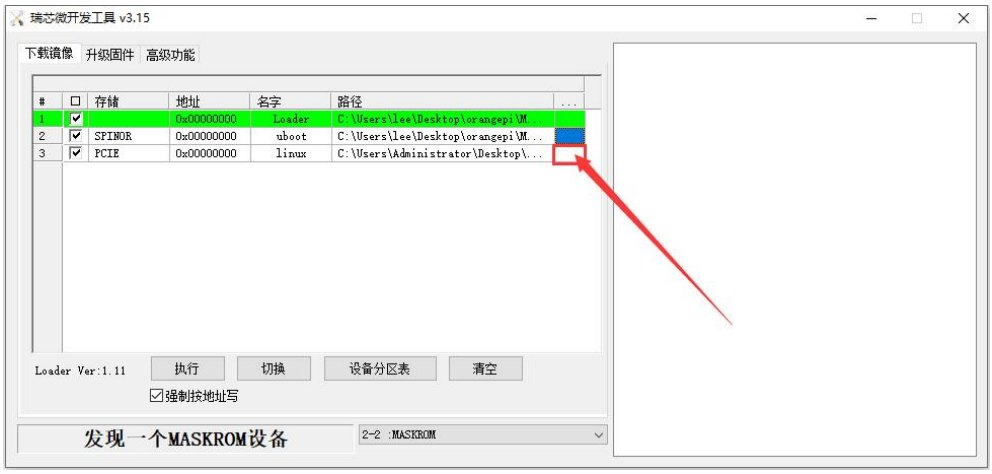
m. Then click on the position shown in the following image.



n. Then go to the **MiniLoader** folder that was downloaded earlier, select **rkspi_loader.img**, and click **Open**.



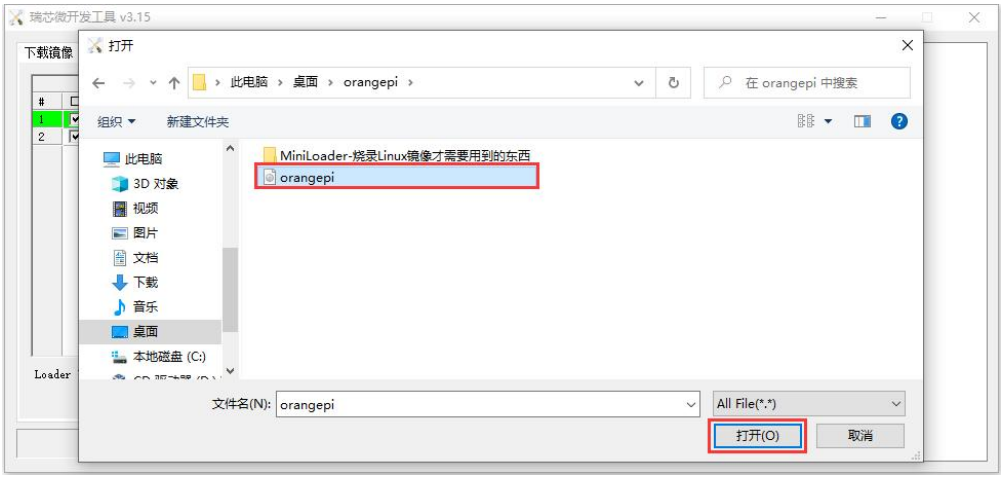
o. Then click on the position shown in the following image.



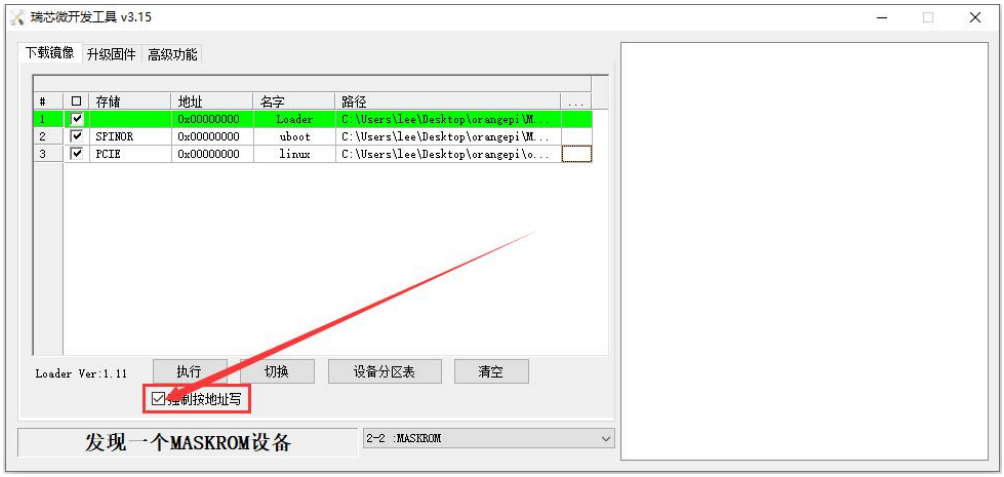
p. Then select the path of the linux image you want to burn, and click **Open**.

Before burning the image, it is recommended to rename the linux image to

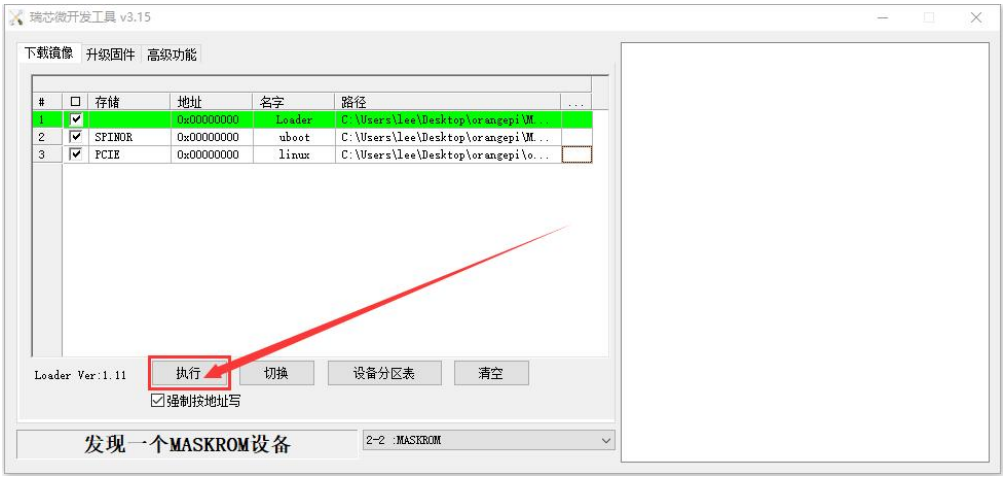
orangepi.img or another shorter name, so that the percentage of burning progress can be seen when burning the image.



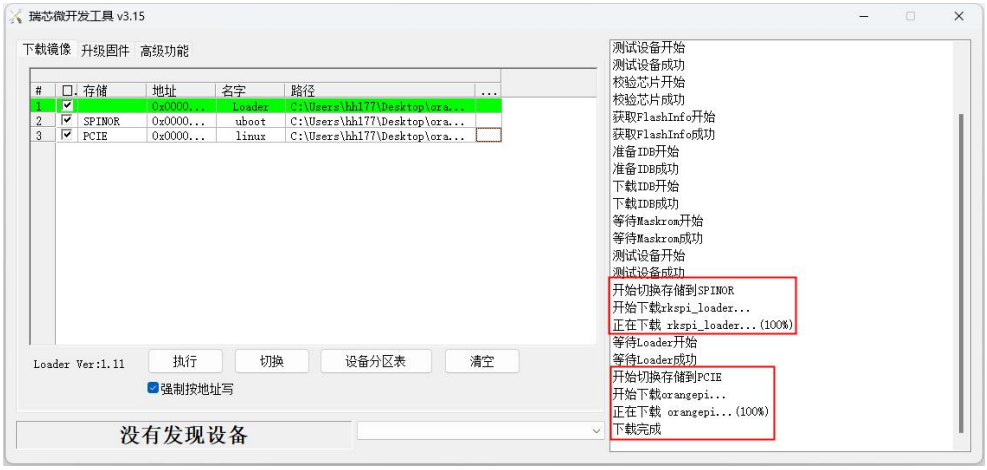
q. Then please check the option to **force writing by address**.



r. Clicking the execute button again will start burning the linux image to the SSD.



s. The display log after burning the linux image is shown in the following figure:



If there is a problem with burning, please clear SPIFlash first and then try burning again. Please refer to the instructions in the section on how to clear SPIFlash using RKDevTool.

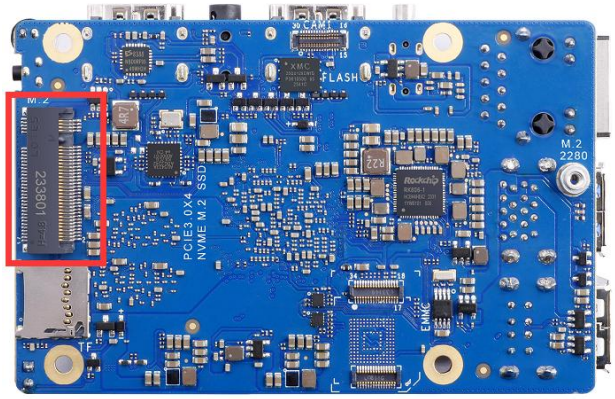
t. After burning the image, the Linux system in SPIFlash+PCIe SSD will automatically start. If it does not start properly, please power it on again and try again.

2. 6. 2. Method of burning with dd command

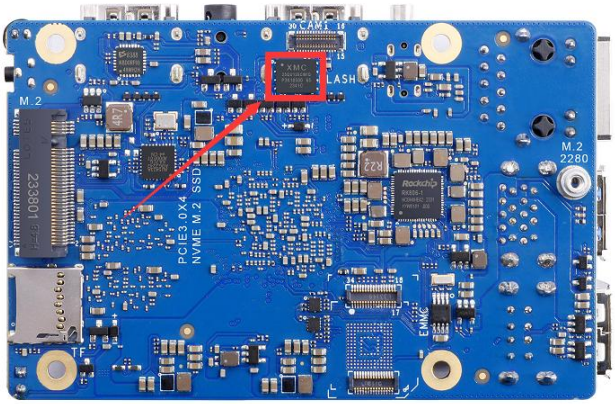
1) Firstly, it is necessary to prepare an NVMe SSD solid state drive with a PCIe interface specification of PCIe 3.0x4 for the M.2 slot of the development board.



2) Then insert the NVMe SSD into the M.2 PCIe interface of the development board and secure it in place.



3) The position of SPI Flash on the development board is shown in the following figure, and no other settings are required before starting to burn.



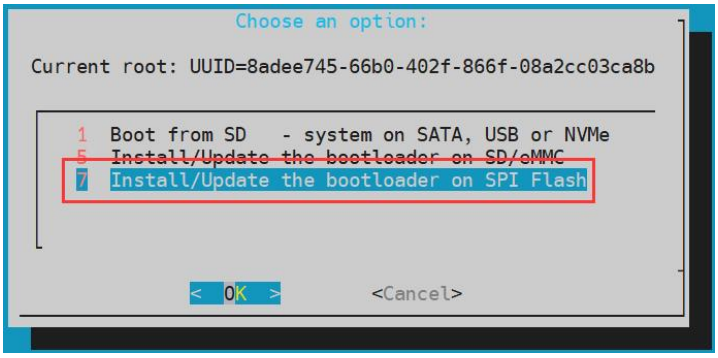
4) Burning a linux image to a SPIFlash+NVMe SSD requires the use of a TF card, so the first step is to burn the linux image onto the TF card, and then use the TF card to boot the development board into the linux system. The method of burning a Linux image to a TF card can be found in the two sections: **the method of burning a Linux image to a TF card based on Windows PC** and **the method of burning a Linux image to a TF card based on Ubuntu PC**.

5) After starting the Linux system with a TF card, we first burn the u-boot image into SPI Flash.

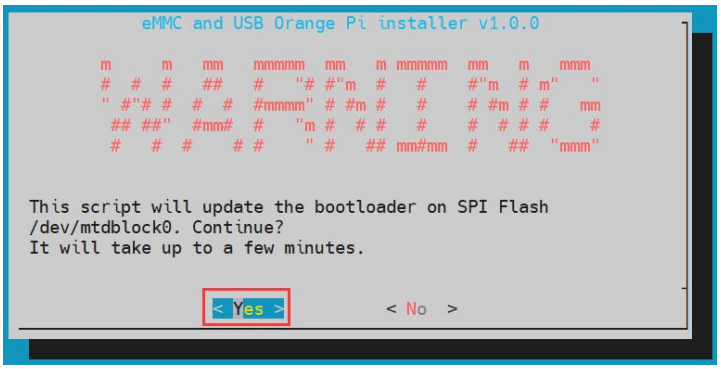
a. First, run **nand-sata-install**. **Ordinary users should remember to grant sudo privileges**.

```
orangepi@orangepi:~$ sudo nand-sata-install
```

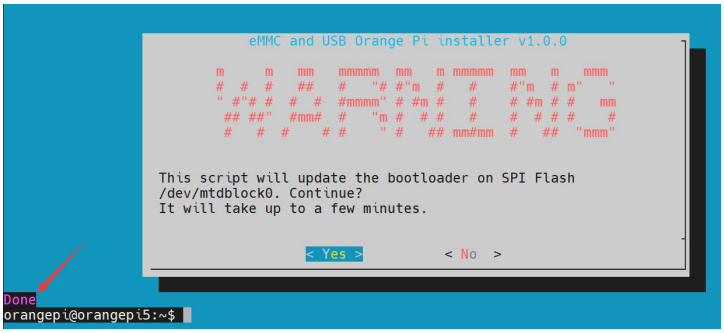
b. Then choose **7 Install/Update ther bootloader on SPI Flash**。



c. Then choose **<Yes>**。



d. Then please be patient and wait for the burning to complete. After the burning is completed, the following will be displayed (a **Done** will appear in the bottom left corner):

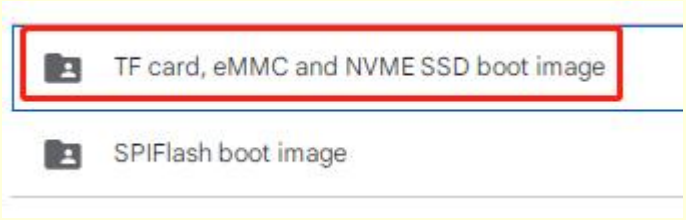


There is no nand sata install script in the OPi OS Arch system. Please use the following command to mirror u-boot to SPI Flash:

[orangepi@orangepi ~]$ **sudo dd if=/boot/rkspi_loader.img of=/dev/mtdblock0**

6) Then upload the Linux image file (downloaded from the official website as a Debian, Ubuntu, or OpenWRT image) to the TF card. Please refer to the instructions in the section on **uploading Linux image files to the development board Linux system** for the method of uploading files to the development board.

Note that if you are downloading an OpenWRT image, you will see the following three types of images in the OpenWRT image download link. Please select the image files in the "TF card, eMMC, and NVME SSD boot images" folder.



7) After uploading the image to the linux system of the development board, we can enter the storage path of the image file in the command line of the linux system of the development board. For example, I stored the linux image of the development board in the **/home/orangepi/Desktop** directory, and then enter the **/home/orangepi/Desktop** directory to see the uploaded image file.

```
orangepi@orangepi:~$ cd /home/orangepi/Desktop
orangepi@orangepi:~/Desktop$ ls
orangepi5ultra_x.x.x_debian_bullseye_desktop_xfce_linux5.10.160.img
```

How to enter the command line of the Linux system on the development board?

1. For the method of using the serial port to log in to the terminal, please refer to the instructions in the section on how to use the debugging serial port.

2. Use ssh to remotely log in to the Linux system, please refer to the instructions in the section of SSH remote login to the development board.

3. If HDMI, LCD or other display screens are connected, a command line terminal can be opened on the desktop.

8) Next, let's confirm that the NVMe SSD has been properly recognized by the Linux development board. If NVMe SSD recognizes it normally, you can use the **sudo fdisk -l** command to see **nvme** related information.

```
orangepi@orangepi:~/Desktop$ sudo fdisk -l | grep "nvme0n1"
Disk /dev/nvme0n1: 1.86 TiB, 2048408248320 bytes, 4000797360 sectors
```

Using the **lspci** command, you can see an NVMe related PCI device.

```
orangepi@orangepi:~/Desktop$ lspci
0004:40:00.0 PCI bridge: Fuzhou Rockchip Electronics Co., Ltd Device 3588 (rev 01)
0004:41:00.0 Non-Volatile memory controller: MAXIO Technology (Hangzhou) Ltd. NVMe SSD Controller MAP1202 (rev 01)
```

9) Then we can use the dd command to clear the NVMe SSD (optional).

```
orangepi@orangepi5ultra:~/Desktop$ sudo dd bs=1M if=/dev/zero of=/dev/nvme0n1 count=2000 status=progress
orangepi@orangepi5ultra:~/Desktop$ sudo sync
```

10) Then you can use the dd command to burn the linux image of the development board

to the NVMe SSD.

a. The **if=** parameter in the following command should be followed by the full path where the Linux image is stored and the name of the Linux image (such as **/home/orangepi/Desktop/Linux image name**). Because we have already entered the path of the Linux image, we only need to fill in the name of the Linux image.

b. Please do not copy the linux image name in the following command, replace it with the actual image name (as the version number of the image may be updated).

```
sudo dd bs=1M if=orangepi5ultra_x.x.x_debian_bullseye_desktop_xfce_linux5.10.160.img of=/dev/nvme0n1 status=progress


sudo sync
```

Note that if you are uploading a compressed Linux image file ending in. 7z,. xz, or. gz, please remember to decompress it before burning it with the dd command.

The detailed explanation and more usage of all parameters of the dd command can be viewed by executing the man dd command in a Linux system.

11) After successfully burning the linux image of the development board to the NVMe SSD, you can use the **poweroff** command to shut down. Then please unplug the TF card and press the power button briefly to start the linux system in SPIFlash+NVMe SSD.

12) After starting the system in NVMe SSD, use the **df -h** command to see the actual hard drive capacity.

a. 128GB NVMe SSD。

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            3.8G  8.0K  3.8G   1% /dev
tmpfs           769M  1.4M  768M   1% /run
/dev/nvme0n1p2  118G  5.8G  111G   5% /
tmpfs           3.8G     0  3.8G   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           3.8G   16K  3.8G   1% /tmp
/dev/nvme0n1p1  256M   90M  166M  36% /boot
```

```
/dev/zram1          194M  9.9M  170M   6% /var/log
tmpfs               769M   60K  769M   1% /run/user/1000
tmpfs               769M   48K  769M   1% /run/user/0
```

b. 2TB NVMe SSD。

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            3.8G  8.0K  3.8G   1% /dev
tmpfs           769M  1.4M  768M   1% /run
/dev/nvme0n1p2  1.9T  4.1G  1.8T   1% /
tmpfs           3.8G     0  3.8G   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
/dev/zram2      3.7G   76K  3.5G   1% /tmp
/dev/nvme0n1p1  256M   90M  166M  36% /boot
/dev/zram1      194M   15M  165M   9% /var/log
tmpfs           769M   60K  769M   1% /run/user/1000
tmpfs           769M   48K  769M   1% /run/user/0
```

13) When the TF card and NVMe SSD burn the same system, **if both the TF card and NVMe SSD are inserted in the development board, power on the development board and u-boot will start the system in the TF card first**. However, since the systems in the TF card and the NVMe SSD are identical, the UUIDs of the /**boot** partition and the **rootfs** partition in the two storage devices are also the same. This may cause the partition in the NVMe SSD to be loaded when the TF card starts. Running the following script can solve this problem.

```
orangepi@orangepi:~$ sudo fix_mmc_ssd.sh
```

An identical system means that the image name is exactly the same. Even with the Debian11 system, different versions are different.

The fix_mmc_ssd.sh script is not available in the OPi OS Arch system.

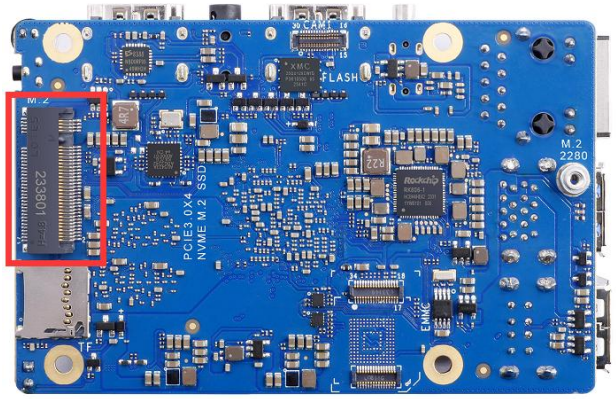
2. 6. 3. Method of burning using balenaEtcher software

Please do not use this method for OPi OS Arch system and OpenWRT system.

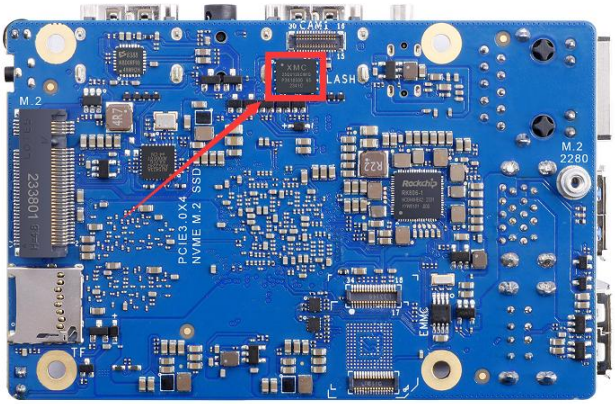
1) Firstly, it is necessary to prepare an NVMe SSD solid state drive with a PCIe interface specification of PCIe 3.0x4 for the M.2 slot of the development board.



2) Then insert the NVMe SSD into the M.2 PCIe interface of the development board and secure it in place.



3) Please ensure that the development board has already been attached with SPI Flash. The location of SPI Flash on the development board is shown in the following figure, and no other settings are required before starting the burning process.



4) Burning a Linux image to a SPIFlash+NVMe SSD requires the use of a TF card, so the first step is to burn the Linux image onto the TF card, and then use the TF card to boot the development board into the Linux system. The method of burning a Linux image to a TF card can be found in the two sections: **the method of burning a Linux image to a TF card based on Windows PC** and **the method of burning a Linux image to a TF card based on Ubuntu PC**.

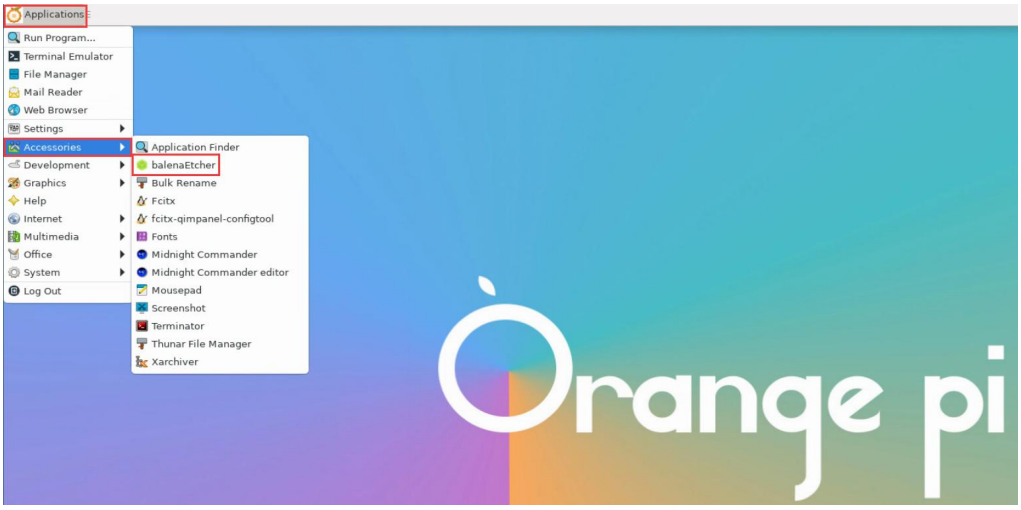
5) After booting into the Linux system on the TF card, please confirm that the NVMe SSD has been recognized by the Linux system on the development board. If NVMe SSD recognizes it normally, you can use the **sudo fdisk -l** command to see **nvme** related information.

```
orangepi@orangepi:~/Desktop$ sudo fdisk -l | grep "nvme0n1"
Disk /dev/nvme0n1: 1.86 TiB, 2048408248320 bytes, 4000797360 sectors
```

Using the **lspci** command, you can see an NVMe related PCI device.

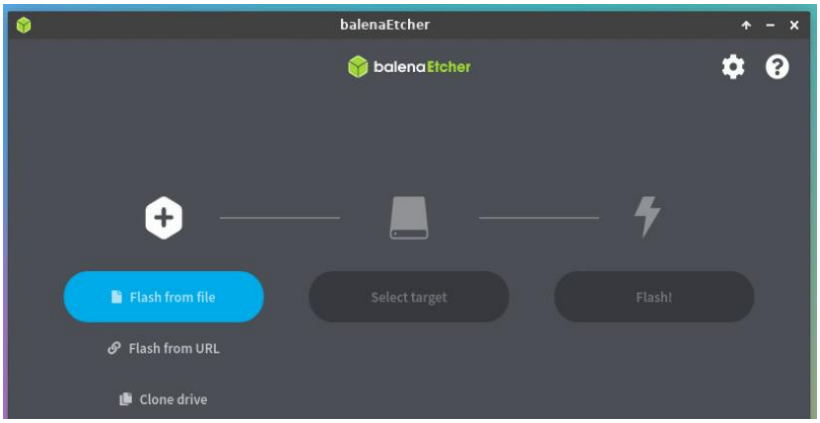
```
orangepi@orangepi:~/Desktop$ lspci
0004:40:00.0 PCI bridge: Fuzhou Rockchip Electronics Co., Ltd Device 3588 (rev 01)
0004:41:00.0 Non-Volatile memory controller: MAXIO Technology (Hangzhou) Ltd. NVMe SSD Controller MAP1202 (rev 01)
```

6) balenaEtcher is already pre installed in the linux image, and the opening method is as follows:



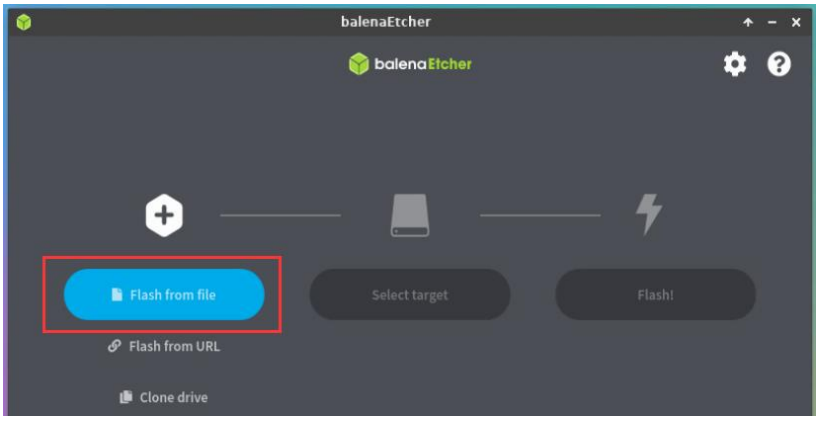
If it is not pre installed, please refer to the instructions in the section on downloading and installing the arm64 version of balenaEtcher.

7) The interface of balenaEtcher after opening is shown below:

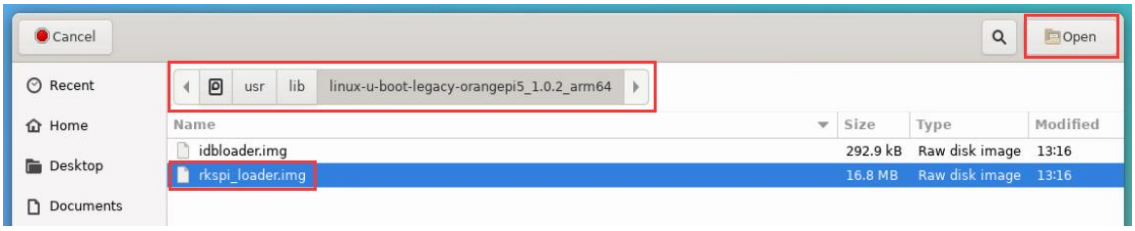


8) The method of burning u-boot to SPI Flash on the development board using balenaEtcher is as follows:

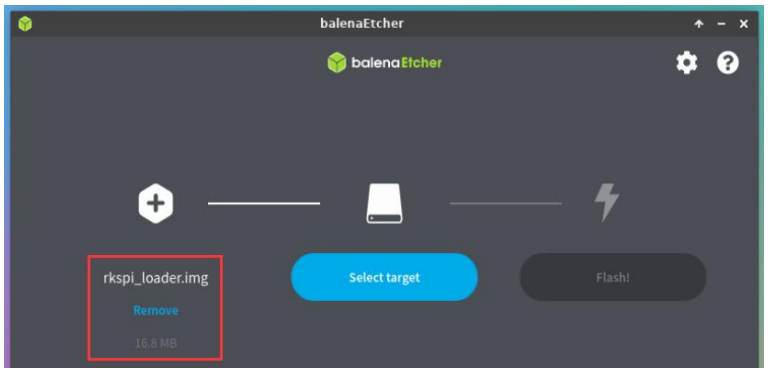
a. Firstly, open the balenaEtcher software and click on **Flash from file**.



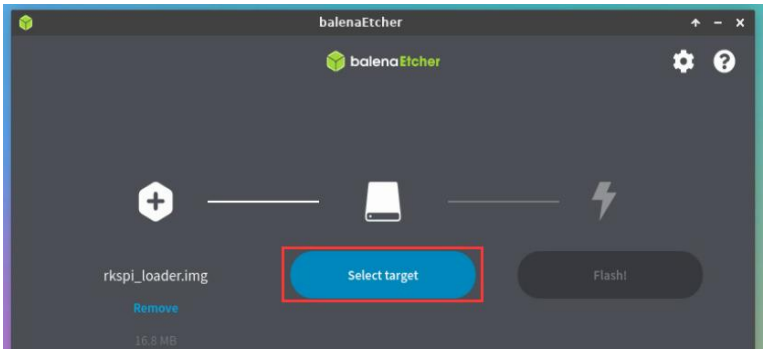
b. Then go to the **/usr/lib/linux-u-boot-legacy-orangepi5ultra_1.0.0_arm64/** directory, select **rkspi_loader.img**, and click **Open** to open it.



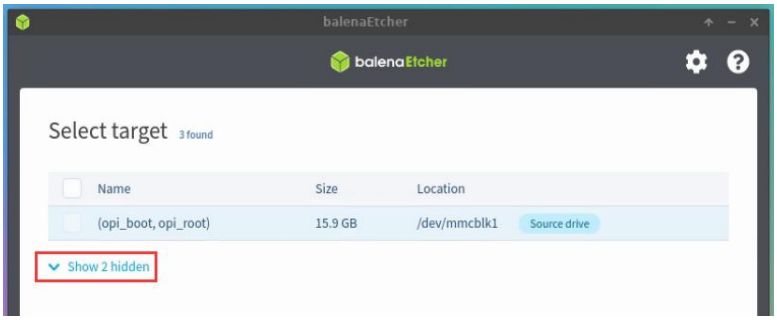
c. The interface after opening **rkspi_loader.img** is shown below:



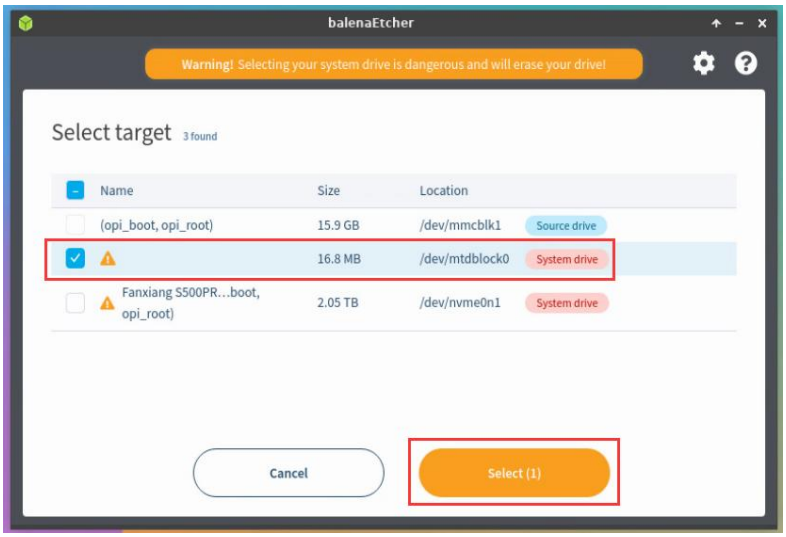
d. Then click on **Select target**.



e. Then click on **Show 2 hidden** to open the option for more storage devices.



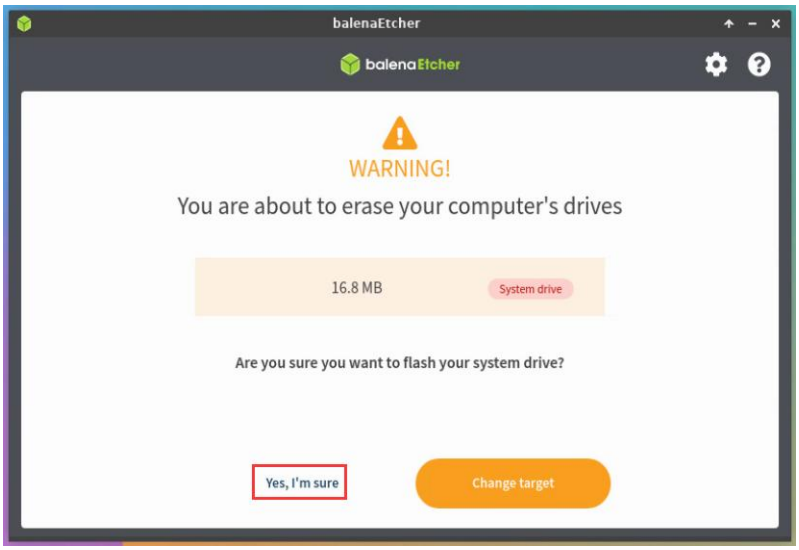
f. Then select the device name **/dev/mtdblock0** for SPI Flash, and click **Select**.



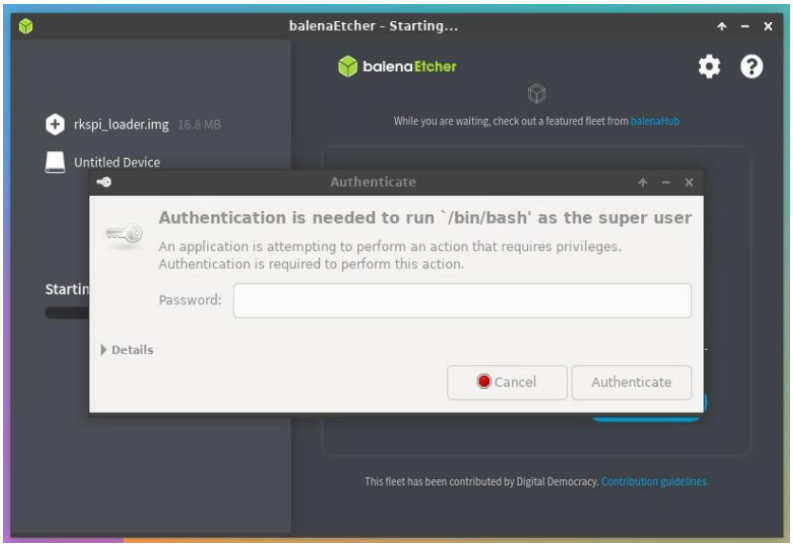
g. Then click on **Flash**.



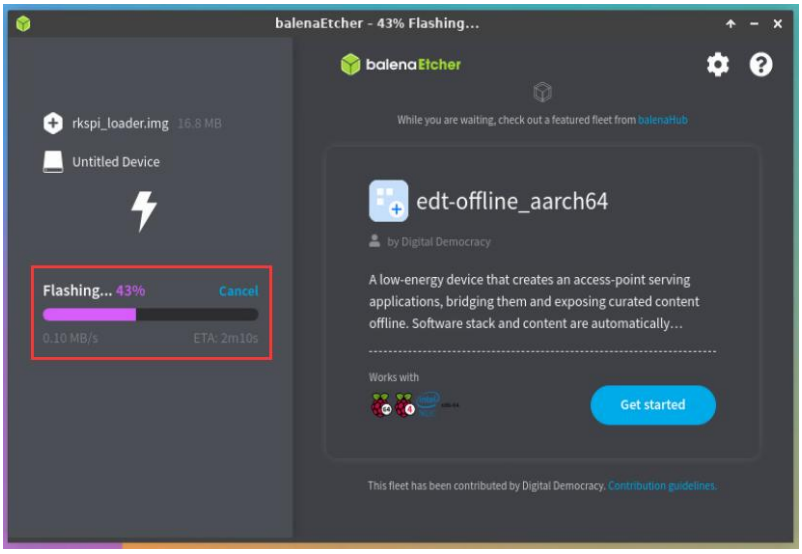
h. Then click **Yes, I'm sure**。



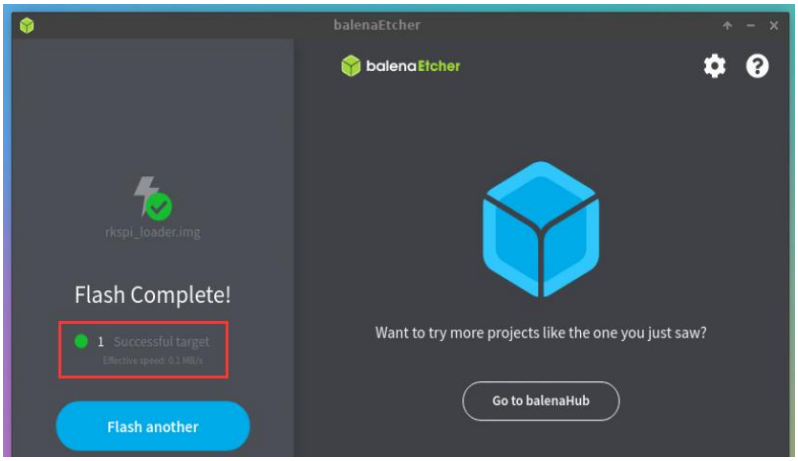
i. Then enter the password **orangepi** for the Linux system on the development board, and the u-boot image will be burned into SPI Flash.



j. The display of the burning process is as follows:

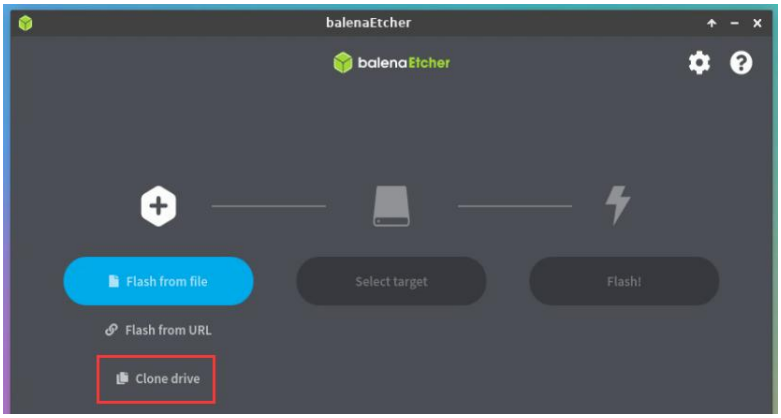


k. The display after burning is as follows:

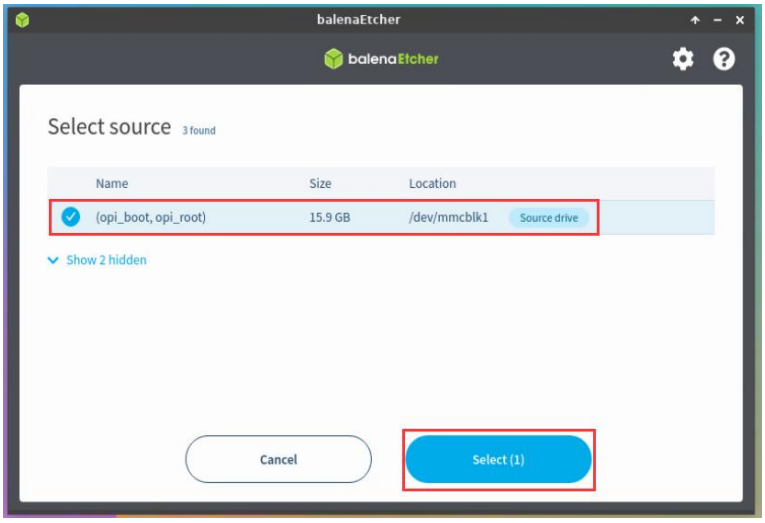


9) The method of burning Linux system from TF card to NVMe SSD (this method is equivalent to cloning the system from TF card to NVMe SSD).

a. First click **Clone drive**。



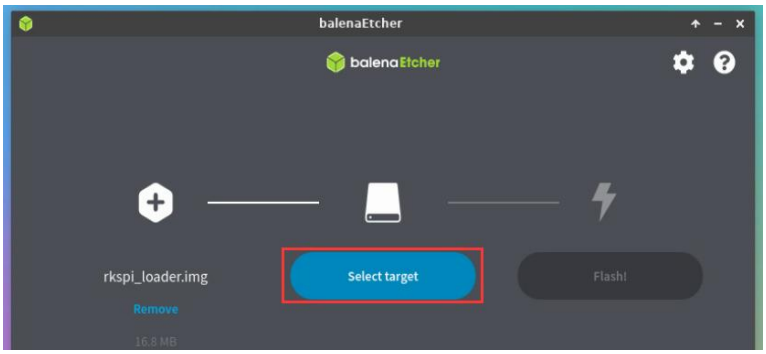
b. Then select the device name **/dev/mmcblk1** for the TF card.



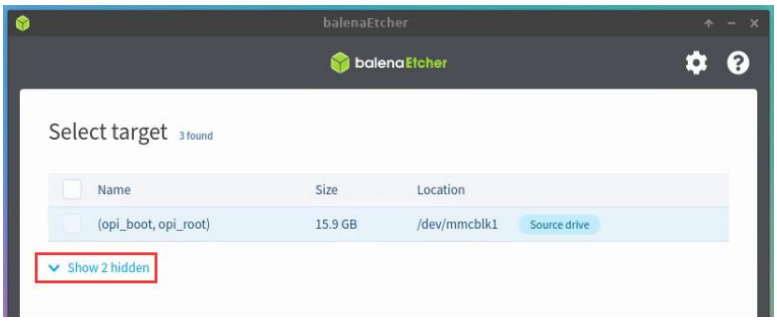
c. The interface after opening the TF card is shown below:



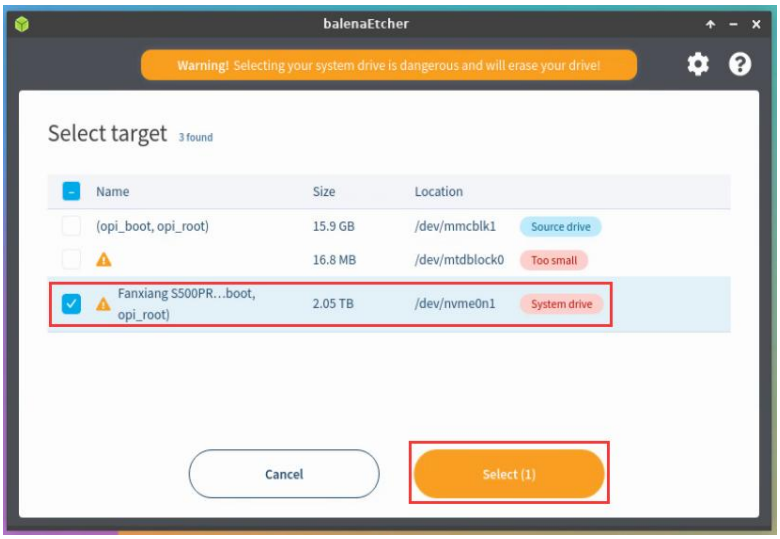
d. Then click **Select target**。



e. Then click on **Show 2 hidden** to open the option for more storage devices.



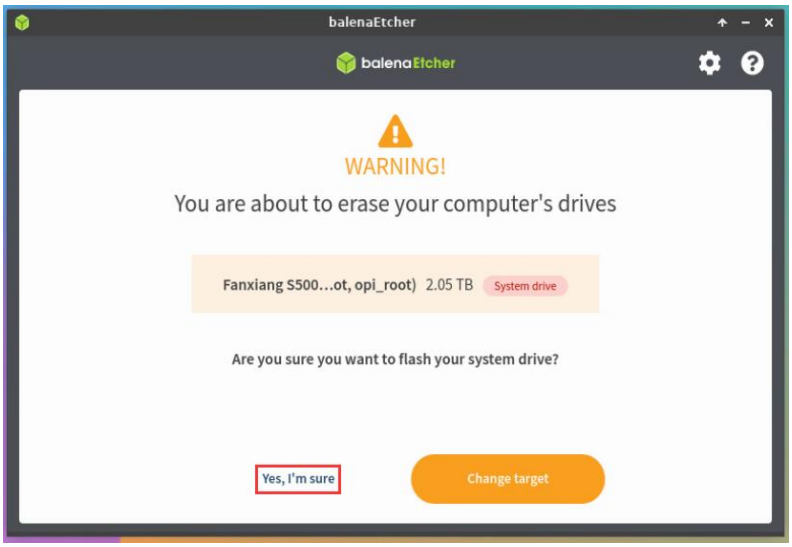
f. Then select the device name **/dev/nvme0n1** for NVMe SSD, and click **Select**.



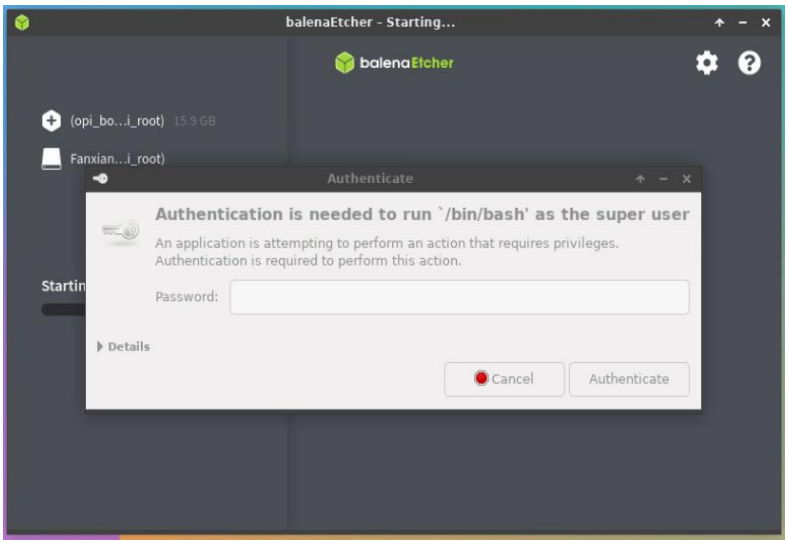
g. Then click **Flash**。



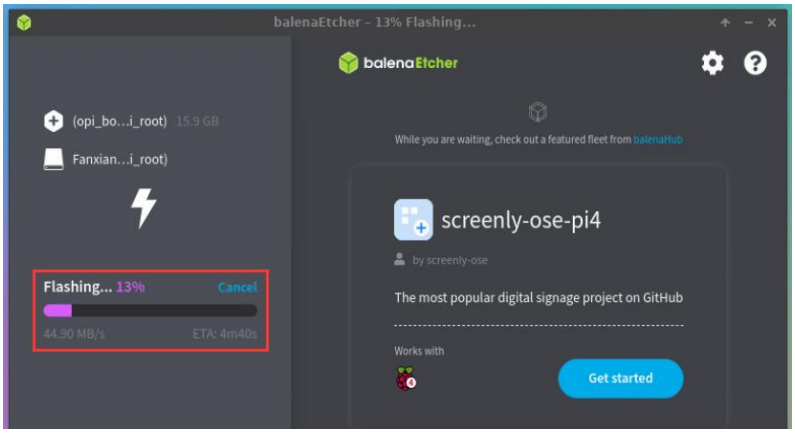
h. Then click **Yes, I'm sure**。

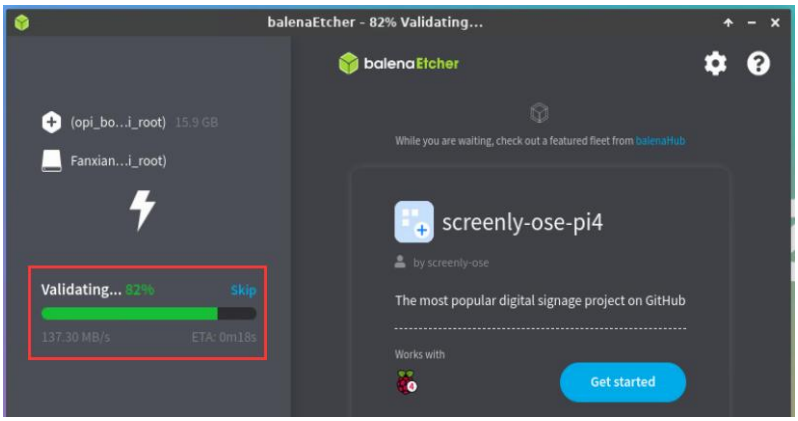


i. Then enter the password orangepi for the linux system on the development board, and the Linux image will be burned to the SSD.

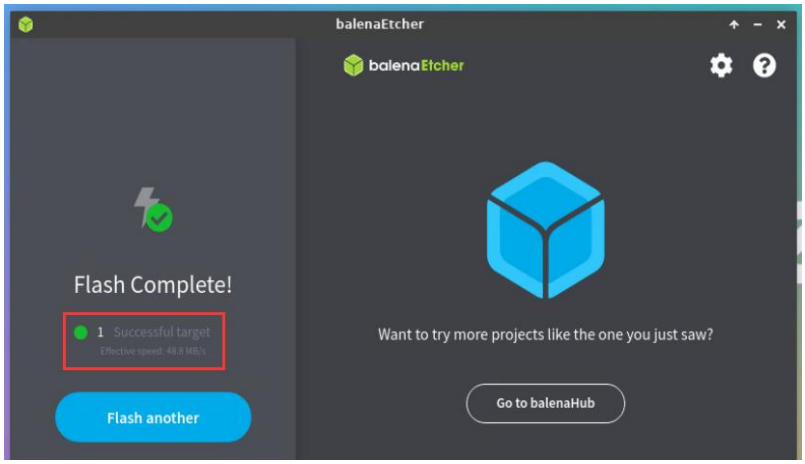


j. The display of the burning process is as follows:





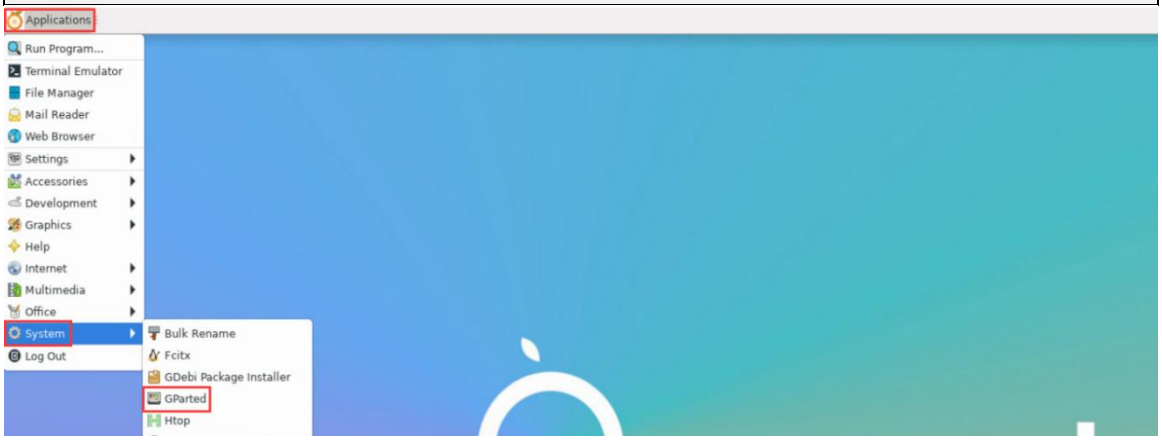
k. The display after burning is as follows:



l. Then it is necessary to expand the capacity of the rootfs partition in the NVMe SSD, as follows:

a) Firstly, open GParted. If **GParted** is not pre installed on the system, please use the apt command to install it.

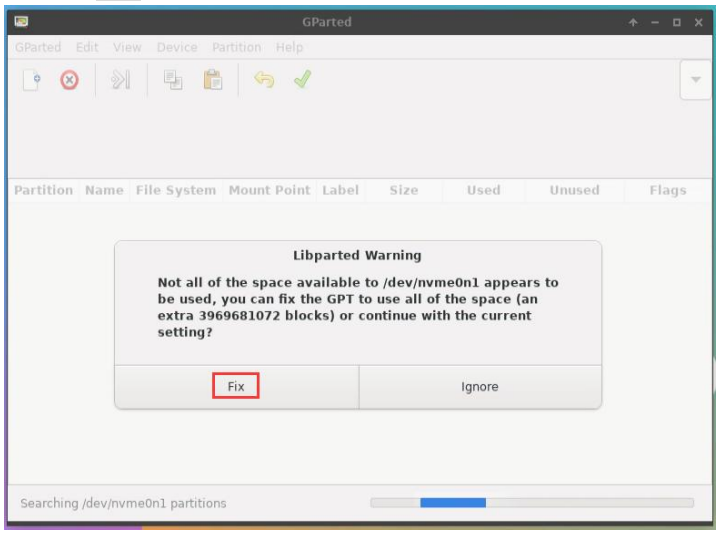
```
orangepi@orangepi:~$ sudo apt-get install -y gparted
```



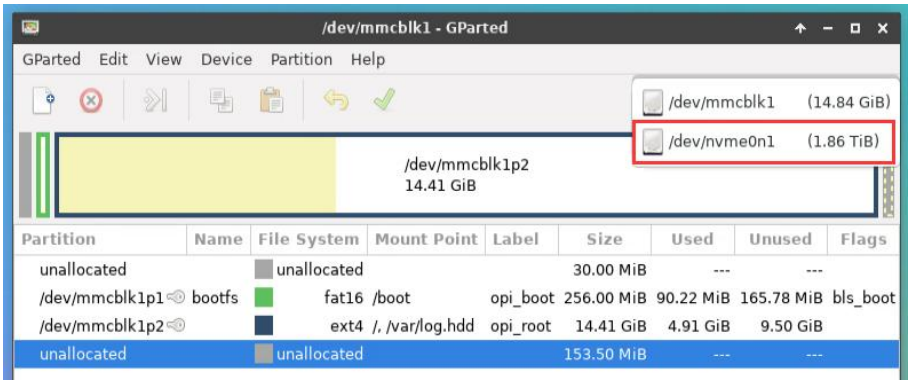
b) Then enter the linux system password **orangepi**, and click **Authenticate**.



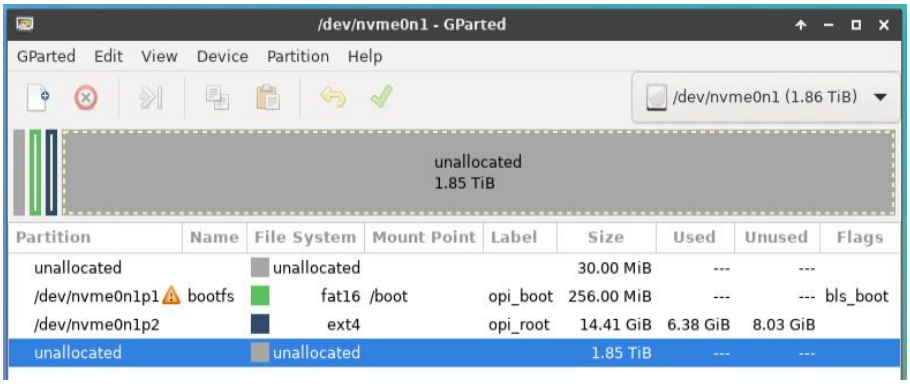
c) Then click **Fix**。



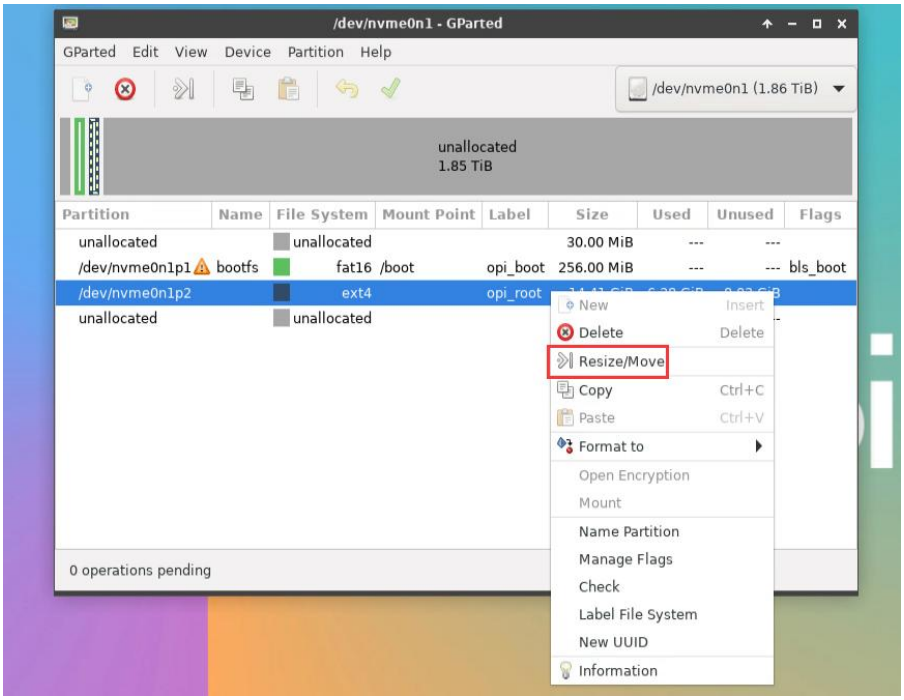
d) Then click NVMe SSD。



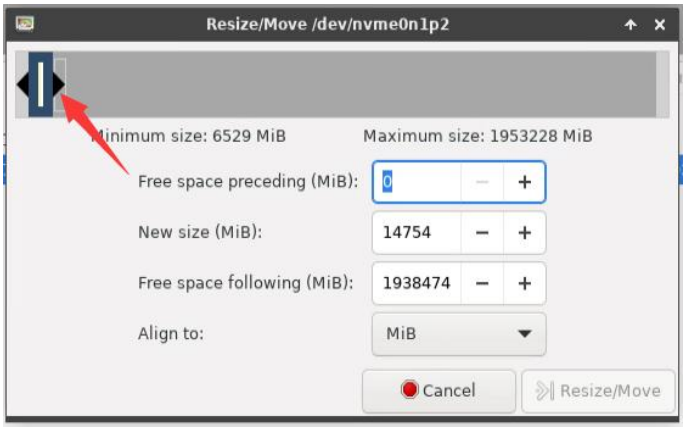
e) The display interface after selecting NVMe SSD is shown below:



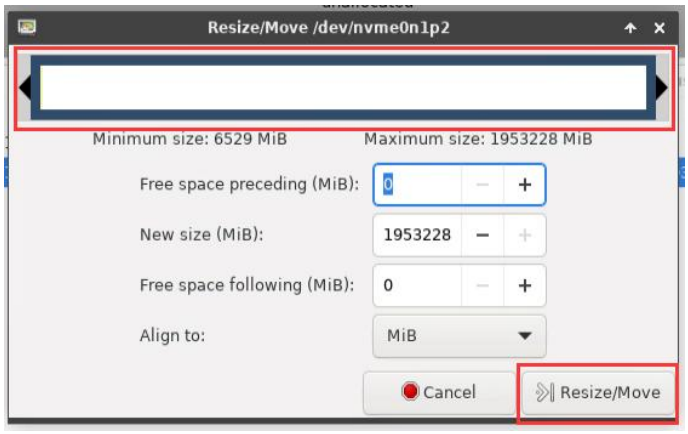
f) Then select the **/dev/nvme0n1p2** partition, right-click, and choose **Resize/Move**.



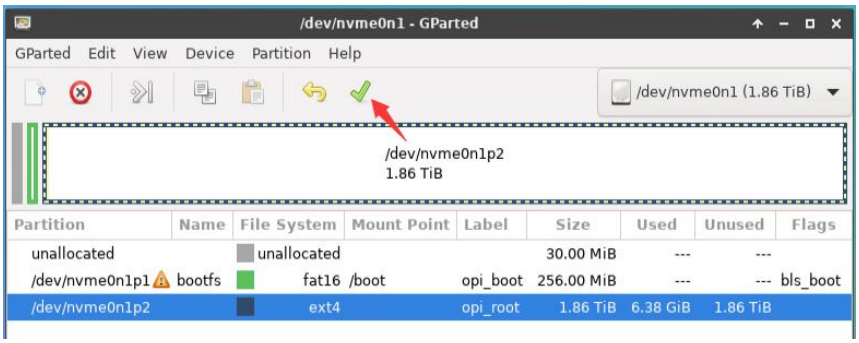
g) Then drag the capacity to its maximum at the position shown in the figure below.



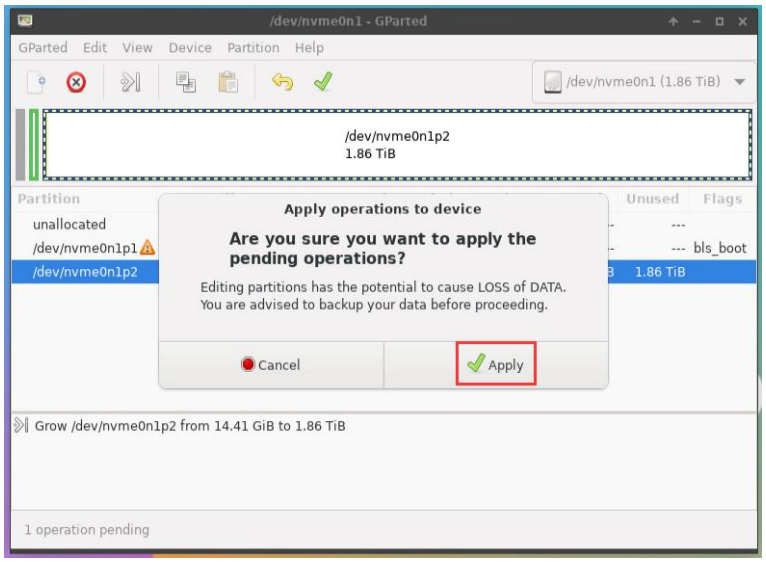
h) Then click **Resize/Move**。



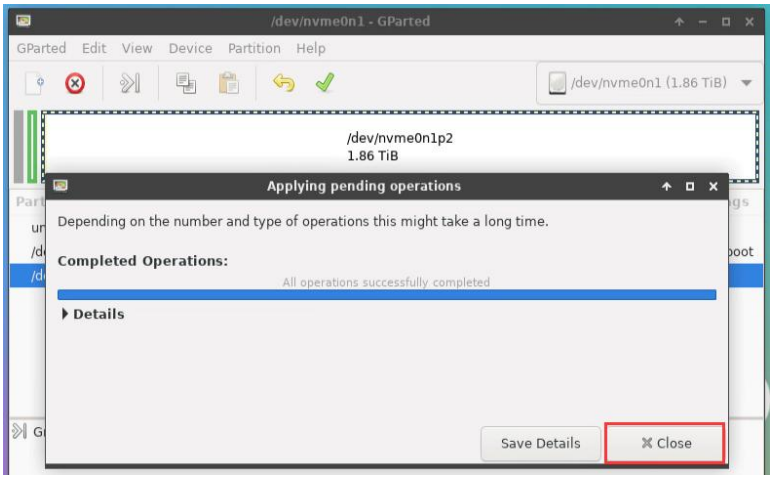
i) Then click on the green √ in the position shown below



j) Then Click **Apply**。



k) Then click **Close** to close it

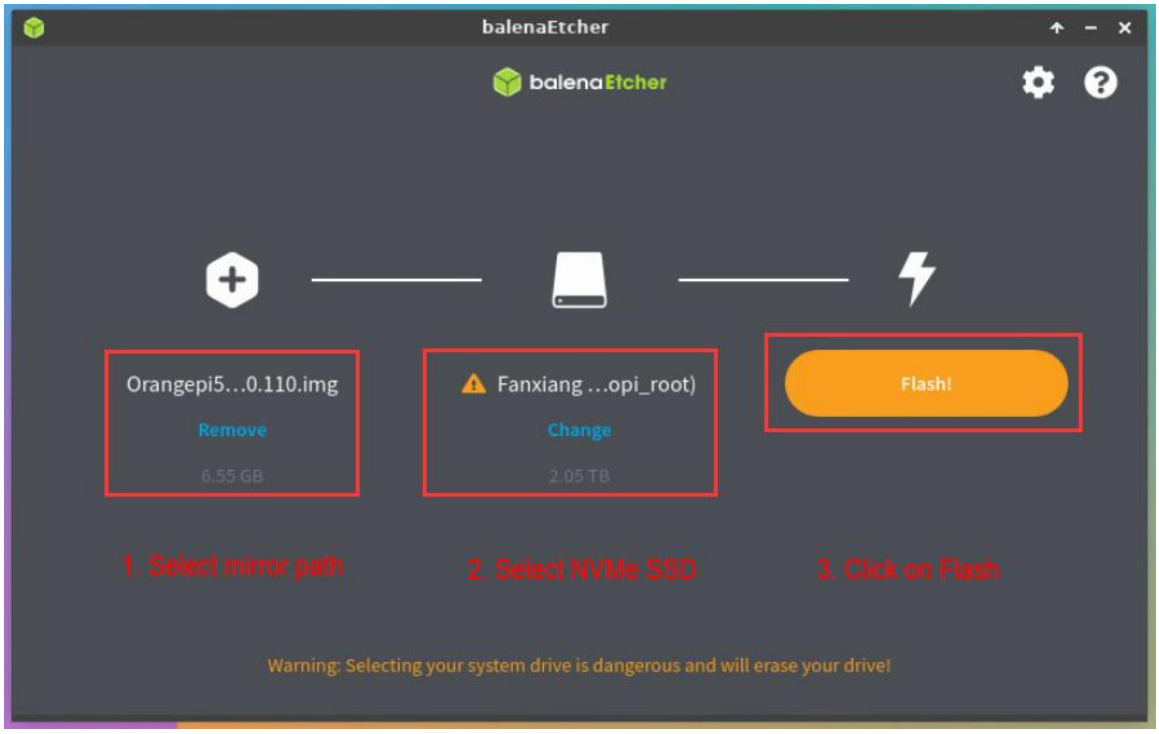


m. At this point, you can use the **sudo poweroff** command to shut down. Then please unplug the TF card and press the power button briefly to start the linux system in SPIFlash+NVMe SSD.

10) Step **9)** is to clone the system from the TF card to the NMVe SSD. We can also directly burn the Linux image file to the NVMe SSD. Here are the general steps:

a. Upload the Linux image file to the Linux system on the development board.

b. Then use balenaEtcher to burn it.



c. **After burning the image using this method, there is no need to manually expand it. The first startup will automatically expand it.**

2.7. Method for burning Linux images to SPIFlash+USB storage devices

Note that the Linux image referred to here specifically refers to Linux distribution images such as Debian, Ubuntu, OpenWRT, or OPi OS Arch downloaded from the Orange Pi data download page.

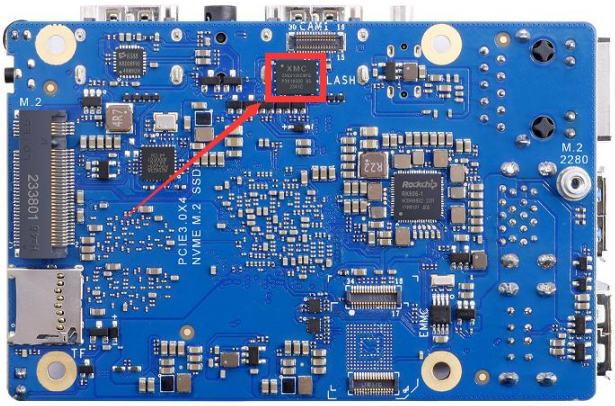
1) Firstly, it is necessary to prepare a USB storage device, such as a USB flash drive.

2) Then please refer to the instructions in two sections: **the method of burning Linux images to TF cards based on Windows PC** and **the method of burning Linux images to TF cards based on Ubuntu PC** to burn Linux images to USB storage devices. There is no difference between burning a Linux image to a USB storage device and burning a Linux image to a TF card (when the TF card is inserted into the card reader, the card reader is actually equivalent to a USB flash drive).

3) Then insert the USB storage device that has burned the Linux system into the USB interface of the development board. **Note that only the two USB 2.0 interfaces shown in the following figure support booting the Linux system, and the blue USB 3.0 interface does not support it**.



4) The position of SPI Flash on the development board is shown in the following figure, and no other settings are required before starting to burn.



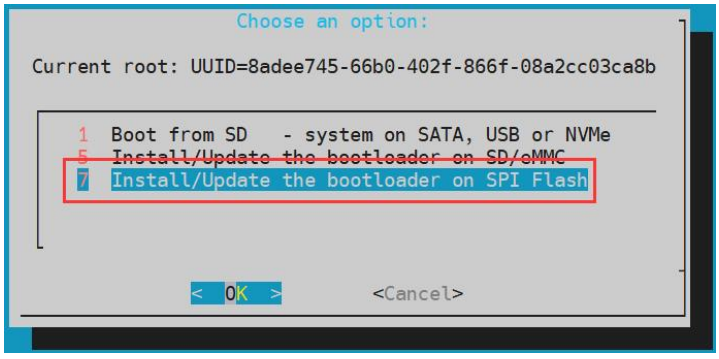
5) Burning the u-boot image to SPIFlash requires the use of a TF card, so the first step is to burn the Linux image onto the TF card, and then use the TF card to boot the development board into the Linux system. The method of burning a Linux image to a TF card can be found in the two sections: **the method of burning a Linux image to a TF card based on Windows PC** and **the method of burning a Linux image to a TF card based on Ubuntu PC**.

6) After starting the linux system with a TF card, you can burn the u-boot image to SPI Flash.

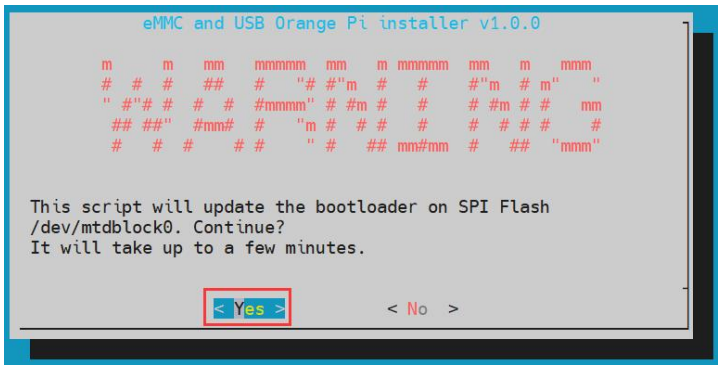
a. First, run **nand-sata-install**. **Regular users should remember to grant sudo privileges**

```
orangepi@orangepi:~$ sudo nand-sata-install
```

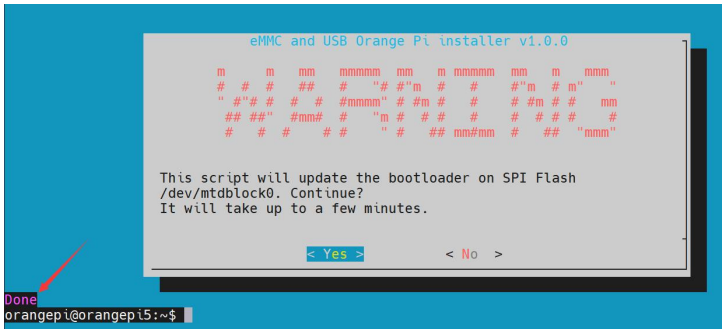
b. Then choose **7 Install/Update ther bootloader on SPI Flash**。



c. Then choose **<Yes>**。



d. Then please be patient and wait for the burning to complete. After the burning is completed, the following will be displayed (a **Done** will appear in the bottom left corner):



There is no nand-sata-install script in the OPi OS Arch system. Please use the following command to mirror u-boot to SPI Flash:

[orangepi@orangepi ~]$ **sudo dd if=/boot/rkspi_loader.img of=/dev/mtdblock0**

7) At this point, you can use the **poweroff** command to shut down. Then please unplug the TF card and press the power button briefly to start the linux system in the SPIFlash+USB storage device.

8) After starting the system in the USB storage device, use the **df -h** command to see the actual capacity of the USB storage device.

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            3.8G  8.0K  3.8G   1% /dev
tmpfs           769M  588K  769M   1% /run
/dev/sda2        15G  1.6G   13G  11% /
tmpfs           3.8G     0  3.8G   0% /dev/shm
```

tmpfs	5.0M	4.0K	5.0M	1% /run/lock
/dev/zram2	3.7G	60K	3.5G	1% /tmp
/dev/sda1	**256M**	**111M**	**146M**	**44% /boot**
/dev/zram1	194M	9.0M	171M	5% /var/log
tmpfs	769M	0	769M	0% /run/user/1000

2. 8. Method of burning Android image to TF card

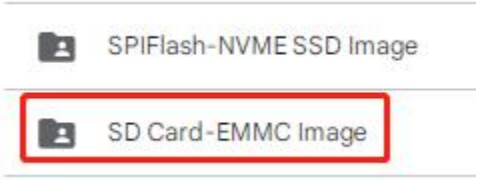
2. 8. 1. Method of burning using RKDevTool

1) Firstly, it is necessary to prepare a high-quality USB2.0 male to male data cable.

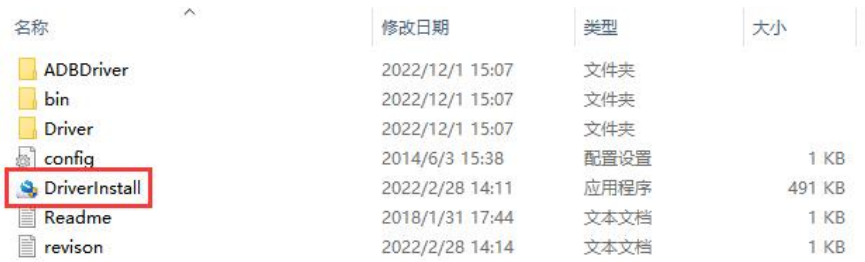


2) Then download the Rockchip driver **DriverAssitant_v5.12.zip** and the burning tool **RKDevTool_Release_v3.15.zip** from the **Orange Pi's download page**.

3) Then download the Android image from the **Orange Pi's data download page**. After opening the download link for the Android image, you can see the following two types of Android images. Please select the images in the **TF card and eMMC startup image** folders to download.

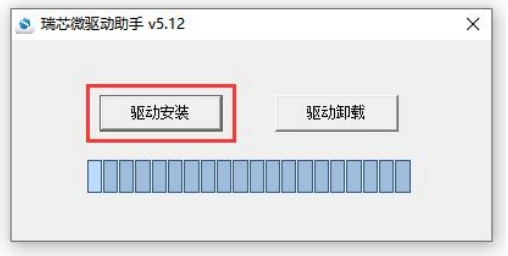


4) Then use decompression software to extract the **DriverAssitant_v5.12.zip** file, and find the **DriverInstall.exe** executable file in the extracted folder and open it.

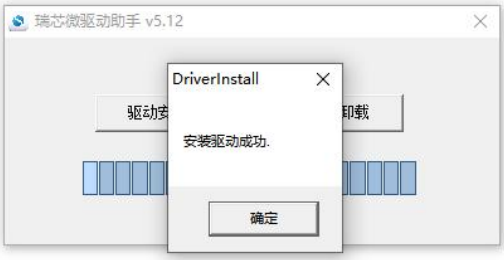


5) The steps to install the Rockchip driver after opening **DriverInstall.exe** are as follows:

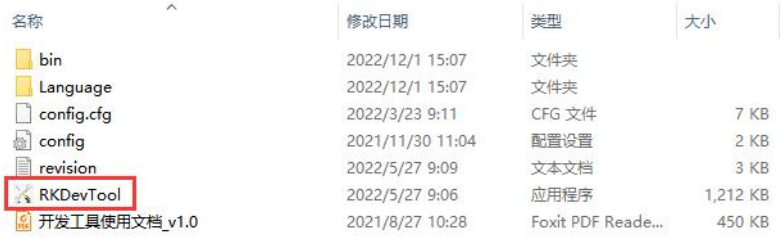
a. Click the "**Driver Install**" button.



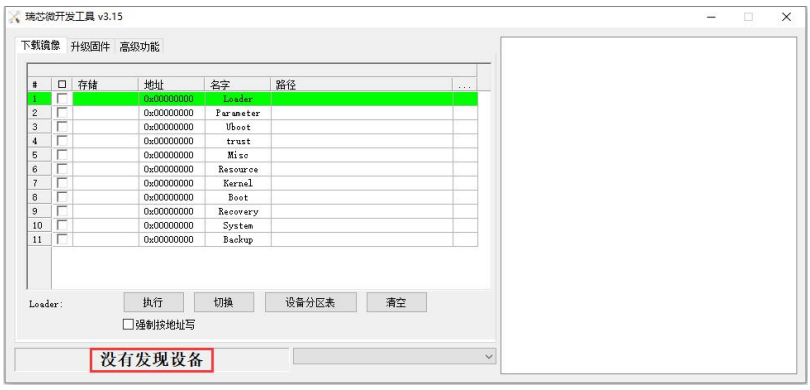
b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.



6) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.

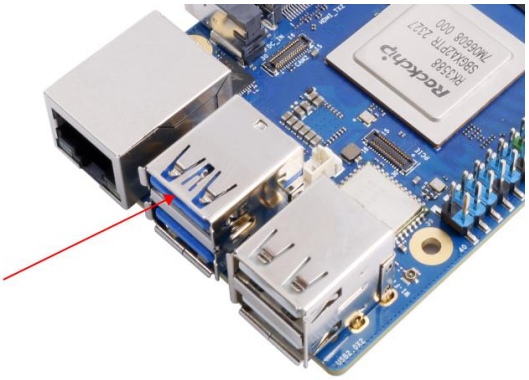


7) After opening the **RKDevTool** burning tool, because the computer has not yet been connected to the development board through a USB2.0 male to male data cable, the bottom left corner will prompt "**Device not found**"



8) Then start burning the Android image onto the TF card.

a. Firstly, connect the development board to the Windows computer via a USB2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:

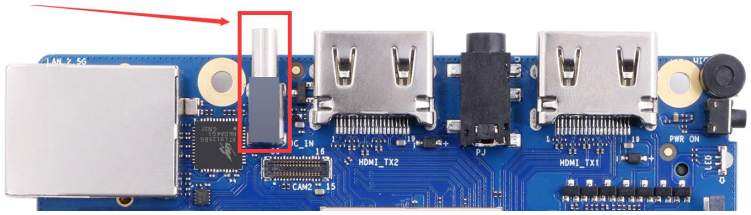


b. Then insert the TF card into the development board and ensure that the board is not connected to a power source.

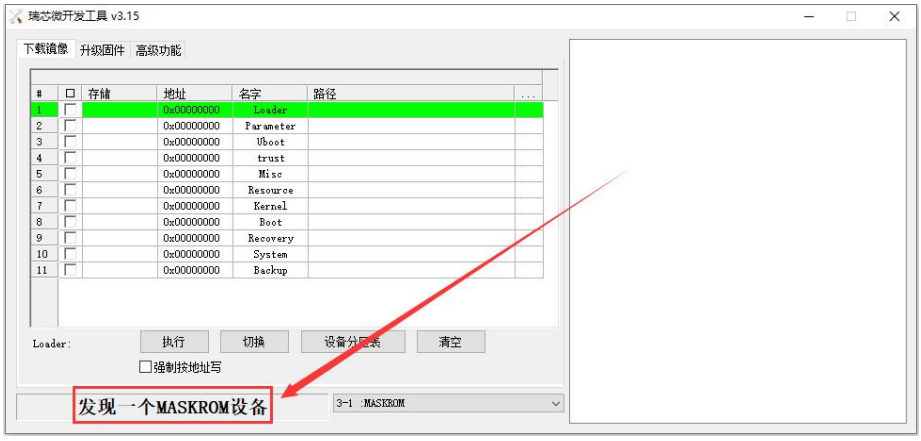
c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is shown in the following figure:



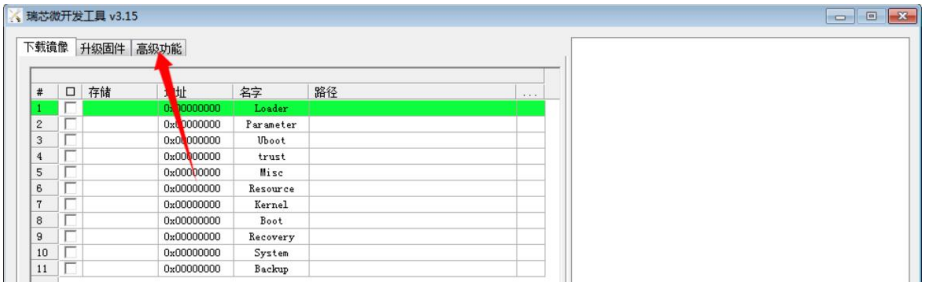
d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



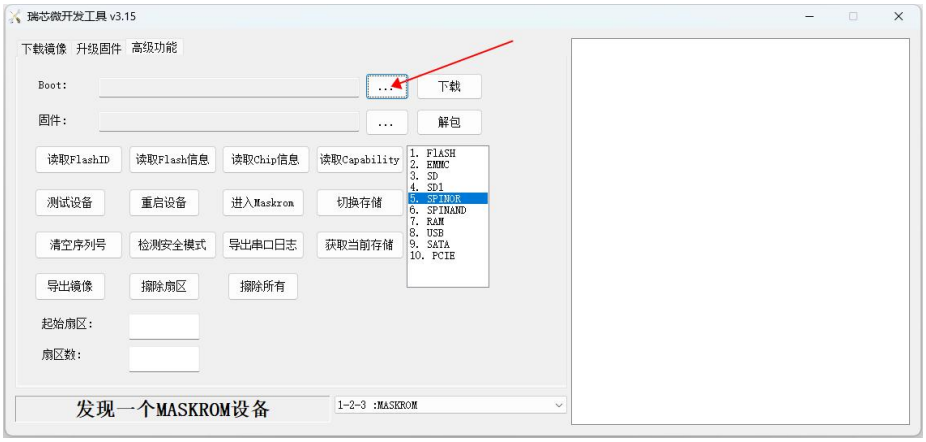
e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**".



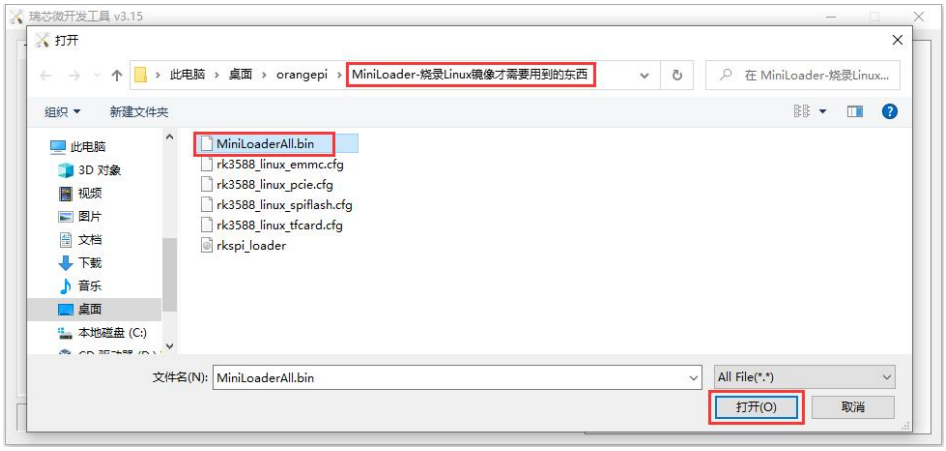
f. Then please select **advanced features**



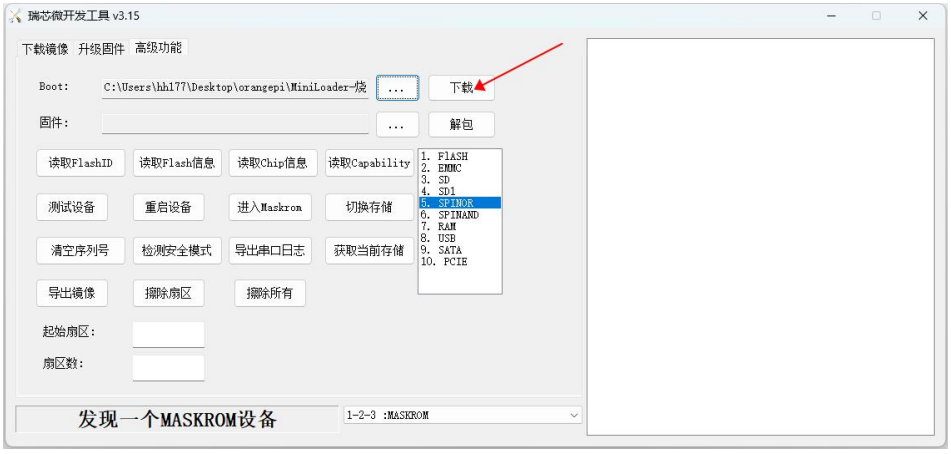
g. Then click on the position shown in the following image.



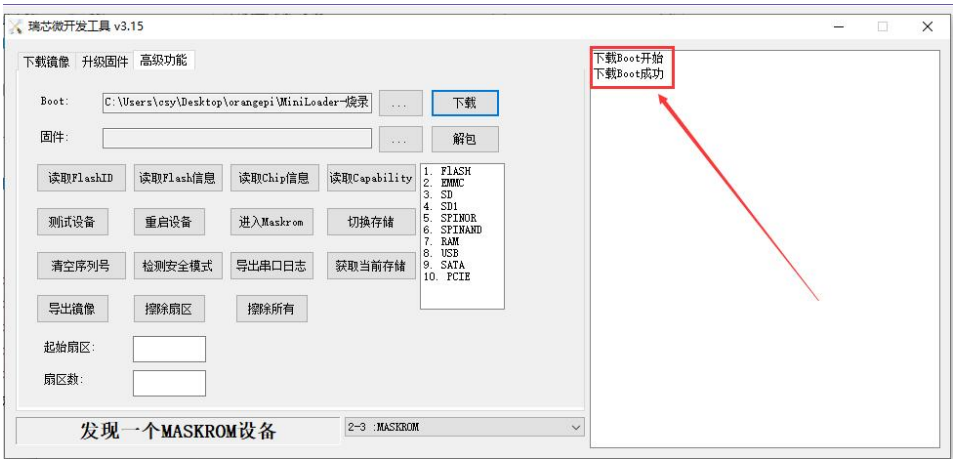
h. Select **MiniLoaderAll.bin** from the **MiniLoader** folder downloaded earlier, and then click Open.



i. Then click on **download**



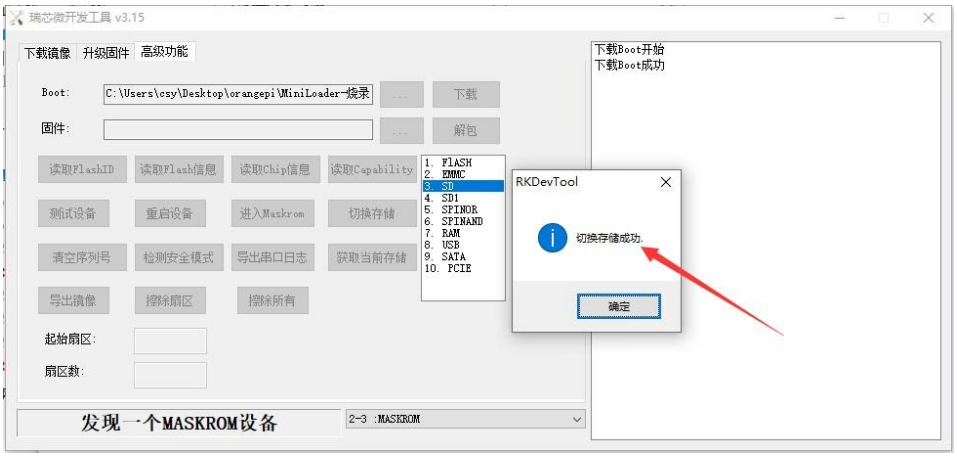
j. After downloading **MiniLoaderAll.bin**, the display is shown in the following figure:



k. Then select the storage device as **SD** and click on **switch storage**.



l. The display of successful switching is shown in the following figure:



m. Then click on the "**Upgrade Firmware**" section of the burning tool.



n. Then click the "**Firmware**" button to select the path of the Android image that needs to be burned.



o. Finally, clicking the "**Upgrade**" button will start burning, and the log during the burning process is shown in the following figure. After the burning is completed, the Android system will automatically start.



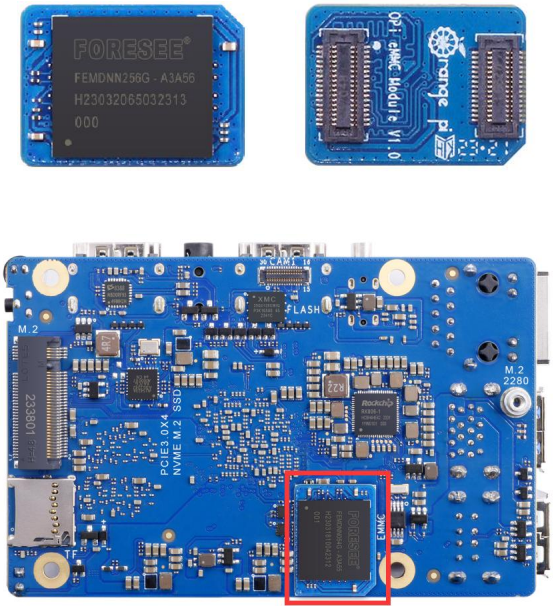
2.9. Method for burning Android images to eMMC

2.9.1. Method of burning using RKDevTool

Note that all the following operations were performed on a Windows computer.

1) The development board has reserved an extension interface for eMMC. Before burning the system to eMMC, it is necessary to purchase an eMMC module that matches the eMMC interface of the development board. Then install the eMMC module onto the development board.

The method of inserting the eMMC module into the development board is as follows:

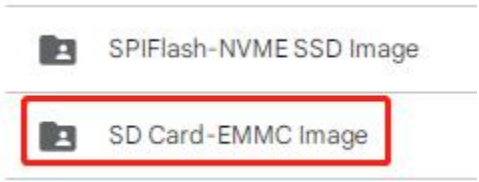


2) We also need to prepare a high-quality USB 2.0 male to male data cable.



3) Then download the Rockchip driver **DriverAssitant_v5.12.zip** and the burning tool **RKDevTool_Release_v3.15.zip** from the **Orange Pi's download page**.

4) Then download the Android image from the **Orange Pi's data download page**. After opening the download link for the Android image, you can see the following two types of Android images. Please select **the images in the TF card and eMMC startup image** folders to download.



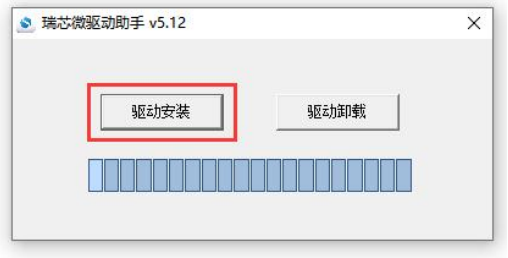
5) Then use decompression software to extract the **DriverAssitant_v5.12.zip** file, and

find the **DriverInstall.exe** executable file in the extracted folder and open it.

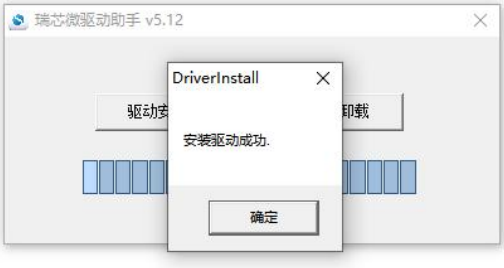


6) The steps to install the Rockchip driver after opening **DriverInstall.exe** are as follows:

a. Click the '**Driver Installation**' button.



b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.

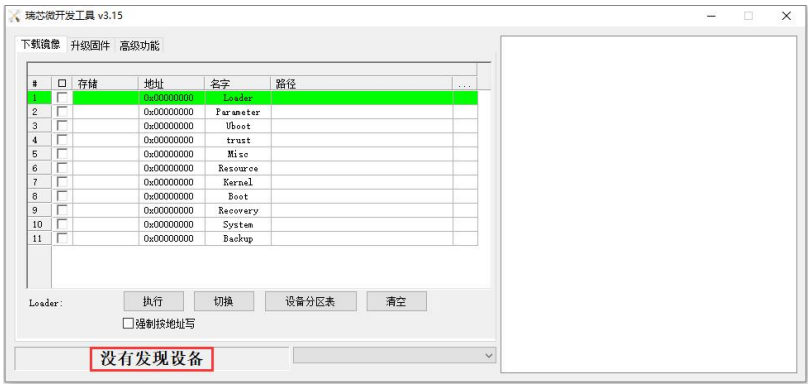


7) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.



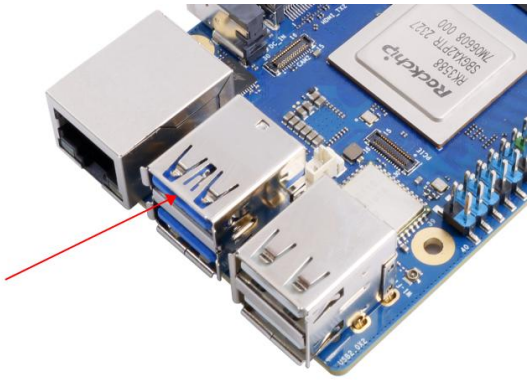
8) After opening the **RKDevTool** burning tool, the bottom left corner will prompt "**Device not found**" because the computer has not yet been connected to the development

board through a USB2.0 male to male data cable.



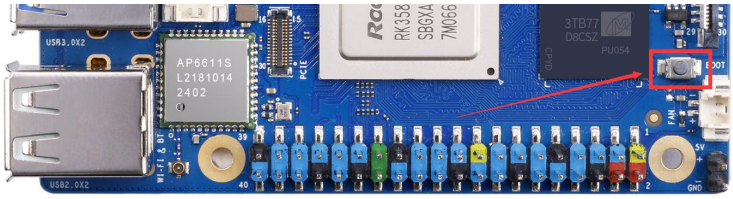
9) Then start burning the Android image into eMMC.

a. Firstly, connect the development board to the Windows computer via a USB2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:

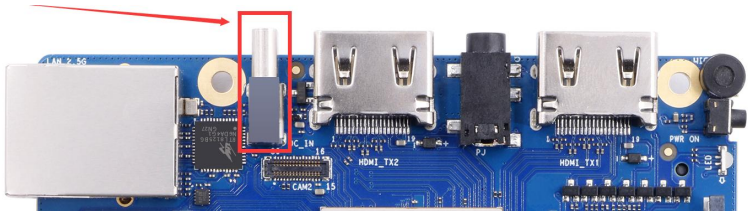


b. Ensure that the development board is not connected to a power source or inserted with a TF card.

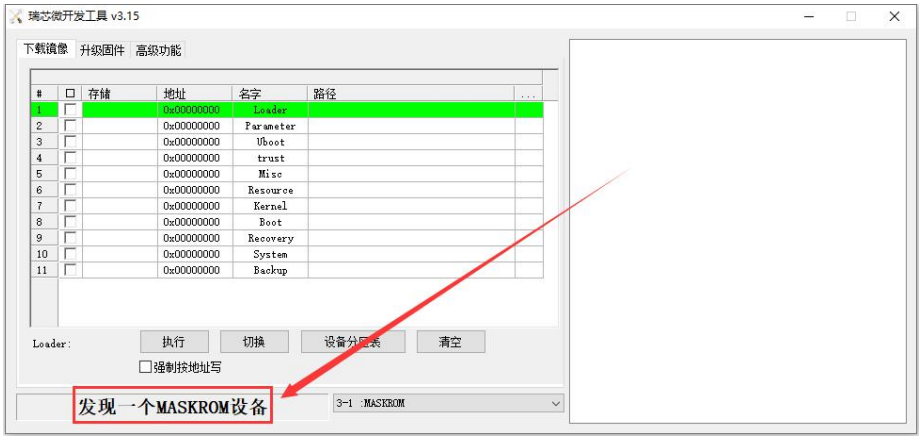
c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is shown in the following figure:



d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**"



f. Then click on the "**Upgrade Firmware**" section of the burning tool.



g. Then click the "**Firmware**" button to select the path of the Android image that needs to be burned.



h. Then click the "**Firmware**" button to select the path of the Android image that needs to be burned.



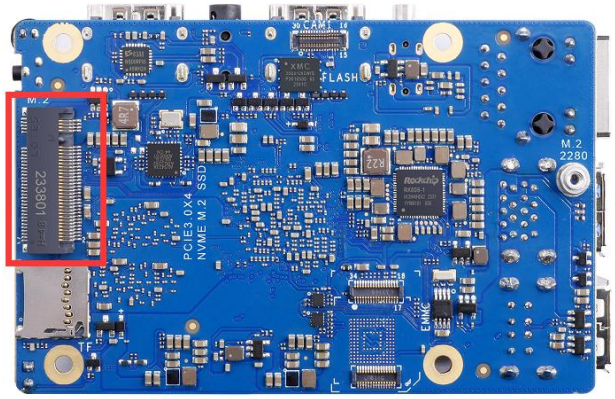
2.10. Method for burning Android images to SPIFlash+NVMe SSD

Note that all the following operations were performed on a Windows computer.

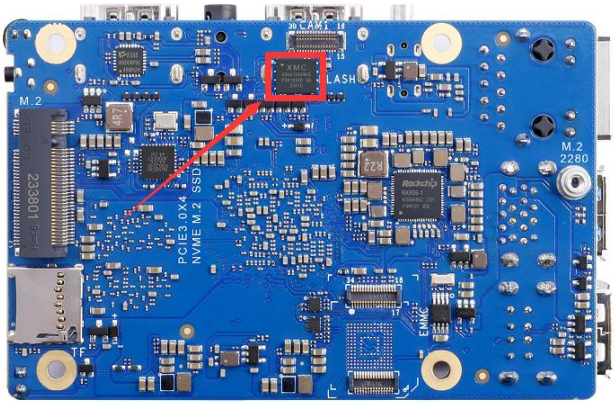
1) First, you need to prepare an NVMe SSD solid state drive.



2) Then insert the NVMe SSD into the M.2 PCIe interface of the development board and secure it in place.



3) Please ensure that the development board has already been attached with SPI Flash. The location of SPI Flash on the development board is shown in the following figure, and no other settings are required before starting the burning process.



4) We also need to prepare a high-quality USB 2.0 male to male data cable.



5) Then download the Rockchip micro driver **DriverAssitant_v5.12.zip** and the burning tool **RKDevTool_Release_v3.15.zip** from the **Orange Pi's download page**.

6) Then download the Android image, open the download link for the Android image, and you will see the following two types of Android images. Please select the image in

the **SPIFlash-NVME SSD** folder to download.

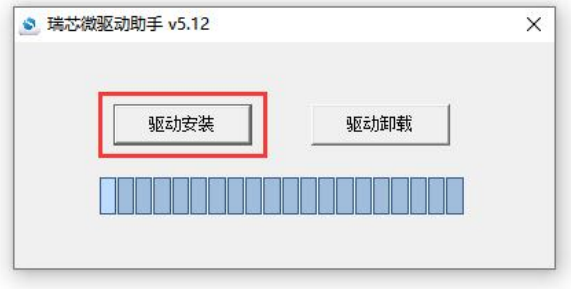


7) Then use decompression software to extract the **DriverAssitant_v5.12.zip**, and find the **DriverInstall.exe** executable file in the extracted folder and open it.

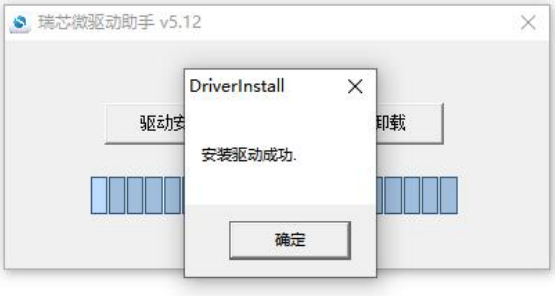


8) The steps to install the Rockchip driver after opening **DriverInstall.exe** are as follows:

a. Click the "**Driver Installation**" button.



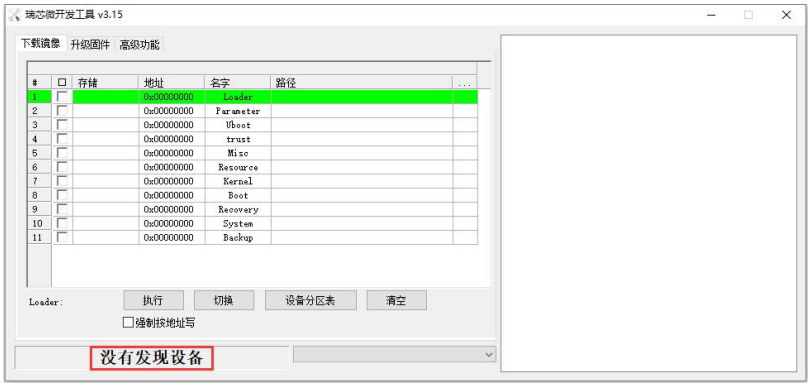
b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.



9) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.

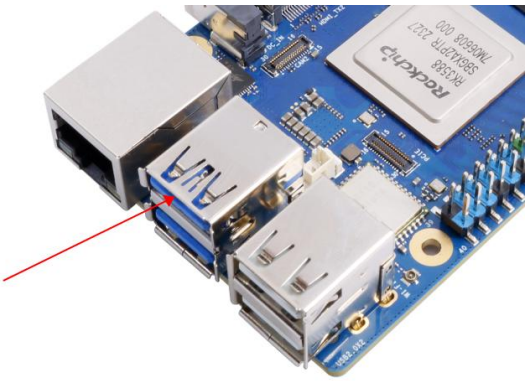


10) After opening the **RKDevTool** burning tool, the bottom left corner will prompt "**Device not found**" because the computer has not yet been connected to the development board through a USB 2.0 male to male data cable.



11) Then start burning the Android image to SPIFlash+NVMe SSD.

a. Firstly, connect the development board to the Windows computer via a USB2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:

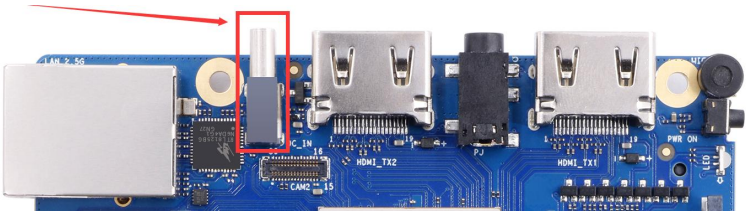


b. Ensure that the development board is not inserted with a TF card or connected to a power source.

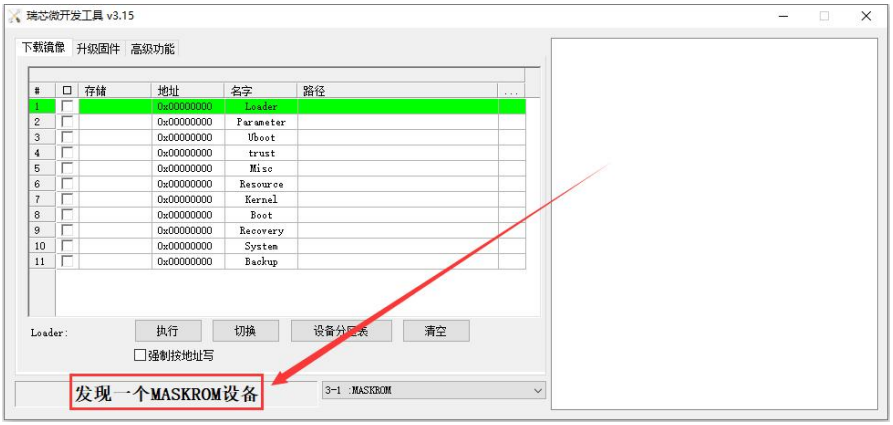
c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is shown in the following figure:



d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**".



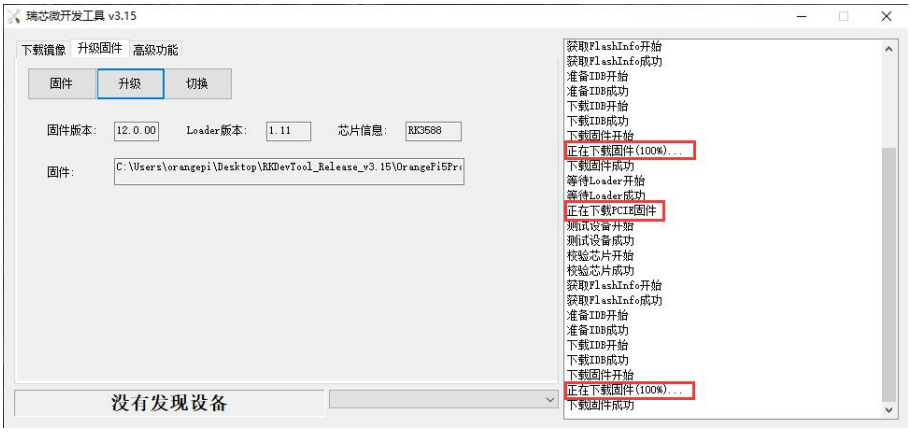
f. Then click on the "**Upgrade Firmware**" section of the burning tool.



g. Then click the "**Firmware**" button to select the Android image that needs to be burned.



h. Finally, clicking the "**Upgrade**" button will start burning. The burning process is shown in the following figure. You can see that the firmware will first be burned to SPIFlash, and then burned to PCIE. After the burning is completed, the Android system will automatically start.



2.11. Method for burning Orange Pi OS (Droid) image to TF card

Note that all the following operations were performed on a Windows computer.

1) Firstly, prepare a TF card with a capacity of 8GB or higher, and the transfer speed of the TF card must be **class10** or above. It is recommended to use TF cards from brands such as SanDisk.

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

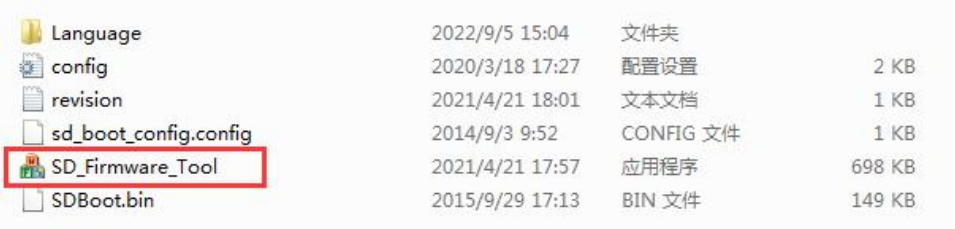
3) Then download the SDDiskTool burning tool from the **Orange Pi's download page**,

ensuring that the SDDiskTool tool version is the latest v1.72.

4) Then download the image of Orange Pi OS (Droid) from the **Orange Pi data download page**.

5) Then use decompression software to extract the compressed file of the downloaded Orange Pi OS (Droid) image. In the decompressed file, the file ending with ".img" is the Orange Pi OS (Droid) image file, with a size of over 1GB.

6) Then use decompression software to extract **SDDiskTool_v1.72.zip**. This software does not need to be installed. Simply find **SD_Firmware_Tool.exe** in the extracted folder and open it.



7) After opening **SDDiskTool**, if the TF card recognition is normal, the inserted disk device will be displayed in the "**Select Removable Disk Device**" column. **Please make sure that the displayed disk device matches the drive letter of the TF card you want to burn.** If it does not display, you can try unplugging the TF card.



8) After confirming the drive letter, you can first format the TF card by clicking the **recover disk** button in SDDiskTool, or you can use the **SD Card Formatter** mentioned earlier to format the TF card.



9) Then start writing the Orange Pi OS (Droid) image to the TF card.

a. First, check "**SD Start**" in the "**Select Function Mode**".

b. Then select the path of the Orange Pi OS (Droid) image in the "**Choose Upgrade Firmware**" column.

c. Finally, clicking the '**Start Creating**' button will start burning the Orange Pi OS (Droid) image onto the TF card.



10) After burning, you can exit the SDDiskTool software and then unplug the TF card from the computer and insert it into the development board to start.



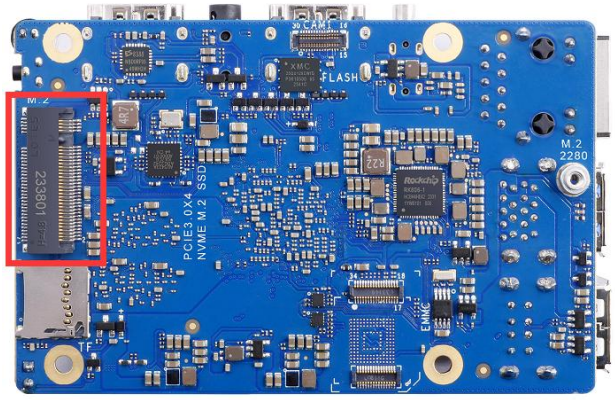
2. 12. Burn Orange Pi OS (Droid) image to SPIFlash+NVMe SSD

Note that all the following operations were performed on a Windows computer.

1) First, you need to prepare an NVMe SSD solid state drive.

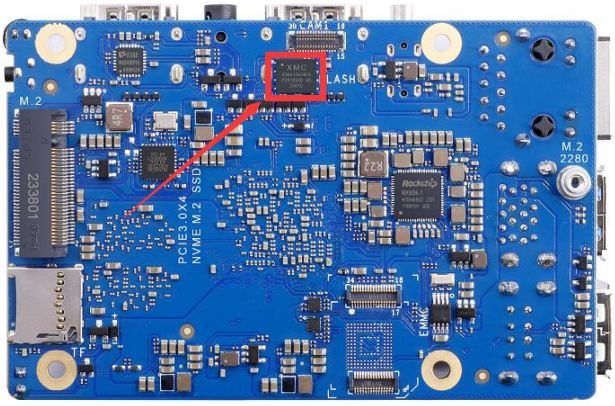


2) Then insert the NVMe SSD into the M.2 PCIe interface of the development board and secure it in place.



3) Please ensure that the development board has already been attached with SPI Flash. The location of SPI Flash on the development board is shown in the following figure, and

no other settings are required before starting the burning process.



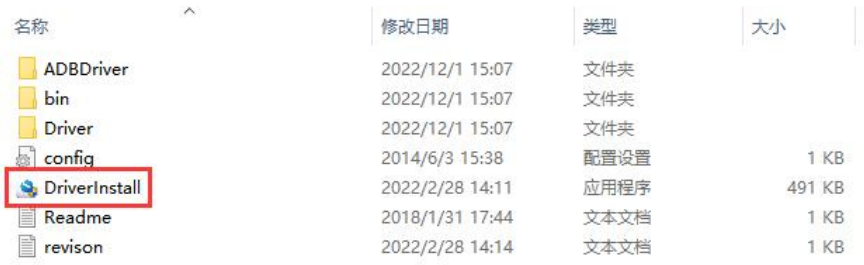
4) We also need to prepare a high-quality USB 2.0 male to male data cable.



5) Then download the Rockchip micro driver **DriverAssitant_v5.12.zip** from the **Orange Pi's download page**, and use the burning tool **RKDevTool_Release_v3.15.zip**.

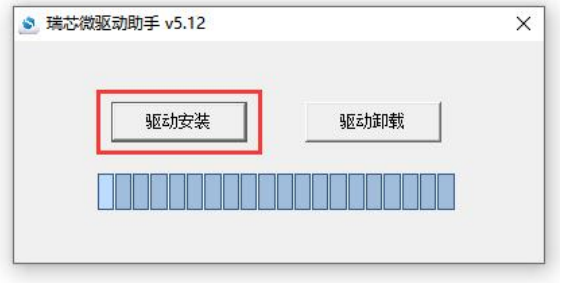
6) Then download the image of Orange Pi OS (Droid).

7) Then use decompression software to extract the **DriverAssitant_v5.12.zip** file, and find the **DriverInstall.exe** executable file in the extracted folder and open it.

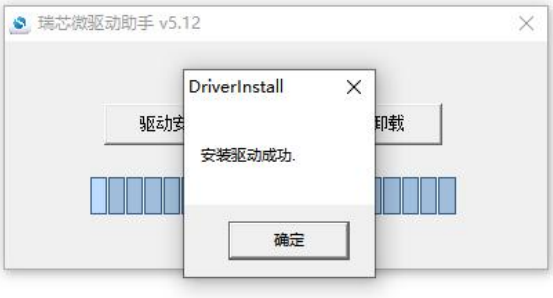


8) The steps to install the Ruixin micro driver after opening **DriverInstall.exe** are as follows:

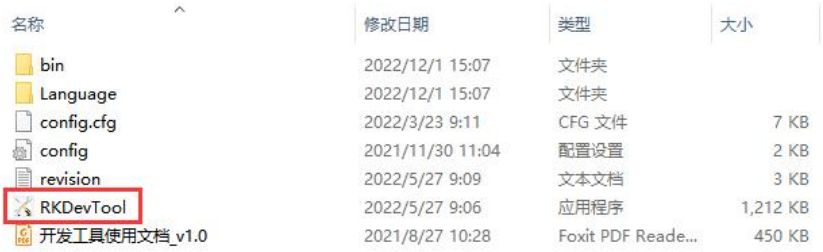
a. Click the "**Driver Installation**" button.



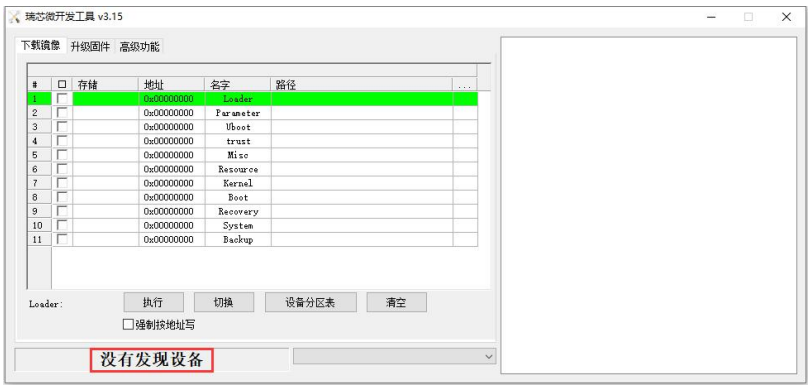
b. After waiting for a period of time, a window will pop up prompting "**Driver installation successful**", and then click the "**OK**" button.



9) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed, just find **RKDevTool** in the decompressed folder and open it.

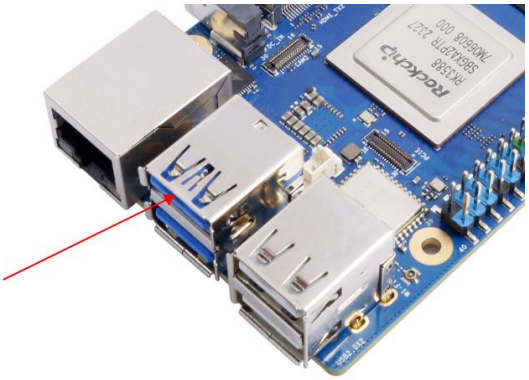


10) After opening the **RKDevTool** burning tool, the bottom left corner will prompt "**Device not found**" because the computer has not yet been connected to the development board through a USB 2.0 male to male data cable.



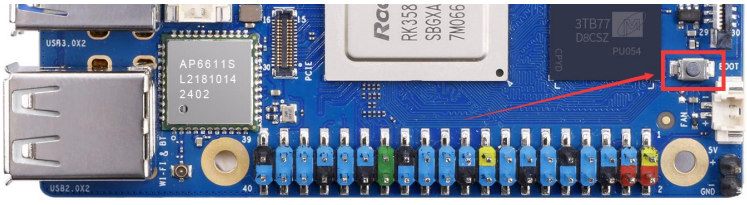
11) Then start burning the Orange Pi OS (Droid) image to SPIFlash+NVMe SSD.

a. Firstly, connect the development board to the Windows computer via a USB 2.0 male to female data cable. The location of the USB burning port on the development board is shown in the following figure:

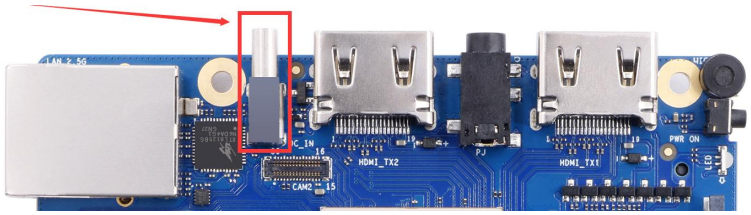


b. Ensure that the development board is not inserted with a TF card or connected to a power source.

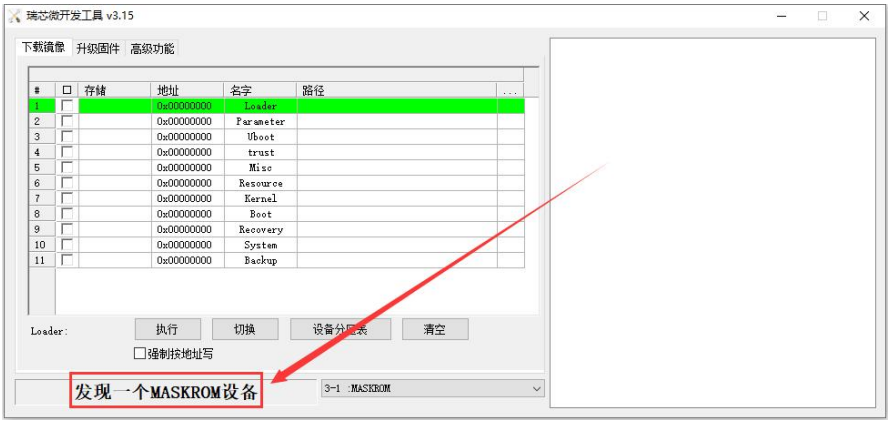
c. Then hold down the MaskROM button on the development board and do not release it. The position of the MaskROM button on the development board is shown in the following figure:



d. Then connect the Type-C interface power to the development board, power it on, and then release the MaskROM button.



e. If the previous steps go smoothly, the development board will enter **MASKROM** mode, and the burning tool interface will prompt "**Found a MASKROM device**".



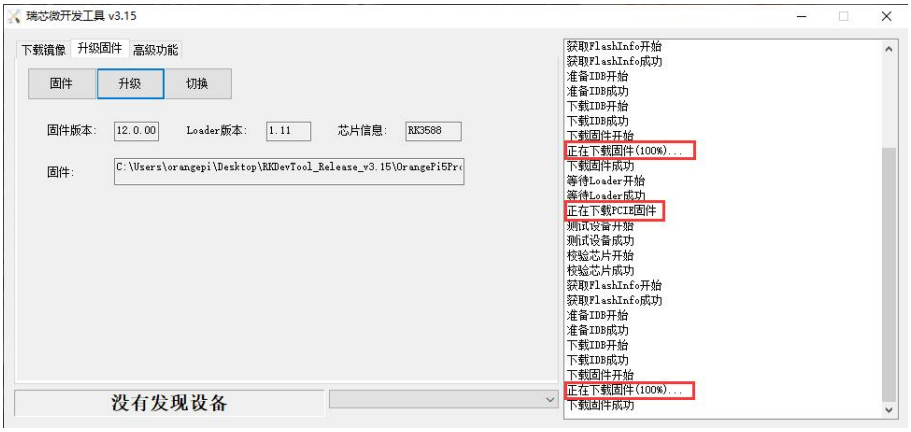
f. Then click on the "**Upgrade Firmware**" section of the burning tool.



g. Then click the "**Firmware**" button to select the Orange Pi OS (Droid) image that needs to be burned.

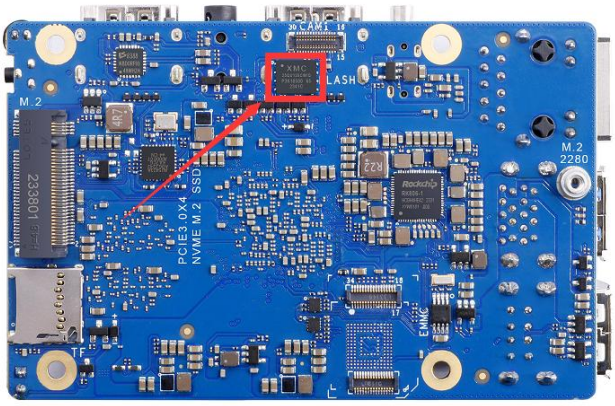


h. Finally, clicking the "**Upgrade**" button will start burning. The burning process is shown in the following figure. You can see that the firmware will first be burned to SPIFlash, and then burned to PCIE. After the burning is completed, the Orange Pi OS (Droid) system will automatically start.



2. 13. How to clear SPIFlash using RKDevTool

1) The location of SPI Flash on the development board is shown in the figure below:

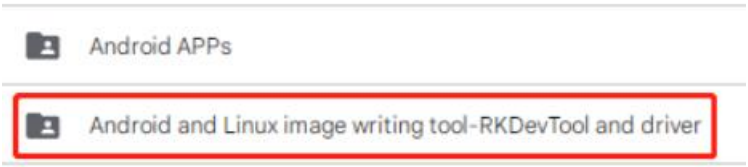


2) First, you need to prepare a good quality USB2.0 male-to-male data cable.

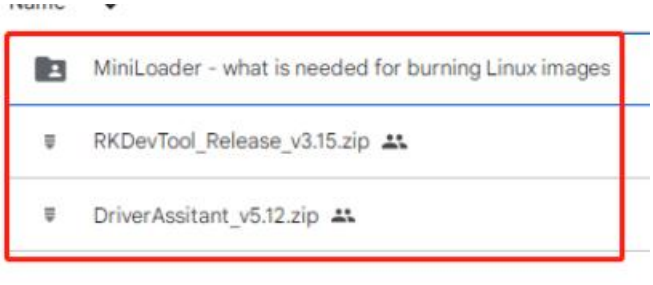


3) Then download Rockchip driver **DriverAssitant_v5.12.zip** and **MiniLoader** and burning tool **RKDevTool_Release_v3.15.zip** from **Orange Pi's download page.**

a. On the Orange Pi download page, first select the official tool, then go to the folder below.



b. Then download all the files below.



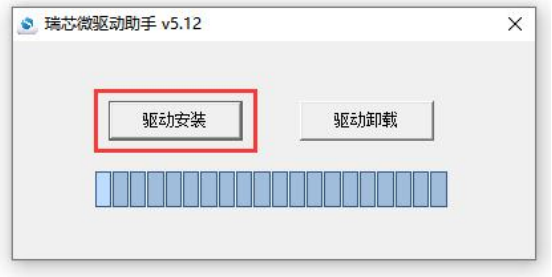
Note that the "MiniLoader-things needed for burning Linux images" folder is referred to as the MiniLoader folder below.

4) Then use the decompression software to decompress **DriverAssitant_v5.12.zip**, then find the **DriverInstall.exe** executable file in the decompressed folder and open it.

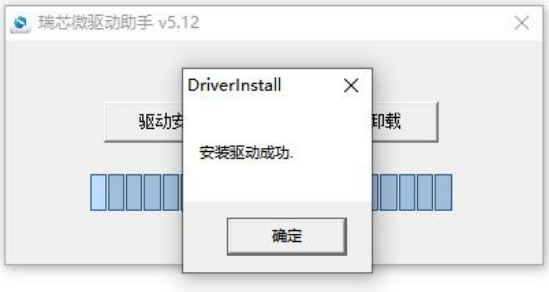


5) After opening **DriverInstall.exe**, the steps to install the Rockchip driver are as follows:

a. Click the "**Driver Installation**" button.



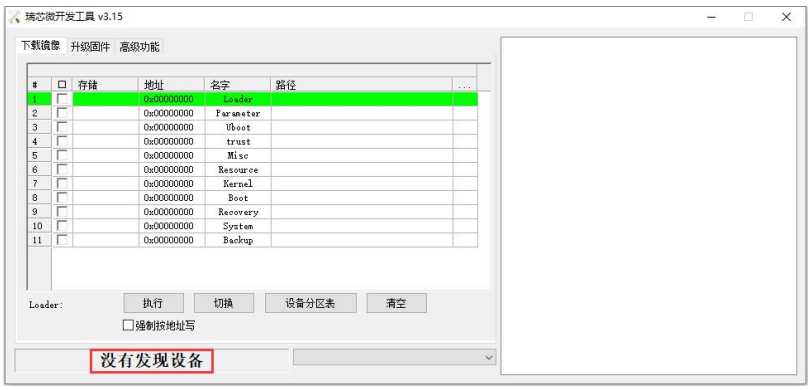
b. After waiting for a while, a window will pop up prompting "**Driver installation successful**", then click the "**OK**" button.



6) Then unzip **RKDevTool_Release_v3.15.zip**. This software does not need to be installed. Just find **RKDevTool** in the unzipped folder and open it.

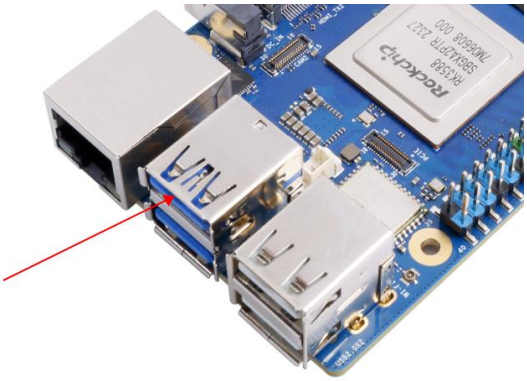


7) After opening the **RKDevTool** burning tool, because the computer has not yet connected to the development board via the USB2.0 male-to-male data cable, the lower left corner will prompt "**No device found**".



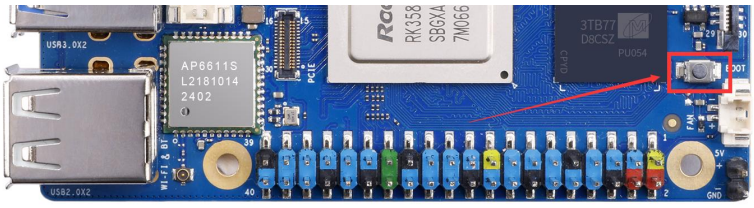
8) Then you can start clearing the contents in SPI FLASH.

a. First, connect the development board to the Windows computer via a USB2.0 male-to-male data cable. The location of the development board's USB interface is shown in the figure below:

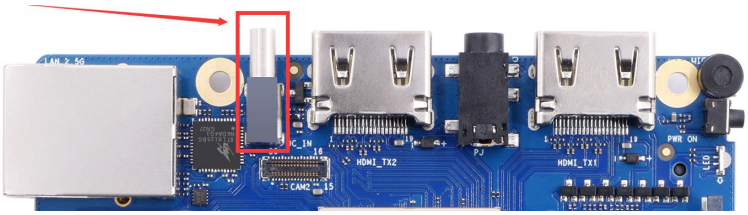


b. Make sure the development board is not plugged into a TF card and is not connected to a power source.

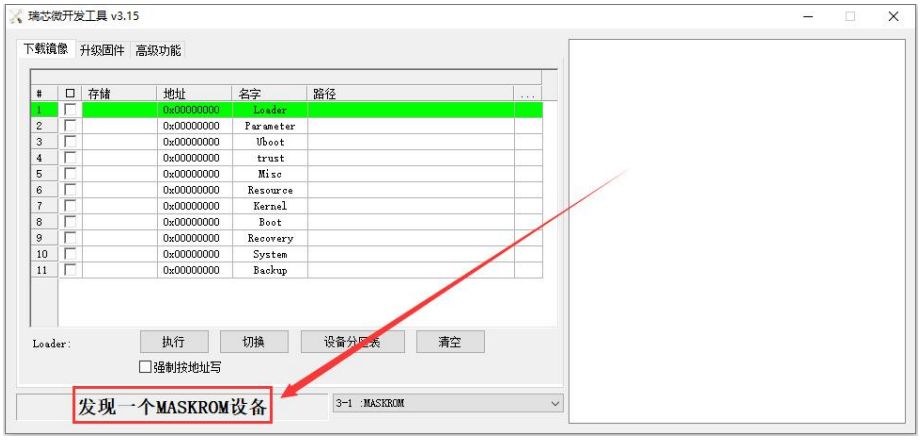
c. Then press and hold the MaskROM button on the development board. The location of the MaskROM button on the development board is shown in the figure below:



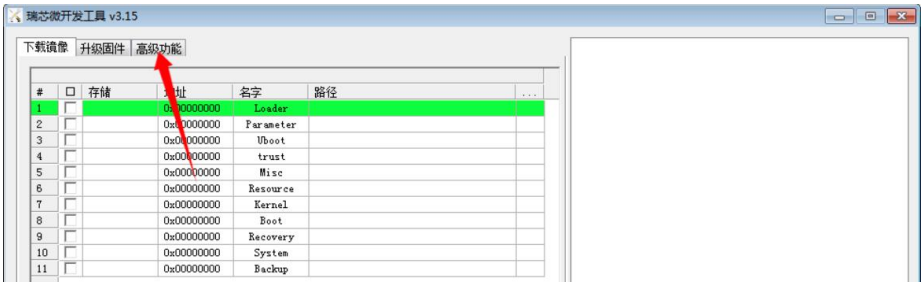
d. Then connect the development board to the power supply of the Type-C interface and power it on. Then you can release the MaskROM button.



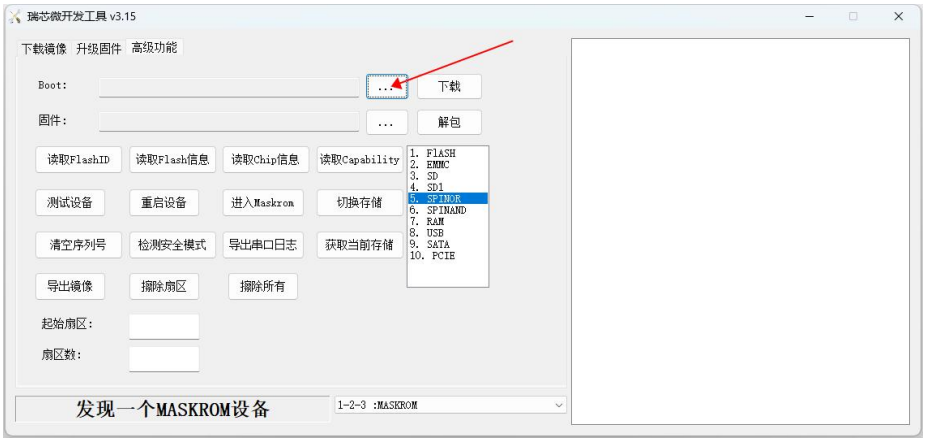
e. If the previous steps are successful, the development board will enter the **MASKROM** mode and the burning tool interface will prompt "**A MASKROM device is found**".



f. Then select **Advanced Features**.

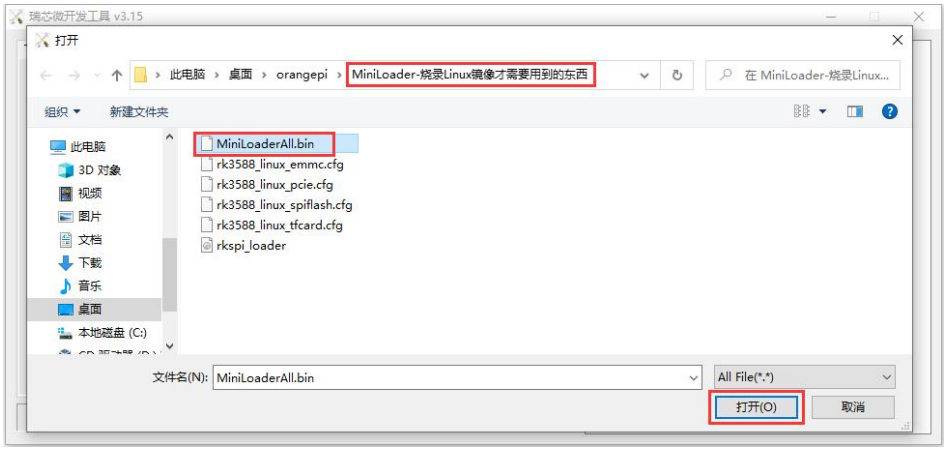


g. Then click the location shown in the picture below.

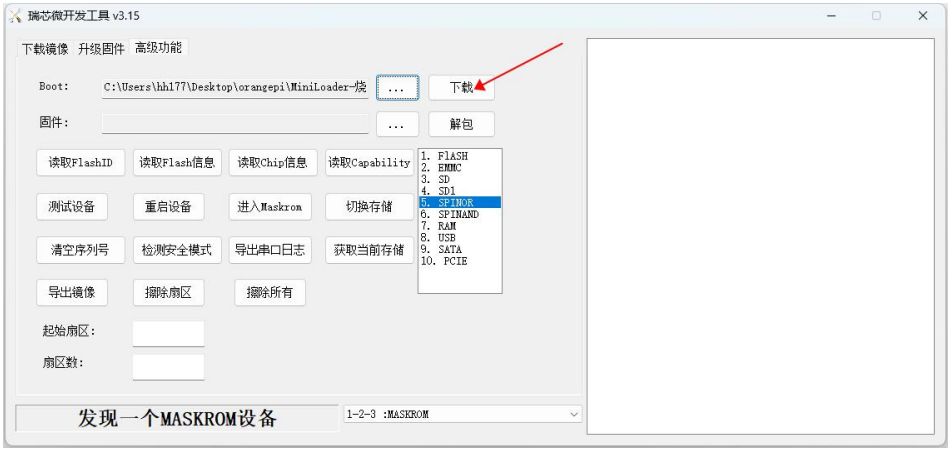


h. Select **MiniLoaderAll.bin** in the **MiniLoader** folder downloaded earlier, and

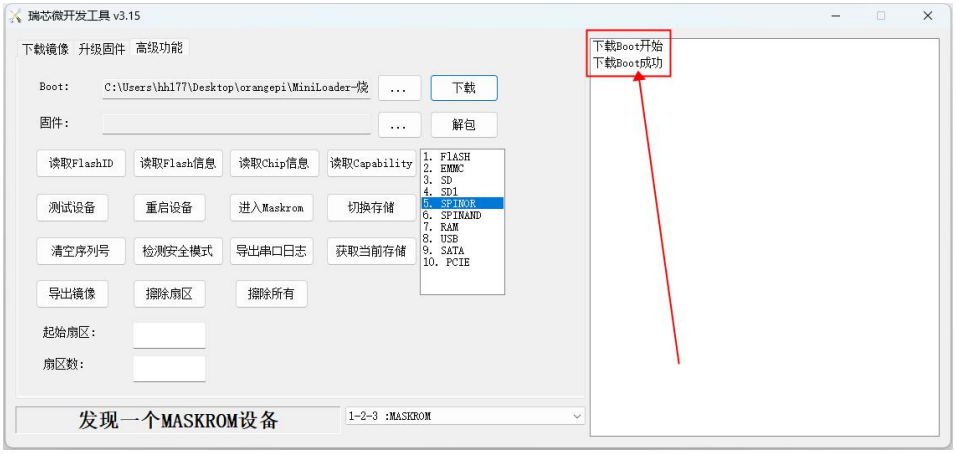
then click Open.



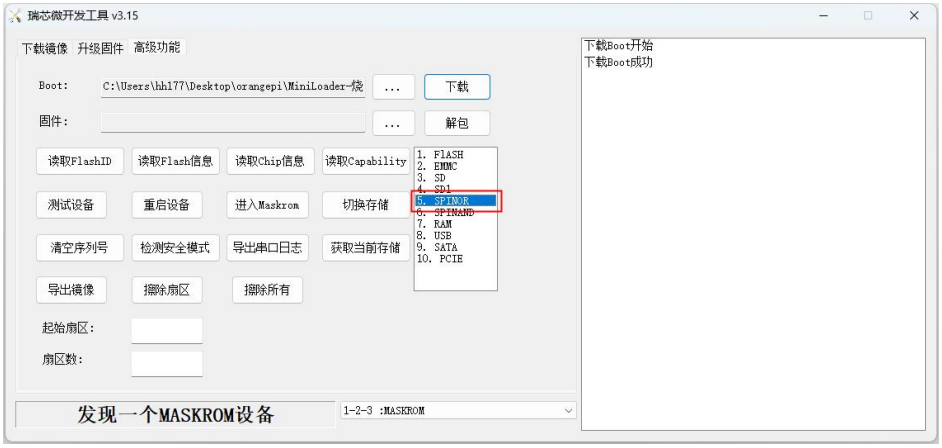
i. Then click Download.



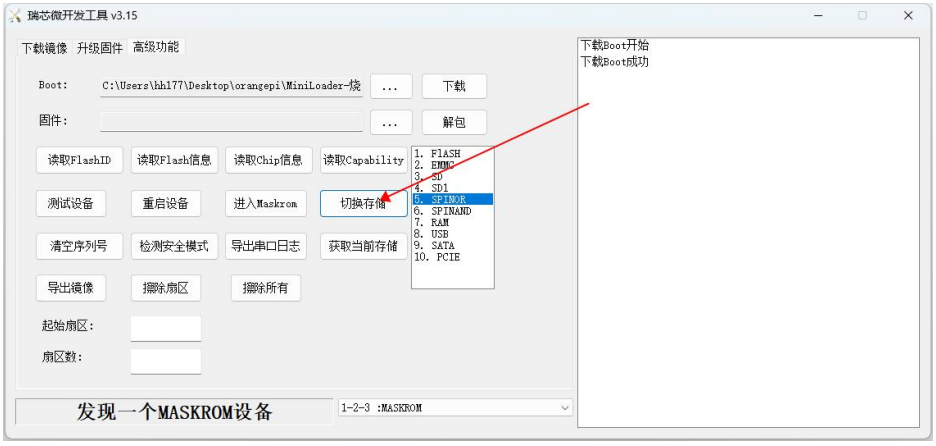
j. After downloading **MiniLoaderAll.bin**, the display is as shown below:



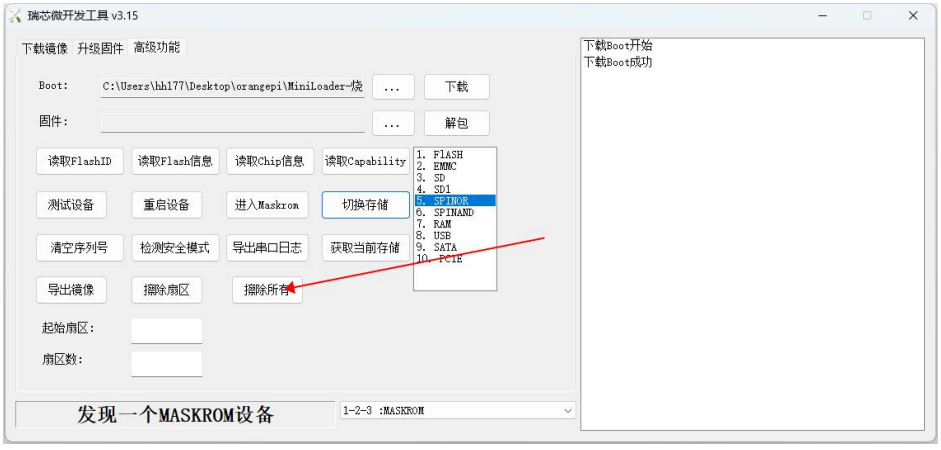
k. Then select the storage device as **SPINOR**.



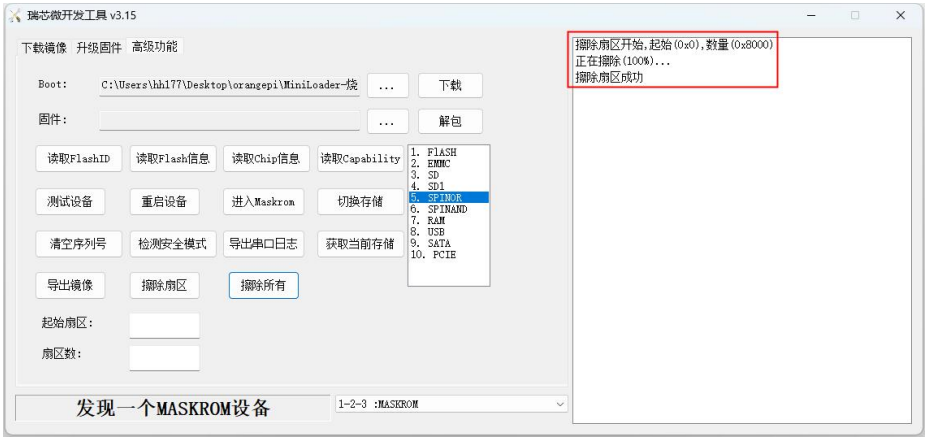
l. Then click **Switch Storage**.



m. Then click Erase All to start erasing SPIFlash.



n. The display log after erasing SPIFlash is as follows:



2. 14. Start the Orange Pi Development Board

1) Insert the TF card with the image burned into the TF card slot of the Orange Pi development board. If the SPIFlash+NVMe SSD or eMMC module has been burned with the image, you do not need to insert the TF card. Just make sure that the NVMe SSD or eMMC module is properly inserted into the development board.

2) The development board has an HDMI interface, which can be connected to a TV or HDMI monitor via an HDMI to HDMI cable. If you have purchased an LCD screen, you can also use it to display the system interface of the development board.

3) Connect a USB mouse and keyboard to control the Orange Pi development board.

4) The development board has an Ethernet port, which can be plugged into a network cable to access the Internet.

5) Connect a high-quality power adapter with a 5V/4A or 5V/5A USB Type-C port.

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

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

The Type-C power interface does not support PD negotiation.

In addition, please do not connect to the USB interface of the computer to power the development board.

6) Then turn on the power adapter. If everything is normal, you can see the system startup screen on the HDMI monitor or LCD screen.

7) If you want to view the system output information through the debug serial port, please use a serial cable to connect the development board to the computer. For the serial port connection method, please refer to the section "**How to use the debug serial port**".

2. 15. How to use the debug serial port

2. 15. 1. Connection Instructions for Debug Serial Port

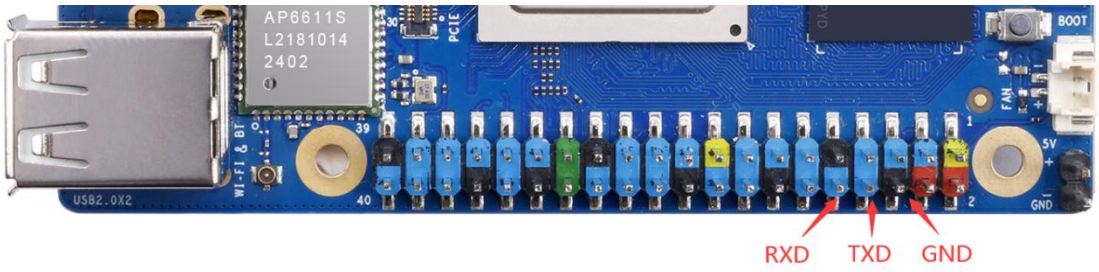
1) First, you need to prepare a 3.3V USB to TTL module, and then insert the USB interface of the USB to TTL module into the USB interface of the computer.

For better compatibility, it is recommended to use the CH340 USB to TTL module. Please do not use the CP2102 or PL2303 type USB to TTL modules.

Before purchasing a USB to TTL module, please confirm that the module supports a baud rate of 1500000.



2) The corresponding relationship between the debugging serial port GND, RXD and TXD pins of the development board is shown in the figure below:



3) The GND, TXD and RXD pins of the USB to TTL module need to be connected to the debug serial port of the development board through DuPont cables.

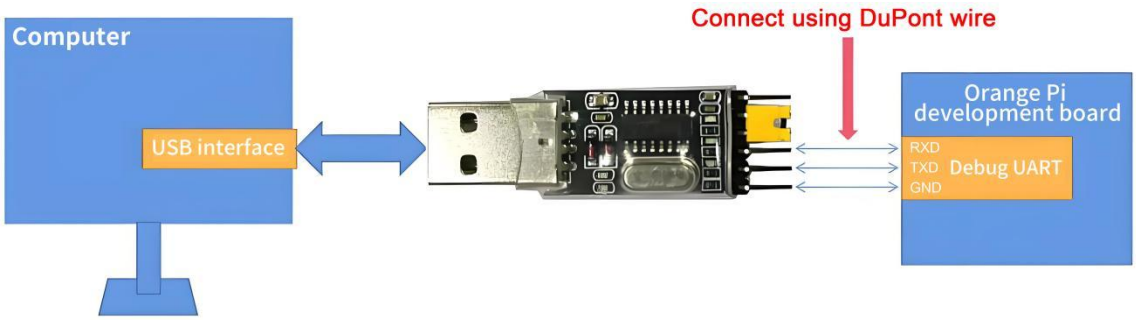
a. **Connect the GND of the USB to TTL module to the GND** of the development board.

b. **Connect the RX of the USB to TTL module to the TX** of the development

board.

c. **Connect the TX of the USB to TTL module 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 as follows:



Schematic diagram of connecting a USB to TTL module to a 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 carefully distinguish the order of TX and RX, you can connect the TX and RX of the serial port randomly. If there is no output in the test, then swap the order of TX and RX. In this way, there will always be one order that is correct.

2. 15. 2. How to use the debug serial port on Ubuntu platform

There are many serial port 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 port of the Ubuntu computer. If the USB to TTL module is connected and recognized normally, you can see the corresponding device node name under **/dev** of the Ubuntu PC. Remember this node name, which will be used when setting up the serial port software later.

```
test@test:~$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

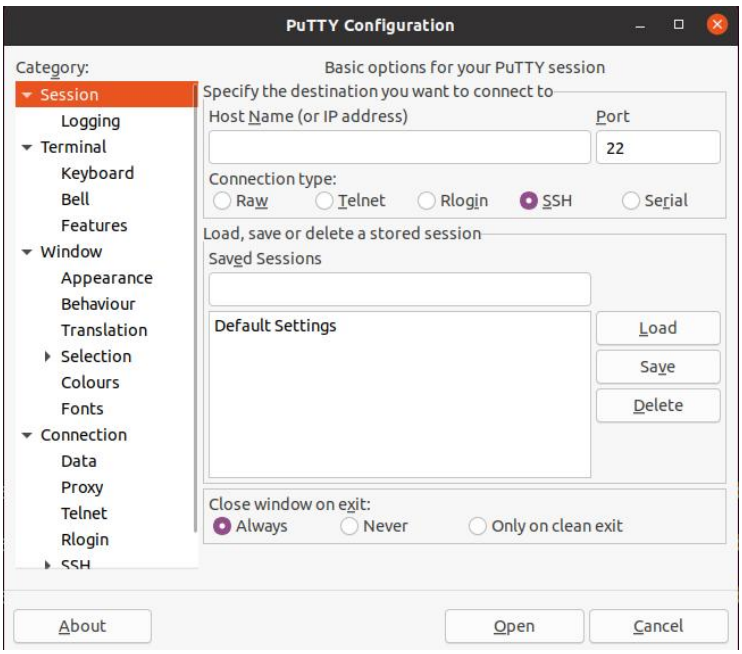
2) Then install putty on your Ubuntu PC using the command below.

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

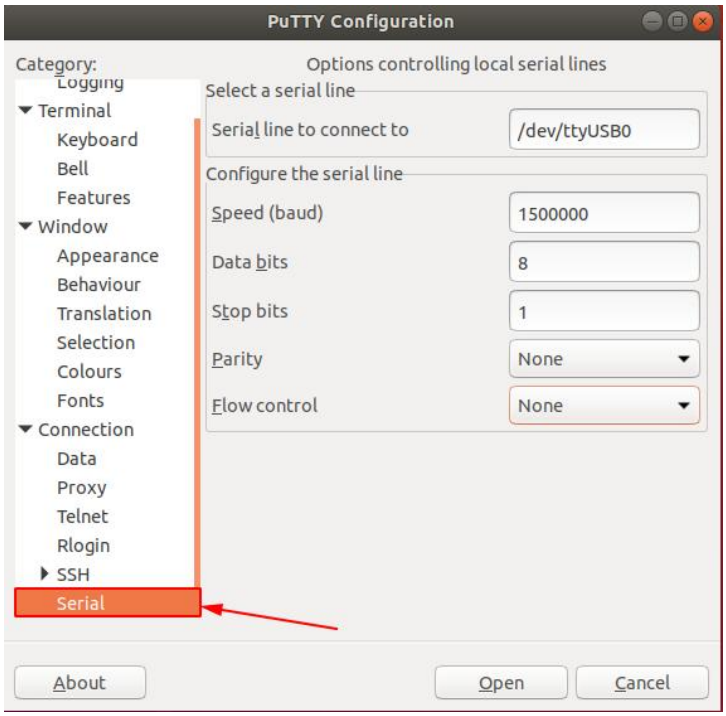
3) Then run putty and **remember to add sudo permissions.**

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

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



5) First select the serial port settings interface.

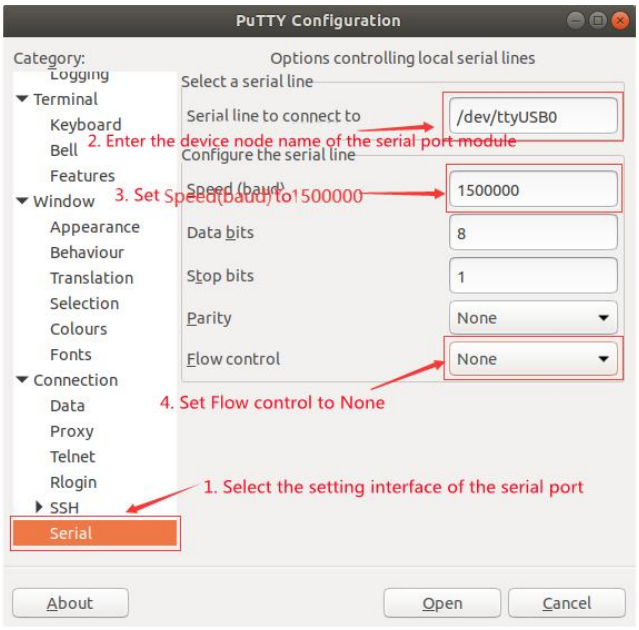


6) Then set the parameters of the serial port.

a. Set **Serial line to connect to** to **/dev/ttyUSB0** (change to the corresponding node name, usually **/dev/ttyUSB0**).

b. Set **Speed(baud)** to 1500000 (the baud rate of the serial port).

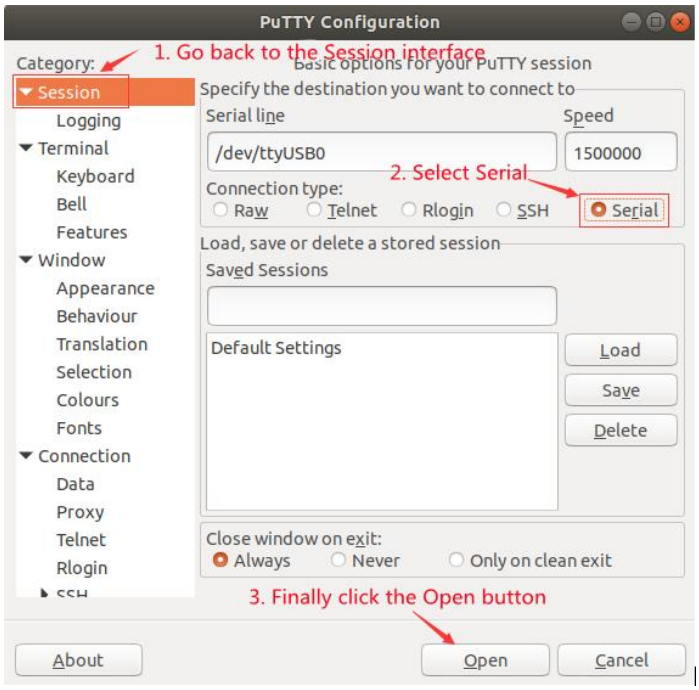
c. Set **Flow control** to **None**.



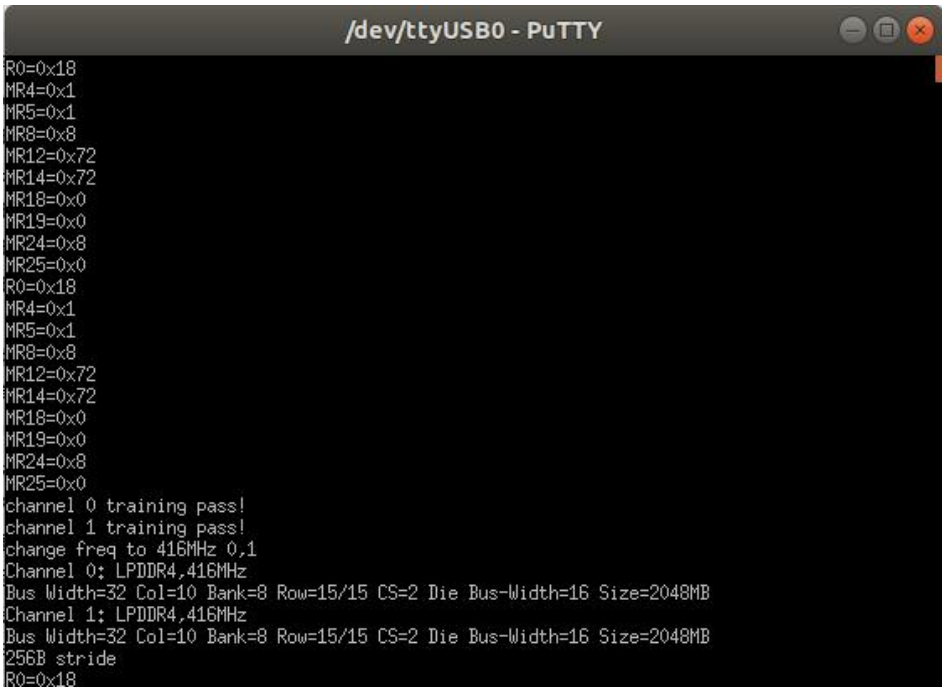
7) After completing the settings on the serial port settings interface, return to the Session interface.

a. First select **Connection type** as Serial.

b. 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 opened serial port terminal.



2. 15. 3. How to use the debug serial port on Windows platform

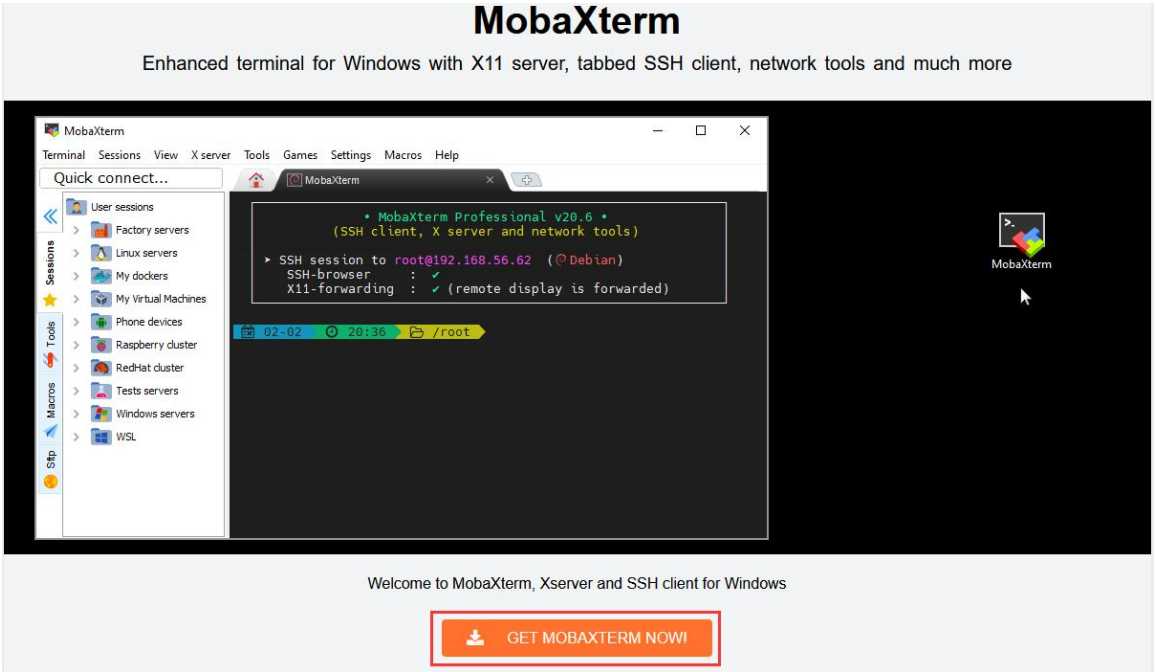
There are many serial port debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following demonstrates how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

1) Download MobaXterm.

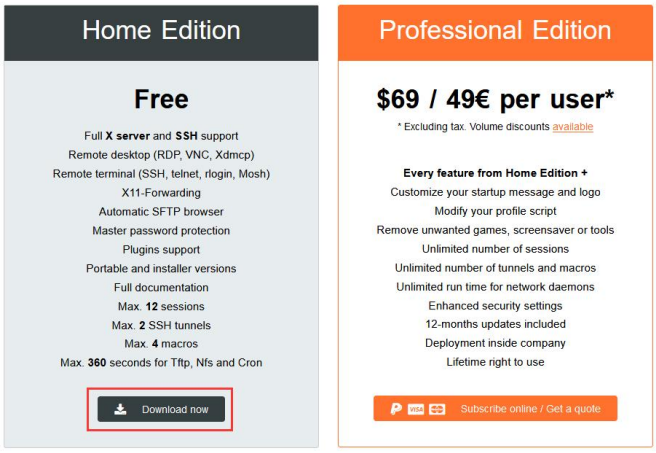
a. Download MobaXterm from the following URL:

https://mobaxterm.mobatek.net

b. to the MobaXterm download page and click **GET XOBATERM NOW!**.

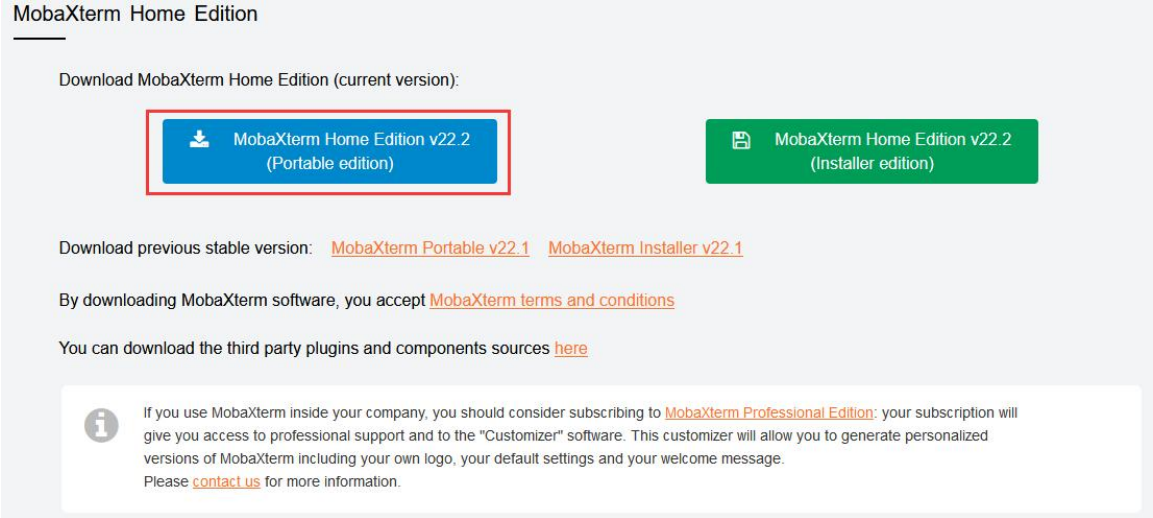


c. Then choose to download the Home version.



d. Then select the Portable version. After downloading, you don’t need to install it.

You can use it directly by opening it.

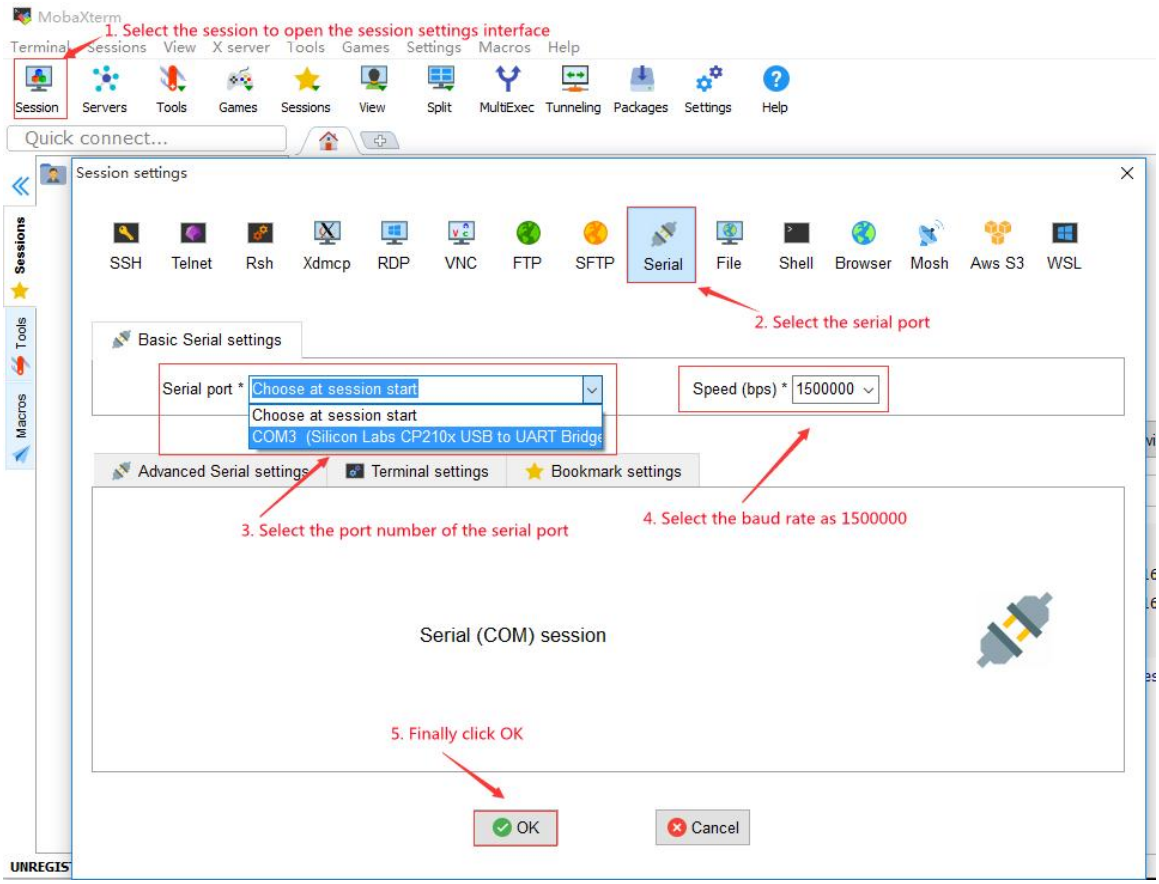


2) After downloading, use decompression software to decompress the downloaded compressed package to get the executable software of MobaXterm, and then double-click to open it.

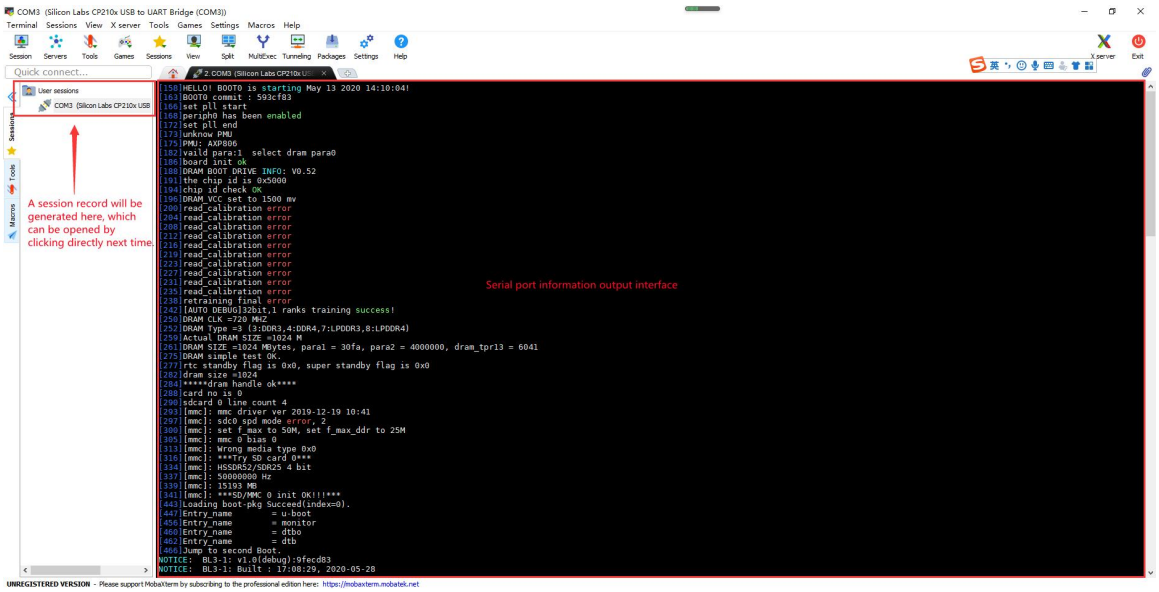


3) After opening the software, the steps to set up the serial port connection are as follows:

a. Open the session settings interface.
b. Select the serial port type.
c. Select the serial port number (select the corresponding port number according to the actual situation). If you cannot see the port number, please use 360 Driver Master to scan and install the USB to TTL serial port chip driver.
d. Select the serial port baud rate as **1500000**.
e. Finally, click the "**OK**" button to complete the settings.



4) Click the "OK" button to enter the following interface. Now start the development board and you can see the output information of the serial port.



2. 16. Power supply instructions using the 5V pin in the 40pin interface of the development board

The power supply method we recommend for the development board is to use a 5V/5A Type C interface power cord plugged into the Type-C power interface of the development board for power supply. If you need to use the 5V pin in the 40pin interface to power the development board, please make sure that the power cord and power adapter used can meet the power supply requirements of the development board. If there is any unstable use, please switch back to 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 purchased on Amazon or Aliexpress. Please search and purchase it by yourself.

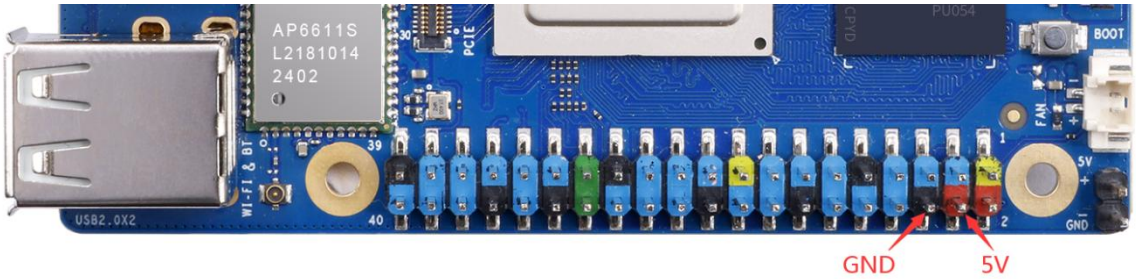
2) Use the 5V pin in the 40-pin interface to power the development board. The power line connection is as follows:

a. The USB A port of the power cable shown in the figure above needs to be plugged into the 5V/5A power adapter connector **(please do not plug it into the USB port of the computer for power supply**).

b. The red DuPont cable needs to be plugged into the 5V pin of the 40pin of the development board.

c. The black DuPont cable needs to be plugged into the GND pin of the 40pin interface.

d. The positions of the 5V pin and GND pin of the 40pin interface in the development board are shown in the figure below. Remember not to connect them in reverse.



3. Instructions for using Ubuntu/Debian Server and Xfce desktop system

This chapter is based on the Linux server version image and the Xfce desktop version image.

If you are using the OPi OS Arch image, please refer to the Orange Pi OS Arch system usage instructions chapter.

3. 1. Supported Linux image types and kernel versions

Linux Image Type	Kernel version	Server Edition	Desktop version
Debian 11 - Bullseye	Linux5.10	Support	Support
Debian 12 - Bookworm	Linux5.10	Support	Support
Ubuntu 20.04 - Focal	Linux5.10	Support	Support
Ubuntu 22.04 - Jammy	Linux5.10	Support	Support
Debian 12 - Bookworm	Linux6.1	Support	Support
Ubuntu 22.04 - Jammy	Linux6.1	Support	Support

3. 2. 3.2. Linux 5.10 system compatibility

Function	Debian11	Debian12	Ubuntu20.04	Ubuntu22.04
HDMI TX Video	OK	OK	OK	OK
HDMI TX Audio	OK	OK	OK	OK
HDMI RX Video	OK	OK	OK	OK
HDMI RX Audio	OK	OK	OK	OK
USB2.0x2	OK	OK	OK	OK
USB3.0x2	OK	OK	OK	OK
2.5G network port	OK	OK	OK	OK
Network port light	OK	OK	OK	OK
WIFI	OK	OK	OK	OK
Bluetooth	OK	OK	OK	OK
Debug serial port	OK	OK	OK	OK
RTC chip	OK	OK	OK	OK
FAN Fan Connector	OK	OK	OK	OK
eMMC interface	OK	OK	OK	OK
GPIO（40pin）	OK	OK	OK	OK
UART（40pin）	OK	OK	OK	OK
SPI（40pin）	OK	OK	OK	OK
I2C（40pin）	OK	OK	OK	OK
CAN（40pin）	OK	OK	OK	OK
PWM（40pin）	OK	OK	OK	OK
OV13850 Camera	OK	OK	OK	OK
OV13855 Camera	OK	OK	OK	OK
SPI+NVME Start	OK	OK	OK	OK
LCD	OK	OK	OK	OK
MIC	OK	OK	OK	OK
Headphone playback	OK	OK	OK	OK
Headphone Recording	OK	OK	OK	OK
Three-color LED light	OK	OK	OK	OK
GPU	OK	OK	OK	OK

NPU	OK	OK	OK	OK
VPU	OK	OK	OK	OK
Power button	OK	OK	OK	OK
Watchdog test	OK	OK	OK	OK
Chromium hard decoding video	OK	OK	OK	OK

3. 3. Linux 6.1 system compatibility

Function	Debian12	Ubuntu22.04
HDMI TX Video	OK	OK
HDMI TX Audio	OK	OK
HDMI RX Video	OK	OK
HDMI RX Audio	OK	OK
USB2.0 x 2	OK	OK
USB3.0 x 2	OK	OK
Gigabit Ethernet	OK	OK
Network port status light	OK	OK
WIFI	OK	OK
Bluetooth	OK	OK
Debug serial port	OK	OK
RTC chip	OK	OK
FAN Fan Connector	OK	OK
eMMC interface	OK	OK
GPIO（40pin）	OK	OK
UART（40pin）	OK	OK
SPI（40pin）	OK	OK
I2C（40pin）	OK	OK
CAN（40pin）	NO	NO
PWM（40pin）	OK	OK
OV13850 Camera	OK	OK

OV13855 Camera	**OK**	**OK**
SPI+NVME Start	**OK**	**OK**
LCD	**OK**	**OK**
MIC	**OK**	**OK**
Headphone playback	**OK**	**OK**
Headphone Recording	**OK**	**OK**
Three-color LED light	**OK**	**OK**
GPU	**OK**	**OK**
NPU	**OK**	**OK**
VPU	**OK**	**OK**
Power button	**OK**	**OK**
Watchdog test	**OK**	**OK**
Chromium hard decoding video	**OK**	**OK**

3. 4. Linux command format description in this manual

1) All commands in this manual that need to be entered in the Linux system will be framed with the following boxes.

```
```

As shown below, the contents in the yellow box indicate the contents that require special attention, except for the commands inside.

```
```

2) Description of the prompt type before the command.

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

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

```
root@test:~# ls
```

b. **root@orangepi:~$** The prompt indicates that this command is entered in the Linux system of the development board. The **$** at the end of the prompt indicates that the current user of the system is a common user. When executing privileged commands, **sudo** is required.

c. **root@orangepi:~#** 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 he wants.

d. **test@test:~$** The prompt indicates that this command is entered in an Ubuntu PC or Ubuntu virtual machine, not in the Linux system of the development board. The **$** at the end of the prompt indicates that the current user of the system is a normal user. When executing privileged commands, **sudo** needs to be added.

e. **root@test:~#** The prompt indicates that this command is entered in an Ubuntu PC or Ubuntu virtual machine, not 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.

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

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

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

b. As shown below, some commands cannot fit in one line and will be placed on the next line. The bold black parts are the commands that need to be entered. When these commands are entered on one line, the "\" at the end of each line needs to be removed, as it is not part of the command. In addition, there are spaces between different parts of the command, so please do not 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. 5. Linux system login instructions

3. 5. 1. 3.5.1.Linux system default login account and password

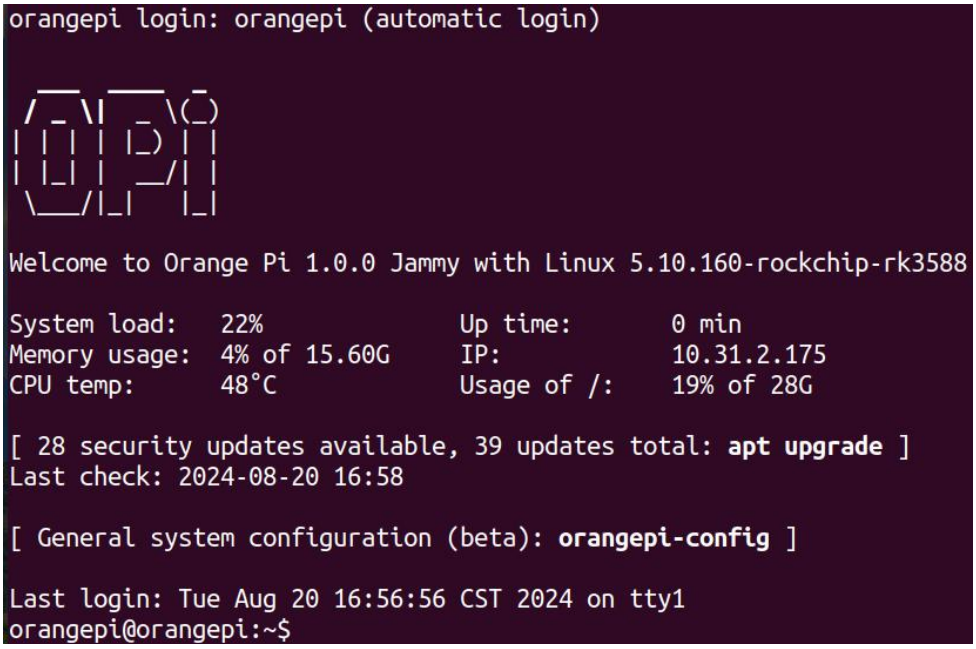
Account	Password
root	orangepi
orangepi	orangepi

Please note that when you enter the password, the specific content of the password will not be displayed on the screen. Please do not think that there is any malfunction. Just press Enter after entering it.

If you get an error message when entering the password, or there is a problem with the ssh connection, please note that as long as you are using the Linux image provided by Orange Pi, do not doubt that the password above is incorrect, but look for other reasons.

3. 5. 2. 3.5.2. How to set up automatic login for Linux system terminal

1) The Linux system automatically logs in to the terminal by default, and the default login username is **orangepi**.



2) Use the following command to set the root user to automatically log in to the terminal.

```
orangepi@orangepi:~$ sudo auto_login_cli.sh root
```

3) Use the following command to disable automatic login to the terminal.

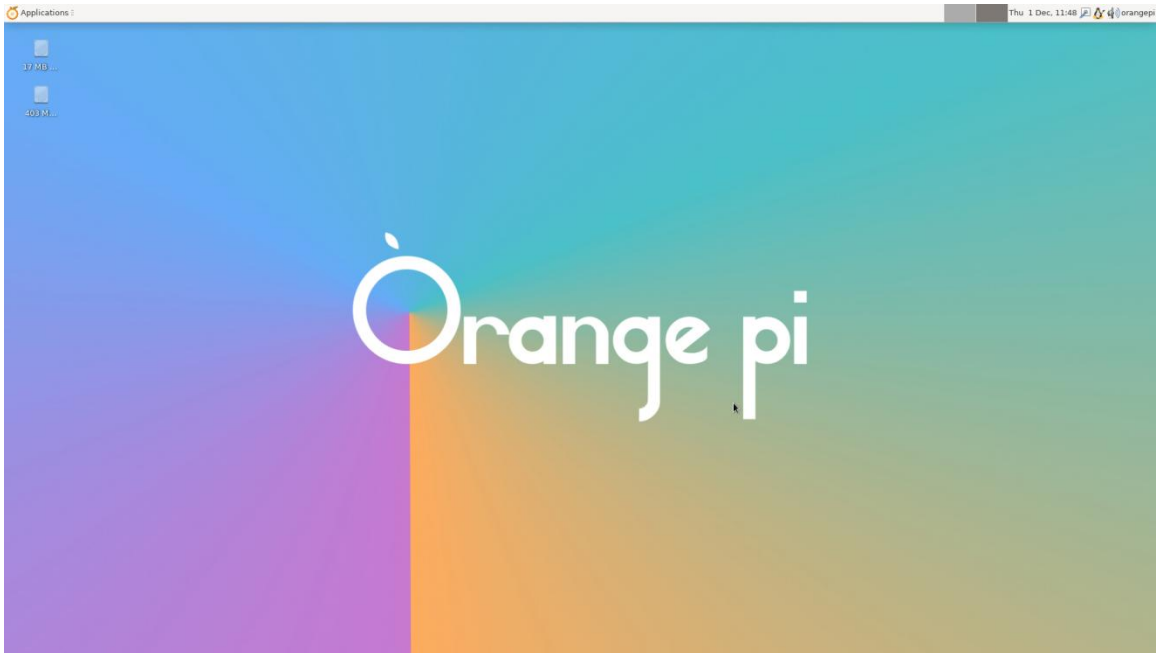
```
orangepi@orangepi:~$ sudo auto_login_cli.sh -d
```

4) Use the following command to set the orangepi user to automatically log in to the terminal again.

```
orangepi@orangepi:~$ sudo auto_login_cli.sh orangepi
```

3. 5. 3. Linux desktop system automatic login instructions

1) After the desktop version system is started, it will automatically log in to the desktop without entering a password.



2) Run the following command to prevent the desktop version of the system from automatically logging into the desktop.

```
orangepi@orangepi:~$ sudo disable_desktop_autologin.sh
```

3) Then restart the system and a login dialog box will appear. You need to enter the **password** to enter the system.



3. 5. 4. How to set up automatic login for root user in Linux desktop system

1) Execute the following command to set the desktop system to automatically log in as root user

```
orangepi@orangepi:~$ sudo desktop_login.sh root
```

2) Then restart the system and the root user will automatically log in to the desktop.



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

Also, please note that this is not a bug, because pulseaudio is not allowed to run under the root user.

3) Run the following command to set the desktop system to automatically log in using the orangepi user again.

```
orangepi@orangepi:~$ sudo desktop_login.sh orangepi
```

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

1) First enter the following command in the command line. **Please remember to add sudo permissions**.

```
orangepi@orangepi:~$ sudo systemctl disable lightdm.service
```

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

```
orangepi@orangepi:~$ sudo reboot
```

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

a. First enter the following command in the command line. **Please remember to add sudo permissions.**

```
orangepi@orangepi:~$ sudo systemctl start lightdm.service
orangepi@orangepi:~$ sudo systemctl enable lightdm.service
```

b. After selecting, the monitor will display the desktop

3. 6. Onboard LED light test instructions

1) There is a red, green and blue light on the development board, and its location is shown in the figure below:



2) As long as the development board is powered on, the red LED light will be always on. This is controlled by hardware and cannot be turned off by software. The red LED light can be used to determine whether the power of the development board has been turned on normally.

3) The green and blue LED lights will keep flashing after the kernel starts, which is controlled by software.

4) The method of setting the green light on and off and flashing is as follows:

Note: The following operations must be performed as the root user.

a. First enter the Green Light settings directory.

```
root@orangepi:~# cd /sys/class/leds/green_led
```

b. The command to set the green light to stop flashing is as follows:

```
root@orangepi:/sys/class/leds/green_led# echo none > trigger
```

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

```
root@orangepi:/sys/class/leds/green_led# echo default-on > trigger
```

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

```
root@orangepi:/sys/class/leds/green_led# echo heartbeat > trigger
```

5) The method of using commands to set the blue light on and off and flashing is as follows:

Note: The following operations must be performed as the root user.

a. First enter the settings directory of Lantern.

```
root@orangepi:~# cd /sys/class/leds/blue_led
```

b. The command to set the blue light to stop flashing is as follows:

```
root@orangepi:/sys/class/leds/blue_led# echo none > trigger
```

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

```
root@orangepi:/sys/class/leds/blue_led# echo default-on > trigger
```

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

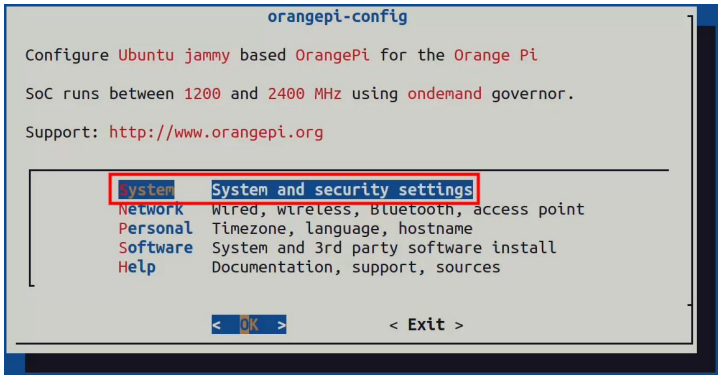
```
root@orangepi:/sys/class/leds/blue_led# echo heartbeat > trigger
```

6) If you do not need the LED light to flash after powering on, you can use the following method to turn off the green and blue lights.

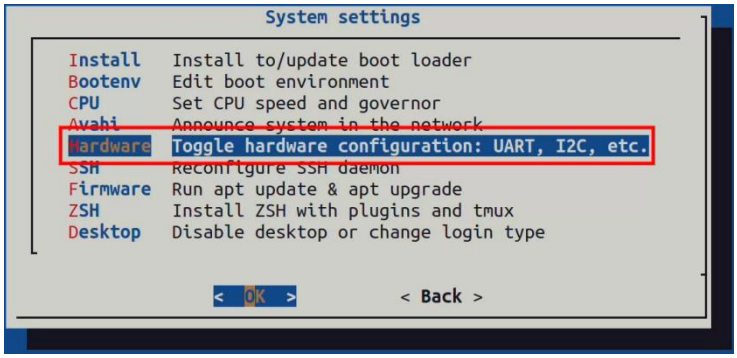
a. First run **orangepi-config**. Remember to add **sudo** permissions as a normal user.

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

b. Then select **System**.



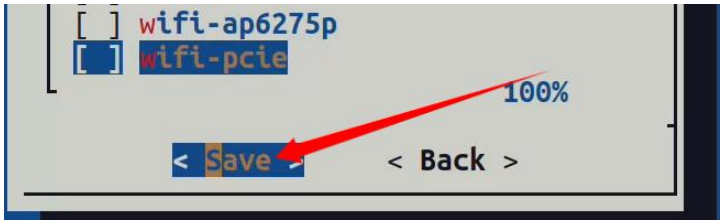
c. Then select **Hardware**.



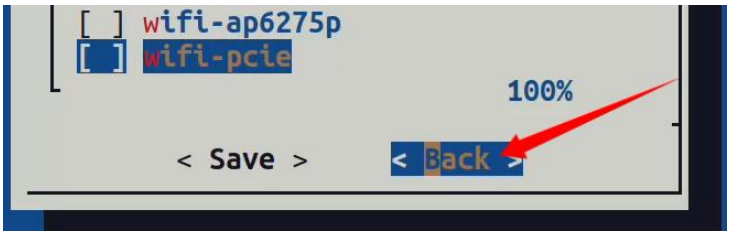
d. Then use the arrow keys on your keyboard to locate the position shown in the figure below, and then use the space bar to select the **opi5ultra-disable-leds** configuration.



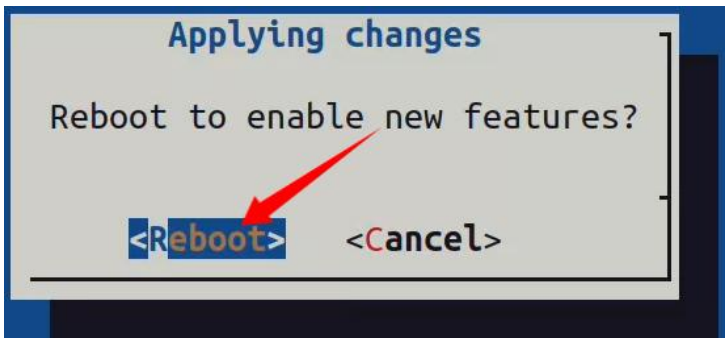
e. Then select **<Save>** to save.



f. Then select **<Back>**.



g. Then select **<Reboot>** to restart the system to make the configuration take effect.



h. After restarting, you can see that only the red light on the development board is always on, and the green and blue lights will not flash.

3. 7. Network connection test

3. 7. 1. Ethernet port test

1) First, plug one end of the network cable into the Ethernet port of the development board, and the other end of the network cable into the router, and make sure the network is unobstructed.

2) After the system starts, the IP address will be automatically assigned to the Ethernet card through **DHCP**, and no other configuration is required.

3) The command to check the IP address in the Linux system of the development board is as follows:

```
orangepi@orangepi:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enP3p49s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:e0:4c:68:00:0f brd ff:ff:ff:ff:ff:ff
    inet 10.31.2.249/16 brd 10.31.255.255 scope global dynamic noprefixroute enP3p49s0
       valid_lft 42670sec preferred_lft 42670sec
    inet6 fe80::d5aa:9a6:cd41:942e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

```
3: wlan0: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state DORMANT group default qlen 1000
    link/ether 50:41:1c:f1:0f:7e brd ff:ff:ff:ff:ff:ff
```

When using ifconfig to check the IP address, if the following message is displayed, it is because sudo is not added. The correct command is: **sudo ifconfig**.

orangepi@orangepi:~$ **ifconfig**
Command 'ifconfig' is available in the following places
 * /sbin/ifconfig
 * /usr/sbin/ifconfig
The command could not be located because '/sbin:/usr/sbin' is not included in the PATH environment variable.
This is most likely caused by the lack of administrative privileges associated with your user account.
ifconfig: command not found

There are three ways to check the IP address after the development board is started:
1. Connect an HDMI display, then log in to the system and use the ip addr show command to view the IP address.
2. Enter the ip addr show command in the debug serial port terminal to view the IP address.
3. If there is no debug serial port and no HDMI display, you can also view the IP address of the development board network port through the router's management interface. However, this method often causes some people to be unable to see the IP address of the development board normally. If you can't see it, the debugging method is as follows:
A) First check whether the Linux system has started normally. If the three-color light on the development board is flashing, it is generally started normally. If only the red light is on, it means that 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 (there are many problems with routers, such as the router cannot allocate IP addresses normally, or the IP address has been allocated normally

but cannot be seen in the router);
D) If there is no router to replace, you can only connect an HDMI display or use the debug serial port to view the IP address.

It should also be noted that the development board DHCP automatically assigns IP addresses without any settings.

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

```
orangepi@orangepi:~$ ping www.baidu.com -I enP3p49s0
PING www.a.shifen.com (183.2.172.185) from 10.31.2.249 enP3p49s0: 56(84) bytes of data.
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=1 ttl=53 time=39.5 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=2 ttl=53 time=33.1 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=3 ttl=53 time=32.4 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
```

3.7.2. WIFI connection test

Please do not connect to WIFI by modifying the /etc/network/interfaces configuration file. This method may cause problems when connecting to the WIFI network.

3.7.2.1. Server version image connects to WIFI through command

When the development board is not connected to Ethernet, not connected to HDMI display, and only connected to the serial port, it is recommended to use the command 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 Ethernet or HDMI display, you can also use the command demonstrated in this section to connect to the WIFI network.

1) Log in to the Linux system first. There are 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 debug serial port, you can use the serial terminal to log in to the Linux system.
   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) First use the **nmcli dev wifi** command to scan the surrounding WIFI hotspots.

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

```
root@orangepi:~# nmcli dev wifi
IN-USE  BSSID              SSID                MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
        28:6C:07:6E:87:2E  [redacted]_orangepi  Infra  9     260 Mbit/s  97            WPA1 WPA2
        D8:D8:66:A5:BD:D1  [redacted]           Infra  10    270 Mbit/s  90            WPA1 WPA2
        A0:40:A0:A1:72:20  [redacted]           Infra  4     405 Mbit/s  82            WPA2
        28:6C:07:6E:87:2F  [redacted]_orangepi_5G Infra  149   540 Mbit/s  80            WPA1 WPA2
        CA:50:E9:89:E2:44  [redacted]           Infra  1     130 Mbit/s  79            WPA1 WPA2
        A0:40:A0:A1:72:31  [redacted]           Infra  100   405 Mbit/s  67            WPA2
        D4:EE:07:08:A9:E0  [redacted]           Infra  4     130 Mbit/s  55            WPA1 WPA2
        88:C3:97:49:25:13  [redacted]           Infra  6     130 Mbit/s  52            WPA1 WPA2
        00:BD:82:51:53:C2  [redacted]           Infra  12    130 Mbit/s  49            WPA1 WPA2
        C0:61:18:FA:49:37  [redacted]           Infra  149   270 Mbit/s  47            WPA1 WPA2
        04:79:70:8D:0C:B8  [redacted]           Infra  153   270 Mbit/s  47            WPA2
        04:79:70:FD:0C:B8  [redacted]           Infra  153   270 Mbit/s  47            WPA2
        9C:A6:15:DD:E6:0C  [redacted]           Infra  10    270 Mbit/s  45            WPA1 WPA2
        B4:0F:3B:45:D1:F5  [redacted]           Infra  48    270 Mbit/s  45            WPA1 WPA2
        E8:CC:18:4F:7B:44  [redacted]           Infra  157   135 Mbit/s  45            WPA1 WPA2
        B0:95:8E:D8:2F:ED  [redacted]           Infra  11    405 Mbit/s  39            WPA1 WPA2
        C0:61:18:FA:49:36  [redacted]           Infra  11    270 Mbit/s  24            WPA1 WPA2
root@orangepi:~#
```

3) Then use the **nmcli** command to connect to the scanned WIFI hotspot, where:
   a. **wifi_name** needs to be replaced with the name of the WIFI hotspot you want to connect to.
   b. **wifi_passwd** needs to be replaced with the password of the WIFI hotspot you want to connect to.

```
orangepi@orangepi:~$ sudo nmcli dev wifi connect wifi_name password wifi_passwd
Device 'wlan0' successfully activated with 'cf937f88-ca1e-4411-bb50-61f402eef293'.
```

4) Use the**ip addr show wlan0** command to view the IP address of the wifi.

```
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 pressing 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
```

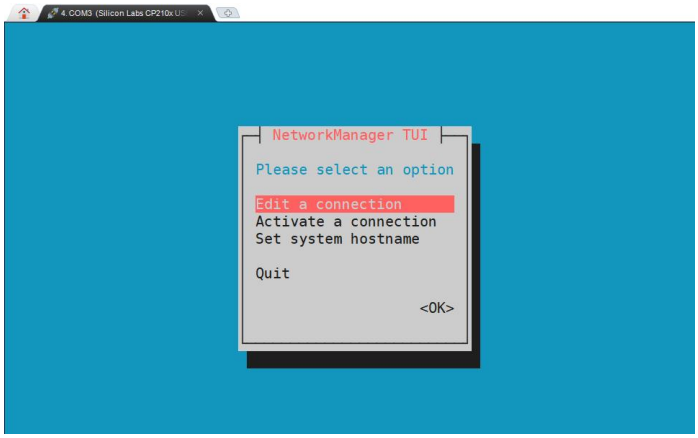
3.7.2.2. The server version image connects to WIFI through a graphical method

1) Log in to the Linux system first. There are 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 debug serial port, you can use the serial terminal to log in to the Linux system (use MobaXterm as the serial software, and minicom cannot display the graphical interface).
   c. If the development board is connected to an HDMI display, you can log in to the Linux system through the HDMI display terminal.

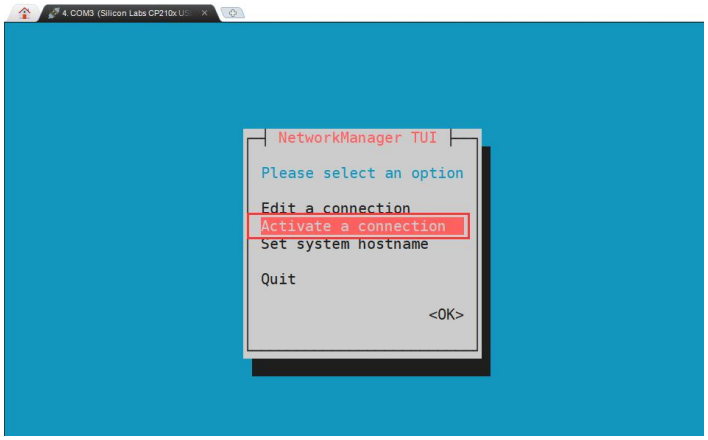
2) Then enter the nmtui command in the command line to open the wifi connection interface.

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

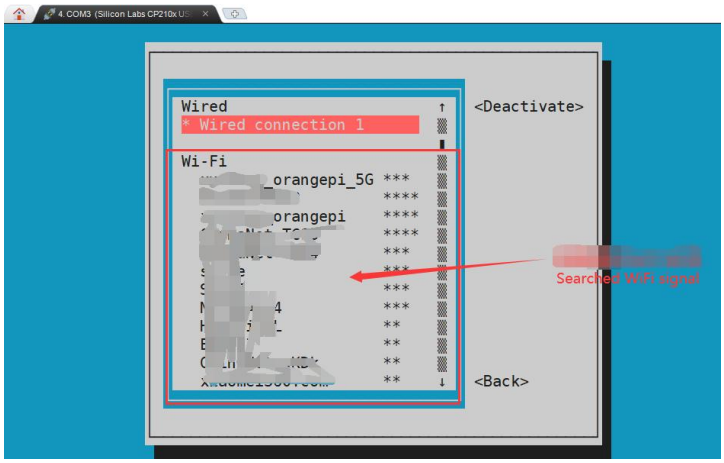
3) Enter the nmtui command to open the interface as shown below:



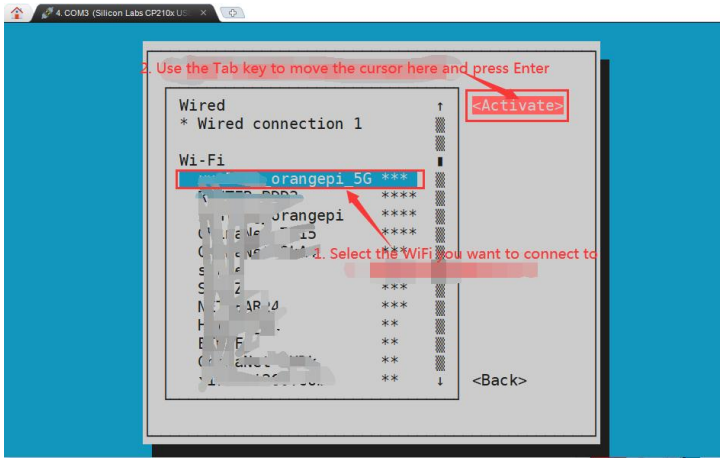
4) Select **Activate a connect** and press Enter.



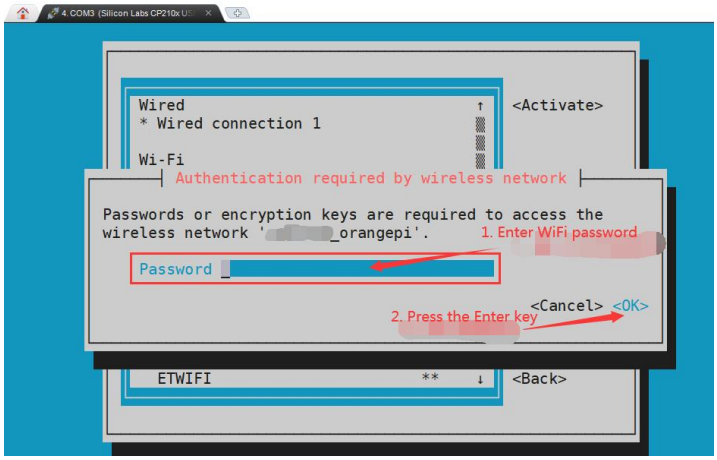
5) Then you can see all the searched WIFI hotspots.



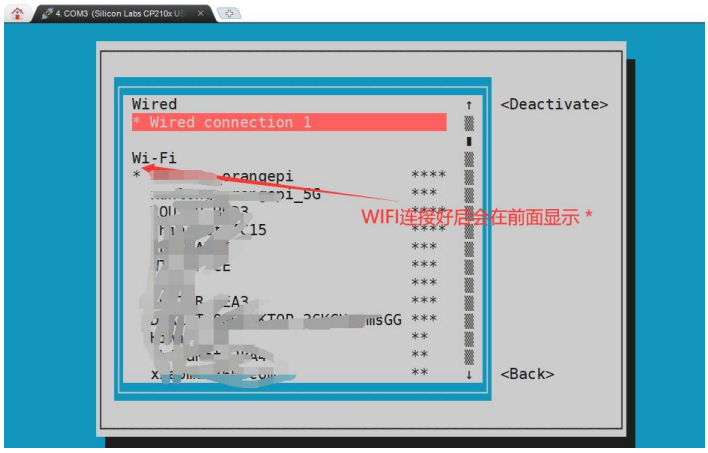
6) Select the WIFI hotspot you want to connect to, then use the Tab key to move the cursor to **Activate** and press Enter.



7) Then a dialog box for entering a password will pop up. Enter the corresponding password in **Password** 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 network through the **ip addr show wlan0** command.

```
orangepi@orangepi:~$ ip addr show wlan0
3: 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
```

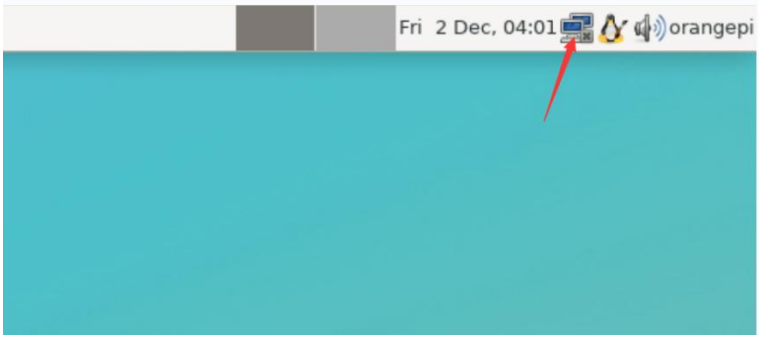
10) Use the **ping** command to test the connectivity of the WiFi network. The **ping** command can be interrupted by pressing the **Ctrl+C** shortcut key.

```
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
```

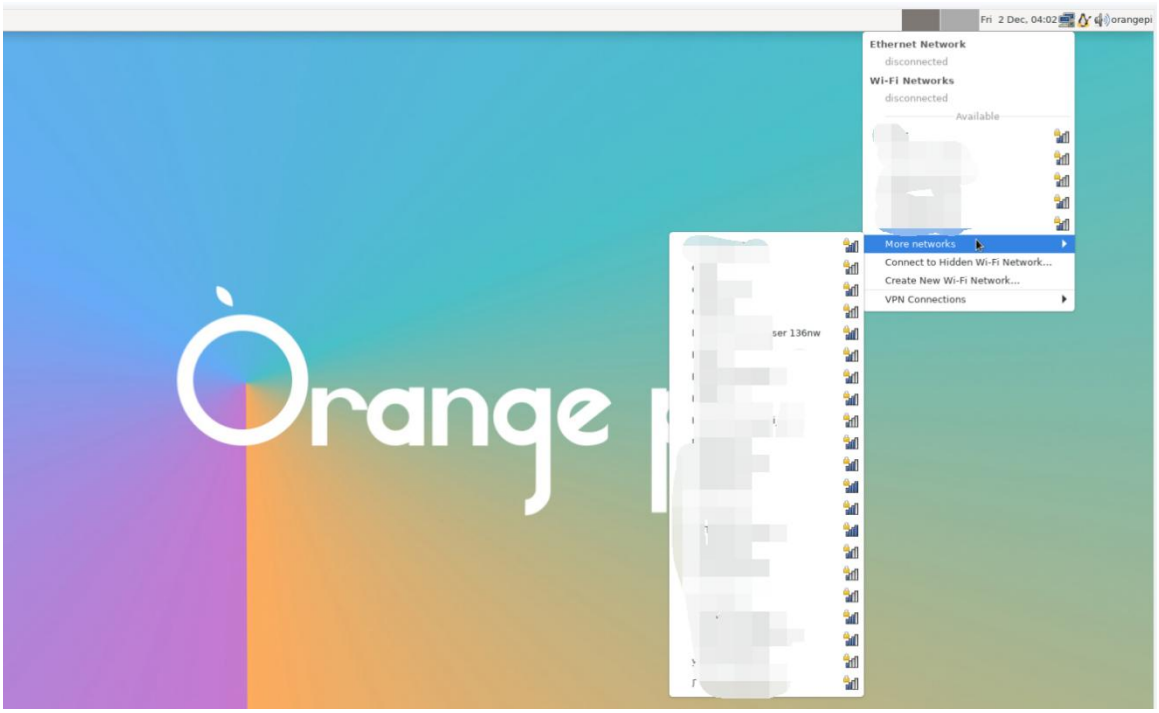
```
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=4 ttl=52 time=45.6 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=5 ttl=52 time=48.8 ms
^C
--- www.orangepi.org ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 41.321/44.864/48.834/2.484 ms
```

3.7.2.3. Testing methods for desktop images

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



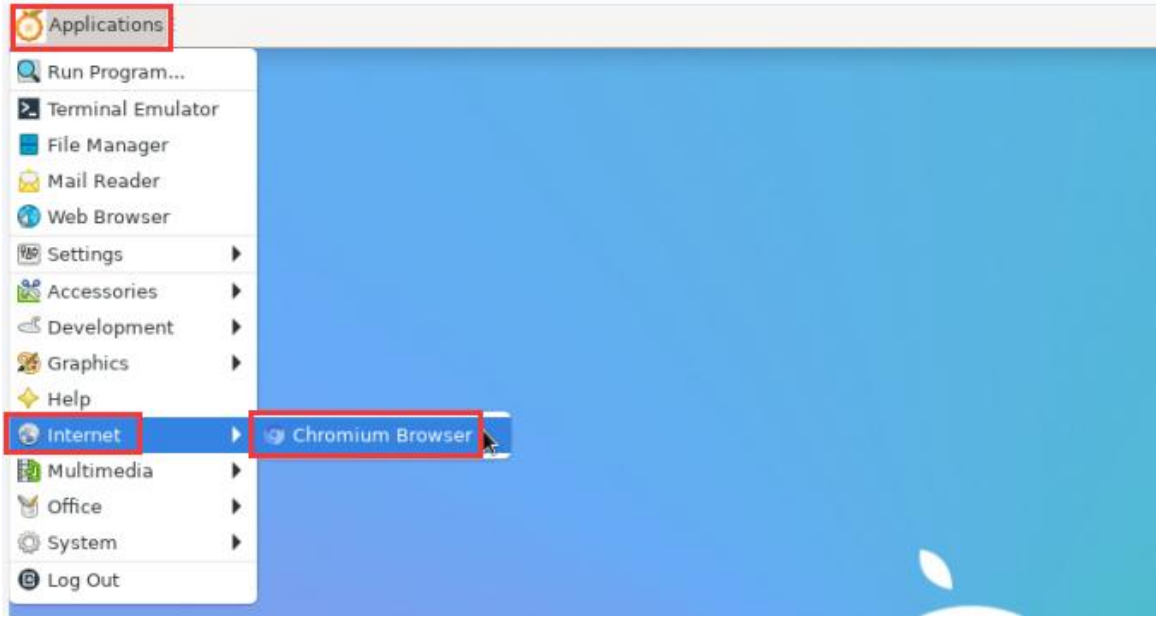
2) Click **More networks** in the pop-up drop-down box to see all scanned WIFI hotspots, and then select the WIFI hotspot you want to connect to.



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



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



5) If you can open other web pages after opening the browser, it means the WIFI connection is normal.



3. 7. 3. How to set a static IP address

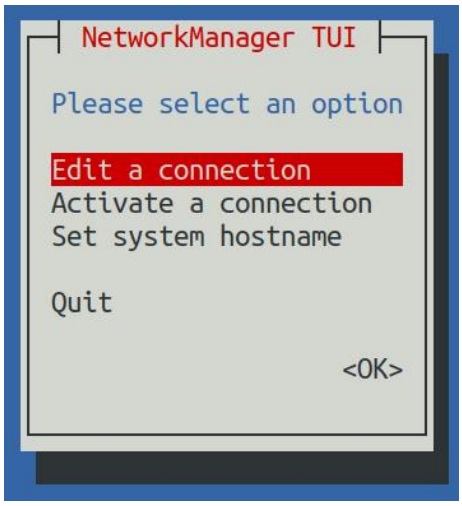
Please do not set a static IP address by modifying the /etc/network/interfaces configuration file.

3. 7. 3. 1. Using nmtui command to set static IP address

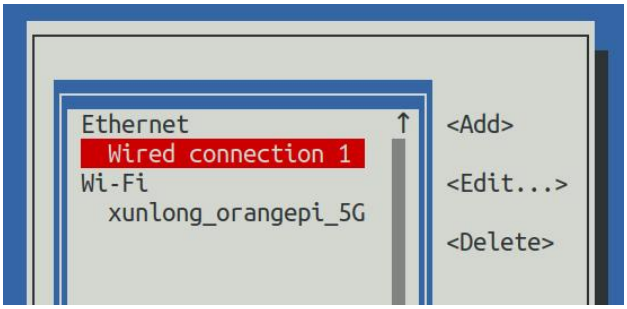
1) First run the **nmtui** command.

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

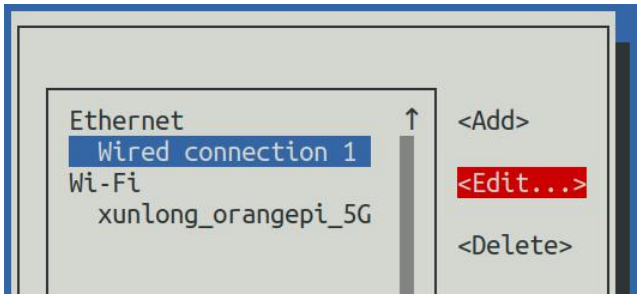
2) Then select **Edit a connection** and press Enter.



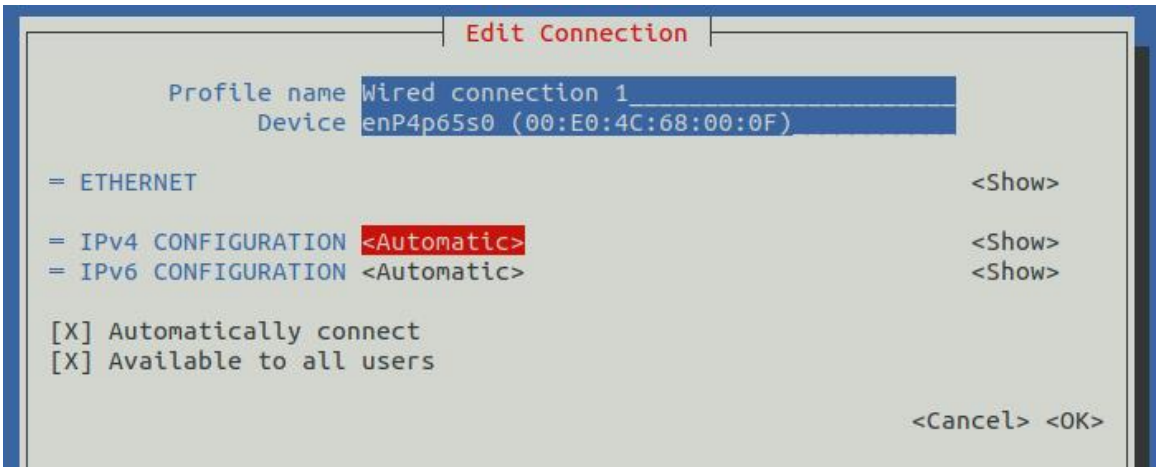
3) Then select the network interface for which you want to set a static IP address. For example, to set a static IP address for an **Ethernet** interface, select **Wired connection 1**.



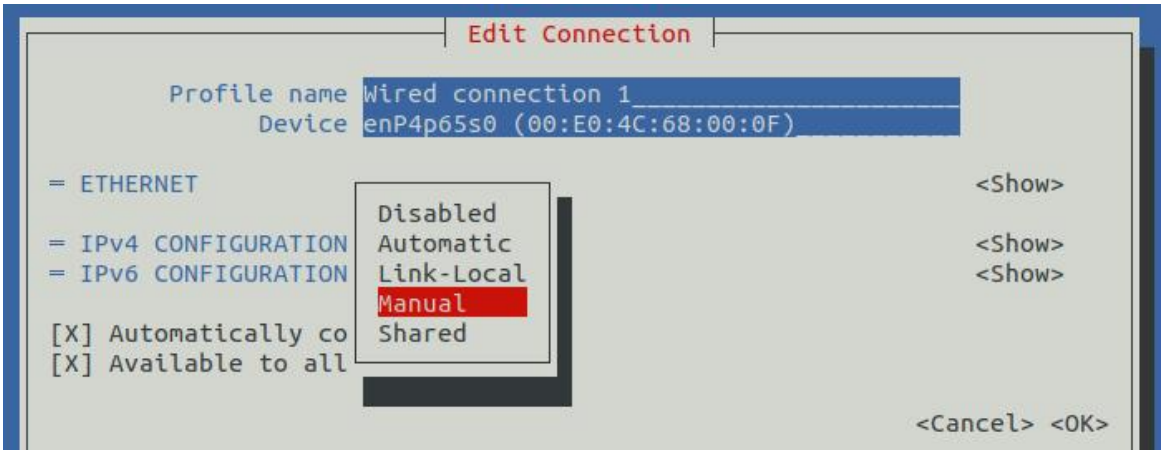
4) Then select **Edit** using the **Tab** key and press Enter.



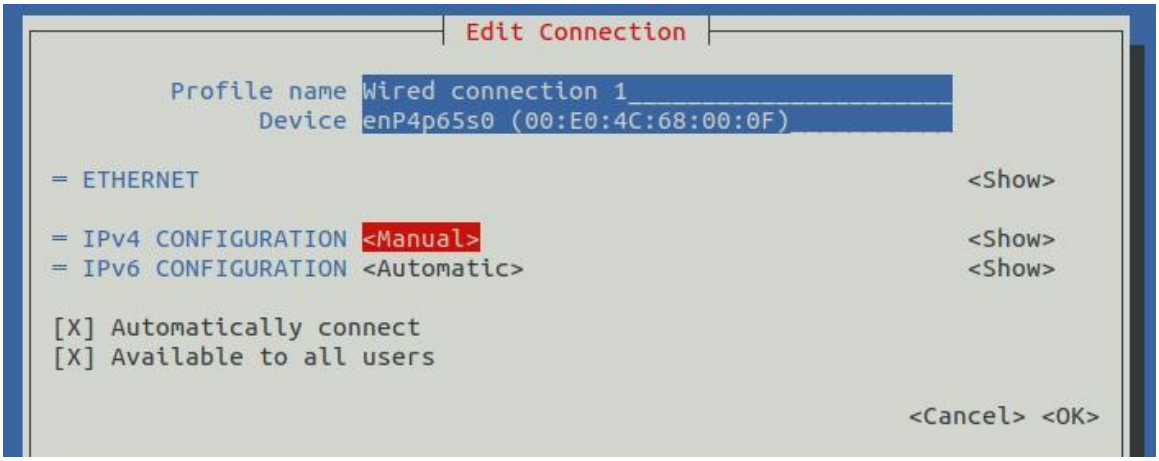
5) Then use the Tab key to move the cursor to the **<Automatic>** position shown in the figure below to configure IPv4.



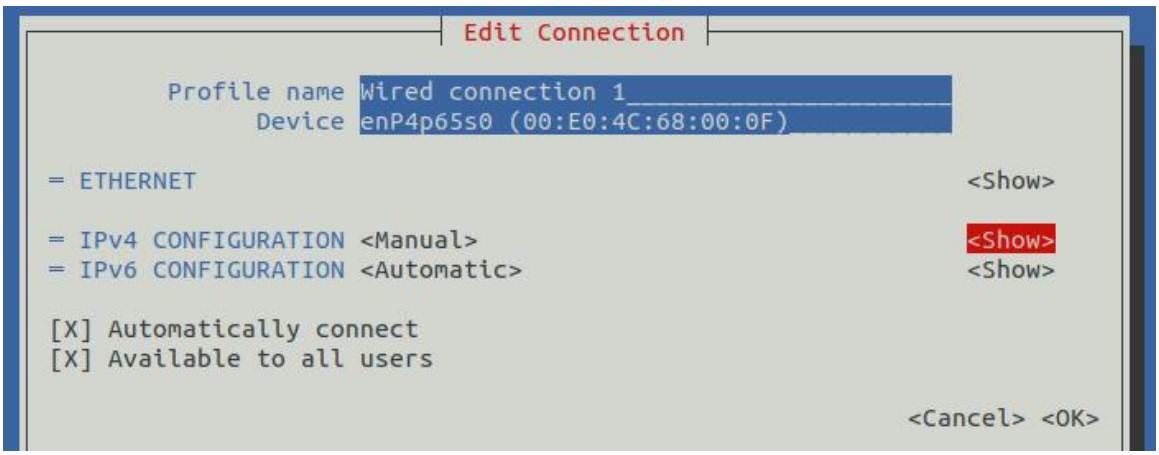
6) Press Enter, use the up and down arrow keys to select **Manual**, and then press Enter to confirm.



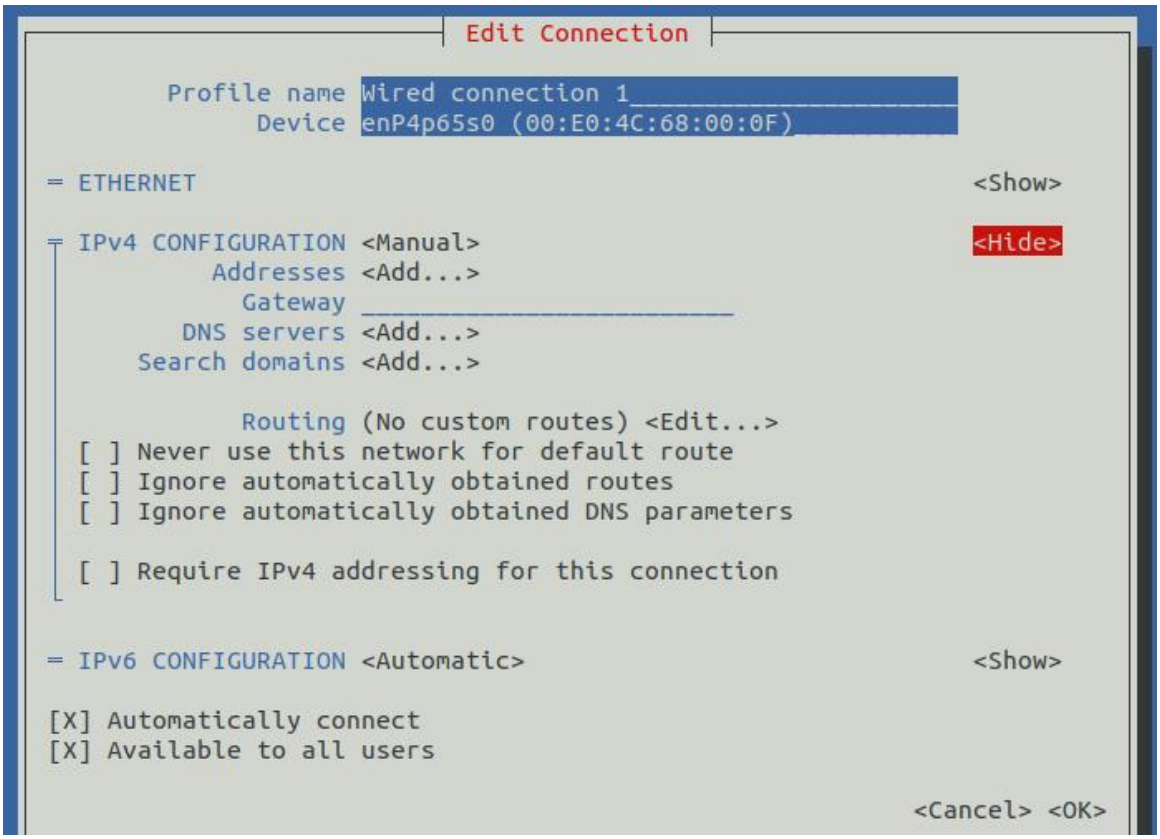
7) The display after selection is as shown below:



8) Then use the Tab key to move the cursor to**<Show>**.

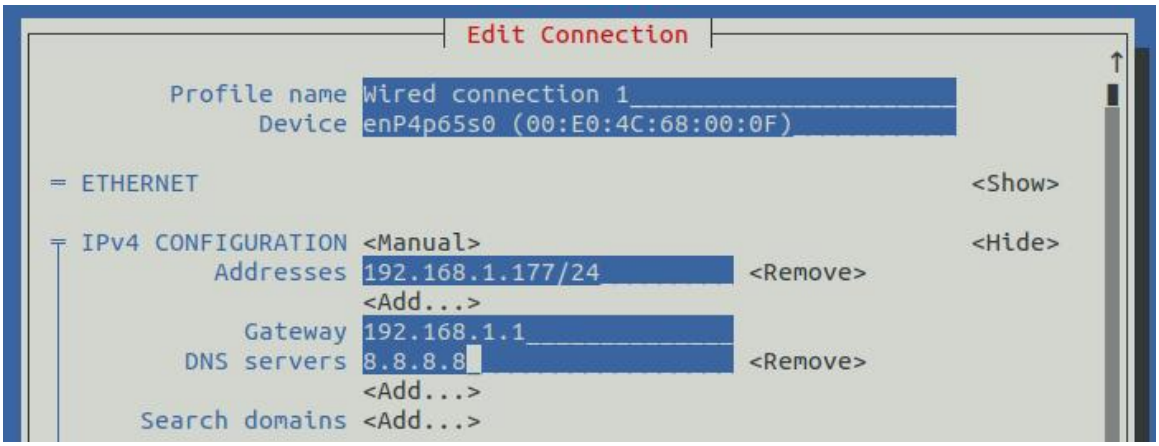


9) Then press Enter, and the following setting interface will pop up.

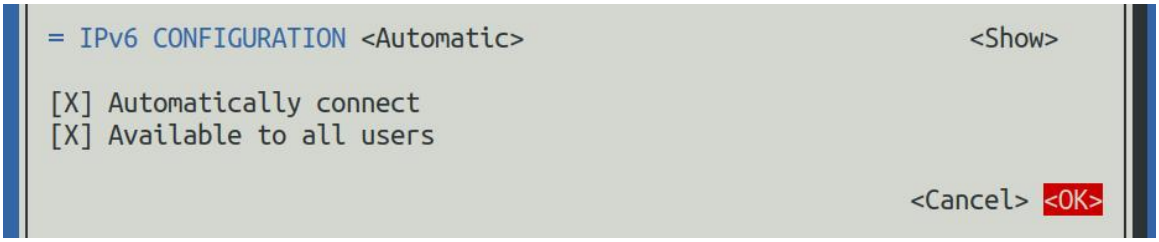


10) Then you can set the IP address (Addresses), gateway (Gateway) and DNS server address as shown in the figure below (there are many other setting options, please explore them yourself). **Please set them according to your specific needs. The value set in the**

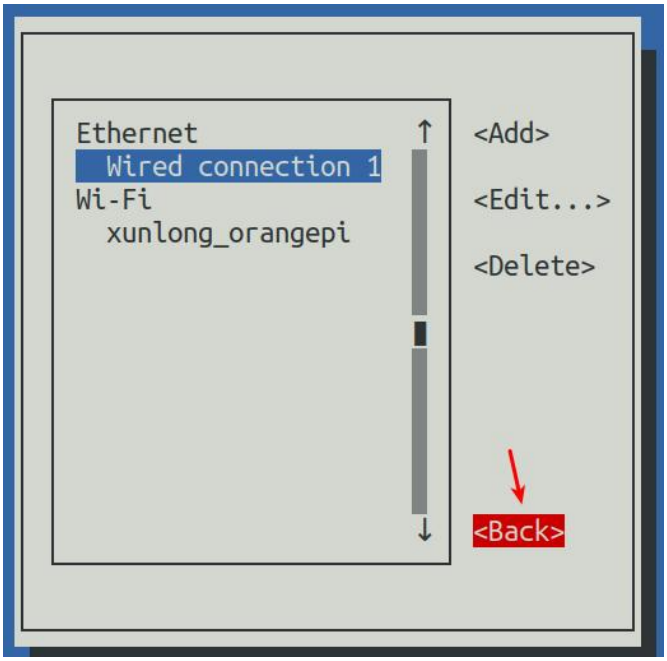
figure below is just an example.



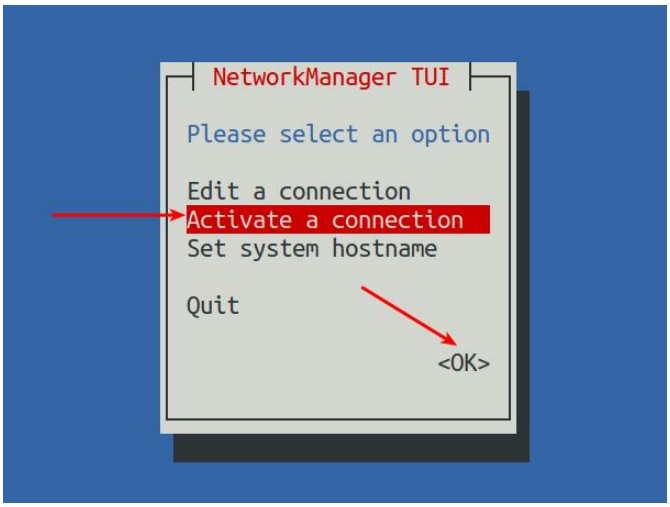
11) After setting, move the cursor to **<OK>** in the lower right corner and press Enter to confirm.



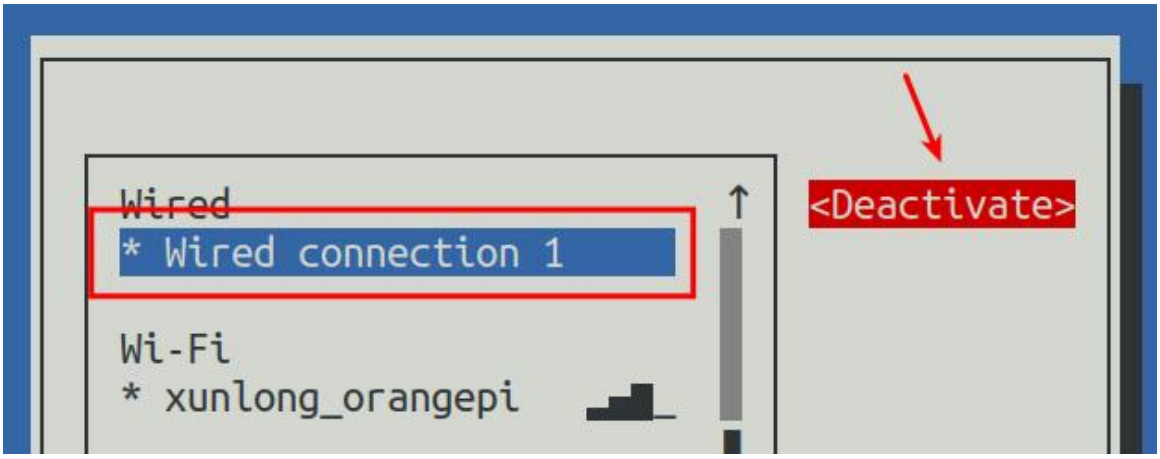
12) Then click **<Back>** to return to the previous selection interface.



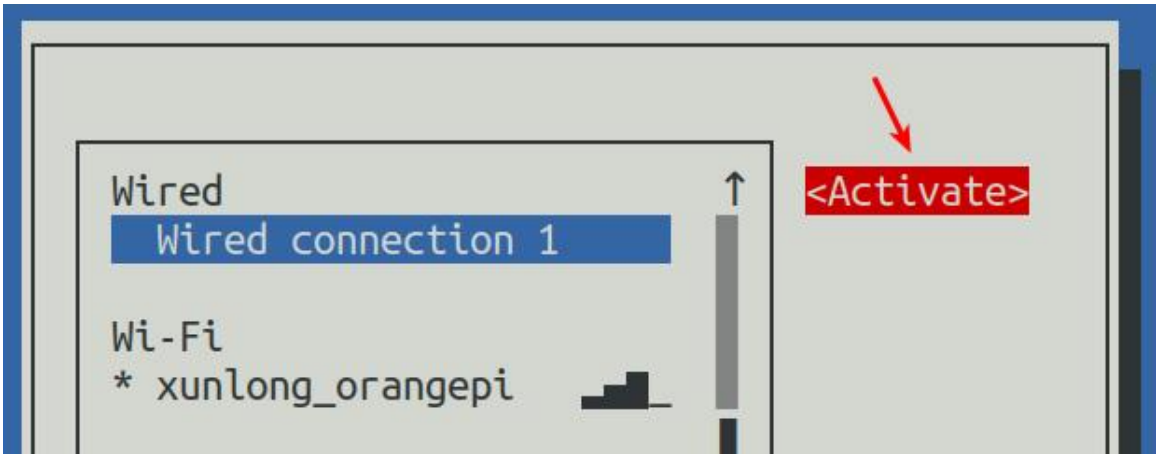
13) Then select **Activate a connection**, move the cursor to **<OK>**, and press Enter.



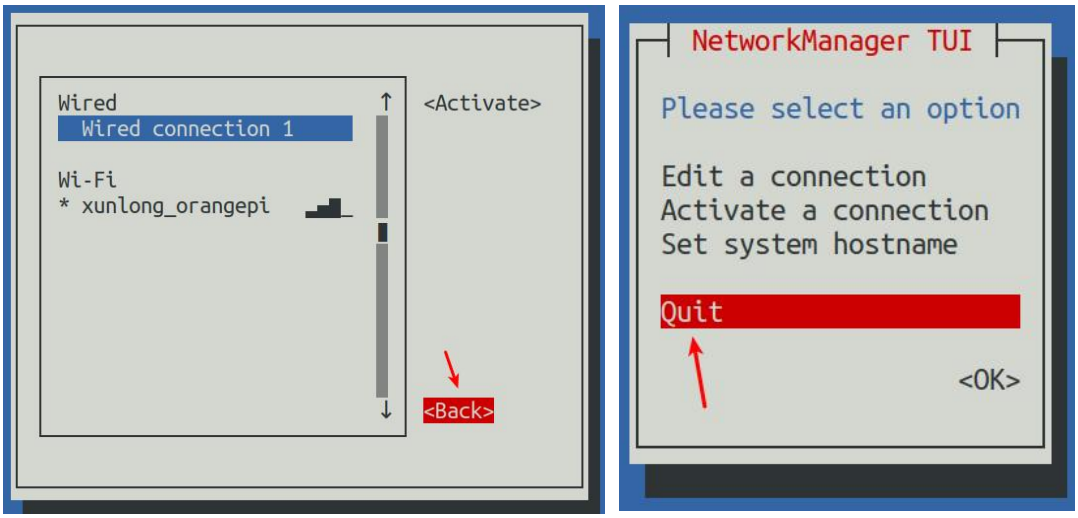
14) Then select the network interface you want to configure, such as **Wired connection 1**, move the cursor to **<Deactivate>**, and press Enter to disable **Wired connection 1**.



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



16) Then you can exit **nmtui** using the **<Back>** and **Quit** buttons.



17) Then use **ip addr show enP3p49s0** to see that the IP address of the network port has become the static IP address set previously.

```
orangepi@orangepi:~$ ip addr show enP3p49s0
2: enP3p49s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state
UP group default qlen 1000
    link/ether 00:e0:4c:68:00:0f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.177/24 brd 192.168.1.255 scope global noprefixroute enP3p49s0
       valid_lft forever preferred_lft forever
    inet6 fe80::d5aa:9a6:cd41:942e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

18) Then you can test the network connectivity to check whether the IP address is

configured OK. The ping command can be interrupted by pressing the **Ctrl+C**shortcut key.

```
orangepi@orangepi:~$ ping 192.168.1.47 -I enP3p49s0
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. 7. 4. How to create a WIFI hotspot through create_ap

create_ap is a script that helps quickly create a WIFI hotspot on Linux. It supports bridge and NAT modes and can automatically combine hostapd, dnsmasq and iptables to complete the setting of WIFI hotspot, avoiding users from making complex configurations. The github address is as follows:

https://github.com/oblique/create_ap

If you are using the latest image, the create_ap script is pre-installed. You can use the create_ap command to create a WIFI hotspot. The basic command format of create_ap is as follows:

create_ap [options] <wifi-interface> [<interface-with-internet>]
[<access-point-name> [<passphrase>]]

*** options: This parameter can be used to specify encryption method, frequency band of WIFI hotspot, bandwidth mode, network sharing method, etc. You can get the specific options through create_ap -h**
*** wifi-interface: the name of the wireless network card**
*** interface-with-internet: the name of the network card that can be connected to the Internet, usually eth0**
*** access-point-name: hotspot name**
*** passphrase: hotspot password**

3.7.4.1. create_ap method to create a WIFI hotspot in NAT mode

1）Enter the following command to create a WiFi hotspot in NAT mode with the name **orangepi** and the password **orangepi**.

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 enP3p49s0 orangepi orangepi
```

2）If the following information is output, it means that the WIFI hotspot is created successfully.

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 enP3p49s0 orangepi orangepi
Config dir: /tmp/create_ap.wlan0.conf.Ks6HobEw
PID: 5405
Network Manager found, set ap0 as unmanaged device... DONE
Creating a virtual WiFi interface... ap0 created.
Sharing Internet using method: nat
hostapd command-line interface: hostapd_cli -p
/tmp/create_ap.wlan0.conf.Ks6HobEw/hostapd_ctrl
ap0: interface state UNINITIALIZED->ENABLED
ap0: AP-ENABLED
```

3）Now take out your mobile phone and find the WIFI hotspot named **orangepi** created by the development board in the searched WIFI list. Then you can click **orangepi** to connect to the hotspot. The password is the **orangepi** set above.



4）The display after successful connection is as shown below:



5）In NAT mode, the wireless device connected to the development board's hotspot requests an IP address from the development board's DHCP service, so there will be two different network segments. For example, the IP of the development board here is 192.168.1.X.

```
orangepi@orangepi:~$ ifconfig enP3p49s0
enP3p49s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.150  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::938f:8776:5783:afa2  prefixlen 64  scopeid 0x20<link>
        ether 4a:a0:c8:25:42:82  txqueuelen 1000  (Ethernet)
        RX packets 25370  bytes 2709590 (2.7 MB)
        RX errors 0  dropped 50  overruns 0  frame 0
        TX packets 3798  bytes 1519493 (1.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 83
```

The DHCP service of the development board will assign an IP address of **192.168.12.0/24** to the device connected to the hotspot by default. At this time, click the connected WIFI hotspot **orangepi**, and then you can see that the IP address of the mobile phone is **192.168.12.X**.





6） If you want to specify a different network segment for the connected device, you can specify it through the -g parameter, such as using the -g parameter to specify the network segment of the access point AP as 192.168.2.1.

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 enP3p49s0 orangepi orangepi -g 192.168.2.1
```

At this time, after connecting to the hotspot through the mobile phone, click the connected WIFI hotspot **orangepi**, and then you can see that the IP address of the mobile phone is **192.168.2.X**.





7）If you do not specify the **--freq-band** parameter, the default hotspot created is the 2.4G band. If you want to create a 5G band hotspot, you can specify it with the **--freq-band 5** parameter. The specific command is as follows:

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 enP3p49s0 orangepi orangepi --freq-band 5
```

8）If you need to hide the SSID, you can specify the **--hidden** parameter. The specific command is as follows:

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 enP3p49s0 orangepi orangepi --hidden
```

At this time, the mobile phone cannot search for the WIFI hotspot. You need to manually specify the WIFI hotspot name and enter the password to connect to the WIFI hotspot.



3.7.4.2. create_ap method to create a WIFI hotspot in bridge mode

1）Enter the following command to create a WiFi hotspot in bridge mode with the name **orangepi** and the password **orangepi**.

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 enP3p49s0 orangepi orangepi
```

2）If the following information is output, it means that the WIFI hotspot is created successfully.

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 enP3p49s0 orangepi orangepi
[sudo] password for orangepi:
Config dir: /tmp/create_ap.wlan0.conf.fg9U5Xgt
PID: 3141
Network Manager found, set ap0 as unmanaged device... DONE
Creating a virtual WiFi interface... ap0 created.
Sharing Internet using method: bridge
Create a bridge interface... br0 created.
hostapd command-line interface: hostapd_cli -p
```

```
/tmp/create_ap.wlan0.conf.fg9U5Xgt/hostapd_ctrl
ap0: interface state UNINITIALIZED->ENABLED
ap0: AP-ENABLED
```

3）Now take out your mobile phone and find the WIFI hotspot named **orangepi** created by the development board in the searched WIFI list. Then you can click **orangepi** to connect to the hotspot. The password is the **orangepi** set above.



4）The display after successful connection is as shown below:



5）In bridge mode, the wireless device connected to the development board's hotspot also requests an IP address from the DHCP service of the main router (the router to which the development board is connected). For example, the IP of the development board here is **192.168.1.X**.

```
orangepi@orangepi:~$ ifconfig enP3p49s0
enP3p49s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.150  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::938f:8776:5783:afa2  prefixlen 64  scopeid 0x20<link>
        ether 4a:a0:c8:25:42:82  txqueuelen 1000  (Ethernet)
        RX packets 25370  bytes 2709590 (2.7 MB)
        RX errors 0  dropped 50  overruns 0  frame 0
```

```
        TX packets 3798  bytes 1519493 (1.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 83
```

The IP address of the device connected to the WIFI hotspot is also assigned by the main router, so the mobile phone and development board connected to the WIFI hotspot are in the same network segment. At this time, click the connected WIFI hotspot **orangepi**, and then you can see that the IP address of the mobile phone is also **192.168.1.X**.



6）If you do not specify the **--freq-band** parameter, the default hotspot created is the 2.4G band. If you want to create a 5G band hotspot, you can specify it with the **--freq-band 5** parameter. The specific command is as follows:

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 enP3p49s0 orangepi orangepi --freq-band 5
```

7）If you need to hide the SSID, you can specify the 定**--hidden** parameter. The specific command is as follows:

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 enP3p49s0 orangepi orangepi --hidden
```

At this time, the mobile phone cannot search for the WIFI hotspot. You need to manually specify the WIFI hotspot name and enter the password to connect to the WIFI hotspot.



3. 8. SSH remote login development board

By default, Linux systems have SSH remote login enabled, and allow the root user to log in to the system. Before SSH login, you must first ensure that the Ethernet or WiFi network is connected, and then use the ip addr command or check the router to obtain the IP address of the development board.

3. 8. 1. SSH remote login to the development board under Ubuntu

1) Get the IP address of the development board.

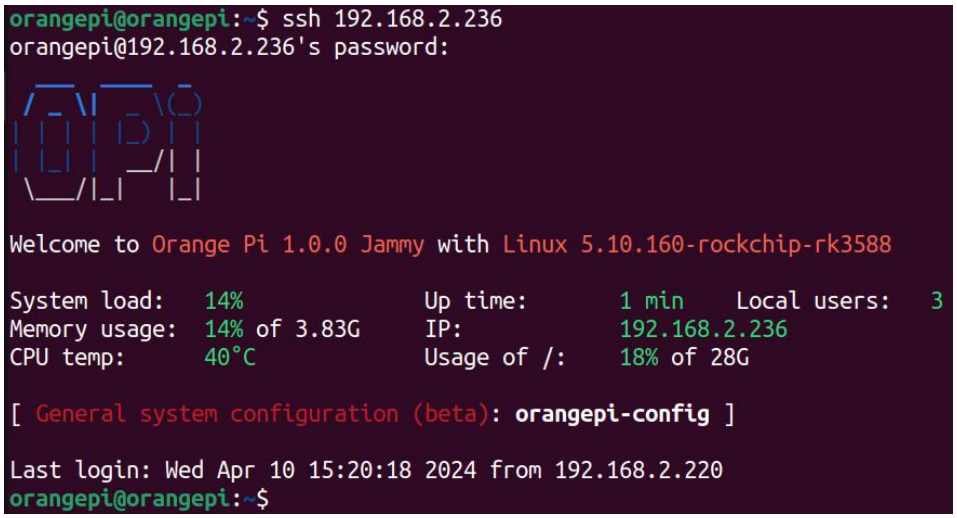
2) Then you can log in to the Linux system remotely through the ssh command.

```
test@test:~$ ssh root@192.168.x.xxx          #Need to be replaced with the IP address of the development board
root@192.168.x.xx's password:                #Enter the password here. The default password is orangepi
```

Note that when you enter the password, the screen will not display the specific content of the password you entered. Please do not think that there is any malfunction. Just press Enter after entering it.

If the prompt refuses to connect, as long as you are using the image provided by Orange Pi, please do not doubt whether the password orangepi is wrong, but look for other reasons.

3) After successfully logging into the system, the display is as shown below:



If ssh cannot log in to the Linux system normally, first check whether the IP address of the development board can be pinged. If the ping is successful, you can log in to the Linux system through the serial port or HDMI display and then enter the following command on the development board to try to connect:

root@orangepi:~# **reset_ssh.sh**

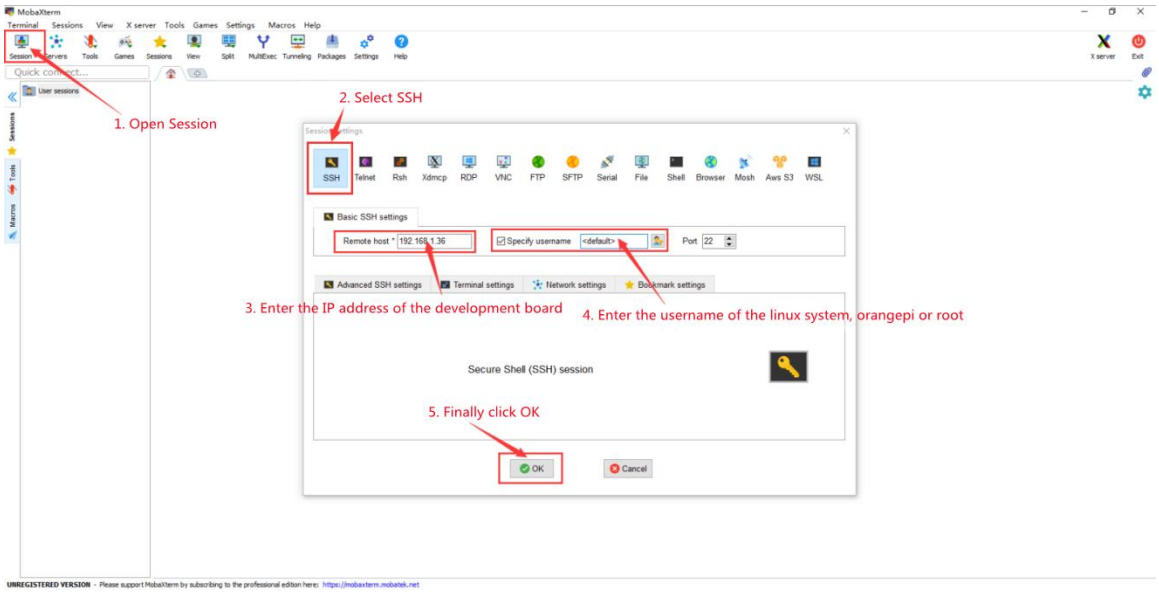
If it still doesn't work, please re-burn the system and try again.

3.8.2. SSH remote login development board under Windows

1) First, obtain the IP address of the development board.

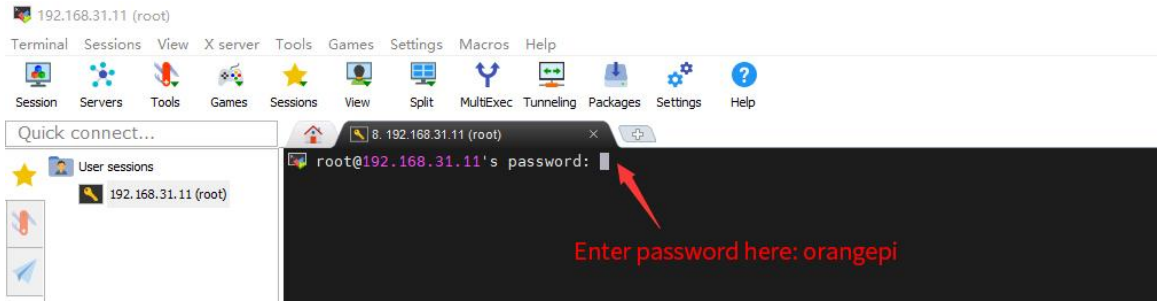
2) Under Windows, you can use MobaXterm to remotely log in to the development board. First, create a new ssh session.

a. Open **Session**.
b. Select **SSH** in **Session Setting**.
c. Enter the IP address of the development board in **Remote host**.
d. Enter the Linux user name **root** or **orangepi** in **Specify username**.
e. Click **OK**.

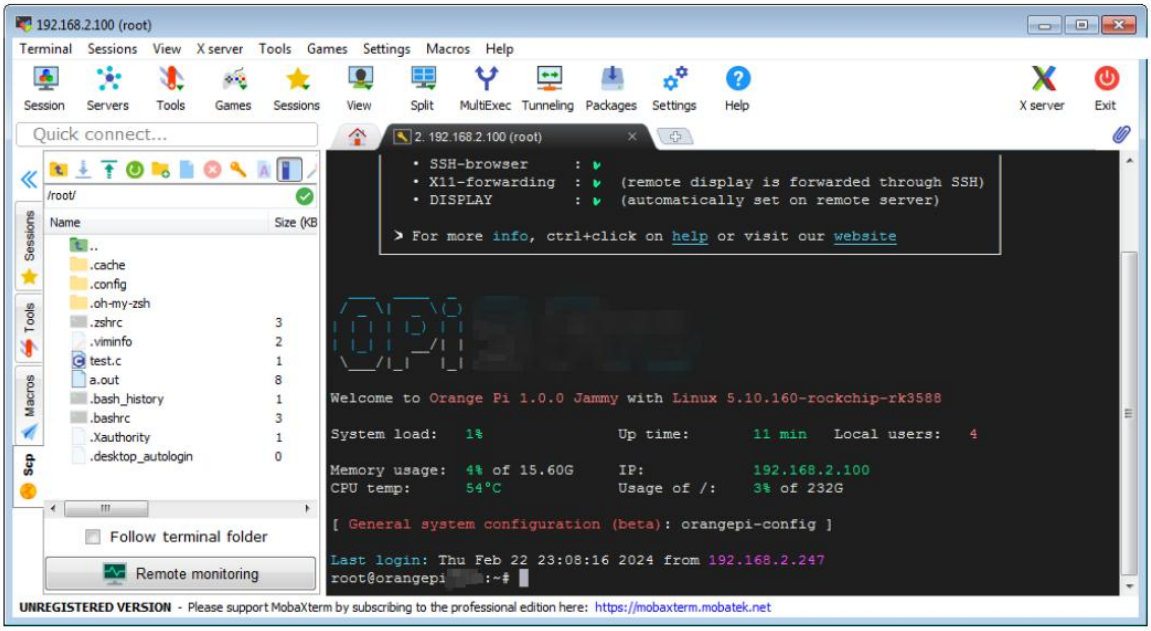


3) You will then be prompted to enter a password. The default password for both root and orangepi users is orangepi.

Please note that when you enter the password, the specific content of the password will not be displayed on the screen. Please do not think that there is any malfunction. Just press Enter after entering it.



4) After successfully logging into the system, the display is as shown below:



3. 9. How to use ADB

3. 9. 1. How to use network adb

1) After the system starts, please make sure that **adbd** has been started.

```
orangepi@orangepi:~$ ps -ax | grep "adbd"
    808 ?        Sl     0:00 /usr/bin/adbd
   3707 ttyFIQ0  S+     0:00 grep --color=auto adbd
```

2) Then check the IP address of the development board and write it down.

3) Then install adb tool on your Ubuntu PC.

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install -y adb
```

4) Then use the following command to connect to the network adb.

```
test@test:~$ adb connect 192.168.1.xx:5555      #Please replace IP address with the IP address of the development board
* daemon not running; starting now at tcp:5037
* daemon started successfully
connected to 192.168.1.xx:5555
```

```
test@test:~$ adb devices
List of devices attached
192.168.1.xx:5555          device
```

5) Then use the following command to log in to the Linux system of the development board.

```
test@test:~$ adb shell
root@orangepi5ultra:/#      <--- Seeing this prompt means you have successfully logged into the development board
```

6) The command to upload files to the development board using adb is as follows:

```
test@test:~$ adb push filename /root
filename: 1 file pushed. 3.7 MB/s (1075091 bytes in 0.277s)
```

7) The command to restart the development board using adb is as follows:

```
test@test:~$ adb reboot
```

If you don't have the adb tool in your Windows system, you can use the adb program in the RKDevTool software (this software is useful in the section on how to burn the Android image to SPIFlash+NVMe SSD).



An example using adb in Windows is shown below:

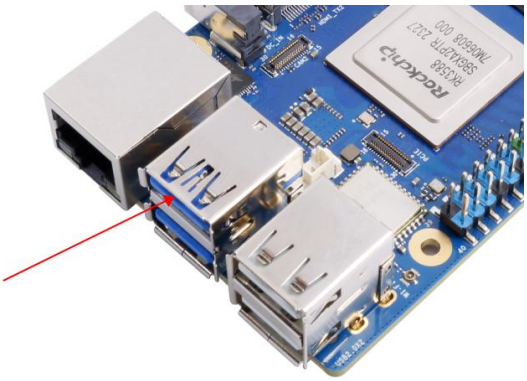
```
命令提示符
Microsoft Windows [版本 10.0.19044.2251]
(c) Microsoft Corporation。保留所有权利。

C:\Users\Administrator>cd C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin

C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin>dir
 驱动器 C 中的卷没有标签。
 卷的序列号是 62AE-5AED

 C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin 的目录

2022/08/09  13:19    <DIR>          .
2022/08/09  13:19    <DIR>          ..
2019/06/24  09:13         1,850,368 adb.exe
2019/06/24  09:13            97,792 AdbWinApi.dll
2019/06/24  09:13            62,976 AdbWinUsbApi.dll
2021/08/23  09:04           894,976 AFPTool.exe
2021/08/16  14:05           890,368 RKImageMaker.exe
               5 个文件      3,796,480 字节
               2 个目录 64,033,034,240 可用字节

C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin>.\adb.exe connect 192.168.1.144
connected to 192.168.1.144:5555

C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin>.\adb.exe devices
List of devices attached
192.168.1.144:5555      device


C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin>.\adb.exe push adb.exe /root
adb.exe: 1 file pushed. 4.1 MB/s (1850368 bytes in 0.427s)

C:\Users\Administrator\Desktop\RKDevTool_Release_v2.92\bin>
```

3.9.2. Use a USB male-to-male data cable to connect to adb

1) First, prepare a good quality USB male-to-male data cable.



2) Then connect the development board to the Ubuntu PC via a USB male-to-male data cable. The location of the USB interface that supports the device function of the development board is shown in the figure below:



3) Then run the following command to set the USB interface to device mode.

```
orangepi@orangepi:~$ sudo set_device.sh
```

If the **set_device.sh** script does not exist in the Linux system, use the following command directly:

```
orangepi@orangepi:~$ sudo bash -c "echo device > /sys/kernel/debug/usb/fc000000.usb/mode"
orangepi@orangepi:~$ sudo systemctl restart usbdevice
```

4) Then please make sure adbd is started.

```
orangepi@orangepi:~$ ps -ax | grep "adbd"
    808 ?        Sl     0:00 /usr/bin/adbd
   3707 ttyFIQ0  S+     0:00 grep --color=auto adbd
```

5) Then install the adb tool on your Ubuntu PC.

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install -y adb
```

6) Then use the following command to check whether the adb device is recognized.

```
test@test:~$ adb devices
List of devices attached
e0f9f71bc343c305    device
```

8) Then use the following command to log in to the Linux system of the development board.

```
test@test:~$ adb shell
root@orangepi5ultra:/#      <---Seeing this prompt means you have successfully logged into the development board
```

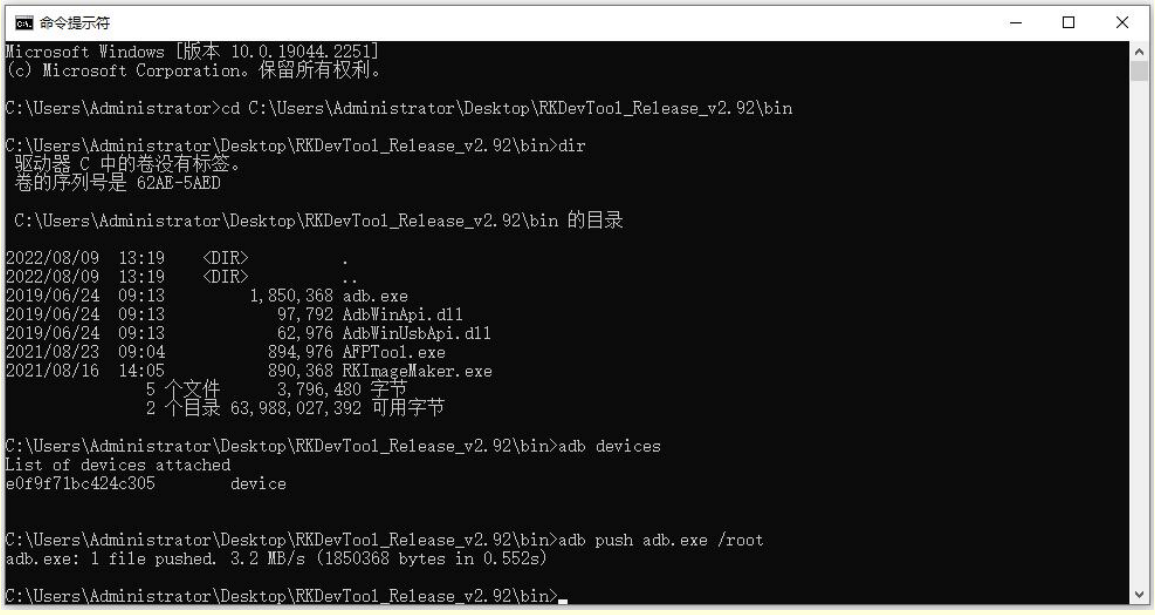
9) The command to upload files to the development board using adb is as follows:

```
test@test:~$ adb push filename /root
filename: 1 file pushed. 3.7 MB/s (1075091 bytes in 0.277s)
```

If you don't have the adb tool in your Windows system, you can use the adb program in the RKDevTool software (this software is useful in the section on how to burn the Android image to SPIFlash+NVMe SSD).



An example using adb in Windows is shown below:



3.10. How to upload files to the Linux system of the development board

3.10.1. How to upload files from Ubuntu PC to the Linux system of the development board

3.10.1.1. How to upload files using the scp command

1) Use the scp command to upload files from the Ubuntu PC to the Linux system of the development board. The specific commands are as follows:

a. **file_path**: needs to be replaced with the path of the file to be uploaded.

b. **orangepi** is the user name of the development board's Linux system, which can also be replaced with other names, such as root.

c. **192.168.xx.xx**: is the IP address of the development board, please modify it according to the actual situation.

d. **/home/orangepi:** is the path in the development board's Linux system, which can also be changed to other paths.

```
test@test:~$ scp    file_path    orangepi@192.168.xx.xx:/home/orangepi/
```

2) If you want to upload a folder, you need to add the -r parameter.

```
test@test:~$ scp    -r    dir_path    orangepi@192.168.xx.xx:/home/orangepi/
```

3) There are more uses for scp. Please use the following command to view the man manual.

```
test@test:~$ man scp
```

3.10.1.2. How to upload files using FileZilla

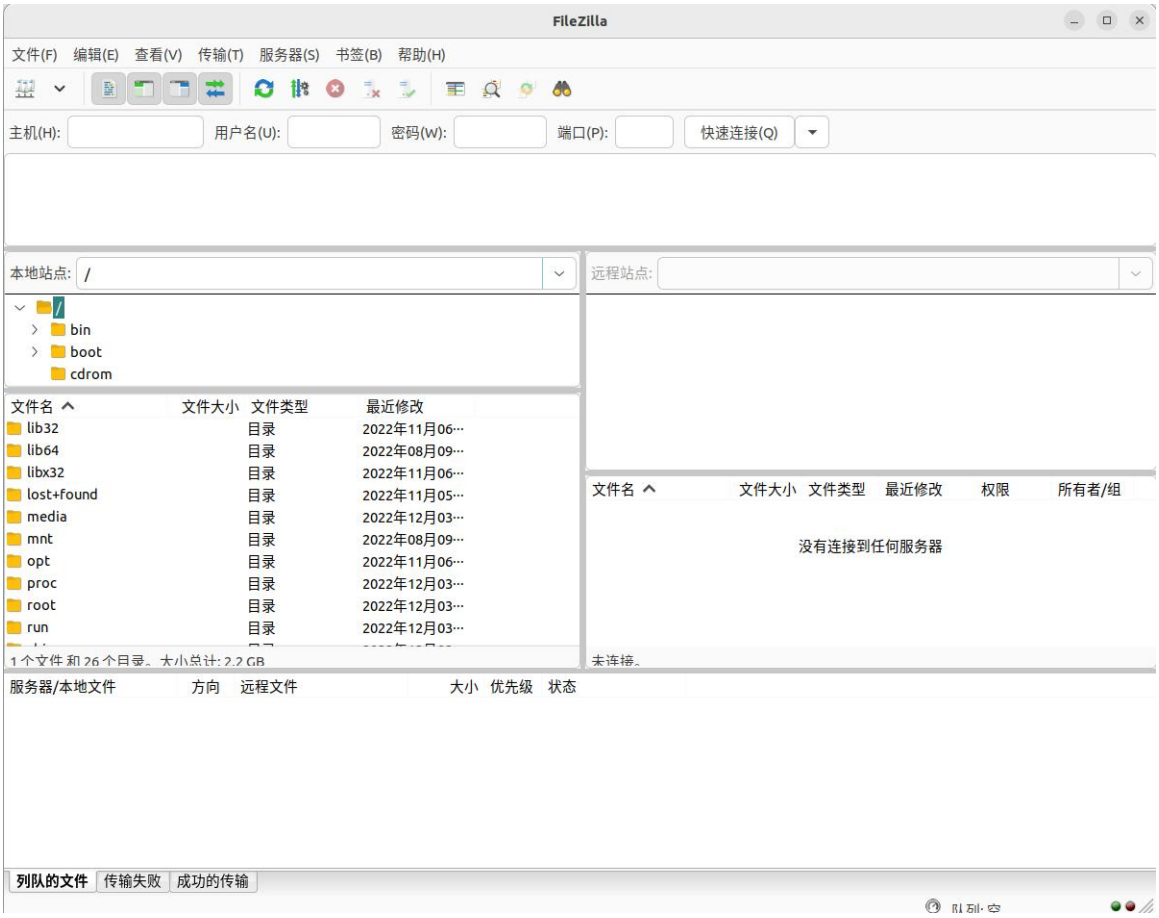
1) First install filezilla in your Ubuntu PC.

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

2) Then open filezilla using the command below.

```
test@test:~$ filezilla
```

3) The interface after opening filezilla is as shown below. At this time, the remote site on the right is empty.



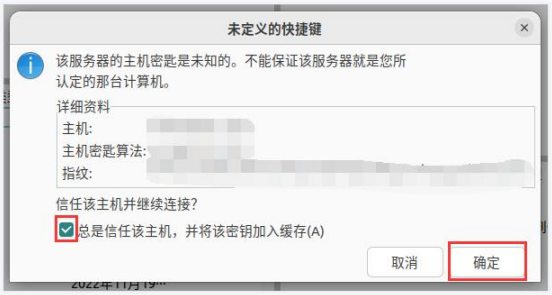
4) The method of connecting the development board is shown in the figure below:



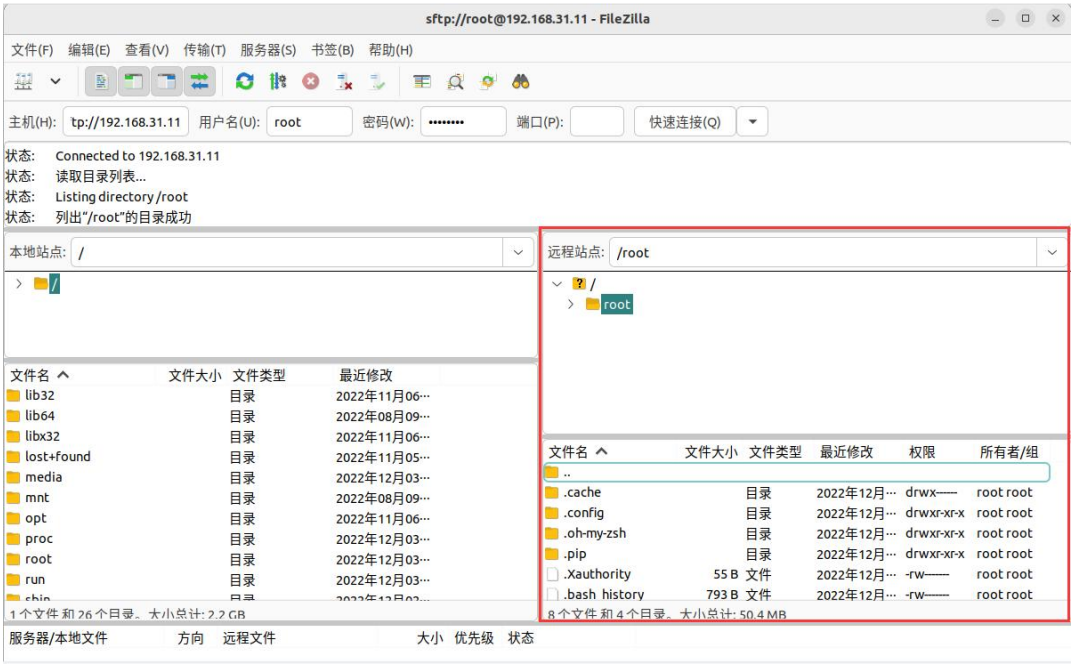
5) Then select **Save** Password and click **OK**.



6) Then select **Always trust this host** and click **OK**.

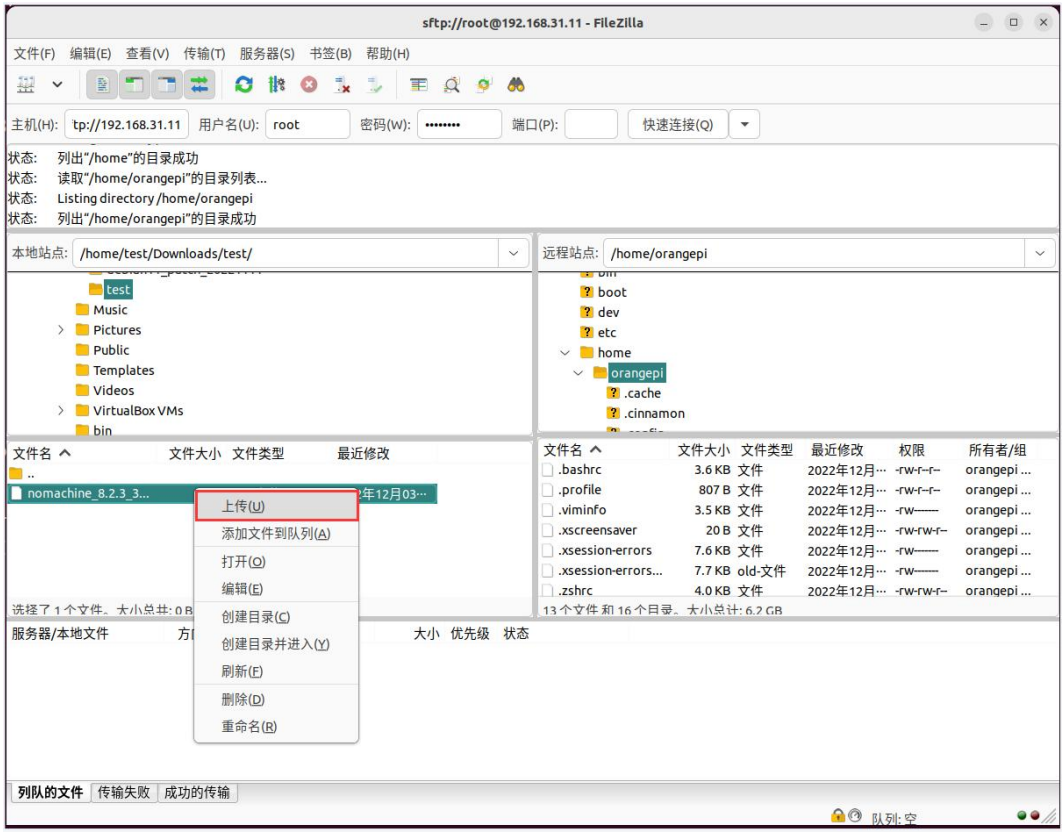


7) After the connection is successful, you can see the directory structure of the development board's Linux file system on the right side of the filezilla software.



8) Then select the path to be uploaded to the development board on the right side of the

filezilla software, then select the file to be uploaded in the Ubuntu PC on the left side of the filezilla software, right-click the mouse, and then click the upload option to start uploading the file to the development board.



9) After uploading is complete, you can go to the corresponding path in the Linux system of the development board to view the uploaded files.

10) The method for uploading a folder is the same as that for uploading a file, so I will not go into details here.

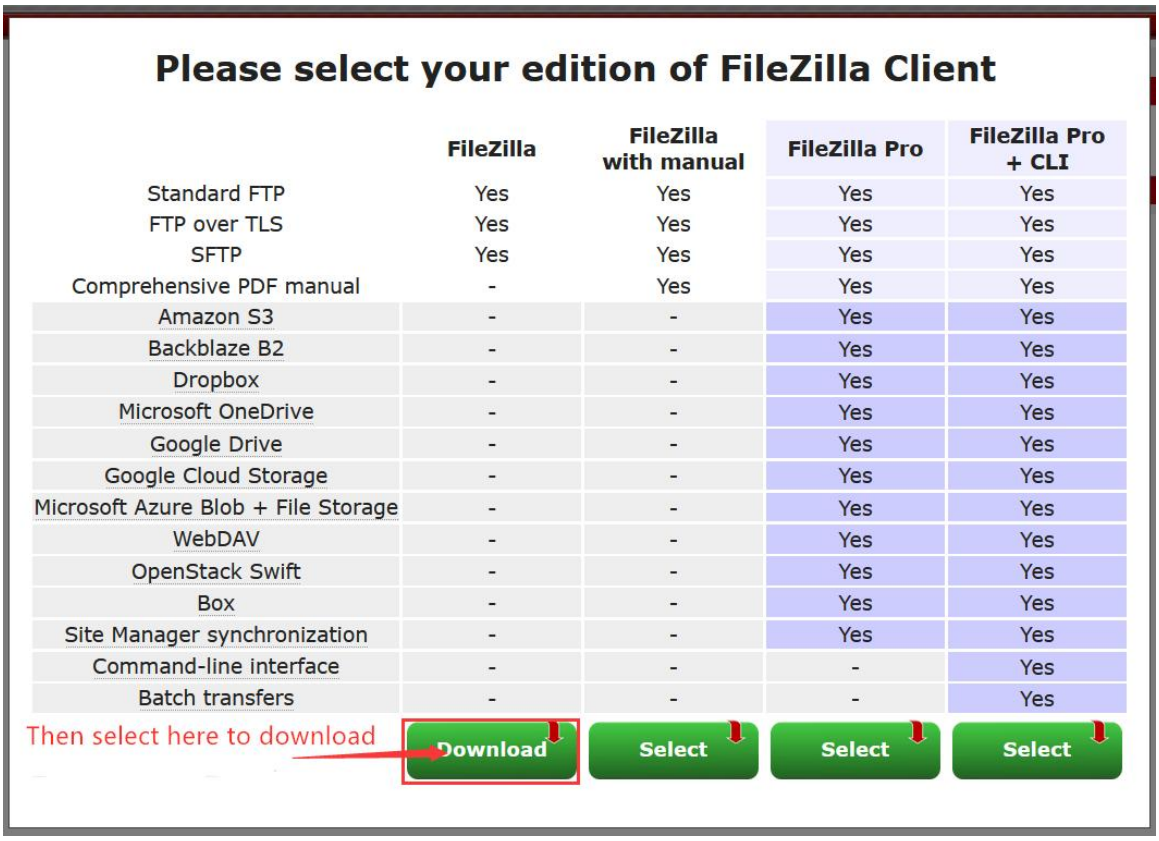
3.10.2. How to upload files from Windows PC to the Linux system of the development board

3.10.2.1. How to upload files using FileZilla

1) First download the installation file of the Windows version of the filezilla software. The download link is as follows:

https://filezilla-project.org/download.php?type=client





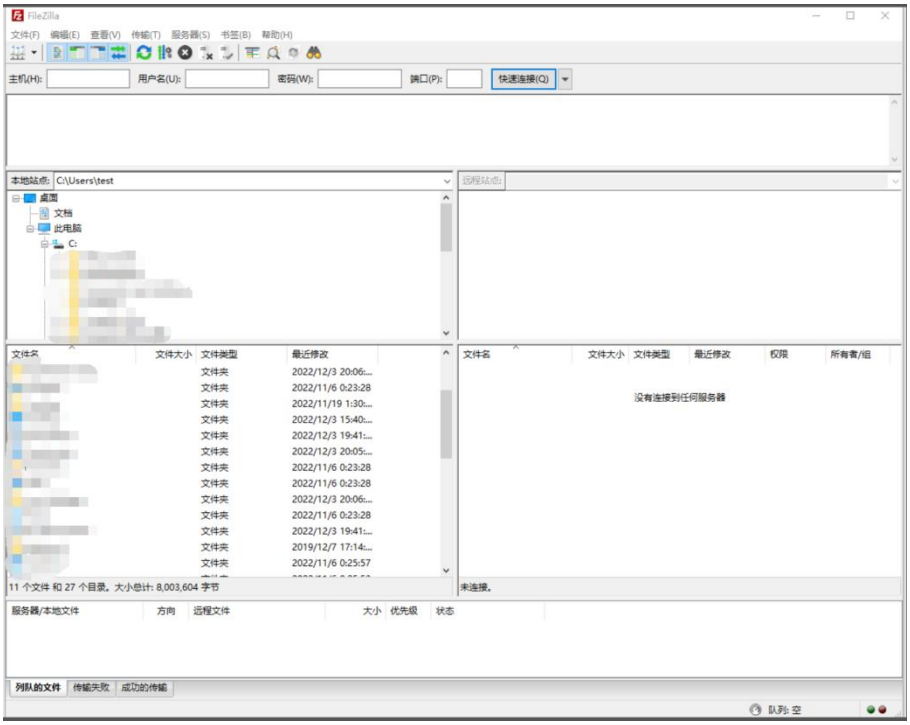
2) The downloaded installation package is as shown below, then double-click to install directly.

FileZilla_Server_1.5.1_win64-setup.exe

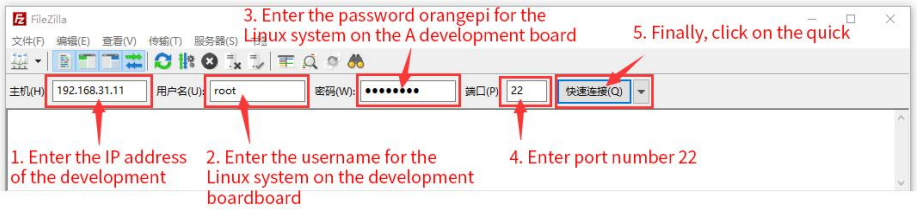
During the installation process, select **Decline** on the following installation interface, and then select **Next>**.



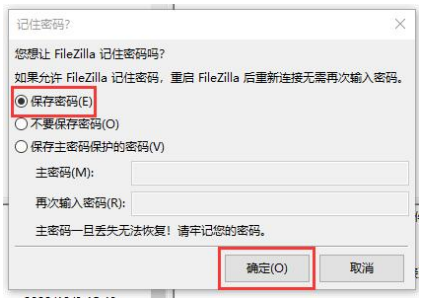
3) The interface after opening filezilla is as shown below. At this time, the remote site on the right is empty.



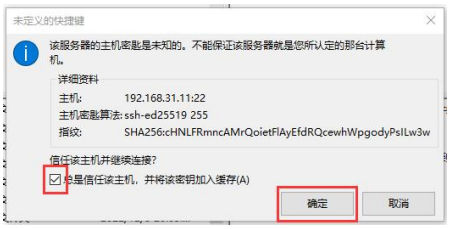
4) The method of connecting the development board is shown in the figure below:



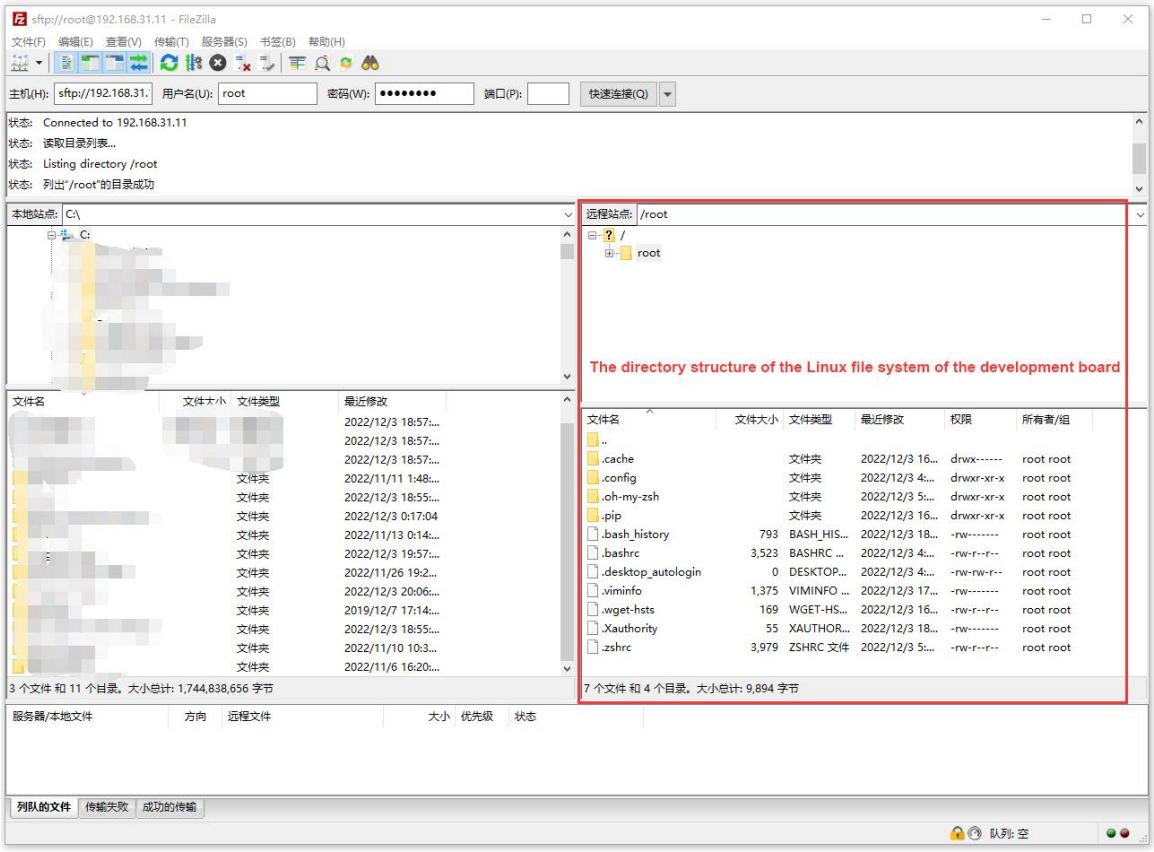
5) Then select **Save Password** and click **OK**.



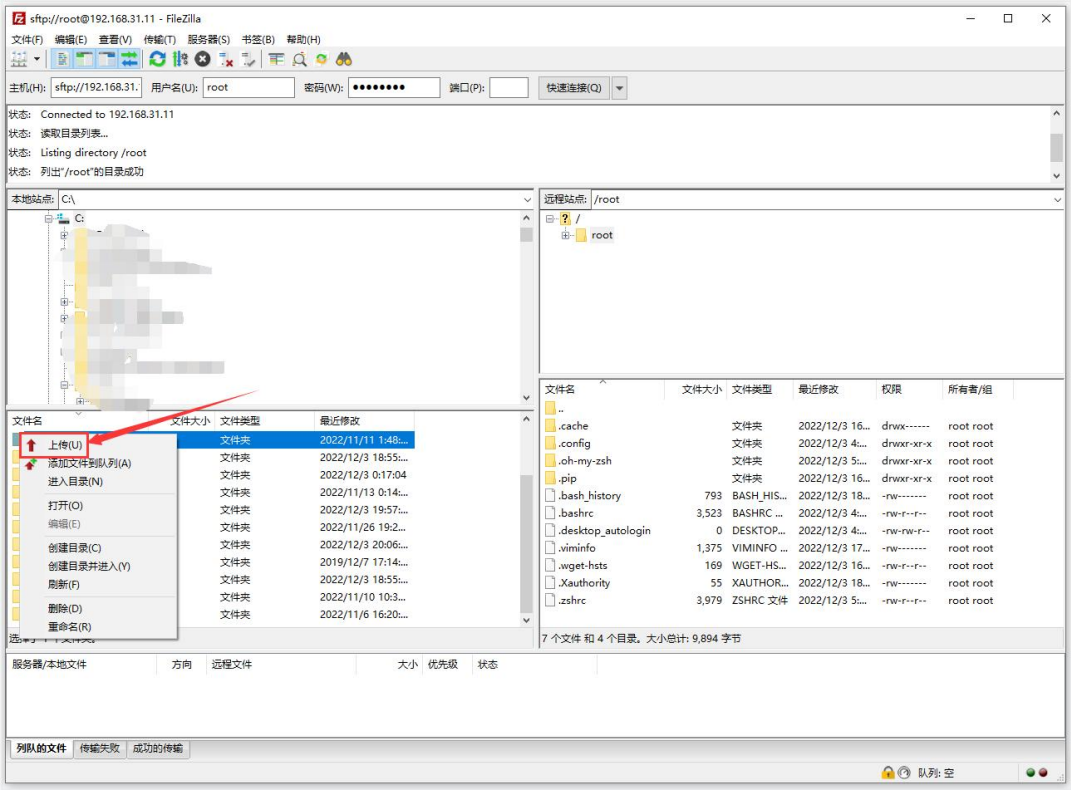
6) Then select **Always trust this host** and click **OK**.



7) After the connection is successful, you can see the directory structure of the development board's Linux file system on the right side of the filezilla software.



8) Then select the path to be uploaded to the development board on the right side of the filezilla software, then select the file to be uploaded in the Windows PC on the left side of the filezilla software, right-click the mouse, and then click the upload option to start uploading the file to the development board.



9) After uploading is complete, you can go to the corresponding path in the Linux system of the development board to view the uploaded files.

10) The method for uploading a folder is the same as that for uploading a file, so I will not go into details here.

3. 11. HDMI test

3. 11. 1. HDMI display test

1) Use an HDMI to HDMI cable to connect the Orange Pi development board and the HDMI display.



2) After starting the Linux system, if the HDMI monitor has image output, it means that the HDMI interface is working properly.

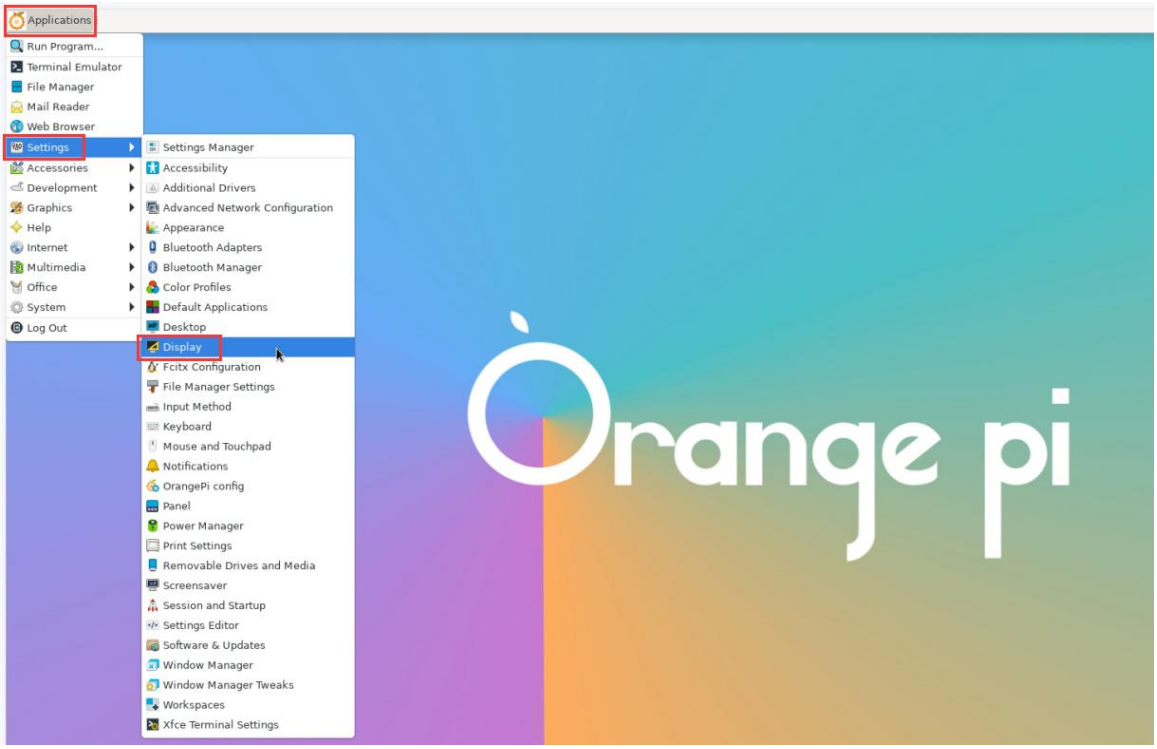
Note that although many laptops have HDMI interfaces, the HDMI interfaces of laptops generally only have output functions and do not have HDMI in functions, which means that the HDMI output of other devices cannot be displayed on the laptop screen.

When you want to connect the HDMI of the development board to the HDMI interface of the laptop, please first confirm that your laptop supports the HDMI in function.

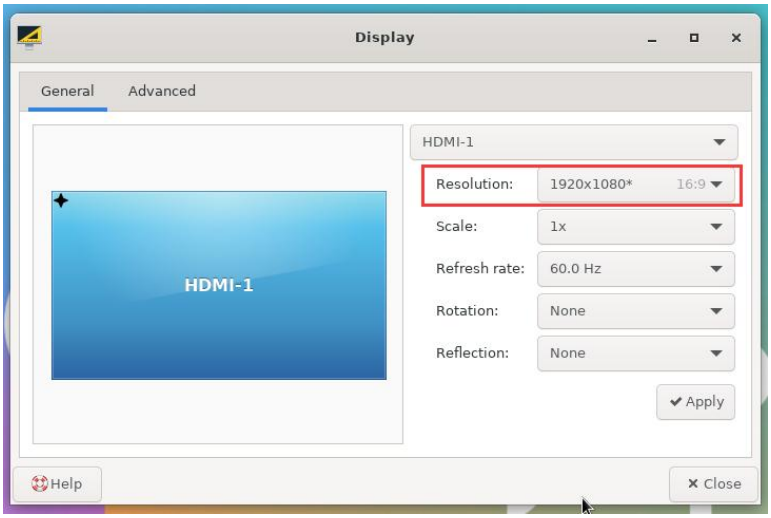
When there is no display on HDMI, please first check whether the HDMI cable is plugged in tightly. After confirming that the connection is OK, you can try a different screen to see if there is any display.

3. 11. 2. HDMI resolution setting method

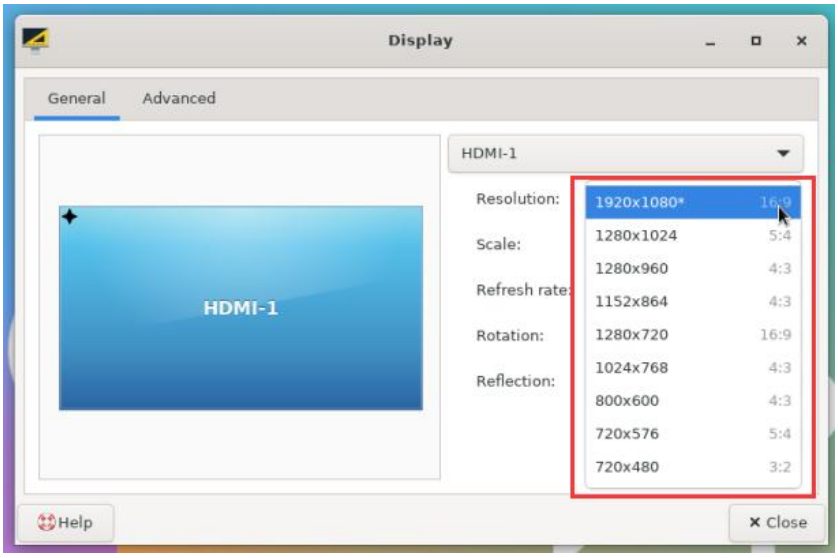
1) First, open **Display** in **Settings**.



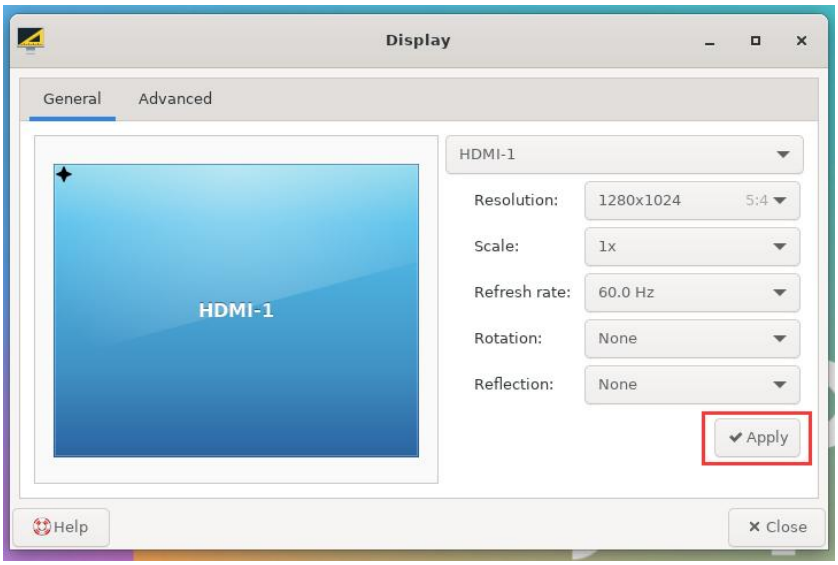
2) Then you can see the current resolution of the system.



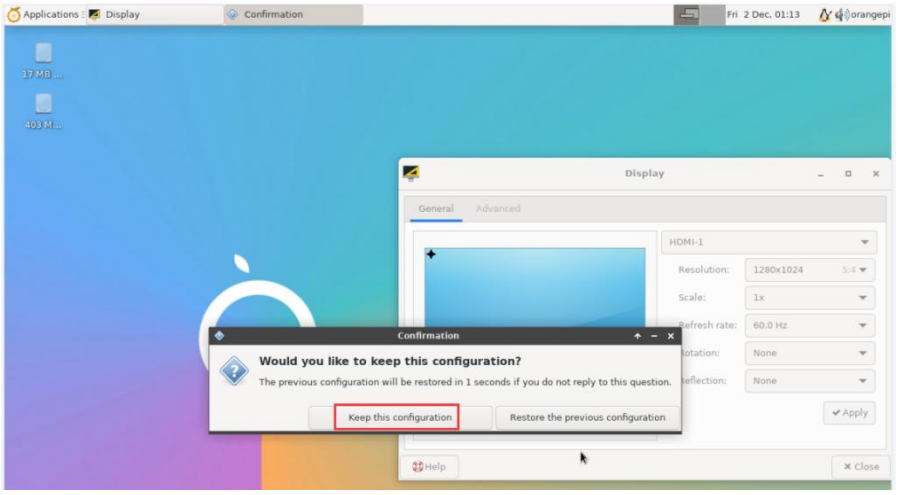
3) Click the drop-down box of Resolution to see all the resolutions currently supported by the monitor.



4) Then select the resolution you want to set and click **Apply**.



5) After the new resolution is set, select **Keep the configuration**.



3. 11. 3. Test method for HDMI RX

1) The location of the HDMI RX interface on the development board is as follows:



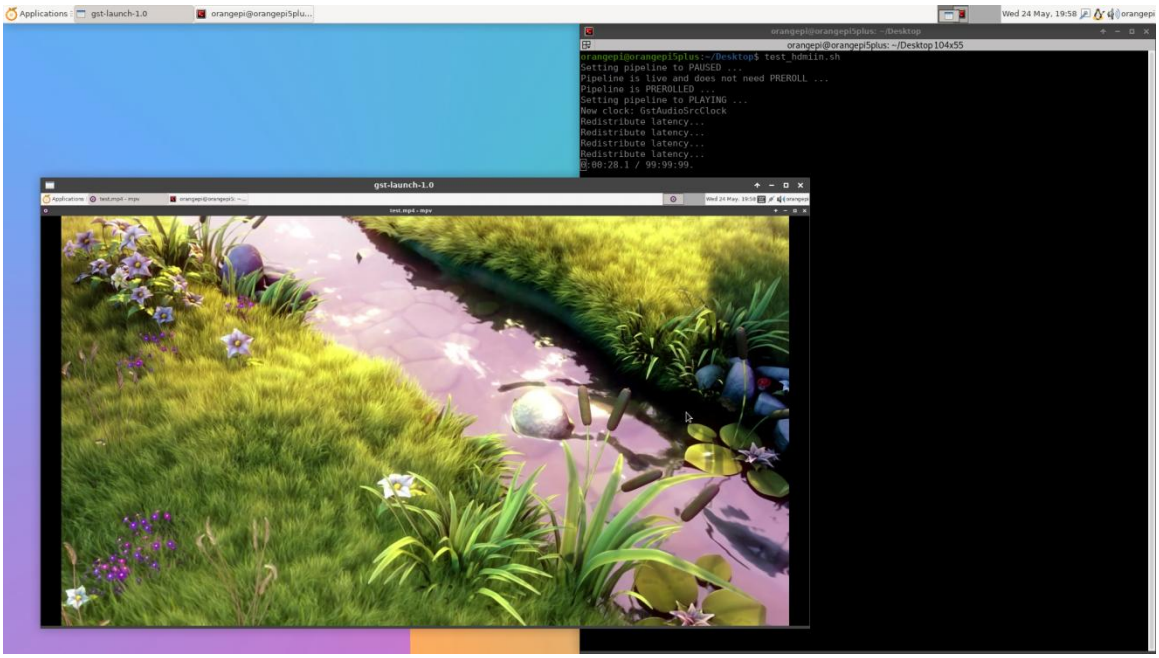
2) First, use the HDMI to HDMI cable shown in the figure below to connect the HDMI output of other devices to the HDMI RX interface of the development board.



3) Then open a terminal on the desktop and run the **test_hdmiin.sh** script

```
[orangepi@orangepi ~]$ sudo test_hdmiin.sh
```

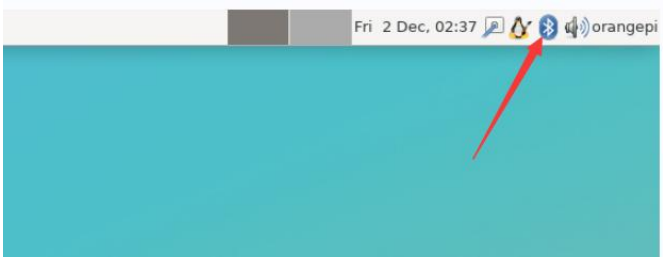
4) Then you can see the input screen of HDMI RX (the HDMI RX in the figure below shows the HDMI output screen of the opi5 development board, and a video is playing at this time). By default, the **test_hdmiin.sh** script will play the audio input from HDMI RX to the HDMI_TX of the development board and the headphones.



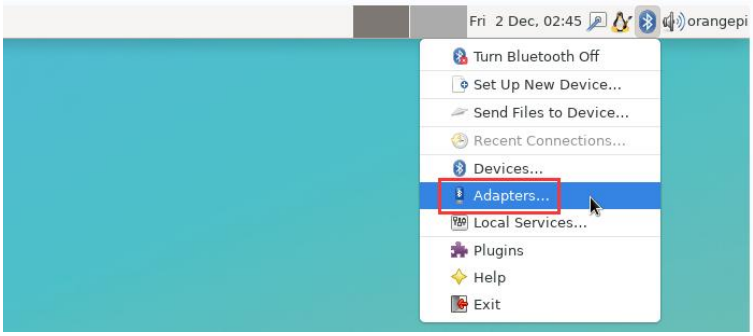
3. 12. How to use Bluetooth

3. 12. 1. Testing methods for desktop images

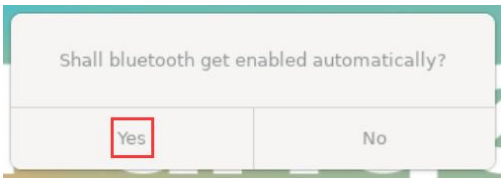
1) Click the Bluetooth icon in the upper right corner of the desktop.



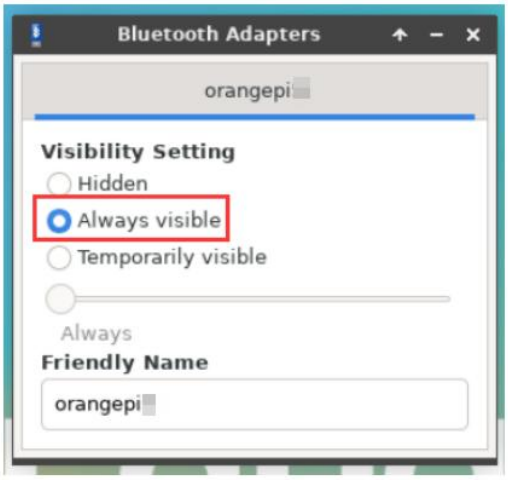
2) Then select the adapter.



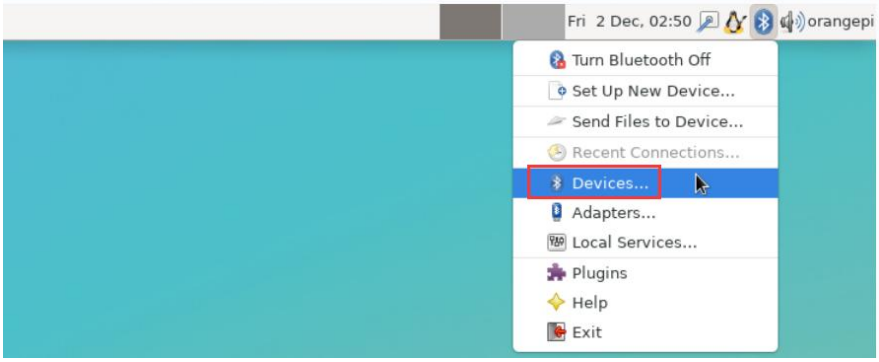
3) If the following interface is displayed, select **Yes.**



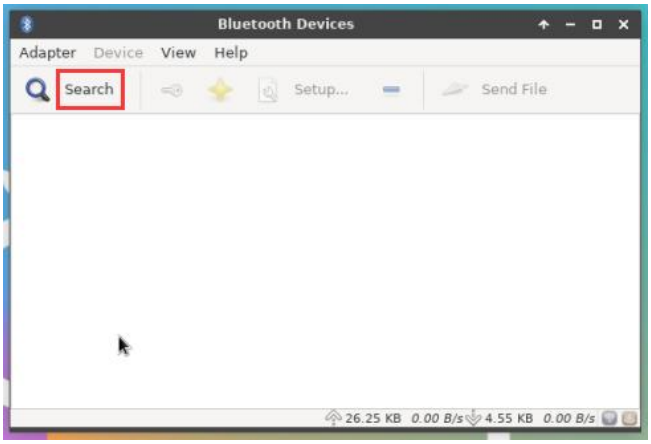
4) Then set the **Visibility Setting** to **Always visible** in the Bluetooth adapter settings interface, and then turn it off.



5) Then open the configuration interface of the Bluetooth device.

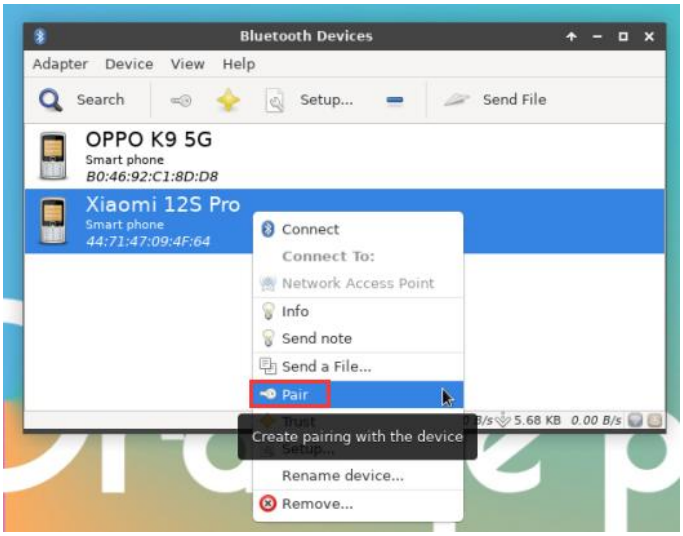


6) Click **Search** to start scanning for surrounding Bluetooth devices.



6) Then select the Bluetooth device you want to connect to, and right-click the mouse to pop up the operation interface for this Bluetooth device. Select **Pair** to start pairing. The

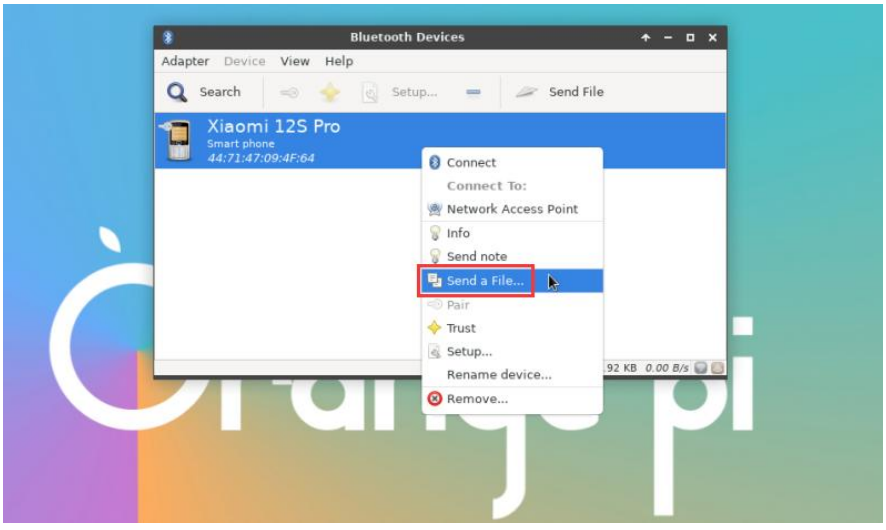
demonstration here is pairing with an Android phone.



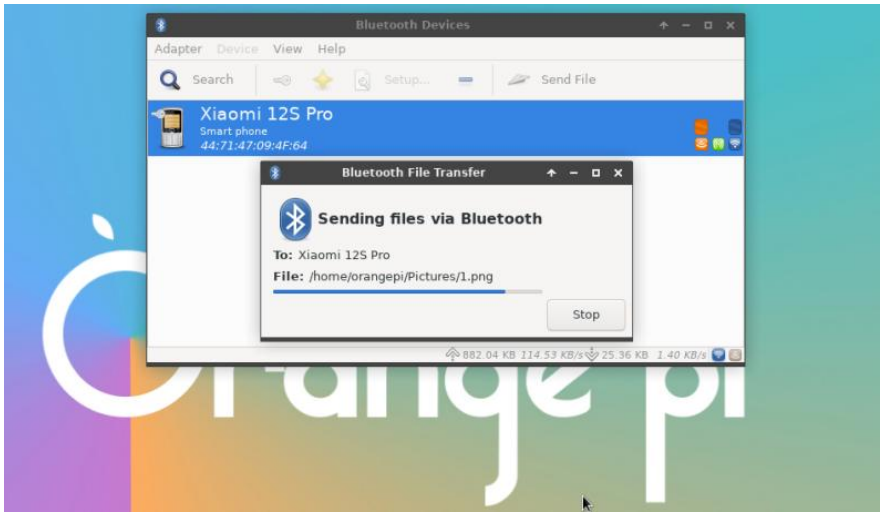
7) During pairing, a pairing confirmation box will pop up in the upper right corner of the desktop. Select **Confirm**. Confirmation is also required on the phone.



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



9) The interface for sending pictures is as follows:



3. 13. USB interface test

The USB port can be connected to a USB hub to expand the number of USB ports.

3. 13. 1. Test by connecting USB mouse or keyboard

1) Insert the USB keyboard into the USB port of the Orange Pi development board.

2) Connect the Orange Pi development board to the HDMI display

3) If the mouse or keyboard can operate the system normally, it means that the USB interface is working properly (the mouse can only be used in the desktop version of the system)

3. 13. 2. Test by connecting USB storage device

1) First, insert the USB flash drive or USB mobile hard disk into the USB port of the Orange Pi development board.

2) Execute the following command. If you can see the output of sdX, it means that the USB disk has been 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 USB drive to **/mnt**, and then you can view the files in the USB drive.

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

4) After mounting, you can use the **df -h** command to view the capacity usage and mount point of the USB drive.

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

3.13.3. USB wireless network card test

The currently tested USB wireless network cards are as follows. Please test other models of USB wireless network cards by yourself. If they cannot be used, you need to transplant the corresponding USB wireless network card driver

Serial number	Model	
1	RTL8723BU Support 2.4G WIFI+BT4.0	
2	RTL8811 Support 2.4G +5G WIFI	
3	RTL8821CU Support 2.4G +5G WIFI Support BT 4.2	

3.13.3.1. RTL8723BU test

1) First, insert the RTL8723BU wireless network card module into the USB port of the development board.

2) Then the Linux system will automatically load the RTL8723BU Bluetooth and WIFI related kernel modules. Through the lsmod command, you can see that the following kernel modules have been automatically loaded

```
orangepi@orangepi:~$ lsmod
Module                  Size  Used by
rfcomm                 57344  16
rtl8xxxu              106496  0
rtk_btusb              61440  0
```

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

```
orangepi@orangepi:~$ dmesg
......
[    83.438901] usb 2-1: new high-speed USB device number 2 using ehci-platform
[    83.588375] usb 2-1: New USB device found, idVendor=0bda, idProduct=b720, bcdDevice= 2.00
[    83.588403] usb 2-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[    83.588422] usb 2-1: Product: 802.11n WLAN Adapter
[    83.588443] usb 2-1: Manufacturer: Realtek
[    83.588460] usb 2-1: SerialNumber: 00e04c000001
[    83.601974] Bluetooth: hci0: RTL: examining hci_ver=06 hci_rev=000b lmp_ver=06 lmp_subver=8723
[    83.603894] Bluetooth: hci0: RTL: rom_version status=0 version=1
[    83.603920] Bluetooth: hci0: RTL: loading rtl_bt/rtl8723b_fw.bin
[    83.610108] Bluetooth: hci0: RTL: loading rtl_bt/rtl8723b_config.bin
[    83.611274] Bluetooth: hci0: RTL: cfg_sz 68, total sz 22564
[    83.658494] rtk_btusb: Realtek Bluetooth USB driver ver 3.1.6d45ddf.20220519-142432
[    83.658651] usbcore: registered new interface driver rtk_btusb
[    83.667124] usb 2-1: This Realtek USB WiFi dongle (0x0bda:0xb720) is untested!
[    83.667137] usb 2-1: Please report results to Jes.Sorensen@gmail.com
[    83.890140] usb 2-1: Vendor: Realtek
[    83.890153] usb 2-1: Product: 802.11n WLAN Adapter
[    83.890159] usb 2-1: rtl8723bu_parse_efuse: dumping efuse (0x200 bytes):
......
[    83.890412] usb 2-1: RTL8723BU rev E (SMIC) 1T1R, TX queues 3, WiFi=1, BT=1,
```

```
GPS=0, HI PA=0
[    83.890417] usb 2-1: RTL8723BU MAC: 00:13:ef:f4:58:ae
[    83.890421] usb 2-1: rtl8xxxu: Loading firmware rtlwifi/rtl8723bu_nic.bin
[    83.895289] usb 2-1: Firmware revision 35.0 (signature 0x5301)
[    84.050893] Bluetooth: hci0: RTL: fw version 0x0e2f9f73
[    84.266905] Bluetooth: RFCOMM TTY layer initialized
[    84.266949] Bluetooth: RFCOMM socket layer initialized
[    84.266999] Bluetooth: RFCOMM ver 1.11
[    84.884270] usbcore: registered new interface driver rtl8xxxu
[    84.912046] rtl8xxxu 2-1:1.2 wlx0013eff458ae: renamed from wlan0
```

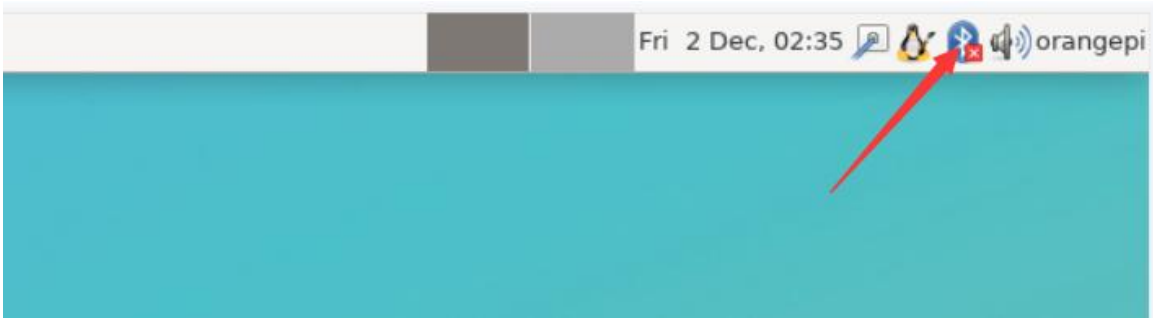
4) Then use the **sudo ifconfig** command to see the device node of RTL8723BU WIFI. For the connection and test methods of WIFI, **please refer to the WIFI connection test section**.

```
orangepi@orangepi:~$ sudo 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
```

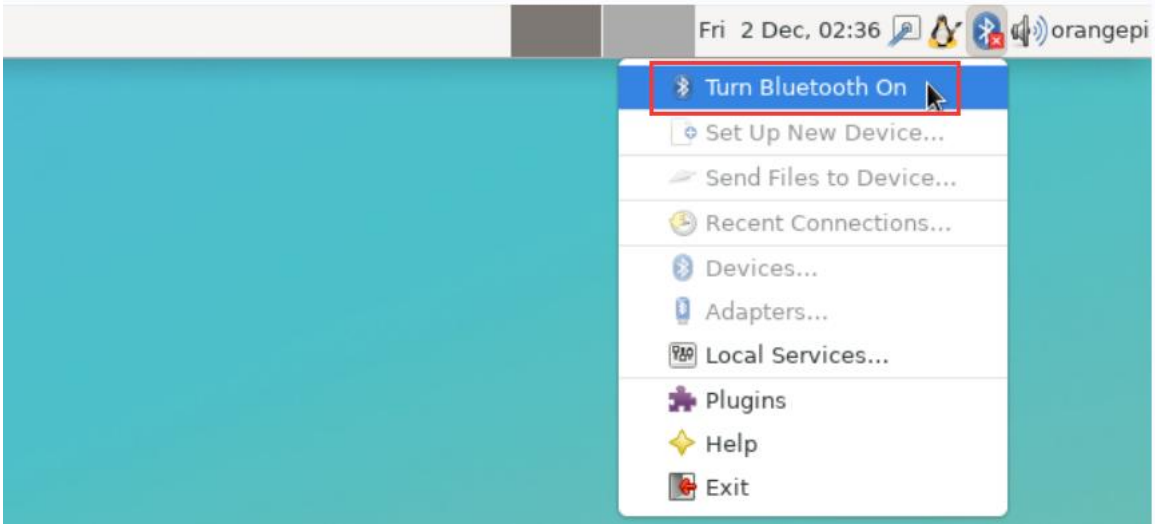
5) Then you can see the USB Bluetooth device through the **hciconfig** command

```
orangepi@orangepi:~$ sudo apt update && sudo apt -y install bluez
orangepi@orangepi:~$ hciconfig
hci0:   Type: Primary  Bus: USB
        BD Address: 00:13:EF:F4:58:AE  ACL MTU: 820:8  SCO MTU: 255:16
        DOWN
        RX bytes:1252 acl:0 sco:0 events:125 errors:0
        TX bytes:23307 acl:0 sco:0 commands:125 errors:0
```

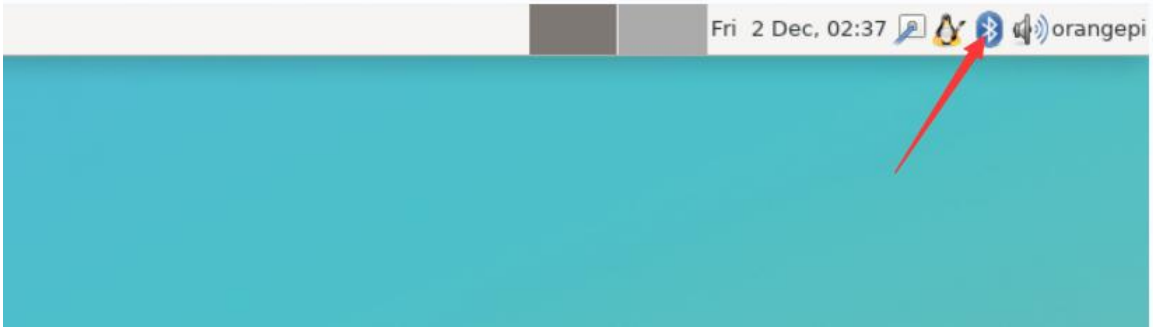
6) You can also see the Bluetooth icon on the desktop. At this time, Bluetooth is not turned on, so a red **x** is displayed.



7) Click **Turn Bluetooth On** to turn on Bluetooth



8) The display after turning on Bluetooth is as follows



9) For the Bluetooth testing method, please refer to the **Bluetooth Usage section**, which will not be repeated here.

3.13.3.2. RTL8811 Test

1) First, insert the RTL8811 wireless network card module into the USB port of the development board.

2) Then the Linux system will automatically load the kernel modules related to RTL8811 WIFI. You can see the following kernel modules have been automatically loaded through the lsmod command:

```
orangepi@orangepi:~$ lsmod
Module                  Size  Used by
8821cu               1839104  0
```

3) You can see the loading information of the RTL8811 module through the dmesg command

```
orangepi@orangepi:~$ dmesg
[  118.618194] usb 2-1: new high-speed USB device number 2 using ehci-platform
[  118.767152] usb 2-1: New USB device found, idVendor=0bda, idProduct=c811, bcdDevice= 2.00
[  118.767181] usb 2-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[  118.767199] usb 2-1: Product: 802.11ac NIC
[  118.767219] usb 2-1: Manufacturer: Realtek
[  118.767235] usb 2-1: SerialNumber: 123456
[  119.500530] usbcore: registered new interface driver rtl8821cu
[  119.525498] rtl8821cu 2-1:1.0 wlx1cbfced9d260: renamed from wlan0
```

4) Then use the **sudo ifconfig** command to see the WIFI device node. For WIFI connection and test methods, please refer to the **WIFI connection test section**. I will not repeat it here.

```
orangepi@orangepi:~$ sudo ifconfig wlx1cbfced9d260
wlx1cbfced9d260: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 1c:bf:ce:d9:d2:60  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
```

3. 13. 3. 3. RTL8821CU test

1) First insert the rtl8821cu wireless network card module into the USB interface of the development board

2) Then use the lsusb command to see the device information of the **lsusb**cu usb wifi module. Please make sure that the USB module is not in Driver CDROM Mode.

```
orangepi@orangepi:~$ lsusb | grep "Realtek"
Bus 002 Device 003: ID 0bda:c820 Realtek Semiconductor Corp. 802.11ac NIC
```

```
orangepi@orangepi:~$ lsusb | grep "Realtek"
Bus 002 Device 002: ID 0bda:1a2b Realtek Semiconductor Corp. RTL8188GU 802.11n WLAN Adapter (Driver CDROM Mode)
```

If the USB WIFI module is in Driver CDROM Mode as seen by the lsusb command, please unplug and re-plug the USB WIFI module. If it still doesn't work, please manually execute the following command to switch the mode:

```
orangepi@orangepi:~$ sudo usb_modeswitch -KW -v 0bda -p 1a2b
```

3) The Linux system will automatically load the rtl8821cu Bluetooth and WiFi related kernel modules. You can see the following kernel modules have been automatically loaded through the lsmod command:

```
orangepi@orangepi:~$ lsmod
Module                  Size  Used by
8821cu                1839104  0
rtk_btusb               61440  0
```

4) You can see the loading information of the rtl8821cu module through the dmesg command

```
orangepi@orangepi:~$ dmesg
......
[    57.083693] usb 2-1: new high-speed USB device number 2 using ehci-platform
[    57.231888] usb 2-1: New USB device found, idVendor=0bda, idProduct=1a2b, bcdDevice= 2.00
```

```
[    57.231916] usb 2-1: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[    57.231937] usb 2-1: Product: DISK
[    57.231956] usb 2-1: Manufacturer: Realtek
[    57.242594] usb-storage 2-1:1.0: USB Mass Storage device detected
[    57.245674] scsi host0: usb-storage 2-1:1.0
[    58.069172] usb 2-1: USB disconnect, device number 2
[    58.440025] usb 2-1: new high-speed USB device number 3 using ehci-platform
[    58.587819] usb 2-1: New USB device found, idVendor=0bda, idProduct=c820, bcdDevice= 2.00
[    58.587827] usb 2-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[    58.587833] usb 2-1: Product: 802.11ac NIC
[    58.587838] usb 2-1: Manufacturer: Realtek
[    58.587844] usb 2-1: SerialNumber: 123456
[    58.610463] rtk_btusb: Realtek Bluetooth USB driver ver 3.1.6d45ddf.20220519-142432
[    58.610656] usbcore: registered new interface driver rtk_btusb
[    58.634631] Bluetooth: hci0: RTL: examining hci_ver=08 hci_rev=000c lmp_ver=08 lmp_subver=8821
[    58.636729] Bluetooth: hci0: RTL: rom_version status=0 version=1
[    58.636740] Bluetooth: hci0: RTL: loading rtl_bt/rtl8821c_fw.bin
[    58.664190] Bluetooth: hci0: RTL: loading rtl_bt/rtl8821c_config.bin
[    58.664746] Bluetooth: hci0: RTL: cfg_sz 10, total sz 31990
[    59.122471] Bluetooth: hci0: RTL: fw version 0x829a7644
[    59.265513] usbcore: registered new interface driver rtl8821cu
[    59.280119] rtl8821cu 2-1:1.2 wlx90de80521825: renamed from wlan0
```

5) Then use the **sudo ifconfig** command to see the device node of rtl8821cu wifi. For the connection and test methods of wifi, please refer to the **WIFI connection test section**. I will not repeat it here.

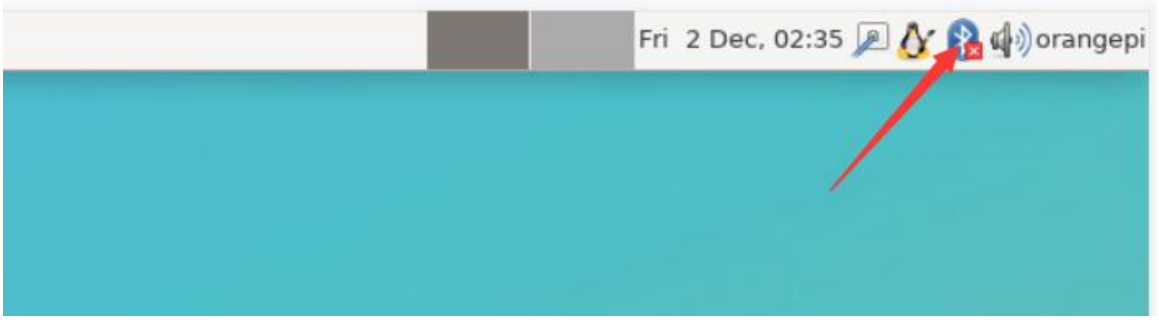
```
orangepi@orangepi:~$ sudo ifconfig wlx90de80521825
wlx90de80521825: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:13:ef:f4:58:ae  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
```

```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

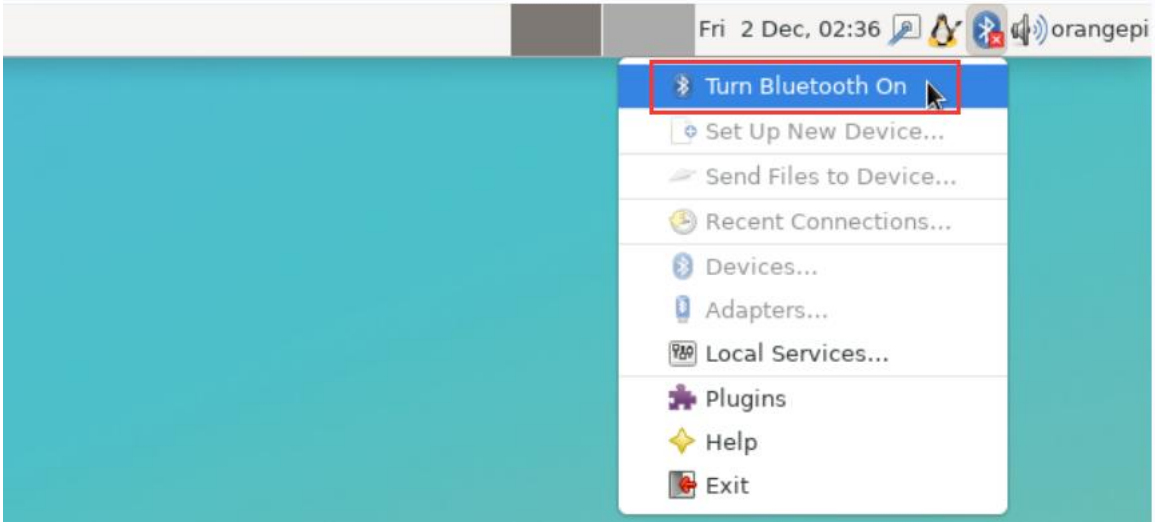
6) Then you can see the USB Bluetooth device through the **hciconfig** command

```
orangepi@orangepi:~$ sudo apt-get update && sudo apt-get install -y bluez
orangepi@orangepi:~$ hciconfig
hci0:   Type: Primary  Bus: USB
        BD Address: 00:13:EF:F4:58:AE  ACL MTU: 820:8  SCO MTU: 255:16
        DOWN
        RX bytes:1252 acl:0 sco:0 events:125 errors:0
        TX bytes:23307 acl:0 sco:0 commands:125 errors:0
```

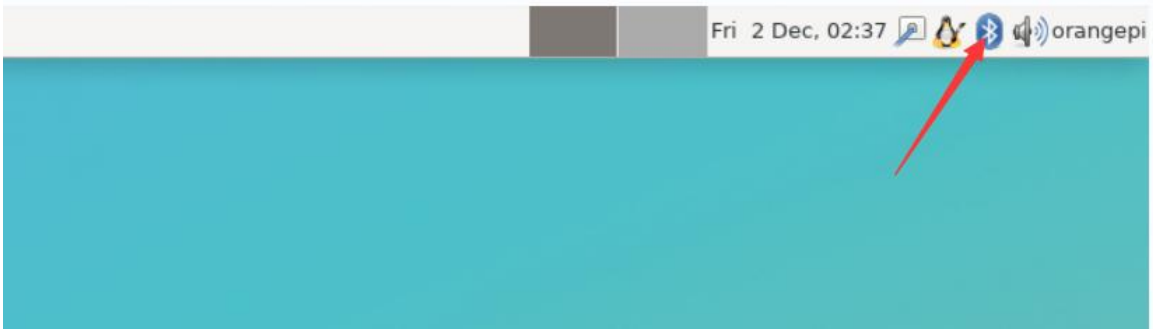
7) You can also see the Bluetooth icon on the desktop. At this time, Bluetooth is not turned on, so a red **x** is displayed.



8) Click **Turn Bluetooth On** to turn on Bluetooth



9) The display after turning on Bluetooth is as follows



10) For the Bluetooth testing method, please refer to the **Bluetooth Usage section**, which will not be repeated here.

3. 13. 4. USB camera test

1) First, you need to prepare a USB camera that supports UVC protocol as shown in the figure below or similar, and then insert the USB camera into the USB port of the Orange Pi development board.



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

```
orangepi@orangepi:~$ v4l2-ctl --list-devices
Q8 HD Webcam: Q8 HD Webcam (usb-fc880000.usb-1):
        /dev/video0
        /dev/video1
        /dev/media0
```

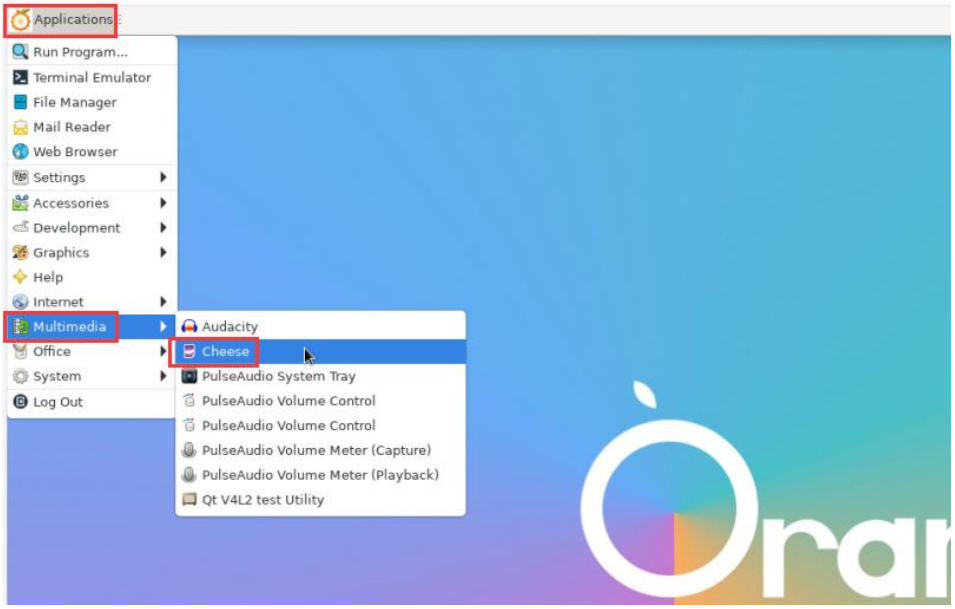
Note that the l in v4l2 is a lowercase letter l, not the number 1.

In addition, the video number is not always video0, please refer to the actual one you see.

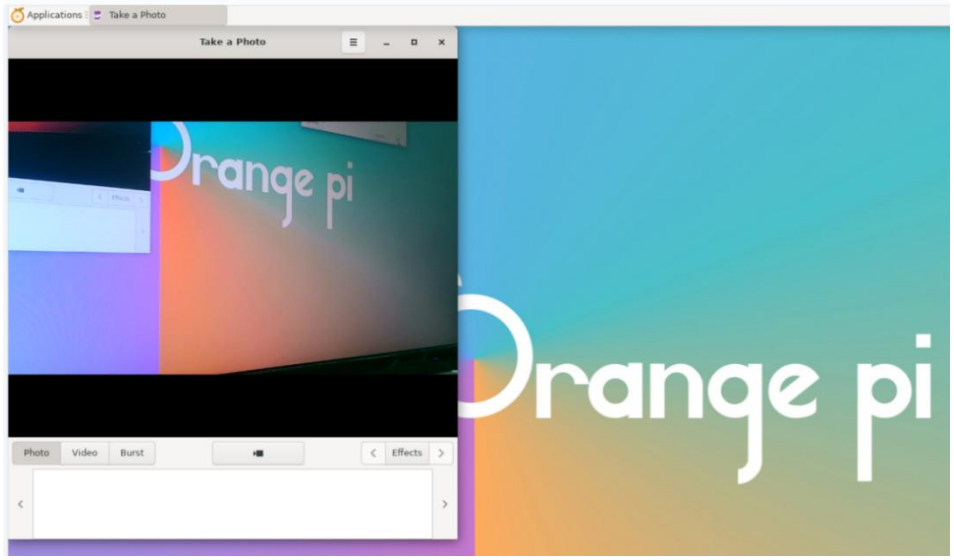
3) In the desktop system, you can use Cheese to directly open the USB camera. The Cheese opening method is shown in the figure below:

If cheese is not pre-installed, you can install it using the following command:

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



The interface after Cheese opens the USB camera is as shown below:



4) How to use fswebcam to test 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 photos
   a) -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 is used to specify the resolution of the photo
   d) -S option is used to set the number of frames to skip in the past
   e) ./image.jpg is used to set the name and path of the generated photo

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

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

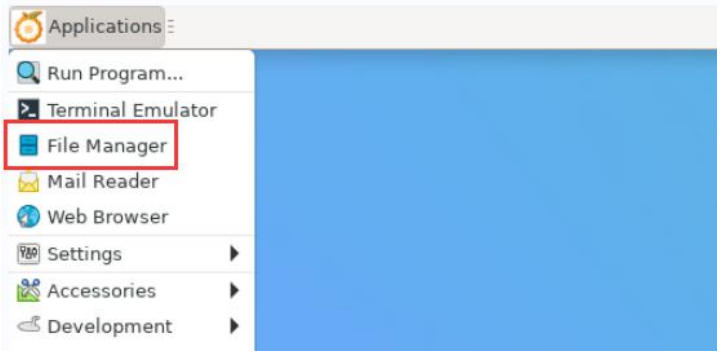
```
orangepi@orangepi:~$ scp image.jpg test@192.168.1.55:/home/test (Modify the IP address and path according to actual situation)
```

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

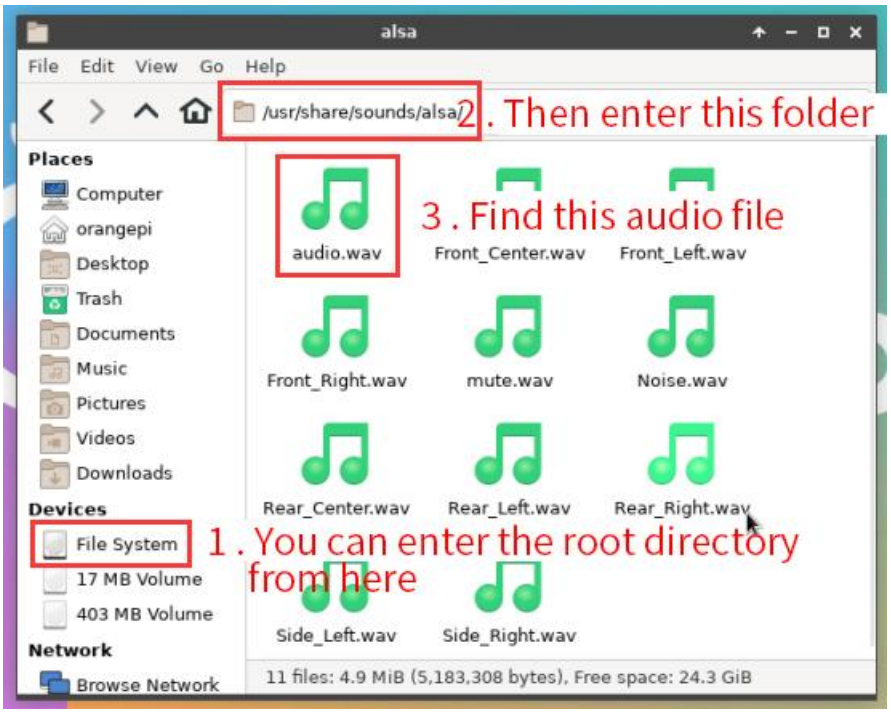
3. 14. Audio Test

3. 14. 1. Testing Audio Methods on Desktop Systems

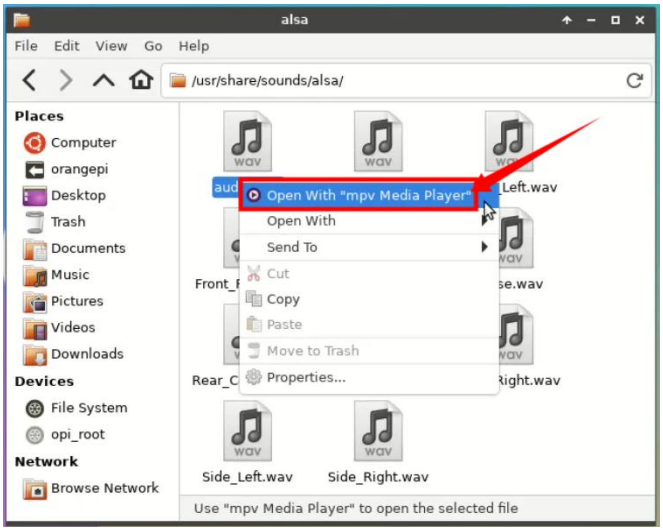
1) First open the file manager.



2) Then find the following file (if there is no such audio file in the system, you can upload an audio file to the system yourself).

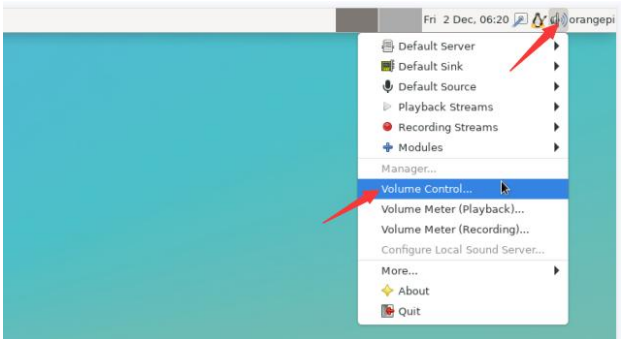


3) Then select the audio.wav file, right-click and choose to open it with vlc or mpv to start playing.

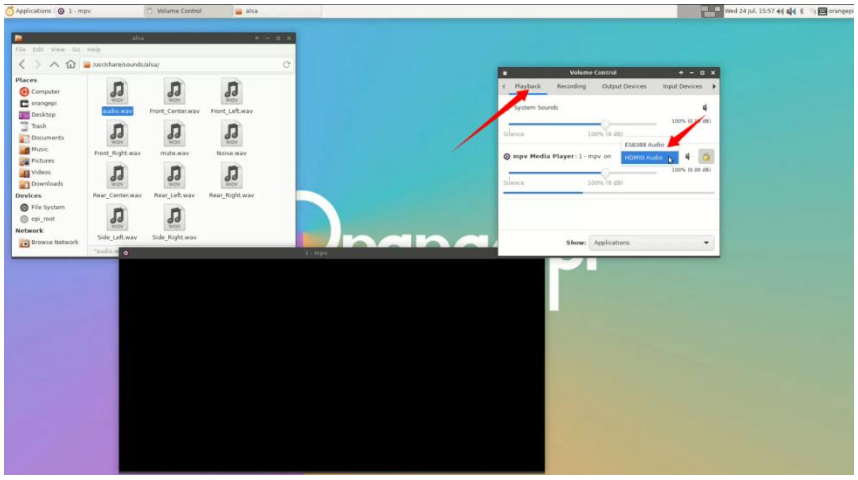


4) Methods for switching between different audio devices such as HDMI playback and headphone playback.

a. First open the volume control interface.



b. When playing audio, the audio device options that the playback software can use will be displayed in **Playback**, as shown in the figure below. Here you can set which audio device you want to play to.

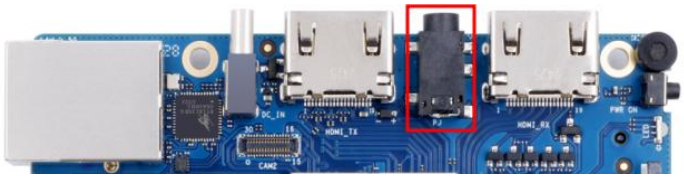


If you don't see the HDMI audio device option, try unplugging the HDMI cable.

3. 14. 2. How to play audio using commands

3. 14. 2. 1. Headphone jack audio playback test

1) First, plug the earphone into the earphone jack of the development board.



2) Then you can use the **aplay -l** command to view the sound card devices supported by the Linux system. From the output below, we can see that **card 2** is the es8388 sound

card device, which is the sound card device of the headset.

```
orangepi@orangepi:~$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: rockchiphdmi1 [rockchip-hdmi1], device 0: rockchip-hdmi1 i2s-hifi-0 [rockchip-hdmi1 i2s-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 2: rockchipes8388 [rockchip,es8388], device 0: dailink-multicodecs ES8323.7-0011-0 [dailink-multicodecs ES8323.7-0011-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

3) Then use the **aplay** command to play the audio file that comes with the system. If the headphones can hear the sound, it means that the hardware can be used normally.

```
orangepi@orangepi:~$ aplay -D hw:2,0 /usr/share/sounds/alsa/audio.wav
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

3.14.2.2. HDMI audio playback test

1) First, use an HDMI to HDMI cable to connect the Orange Pi development board to the TV (other HDMI displays need to ensure that they can play audio)

2) Then check the HDMI sound card serial number. From the output below, we can know that the HDMI TX sound card is **card 0**

```
orangepi@orangepi:~$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: rockchiphdmi1 [rockchip-hdmi1], device 0: rockchip-hdmi1 i2s-hifi-0 [rockchip-hdmi1 i2s-hifi-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
.....
```

3) Then use the **aplay** command to play the audio file that comes with the system. If the HDMI display or TV can hear the sound, it means that the hardware can be used normally.

```
orangepi@orangepi:~$ aplay -D hw:0,0 /usr/share/sounds/alsa/audio.wav
```

3.14.3. How to test recording using commands

1) There is an onboard MIC on the development board, the location is as follows:



2) Run the **test_record.sh main** command to record an audio clip through the onboard MIC and then play it to the HDMI and headphones.

```
orangepi@orangepi:~$ test_record.sh main
Start recording: /tmp/test.wav
Recording WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Start playing
Playing WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Playing WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

3) In addition to the onboard MIC, we can also record audio through headphones with MIC function. After inserting the headphones with MIC function into the development board, running the **test_record.sh headset** command will record an audio through the headphones and then play it to HDMI and headphones.

```
orangepi@orangepi:~$ test_record.sh headset
Start recording: /tmp/test.wav
Recording WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Start playing
Playing WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
Playing WAVE '/tmp/test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

3.15. Temperature sensor

1) The command to view the system temperature sensor is:

```
orangepi@orangepi:~$ sensors
gpu_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C
```

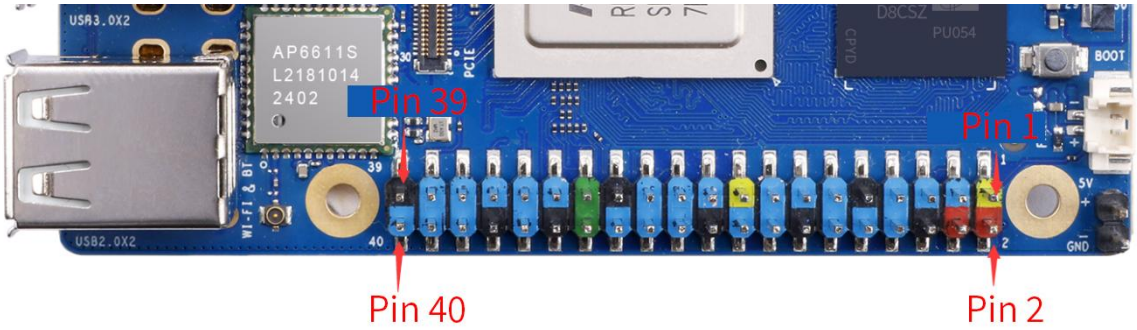
```
littlecore_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C

bigcore0_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C

tcpm_source_psy_6_0022-i2c-6-22
Adapter: rk3x-i2c
in0:             0.00 V  (min =  +0.00 V, max =  +0.00 V)
curr1:           0.00 A  (max =  +0.00 A)

npu_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C

center_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C

bigcore1_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C

soc_thermal-virtual-0
Adapter: Virtual device
temp1:          +47.2°C  (crit = +115.0°C)
```

2) The command to check the current temperature of the nvme ssd solid state drive is:

```
orangepi@orangepi:~$ sudo smartctl -a /dev/nvme0 | grep "Temperature:"
Temperature:                        40 Celsius
```

3. 16. 40 Pin Interface Pin Description

1) Please refer to the following figure for the order of the 40 pin interface pins of the Orange Pi 5 Ultra development board



2) The functions of the 40 pin interface pins of the Orange Pi 5 Ultra development board are shown in the following table

a. Below is the complete pin diagram of 40pin

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

b. The table below is a picture of the left half of the complete table above, which can be seen more clearly

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号
			3.3V		1
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7
			GND		9
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15
			3.3V		17
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19
		SPI0_MISO_M2	GPIO1_B1	41	21
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23
			GND		25
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27
		SPI1_CLK_M1	GPIO3_C1	113	29
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35
I2C5_SDA_M2			GPIO4_A7	135	37
			GND		39

c. The table below is the right half of the complete table above, which can be seen more clearly

引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
2		5V			
4		5V			
6		GND			
8	13	GPIO0_B5	UART2_TX_M0		
10	14	GPIO0_B6	UART2_RX_M0		
12	134	GPIO4_A6			I2C5_SCL_M2
14		GND			
16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
18	36	GPIO1_A4			
20		GND			
22	40	GPIO1_B0			
24	44	GPIO1_B4	SPI0_CS0_M2		
26	45	GPIO1_B5	SPI0_CS1_M2		
28	144	GPIO4_C0	I2C8_SCL_M3		
30		GND			
32	139	GPIO4_B3			
34		GND			
36	143	GPIO4_B7			
38	112	GPIO3_C0	SPI1_MISO_M1		
40	111	GPIO3_B7	SPI1_MOSI_M1		

In the table above, the base addresses of the corresponding registers are marked for pwm, which is useful for checking which pwmchip in /sys/class/pwm/ corresponds to which pwm pin in the 40-pin header.

3) There are a total of **28** GPIO ports in the 40pin interface, and the voltage of all GPIO ports is **3.3v**

3. 17. How to install wiringOP

Note that wiringOP is pre-installed in the Linux image released by Orange Pi. Unless the wiringOP code is updated, you do not need to download, compile and install it again. You can use it directly.

The storage path of the compiled wiringOP deb package in orangepi-build is:
orangepi-build/external/cache/debs/arm64/wiringpi_x.xx.deb

After entering the system, you can run the gpio readall command. If you can see the following output, it means wiringOP has been pre-installed and can be used normally.

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |     IN | 1 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
 |   32 |   5 |    RXD.6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO4_A6 | 6   | 134  |
 |   33 |   7 |    TXD.6 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |   34 |   8 | GPIO1_A2 |     IN | 0 | 15 || 16 | 0 | IN     | GPIO1_A3 | 9   | 35   |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | IN     | GPIO1_A4 | 10  | 36   |
 |   42 |  11 | SPI0_TXD |     IN | 0 | 19 || 20 |   |        | GND      |     |      |
 |   41 |  12 | SPI0_RXD |     IN | 0 | 21 || 22 | 1 | IN     | GPIO1_B0 | 13  | 40   |
 |   43 |  14 | SPI0_CLK |     IN | 0 | 23 || 24 | 1 | IN     | SPI0_CS0 | 15  | 44   |
 |      |     |      GND |        |   | 25 || 26 | 1 | IN     | SPI0_CS1 | 16  | 45   |
 |  145 |  17 |    SDA.8 |     IN | 0 | 27 || 28 | 1 | IN     | SCL.8    | 18  | 144  |
 |  113 |  19 | GPIO3_C1 |     IN | 1 | 29 || 30 |   |        | GND      |     |      |
 |  109 |  20 |  CAN1_RX |     IN | 1 | 31 || 32 | 1 | IN     | GPIO4_B3 | 21  | 139  |
 |  110 |  22 |  CAN1_TX |     IN | 1 | 33 || 34 |   |        | GND      |     |      |
 |  114 |  23 | GPIO3_C2 |     IN | 1 | 35 || 36 | 1 | IN     | GPIO4_B7 | 24  | 143  |
 |  135 |  25 | GPIO4_A7 |     IN | 0 | 37 || 38 | 1 | IN     | GPIO3_C0 | 26  | 112  |
 |      |     |      GND |        |   | 39 || 40 | 1 | IN     | GPIO3_B7 | 27  | 111  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
orangepi@orangepi5ultra:~$
```

1) Download the wiringOP code

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y git
orangepi@orangepi:~$ git clone https://github.com/orangepi-xunlong/wiringOP.git -b next
```

Note that Orange Pi 5 Ultra needs to download the wiringOP next branch code, please do not miss the -b next parameter.

If you have problems downloading the code from GitHub, you can directly use the wiringOP source code that comes with the Linux image, which is stored in /usr/src/wiringOP.

2) Compile and install wiringOP

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

3) Test the output of the gpio readall command as follows

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |     IN | 1 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
 |   32 |   5 |    RXD.6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO4_A6 | 6   | 134  |
 |   33 |   7 |    TXD.6 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |   34 |   8 | GPIO1_A2 |     IN | 0 | 15 || 16 | 0 | IN     | GPIO1_A3 | 9   | 35   |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | IN     | GPIO1_A4 | 10  | 36   |
 |   42 |  11 | SPI0_TXD |     IN | 0 | 19 || 20 |   |        | GND      |     |      |
 |   41 |  12 | SPI0_RXD |     IN | 0 | 21 || 22 | 1 | IN     | GPIO1_B0 | 13  | 40   |
 |   43 |  14 | SPI0_CLK |     IN | 0 | 23 || 24 | 1 | IN     | SPI0_CS0 | 15  | 44   |
 |      |     |      GND |        |   | 25 || 26 | 1 | IN     | SPI0_CS1 | 16  | 45   |
 |  145 |  17 |    SDA.8 |     IN | 0 | 27 || 28 | 1 | IN     | SCL.8    | 18  | 144  |
 |  113 |  19 | GPIO3_C1 |     IN | 1 | 29 || 30 |   |        | GND      |     |      |
 |  109 |  20 |  CAN1_RX |     IN | 1 | 31 || 32 | 1 | IN     | GPIO4_B3 | 21  | 139  |
 |  110 |  22 |  CAN1_TX |     IN | 1 | 33 || 34 |   |        | GND      |     |      |
 |  114 |  23 | GPIO3_C2 |     IN | 1 | 35 || 36 | 1 | IN     | GPIO4_B7 | 24  | 143  |
 |  135 |  25 | GPIO4_A7 |     IN | 0 | 37 || 38 | 1 | IN     | GPIO3_C0 | 26  | 112  |
 |      |     |      GND |        |   | 39 || 40 | 1 | IN     | GPIO3_B7 | 27  | 111  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
orangepi@orangepi5ultra:~$
```

3. 18. 40pin interface GPIO, I2C, UART, SPI, CAN and PWM test

3. 18. 1. 40pin GPIO port test

The Linux system released by Orange Pi has a pre-installed blink_all_gpio program, which will set all 28 GPIO ports in the 40-pin to switch high and low levels continuously.

After running the blink_all_gpio program, when you use a multimeter to measure the voltage level of the GPIO port, you will find that the GPIO pin will switch between 0 and 3.3v. Using this program, we can test whether the GPIO port can work properly.

The way to run the blink_all_gpio program is as follows:

orangepi@orangepi:~$ **sudo blink_all_gpio** #Remember to add sudo permissions

[sudo] password for orangepi: #You need to enter a password here

1) There are a total of 28 GPIO ports available in the 40 pins of the development board. The following uses pin 7, which corresponds to GPIO GPIO1_A7 and wPi serial number 2, as an example to demonstrate how to set the high and low levels of the GPIO port.

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |     IN | 1 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
```

2) First set the GPIO port to output mode, where the third parameter needs to input the wPi number corresponding to the pin

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

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 set successfully.

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

Using gpio readall, you can see that the value of pin 7 (V) has changed to 0

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |    OUT | 0 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
```

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 set successfully.

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

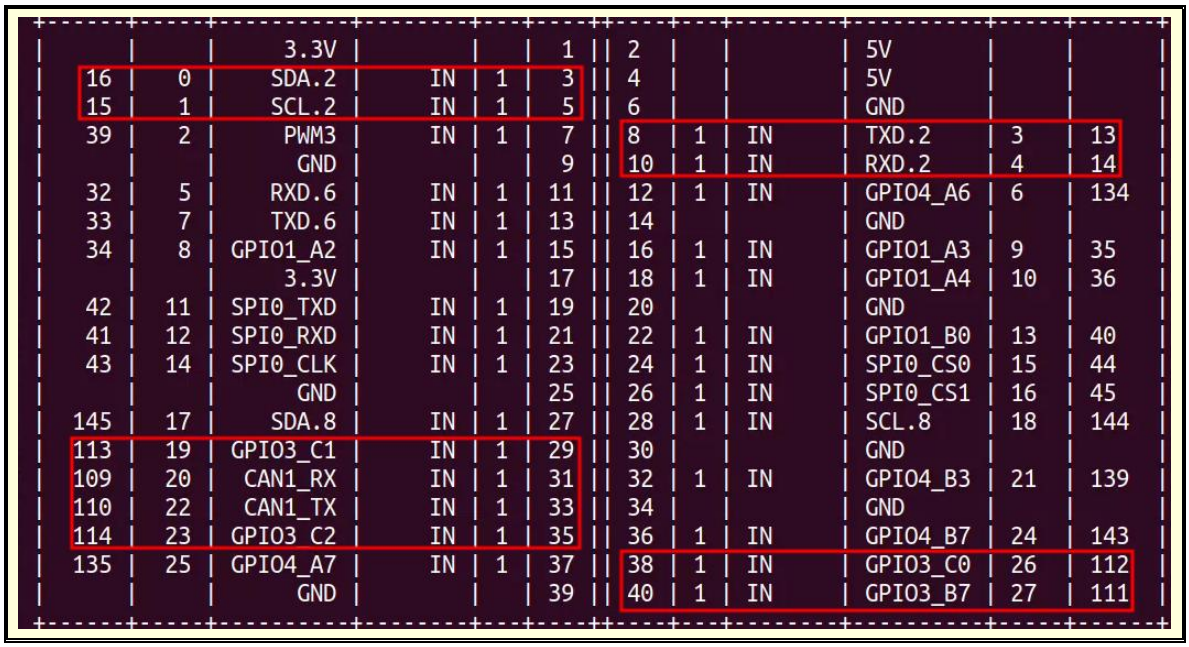
Using gpio readall, you can see that the value of pin 7 (V) has changed to 1

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |    OUT | 1 |  7 || 8  | 0 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
```

5) The setting method of other pins is similar. Just change the serial number of wPi to the serial number corresponding to the pin.

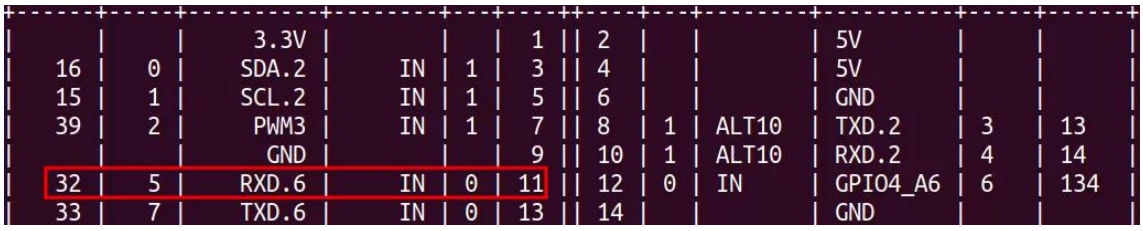
3.18.2. 40 How to set pull-up and pull-down resistors on GPIO pin

Note that the following 10 GPIO pins of Orange Pi 5 Ultra have external 3.3V pull-up, so setting the pull-down is invalid.



		3.3V			1	2			5V		
16	0	SDA.2	IN	1	3	4			5V		
15	1	SCL.2	IN	1	5	6			GND		
39	2	PWM3	IN	1	7	8	1	IN	TXD.2	3	13
		GND			9	10	1	IN	RXD.2	4	14
32	5	RXD.6	IN	1	11	12	1	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	1	13	14			GND		
34	8	GPIO1_A2	IN	1	15	16	1	IN	GPIO1_A3	9	35
		3.3V			17	18	1	IN	GPIO1_A4	10	36
42	11	SPI0_TXD	IN	1	19	20			GND		
41	12	SPI0_RXD	IN	1	21	22	1	IN	GPIO1_B0	13	40
43	14	SPI0_CLK	IN	1	23	24	1	IN	SPI0_CS0	15	44
		GND			25	26	1	IN	SPI0_CS1	16	45
145	17	SDA.8	IN	1	27	28	1	IN	SCL.8	18	144
113	19	GPIO3_C1	IN	1	29	30			GND		
109	20	CAN1_RX	IN	1	31	32	1	IN	GPIO4_B3	21	139
110	22	CAN1_TX	IN	1	33	34			GND		
114	23	GPIO3_C2	IN	1	35	36	1	IN	GPIO4_B7	24	143
135	25	GPIO4_A7	IN	1	37	38	1	IN	GPIO3_C0	26	112
		GND			39	40	1	IN	GPIO3_B7	27	111

1) Below, we take pin 11, which corresponds to GPIO GPIO1_A0 and wPi number 5, as an example to demonstrate how to set the pull-up and pull-down resistors of the GPIO port.



		3.3V			1	2			5V		
16	0	SDA.2	IN	1	3	4			5V		
15	1	SCL.2	IN	1	5	6			GND		
39	2	PWM3	IN	1	7	8	1	ALT10	TXD.2	3	13
		GND			9	10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11	12	0	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	0	13	14			GND		

2) First, you need to set the GPIO port to input mode. The third parameter needs to enter the wPi number corresponding to the pin.

```
root@orangepi:~# gpio mode 5 in
```

3) After setting to input mode, execute the following command to set the GPIO port to pull-up mode

```
root@orangepi:~# gpio mode 5 up
```

4) Then enter the following command to read the level of the GPIO port. If the level is 1, it means that the pull-up mode is set successfully.

```
root@orangepi:~# gpio read 5
1
```

5) Then execute the following command to set the GPIO port to pull-down mode

```
root@orangepi:~# gpio mode 5 down
```

6) Then enter the following command to read the level of the GPIO port. If the level is 0, it means that the pull-down mode is set successfully.

```
root@orangepi:~# gpio read 5
0
```

3. 18. 3. 40pin SPI test

1) As shown in the figure below, the available spis for Orange Pi 5 Ultra are spi0, spi1 and spi4

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of SPI0, SPI1 and SPI4 in 40 pins are shown in the following table.

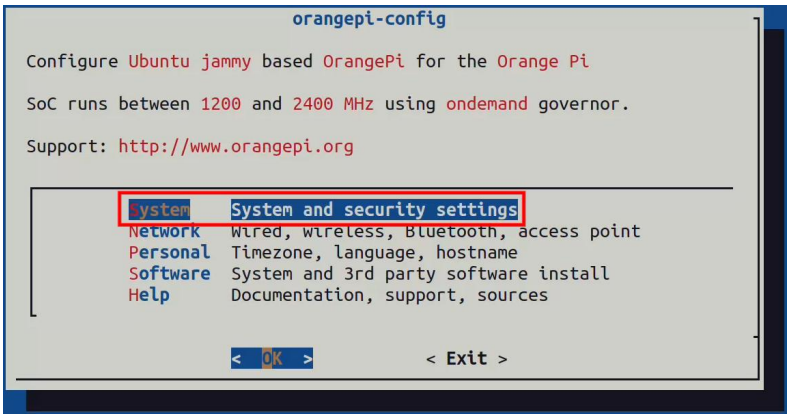
	SPI0_M2 corresponds to 40pin	**SPI1_M1 corresponds to 40pin**	**SPI4_M2 corresponds to 40pin**
MOSI	**Pin 19**	**Pin 40**	**Pin 13**
MISO	**Pin 21**	**Pin 38**	**Pin 14**
CLK	**Pin 23**	**Pin 29**	**Pin 15**
CS0	**Pin 24**	**Pin 1935**	**Pin 16**
CS1	**Pin 26**	**None**	**None**
Dtbo configuration	**spi0-m2-cs0-spidev** **spi0-m2-cs1-spidev** **spi0-m2-cs0-cs1-spidev**	**spi1-m1-cs0-spidev**	**spi4-m2-cs0-spidev**

3) In Linux system, the SPI in 40 pin is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

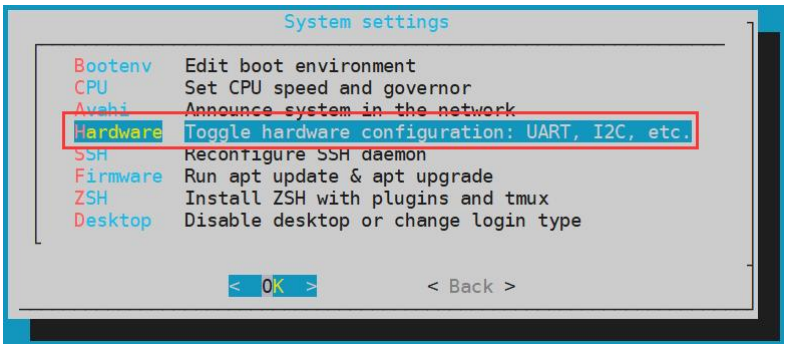
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**



c. Then select **Hardware**



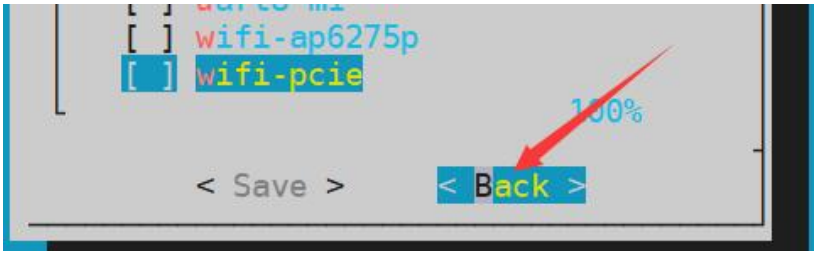
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the SPI configuration you want to open



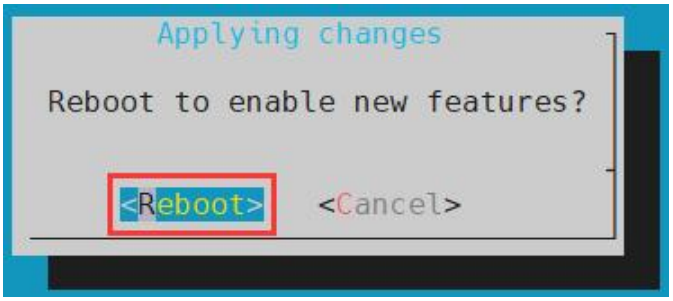
e. Then select **<Save>** to save



f. Then Select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After restarting, enter the system and check whether there is a device node of **spidevx.x** in the Linux system. If it exists, it means that SPI has been set up and can be used directly.

```
orangepi@orangepi:~$ ls /dev/spidev*
/dev/spidev0.0  /dev/spidev0.1  /dev/spidev1.0  /dev/spidev4.0
```

The above is the result after opening spi0-m2-cs0-cs1-spidev, spi1-m1-cs0-spidev and spi4-m2-cs0-spidev.

5) Do not short the mosi and miso pins of SPI0, SPI1 or SPI4. The output of running spidev_test is as follows. You can see that the data of TX and RX are inconsistent.

```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev0.0          #The command for spi0
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev1.0          #The command
```

```
for spi1
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev4.0          #The command
for spi4
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 FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.................▒.
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF    | ................................
```

6) Then short the mosi and miso pins of SPI0 or SPI4 and run spidev_test. The output is as follows. You can see that the data sent and received are the same.

```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev0.0          #The command
for spi0
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev1.0          #The command
for spi1
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev4.0          #The command
for spi4
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 FF FF FF 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 FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.................▒.
```

3. 18. 4. 40pin I2C test

1) As can be seen from the table below, Orange Pi 5 Ultra has four i2c buses: i2c2, i2c4, i2c5 and i2c8

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the 4 groups of I2C buses in 40 pins are shown in the following table. I2C2_M0 and I2C2_M4 can only use one of them at the same time, not both. They are the same I2C, just connected to different pins. Please do not think that they are two different I2C buses.

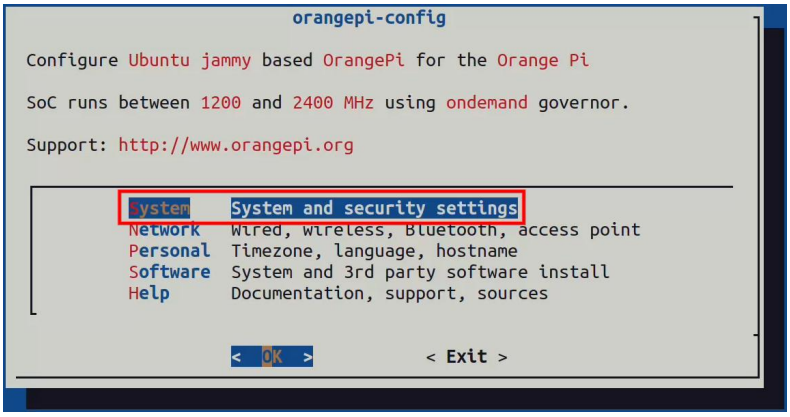
I2C Bus	SDA corresponds to 40pin	SCL corresponds to 40pin	dtbo corresponding configuration
I2C2_M0	**Pin 3**	**Pin 5**	**i2c2-m0**
I2C2_M4	**Pin 11**	**Pin 13**	**i2c2-m4**
I2C4_M3	**Pin 15**	**Pin 16**	**i2c4-m3**
I2C5_M2	**Pin 37**	**Pin 12**	**i2c5-m2**
I2C8_M3	**Pin 27**	**Pin 28**	**i2c8-m3**

3) In Linux system, the I2C bus in 40 pins is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

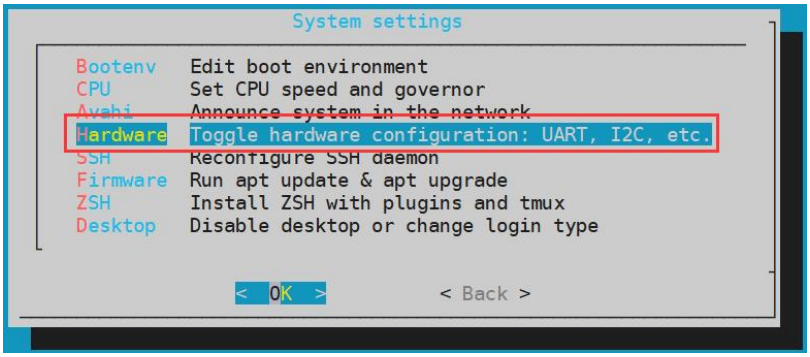
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

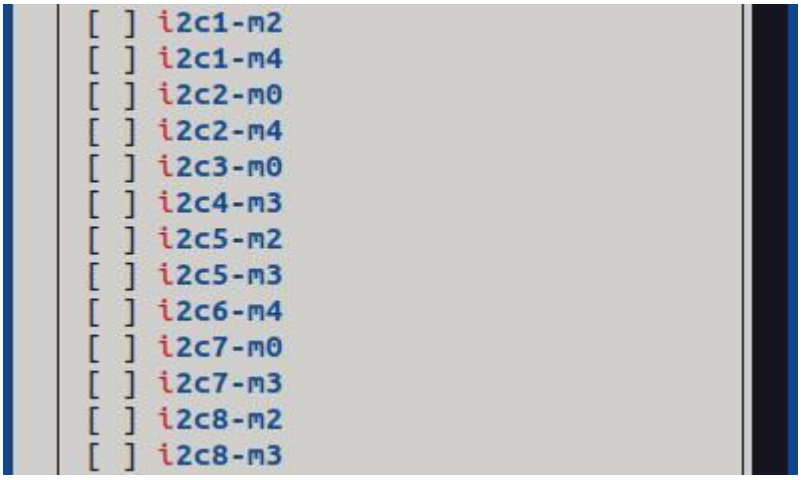
b. Then Select **System**



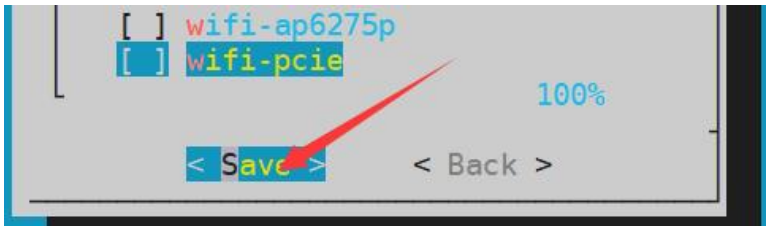
c. Then Select **Hardware**



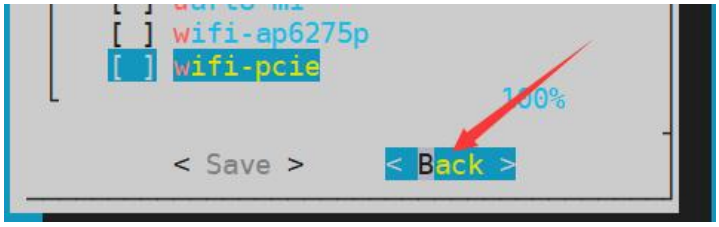
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the I2C configuration you want to open



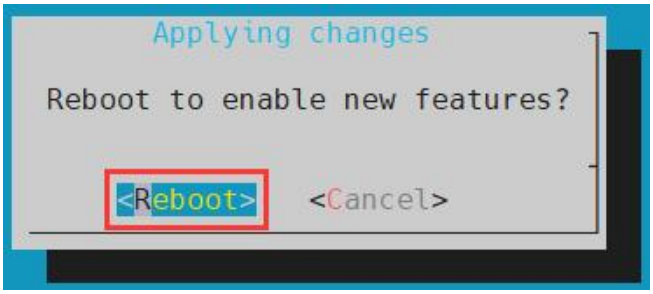
e. Then select **<Save>** to save



f. Then Select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After starting the Linux system, first confirm that there is a device node that needs to use I2C under **/dev**

```
orangepi@orangepi:~$ ls /dev/i2c-*
```

5) Then connect an i2c device to the i2c pin of the 40pin connector

6) Then use the **i2cdetect -y** command. If the address of the connected i2c device can be detected, it means that i2c can be used normally.

```
orangepi@orangepi:~$ sudo i2cdetect -y 2      #i2c2 commands
orangepi@orangepi:~$ sudo i2cdetect -y 4      #i2c4 commands
orangepi@orangepi:~$ sudo i2cdetect -y 5      #i2c5 commands
orangepi@orangepi:~$ sudo i2cdetect -y 8      #i2c8 commands
```

```
root@orangepi:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@orangepi:~#
```

3. 18. 5. 40pin UART test

1) As can be seen from the table below, the available UARTs of Orange Pi 5 Ultra are uart3, uart4 and uart6, a total of three UART buses. Uart2 is used for debugging serial port functions and is not included.

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the four UART bus groups in 40 pins are shown in the following table.

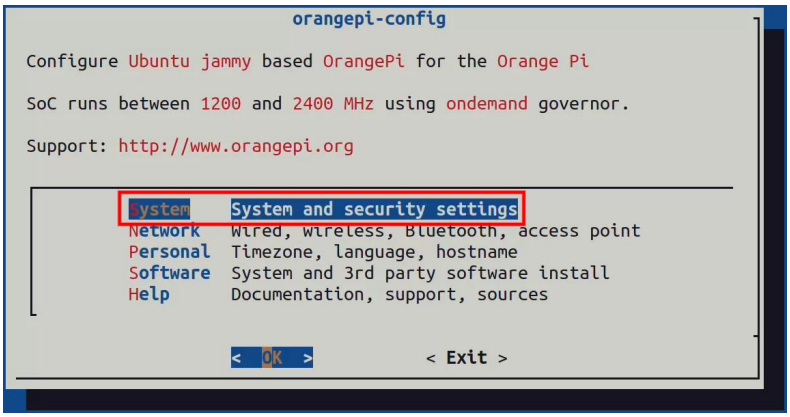
UART Bus	RX corresponds to 40pin	TX corresponds to 40pin	Dtbo corresponding configuration
UART3_M1	**Pin 33**	**Pin 31**	**uart3-m1**
UART4_M2	**Pin 19**	**Pin 23**	**uart4-m2**
UART6_M1	**Pin 11**	**Pin 13**	**uart6-m1**

3) In Linux system, the UART in 40 pin is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

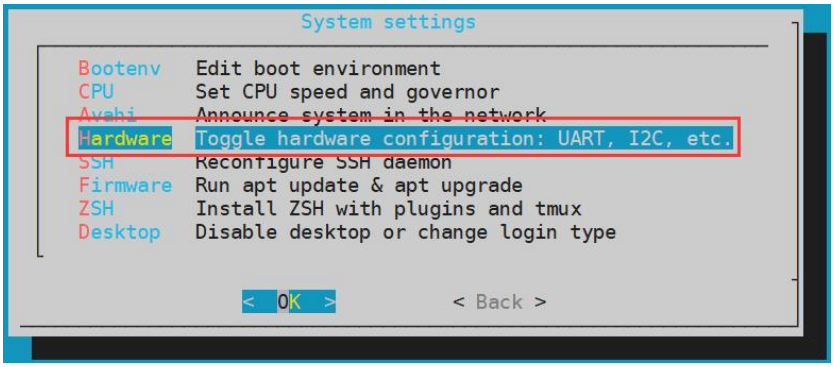
a. Ordinary users should remember to add **sudo** permissions.

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

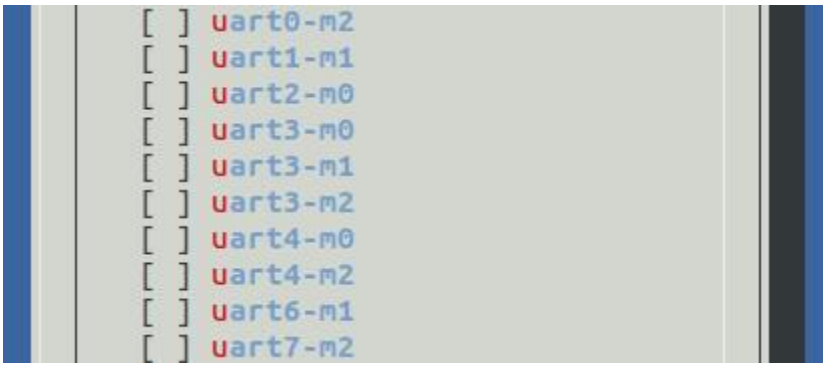
b. Then Select **System**



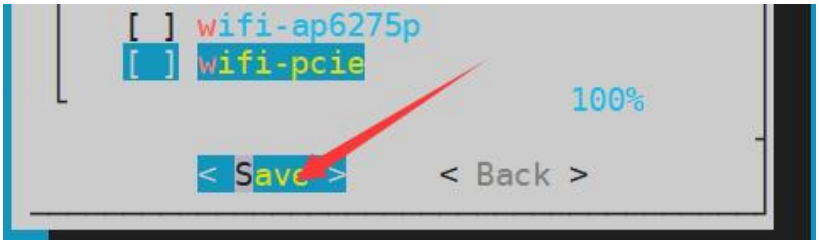
c. Then Select **Hardware**



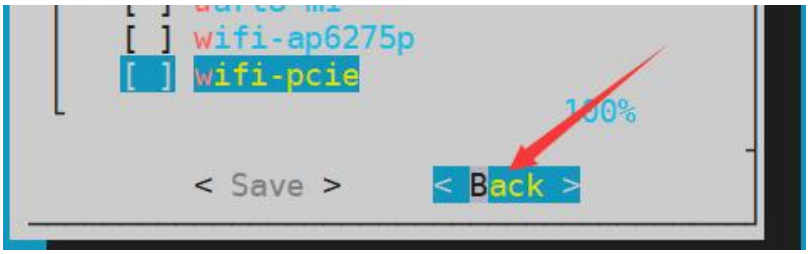
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the UART configuration you want to open



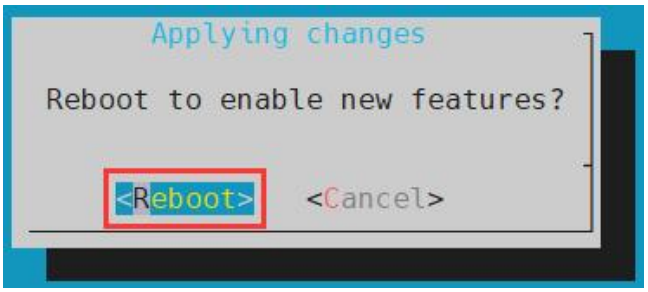
e. Then select **<Save>** to save



f. Then Select**<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After entering the Linux system, first confirm whether there is a device node corresponding to uart under **/dev**

```
orangepi@orangepi:~$ ls /dev/ttyS*
```

5) Then start testing the UART interface. First use the Dupont line to short-circuit the rx and tx of the UART interface to be tested.

6) Use the **gpio serial** command 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 (ttySX needs to be replaced with the corresponding uart node name, please do not copy it)

```
orangepi@orangepi:~$ sudo gpio serial /dev/ttySX
Out:    0:  ->  0
Out:    1:  ->  1
Out:    2:  ->  2
Out:    3:  ->  3
Out:    4:  ->  4
Out:    5:  ->  5^C
```

3. 18. 6. How to test PWM using /sys/class/pwm

1) As can be seen from the table below, Orange Pi 5 Ultra has seven PWM channels: pwm0, pwm1, pwm3, pwm6, pwm12, pwm13 and pwm14

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of PWM in 40pin are shown in the following table. Only one of PWM0_M0 and PWM0_M2, PWM1_M0 and PWM1_M2 can be used at the same time, and they cannot be used at the same time. They are the same PWM, just connected to different pins. Please do not think that they are two different PWM buses.

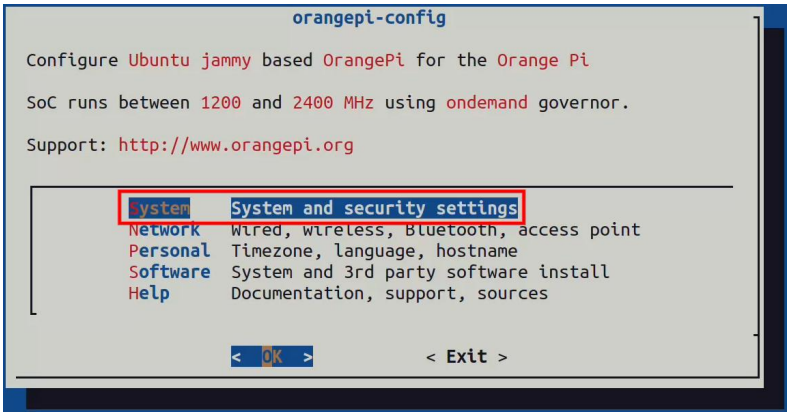
PWM Bus	Corresponding to 40pin	Dtbo corresponding configuration
PWM0_M0	**Pin 5**	**pwm0-m0**
PWM0_M2	**Pin 15**	**pwm0-m2**
PWM1_M0	**Pin 3**	**pwm1-m0**
PWM1_M2	**Pin 16**	**pwm1-m2**
PWM3_M3	**Pin 7**	**pwm3-m3**
PWM6_M1	**Pin 27**	**pwm6-m1**
PWM12_M0	**Pin 31**	**pwm12-m0**
PWM13_M0	**Pin 33**	**pwm13-m0**
PWM14_M0	**Pin 35**	**pwm14-m0**

3) In Linux system, PWM in 40 pin is disabled by default and needs to be enabled manually. The detailed steps are as follows:

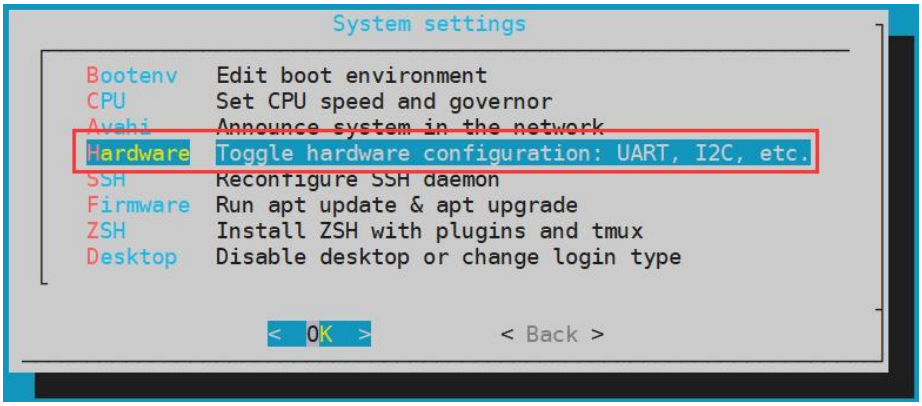
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

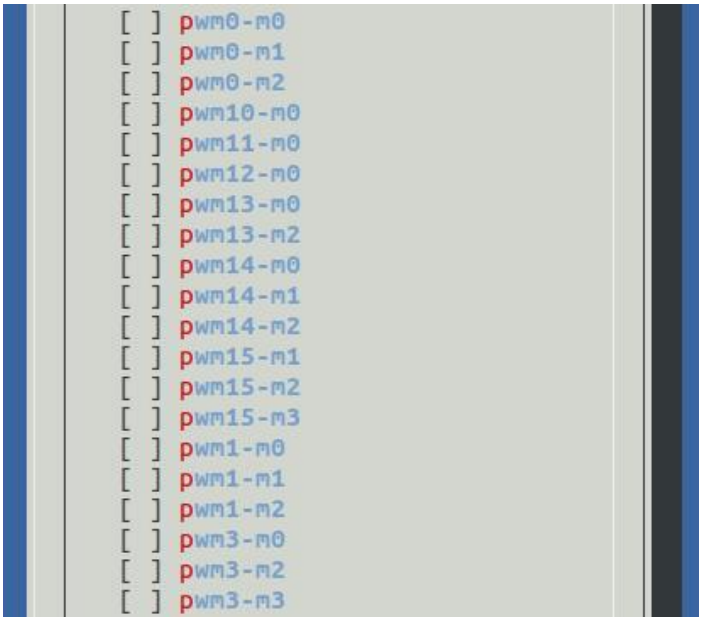
b. Then select **System**



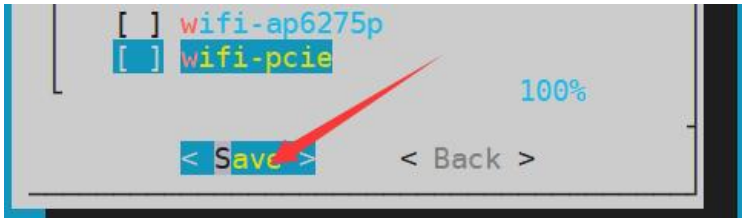
c. Then select **Hardware**



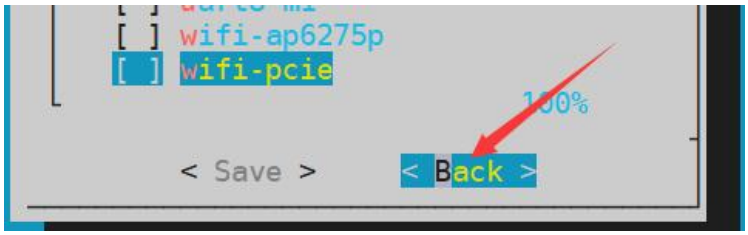
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the PWM configuration you want to open



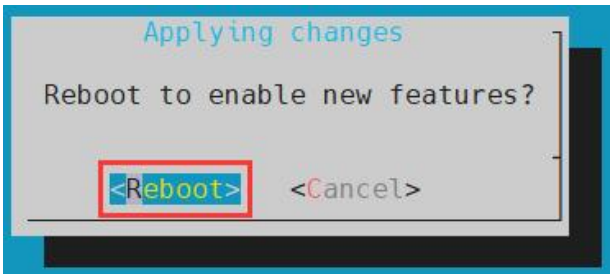
e. Then select **<Save>** to save



f. Then Select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) When a pwm is turned on, there will be an additional pwmchipX in **/sys/class/pwm/** (X is a specific number). For example, after turning on pwm3, the pwmchipX under**/sys/class/pwm/**will change from two to three.

```
orangepi@orangepi:~$ ls /sys/class/pwm/
pwmchip0  pwmchip1  pwmchip2
```

5) Which pwmchip above corresponds to pwm3? Let's first check the output of the **ls /sys/class/pwm/ -l** command, as shown below:

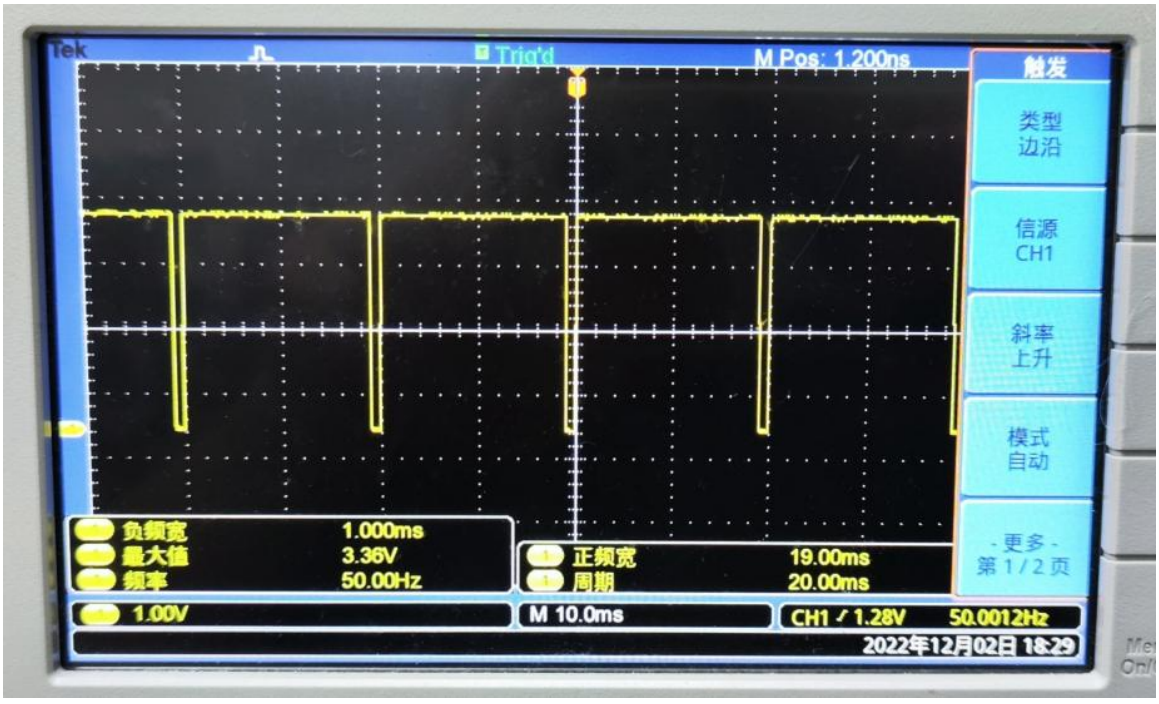
```
root@orangepi:~# ls /sys/class/pwm/ -l
total 0
lrwxrwxrwx 1 root root 0 Aug 20 19:43 pwmchip0 -> ../../devices/platform/fd8b0030.pwm/pwm/pwmchip0
lrwxrwxrwx 1 root root 0 Aug 20 19:43 pwmchip1 -> ../../devices/platform/febe0010.pwm/pwm/pwmchip1
lrwxrwxrwx 1 root root 0 Aug 20 19:43 pwmchip2 -> ../../devices/platform/febf0030.pwm/pwm/pwmchip2
root@orangepi:~#
```

6) Then from the table below, we can see that the base address of the pwm3 register is fd8b0030. Looking at the output of the ls /sys/class/pwm/ -l command, we can see that pwmchip0 is linked to fd8b0030.pwm, so the pwmchip corresponding to pwm3 is pwmchip0.

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号
			3.3V		1
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7
			GND		9
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15
			3.3V		17
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19
		SPI0_MISO_M2	GPIO1_B1	41	21
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23
			GND		25
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27
		SPI1_CLK_M1	GPIO3_C1	113	29
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35
I2C5_SDA_M2			GPIO4_A7	135	37
			GND		39

7) Then use the following command to make pwm3 output a 50Hz square wave (please switch to the root user first, then execute the following command)

```
root@orangepi:~# echo 0 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm0/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
```



8) The pwm3 test method demonstrated above is similar to other pwm test methods.

3. 18. 7. CAN test methods

Only the Linux 5.10 kernel image supports the CAN function, while the Linux 6.1 kernel image does not support it.

3. 18. 7. 1. How to open CAN

1) As can be seen from the table below, the available CAN buses for Orange Pi 5 Ultra are CAN0 and CAN1

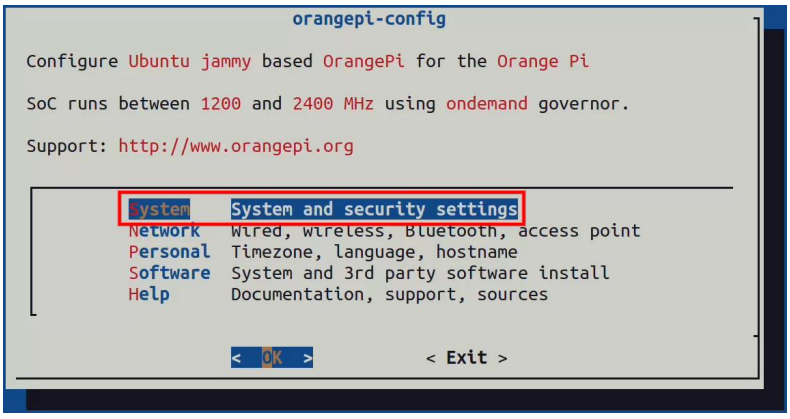
复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) In Linux system, CAN in 40 pin is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

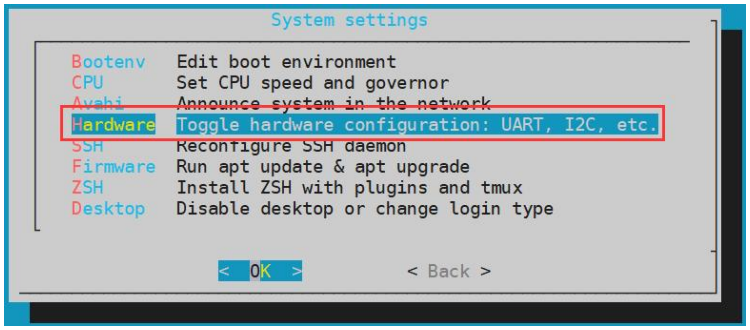
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**



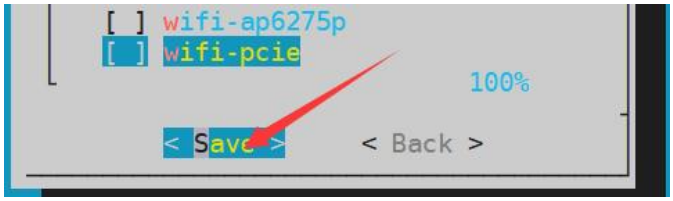
c. Then select **Hardware**



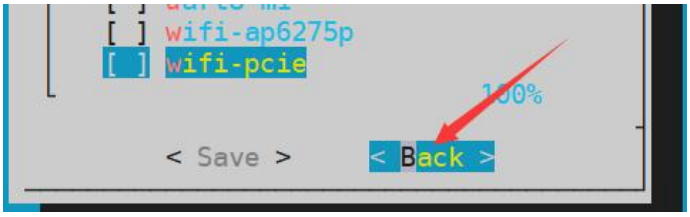
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the **spacebar** to select the configuration you want to open



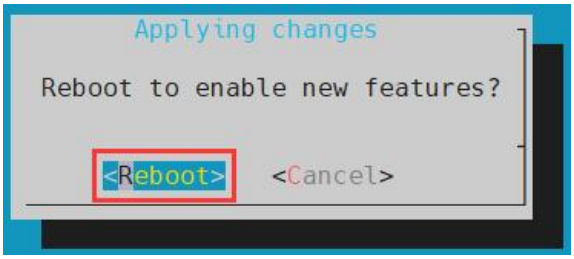
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system to make the configuration take effect



3) After entering the Linux system, use the **sudo ifconfig -a** command. If you can see the CAN device, it means that CAN has been correctly opened.

```
orangepi@orangepi:~$ sudo ifconfig -a
can0: flags=128<NOARP>  mtu 16
```

```
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00    txqueuelen 10    (UNSPEC)
        RX packets 0    bytes 0 (0.0 B)
        RX errors 0    dropped 0    overruns 0    frame 0
        TX packets 0    bytes 0 (0.0 B)
        TX errors 0    dropped 0 overruns 0    carrier 0    collisions 0
        device interrupt 91

can1: flags=128<NOARP>    mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00    txqueuelen 10    (UNSPEC)
        RX packets 0    bytes 0 (0.0 B)
        RX errors 0    dropped 0    overruns 0    frame 0
        TX packets 0    bytes 0 (0.0 B)
        TX errors 0    dropped 0 overruns 0    carrier 0    collisions 0
        device interrupt 92
```

4) The pins corresponding to CAN0 and CAN1 are

	CAN0	CAN1
TX Pin	**Corresponding to pin 5**	**Corresponding to pin 33**
RX Pin	**Corresponding to pin 3**	**Corresponding to pin 31**

3.18.7.2. Using CANalyst-II analyzer to test the sending and receiving messages

1) The CANalyst-II analyzer used in the test is shown in the figure below



2) CANalyst-II analyzer data download link

https://www.zhcxgd.com/3.html

3) First, you need to install the USBCANToolSetup software



4) The shortcut after USBCANToolSetup is installed is:



5) You also need to install the USB driver



6) The USB port of CANalyst-II analyzer needs to be connected to the USB port of the computer.



7) To test the CAN function, you also need to prepare a CAN transceiver as shown in the figure below. The main function of the CAN transceiver is to convert the TTL signal of the CAN controller into the differential signal of the CAN bus.

a. The 3.3V pin of the CAN transceiver needs to be connected to the 3.3V pin of

the 40pin of the development board

b. The GND pin of the CAN transceiver needs to be connected to the GND pin of the 40pin of the development board
c. The CAN TX pin of the CAN transceiver needs to be connected to the TX pin of the CAN bus in the 40pin of the development board
d. The CAN RX pin of the CAN transceiver needs to be connected to the RX pin of the CAN bus in the 40pin of the development board
e. The CANH pin of the CAN transceiver needs to be connected to the H interface of the analyzer
f. The CANL pin of the CAN transceiver needs to be connected to the L interface of the analyzer



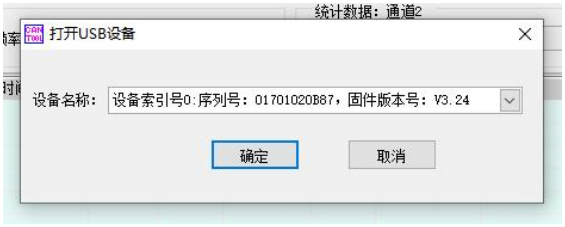
8) Then you can open the USB-CAN software



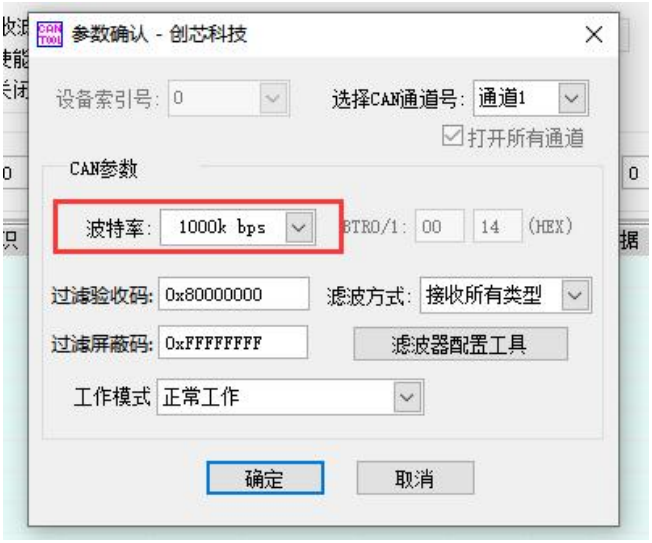
9) Then click Start Device



10) Then click OK



11) Set the baud rate to 1000k bps



12) After successfully opening, the USB-CAN software will display the serial number and other information



13) Development board receives CAN message test

a. First, set the baud rate of the CAN bus to **1000kbps** in the Linux system of the development board

```
orangepi@orangepi:~$ sudo ip link set can0 down
orangepi@orangepi:~$ sudo ip link set can0 type can bitrate 1000000
orangepi@orangepi:~$ sudo ip link set can0 up
```

b. Then run the **candump can0** command to prepare to receive messages.

```
orangepi@orangepi:~$ sudo candump can0
```

c. Then send a message to the development board in the USB-CAN software



d. If the development board can receive the message sent by the analyzer, it means that the CAN bus can be used normally.

```
orangepi@orangepi5ultra:~$ sudo candump can0
  can0  001   [8]  01 02 03 04 05 06 07 08
```

14) Development board sends CAN message test

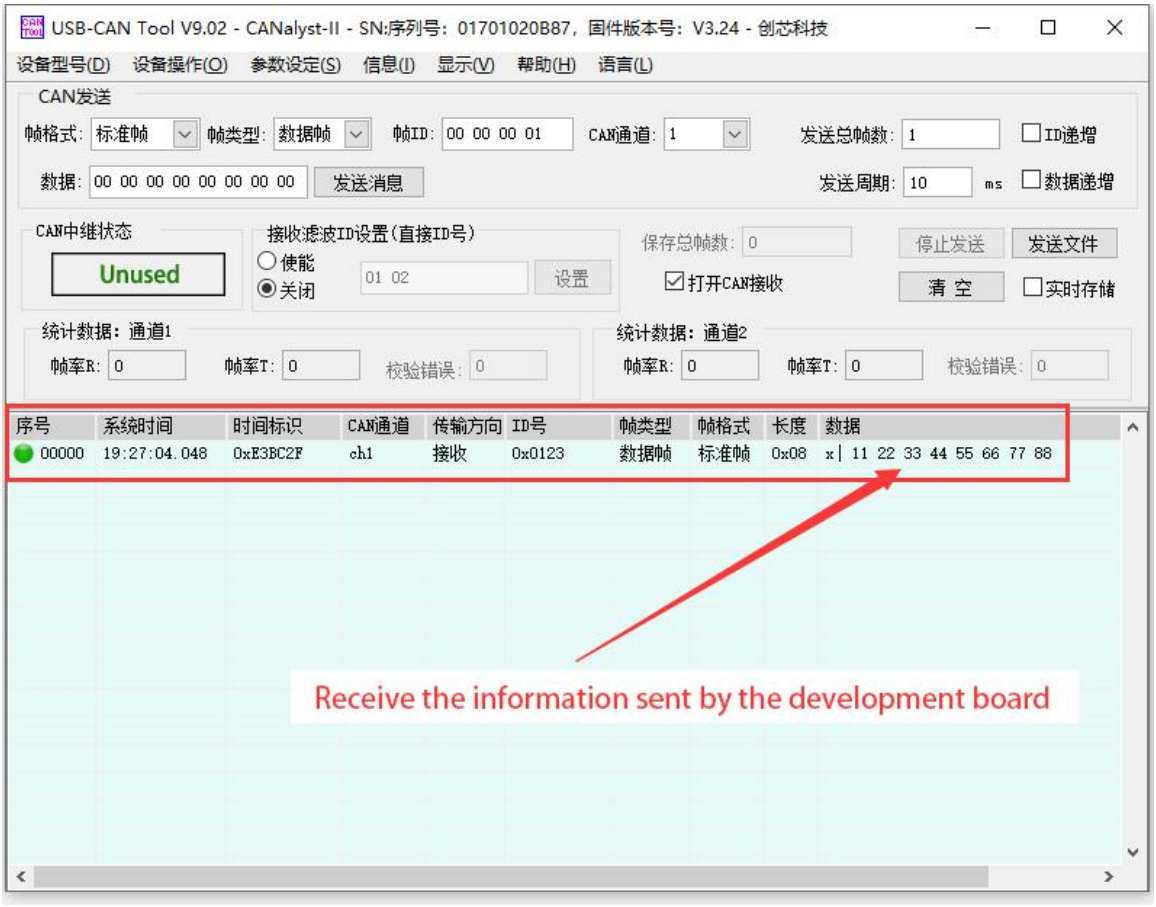
a. First, set the CAN baud rate to **1000kbps** in the Linux system

```
orangepi@orangepi:~$ sudo ip link set can0 down
orangepi@orangepi:~$ sudo ip link set can0 type can bitrate 1000000
orangepi@orangepi:~$ sudo ip link set can0 up
```

b. Execute the **cansend** command in the development board to send a message

```
orangepi@orangepi:~$ sudo cansend can0 123#1122334455667788
```

c. If the USB-CAN software can receive the message sent by the development board, it means the communication is successful



3.19. How to use wiringOP hardware PWM

Before using wiringOP to operate PWM, please make sure that wiringOP has been installed on the Linux system. If the gpio readall command can be used normally, it means that wiringOP has been installed. If it prompts that the command cannot be found, please refer to the instructions in the section "How to install

wiringOP" to install wiringOP first.

3. 19. 1. How to set PWM using wiringOP's gpio command

3. 19. 1. 1. Set the corresponding pin to PWM mode

1) As shown in the following table, the development board can use seven PWMs, namely PWM0, PWM1, PWM3, PWM6, PWM12, PWM13 and PWM14. Only one of PWM0_M0 and PWM0_M2, PWM1_M0 and PWM1_M2 can be used at the same time, and they cannot be used at the same time. They are the same PWM, but connected to different pins. Please do not think that they are two different PWM buses.

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The wPi numbers corresponding to the PWM pins are as follows:

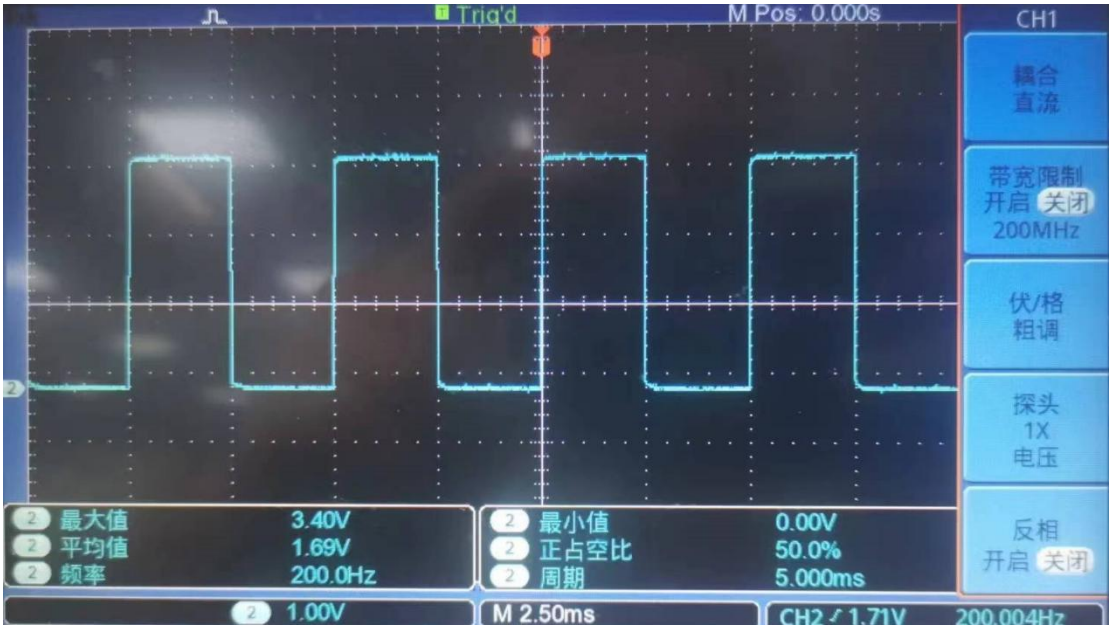
PWM Pin	wPi Serial number	Pin number	GPIO number
PWM0_M0	1	5	15
PWM0_M2	8	15	34
PWM1_M0	0	3	16
PWM1_M2	9	16	35
PWM3_M3	2	7	39
PWM6_M1	17	27	145
PWM12_M0	20	31	109
PWM13_M0	22	33	110
PWM14_M0	23	35	114

3) The command to set the pin to PWM mode is as follows, taking PWM0_M0 as an

example, where the third parameter needs to enter the wPi number corresponding to the PWM0_M0 pin.

```
orangepi@orangepi:~$ gpio mode 1 pwm
```

4) After the pin is set to PWM mode, it will output a square wave with a frequency of 200Hz, a period of 5ms, and a duty cycle of 50% by default. At this time, we use an oscilloscope to measure the corresponding PWM pin and we can see the following waveform.



3.19.1.2. Methods for adjusting PWM frequency

The calculation formula of PWM frequency is as follows:

PWM frequency = clock frequency / (frequency division factor * value of period register)

Among them:

1. The default value of the clock frequency is 24000000Hz.

2. The value range of the frequency division coefficient is an even number between 2 and 512, and the default value is 120. If the frequency division coefficient is set to an odd number, the actual frequency division coefficient is the set value minus one.

3. The default value of the period register is 1000.
4. The default value of the PWM frequency is 24000000 / (120 * 1000) = 200Hz.

3.19.1.2.1. Method of adjusting PWM frequency by setting the frequency division factor

1) We can use the following command to set the division factor of the PWM0_M0 pin to 4.

```
orangepi@orangepi:~$ gpio pwmc 1 4
```

2) According to the above formula, the calculated value of PWM frequency is 6000Hz. Through the oscilloscope, it can be observed that the measured value of PWM frequency is 6010Hz, and the error can be ignored.



3.19.1.2.2. Direct method to set PWM frequency

1) We can use the **gpio pwmTone** command to set the frequency of the PWM pin. For example, the following command can be used to set the PWM frequency of the PWM0_M0 pin to 500Hz.

```
orangepi@orangepi:~$ gpio pwmTone 1 500
```

When setting the PWM frequency, you need to ensure that:

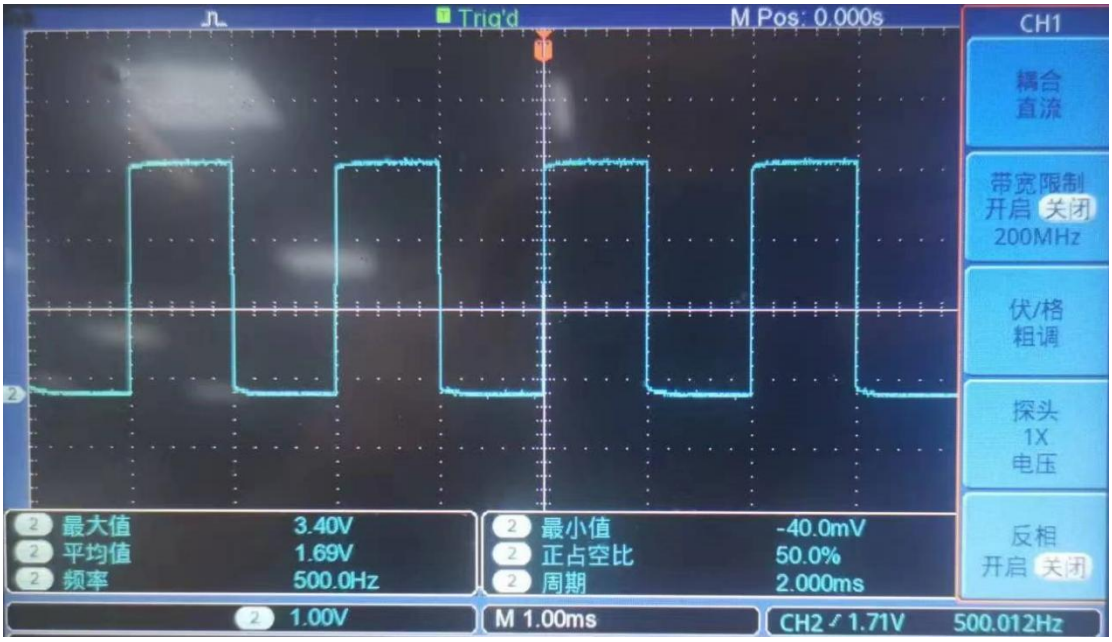
The set frequency value is < 24000000 / (frequency division factor * 2).

For example, the default division factor is 120. If the division factor is not modified, the set frequency value should be less than 100000 Hz.

If the value is set too large, the following error message will appear:

gpio: The PWM frequency you set is too high to be possible

2) Then through the oscilloscope, it can be observed that the PWM frequency becomes 500Hz.



3. 19. 1. 3. Methods for adjusting PWM duty cycle

1) The calculation formula of PWM duty cycle is as follows. We can adjust the PWM duty cycle by setting the value of the duty cycle register and the value of the period register.

PWM duty cycle = value of duty cycle register / value of period register

Where:

The default value of the duty cycle register is 500.

The default value of the period register is 1000.

It should be noted that the value of the duty cycle register needs to be smaller than the value of the period register because the duty cycle cannot be greater than 1.

When the value of the duty cycle register is set to be greater than the value of the period register, the following error message will be displayed:

gpio: CCR should be less than or equal to ARR (XXX)

When the value of the period register is set to < the value of the duty cycle register, the following error message will be prompted:

gpio: ARR should be greater than or equal to CRR (XXX)

2) We can use the following command to set the value of the period register of the PWM0_M0 pin to 2000.

```
orangepi@orangepi:~$ gpio pwmr 1 2000
```

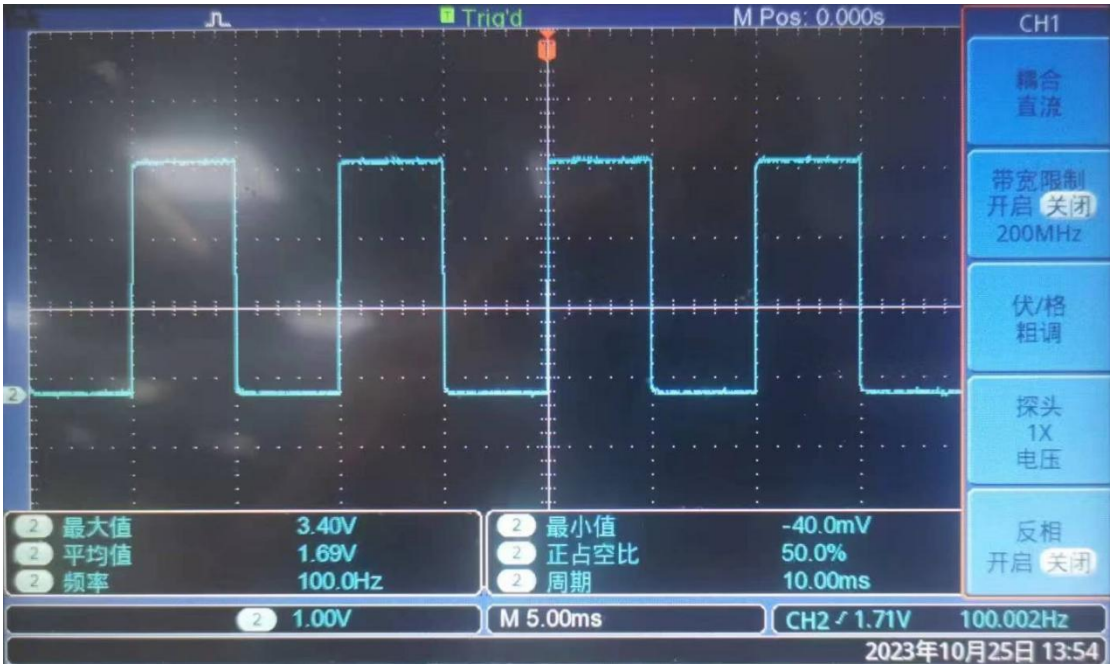
3) After running the above command, you can observe through the oscilloscope that the PWM duty cycle changes from the default 50% (500/1000) to 25% (500/2000).



4) We can use the following command to set the duty cycle register value of the PWM0_M0 pin to 1000.

```
orangepi@orangepi:~$ gpio pwm 1 1000
```

5) After running the above command, you can observe through the oscilloscope that the PWM duty cycle changes from 25% (500/2000) to 50% (1000/2000).



3. 19. 2. How to use the PWM test program

1) In the example directory of wiringOP, there is a program called pwm.c, which demonstrates how to use the PWM-related API in wiringOP to operate PWM.

```
orangepi@orangepi:~$ cd /usr/src/wiringOP/examples/
orangepi@orangepi:/usr/src/wiringOP/examples$ ls pwm.c
pwm.c
```

2) The command to compile **pwm.c** into an executable program is as follows:

```
orangepi@orangepi:/usr/src/wiringOP/examples$ gcc -o pwm pwm.c -lwiringPi
```

3) Then you can execute the PWM test program. When executing the PWM test program, you need to specify the PWM pin. For example, you can use the following command to test the PWM0_M0 pin:

```
orangepi@orangepi:/usr/src/wiringOP/examples$ sudo ./pwm 1
```

4) After the pwm program is executed, the following contents will be tested in sequence:

a. Adjust the PWM duty cycle by setting the value of the period register.

b. Adjust the PWM duty cycle by setting the value of the duty cycle register.
c. Adjust the PWM frequency by setting the division factor.
d. Set the PWM frequency directly.

5) After each test is completed, the PWM waveform output will stop for 5 seconds. After all test contents are completed, a new round of testing will start again.

6) The detailed execution process of the PWM test program is as follows:
a. Adjust the PWM duty cycle by setting the value of the period register: Through the oscilloscope, it can be observed that the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 50% to 25% and remains for 5 seconds. Then the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 25% to 50% and remains for 5 seconds.
b. Adjust the PWM duty cycle by setting the value of the duty cycle register: Through the oscilloscope, it can be observed that the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 50% to 100% and remains for 5 seconds. Then the PWM waveform changes every 0.5 seconds. After changing 8 times, the PWM duty cycle changes from 100% to 50% and remains for 5 seconds.
c. Adjust the PWM frequency by setting the division factor: Through the oscilloscope, it can be observed that the PWM waveform changes every 0.5 seconds. After changing 9 times, the PWM frequency will change from 2000hz to 200hz and maintain for 5 seconds. Then the PWM waveform changes every 0.5 seconds. After changing 9 times, the PWM frequency will become 2000Hz and maintain for 5 seconds.
d. Directly set the PWM frequency: Through the oscilloscope, it can be observed that the PWM frequency first changes to 2000Hz, and then increases by 2000Hz every two seconds. After changing 9 times, the PWM frequency becomes 20000Hz and remains for 5 seconds.

3. 20. Installation and use of wiringOP-Python

wiringOP-Python is the Python version of wiringOP, which is used to operate the GPIO, I2C, SPI, UART and other hardware resources of the development board

in Python programs.

Also note that some of the following commands are performed under the root user.

3.20.1. Installation of wiringOP-Python

1) First install the dependency package

```
root@orangepi:~# sudo apt-get update
root@orangepi:~# sudo apt-get -y install git swig python3-dev python3-setuptools
```

2) Then use the following command to download the source code of wiringOP-Python

Note that the following git clone --recursive command will automatically download the source code of wiringOP, because wiringOP-Python depends on wiringOP. Please make sure that there are no errors during the download process due to network problems.

If you have problems downloading the code from GitHub, you can directly use the wiringOP-Python source code that comes with the Linux image, which is stored in **/usr/src/wiringOP-Python**.

```
root@orangepi:~# git clone --recursive https://github.com/orangepi-xunlong/wiringOP-Python -b next
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# git submodule update --init --remote
```

3) Then use the following command to compile wiringOP-Python and install it into the Linux system of the development board

```
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# python3 generate-bindings.py > bindings.i
root@orangepi:~/wiringOP-Python# sudo python3 setup.py install
```

4) Then enter the following command. If help information is output, it means wiringOP-Python has been successfully installed. Press the **q** key to exit the help information interface.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi;  help(wiringpi)"
Help on module wiringpi:

NAME
```

```
    wiringpi

DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.
```

5) The steps to test whether wiringOP-Python is successfully installed in the python command line are as follows:

a. First use the python3 command to enter the python3 command line mode

```
root@orangepi:~# python3
```

b. Then import the Python module of wiringPi

```
>>> import wiringpi;
```

c. Finally, enter the following command to view the help information of wiringOP-Python. Press the **q** key to exit the help information interface.

```
>>> help(wiringpi)
Help on module wiringpi:

NAME
    wiringpi

DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.

CLASSES
    builtins.object
        GPIO
        I2C
        Serial
```

```
        nes


    class GPIO(builtins.object)
     |  GPIO(pinmode=0)
     |

>>>
```

3. 20. 2. 40pin GPIO port test

WiringOP-Python is the same as wiringOP. It can also determine which GPIO pin to operate by specifying the wPi number. Because there is no command to view the wPi number in wiringOP-Python, the correspondence between the board's wPi number and the physical pin can only be viewed through the gpio command in wiringOP.

```
orangepi@orangepi5ultra:~$ gpio readall
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |     IN | 1 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
 |   32 |   5 |    RXD.6 |     IN | 0 | 11 || 12 | 0 | IN     | GPIO4_A6 | 6   | 134  |
 |   33 |   7 |    TXD.6 |     IN | 0 | 13 || 14 |   |        | GND      |     |      |
 |   34 |   8 | GPIO1_A2 |     IN | 0 | 15 || 16 | 0 | IN     | GPIO1_A3 | 9   | 35   |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | IN     | GPIO1_A4 | 10  | 36   |
 |   42 |  11 | SPI0_TXD |     IN | 0 | 19 || 20 |   |        | GND      |     |      |
 |   41 |  12 | SPI0_RXD |     IN | 0 | 21 || 22 | 1 | IN     | GPIO1_B0 | 13  | 40   |
 |   43 |  14 | SPI0_CLK |     IN | 0 | 23 || 24 | 1 | IN     | SPI0_CS0 | 15  | 44   |
 |      |     |      GND |        |   | 25 || 26 | 1 | IN     | SPI0_CS1 | 16  | 45   |
 |  145 |  17 |    SDA.8 |     IN | 0 | 27 || 28 | 1 | IN     | SCL.8    | 18  | 144  |
 |  113 |  19 | GPIO3_C1 |     IN | 1 | 29 || 30 |   |        | GND      |     |      |
 |  109 |  20 |  CAN1_RX |     IN | 1 | 31 || 32 | 1 | IN     | GPIO4_B3 | 21  | 139  |
 |  110 |  22 |  CAN1_TX |     IN | 1 | 33 || 34 |   |        | GND      |     |      |
 |  114 |  23 | GPIO3_C2 |     IN | 1 | 35 || 36 | 1 | IN     | GPIO4_B7 | 24  | 143  |
 |  135 |  25 | GPIO4_A7 |     IN | 0 | 37 || 38 | 1 | IN     | GPIO3_C0 | 26  | 112  |
 |      |     |      GND |        |   | 39 || 40 | 1 | IN     | GPIO3_B7 | 27  | 111  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+OPI5-ULTRA+---+--------+----------+-----+------+
orangepi@orangepi5ultra:~$
```

1) Below, we take pin 7, which corresponds to GPIO GPIO1_A7 and wPi number 2, as an example to demonstrate how to set the high and low levels of the GPIO port.

```
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   16 |   0 |    SDA.2 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   15 |   1 |    SCL.2 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   39 |   2 |     PWM3 |     IN | 1 |  7 || 8  | 1 | ALT10  | TXD.2    | 3   | 13   |
 |      |     |      GND |        |   |  9 || 10 | 1 | ALT10  | RXD.2    | 4   | 14   |
```

2) The steps for testing directly using commands are as follows:

a. First, set the GPIO port to output mode. The first parameter of the **pinMode** function is the wPi number corresponding to the pin, and the second parameter is the GPIO mode.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
from wiringpi import GPIO; wiringpi.wiringPiSetup() ; \
wiringpi.pinMode(2, GPIO.OUTPUT) ;  "
```

b. 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 set successfully.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.LOW)"
```

c. 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 set successfully.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.HIGH)"
```

3) The steps to test in the python3 command line are as follows:

a. First use the python3 command to enter the python3 command line mode

```
root@orangepi:~# python3
```

b. Then import the Python module of wiringPi

```
>>> import wiringpi
>>> from wiringpi import GPIO
```

c. Then set the GPIO port to output mode, where the first parameter of the **pinMode** function is the wPi number corresponding to the pin, and the second parameter is the GPIO mode

```
>>> wiringpi.wiringPiSetup()
0
>>> wiringpi.pinMode(2, GPIO.OUTPUT)
```

d. 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 set successfully.

```
>>> wiringpi.digitalWrite(2, GPIO.LOW)
```

e. 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 set successfully.

```
>>> wiringpi.digitalWrite(2, GPIO.HIGH)
```

4) wiringOP-Python For setting the GPIO high and low levels in Python code, please refer to the **blink.py** test program in the examples. The **blink.py** test program will set the voltage of all GPIO ports in the 26 pins of the development board to change continuously.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# ls blink.py
blink.py
root@orangepi:~/wiringOP-Python/examples# python3 blink.py
```

3. 20. 3. 40pin SPI test

1) As shown in the figure below, the available spis for Orange Pi 5 Ultra are spi0, spi1 and spi4

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of SPI0, SPI1 and SPI4 in 40 pins are shown in the following table.

	SPI0_M2 correspond 40pin	**SPI1_M1 correspond 40pin**	**SPI4_M2 correspond 40pin**
MOSI	**Pin 19**	**Pin 40**	**Pin 13**
MISO	**Pin 21**	**Pin 38**	**Pin 11**
CLK	**Pin 23**	**Pin 29**	**Pin 15**
CS0	**Pin 24**	**Pin 35**	**Pin 16**

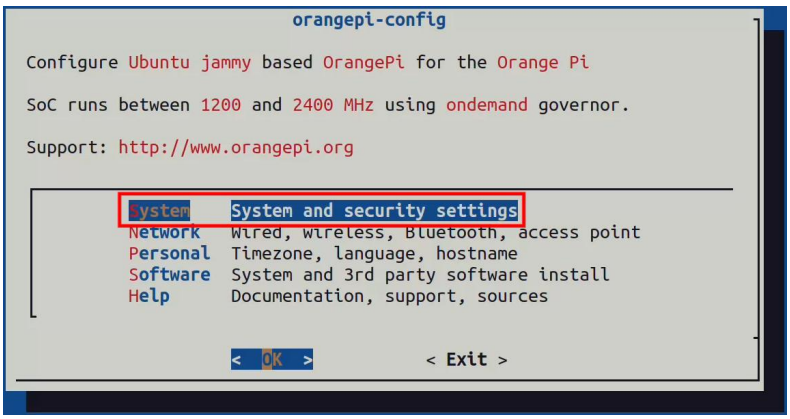
CS1	Pin 26	None	None
Dtbo configuration	spi0-m2-cs0-spidev spi0-m2-cs1-spidev spi0-m2-cs0-cs1-spidev	spi1-m1-cs0-spidev	spi4-m2-cs0-spidev

3) In Linux system, the SPI in 40 pin is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

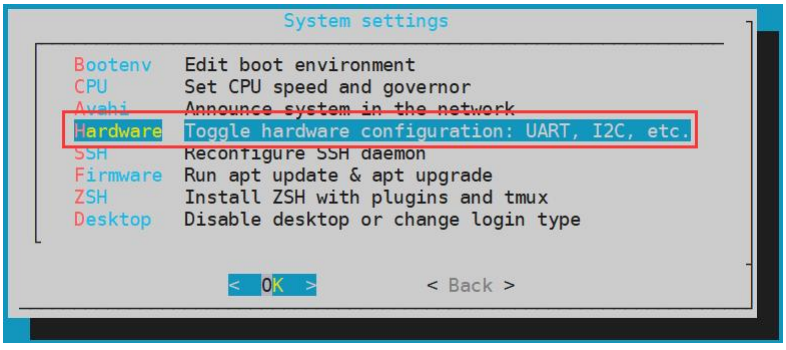
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**



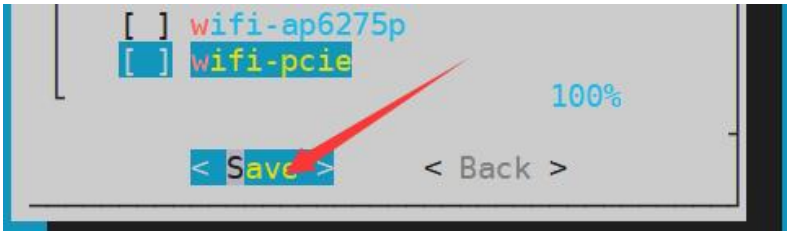
c. Then select **Hardware**



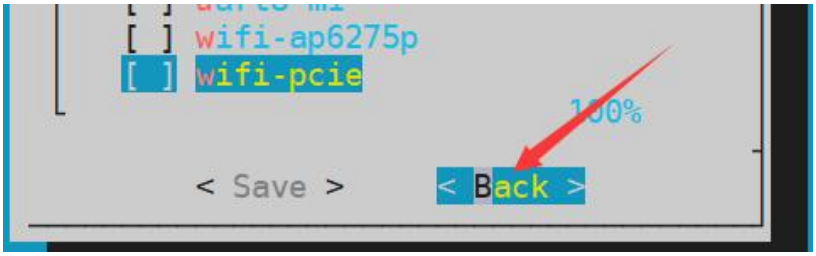
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the **spacebar** to select the SPI configuration you want to open



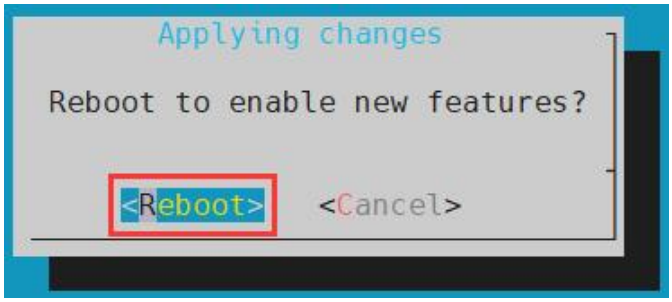
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After restarting, enter the system and check whether there is a device node of **spidevx.x** in the Linux system. If it exists, it means that SPI has been set up and can be used directly.

```
orangepi@orangepi:~$ ls /dev/spidev*
/dev/spidev0.0  /dev/spidev0.1  /dev/spidev4.0
```

The above is the result displayed after opening spi0-m2-cs0-cs1-spidev and

spi4-m2-cs0-spidev.

5) Then you can use the **spidev_test.py** program in the examples to test the SPI loopback function. The **spidev_test.py** program needs to specify the following two parameters:

a. **--channel：** Specify the SPI channel number

b. **--port：** Specify the SPI port number

6) Without shorting the mosi and miso pins of SPI, the output of running spidev_test.py is as follows. It can be seen that the data of TX and RX are inconsistent.

The x after the --channel and --port parameters needs to be replaced with the specific SPI channel number and SPI port number.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py --channel x --port x
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev0.0
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF F0 0D    |......@.........|
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF    |................|
```

7) Then use the Dupont line to short the SPI's txd and rxd pins and run spidev_test.py. The output is as follows. You can see that the sent and received data are the same, indicating that the SPI loopback test is normal.

The x after the --channel and --port parameters needs to be replaced with the specific SPI channel number and SPI port number.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py --channel x --port x
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev0.0
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF 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 FF FF FF FF FF FF FF FF FF
FF FF FF F0 0D    |......@.........|
```

3. 20. 4. 4.40pin I2C test

1) As can be seen from the table below, Orange Pi 5 Ultra has four i2c buses: i2c2, i2c4, i2c5 and i2c8

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the 4 groups of I2C buses in 40 pins are shown in the following table. I2C2_M0 and I2C2_M4 can only use one of them at the same time, not both. They are the same I2C, just connected to different pins. Please do not think that they are two different I2C buses.

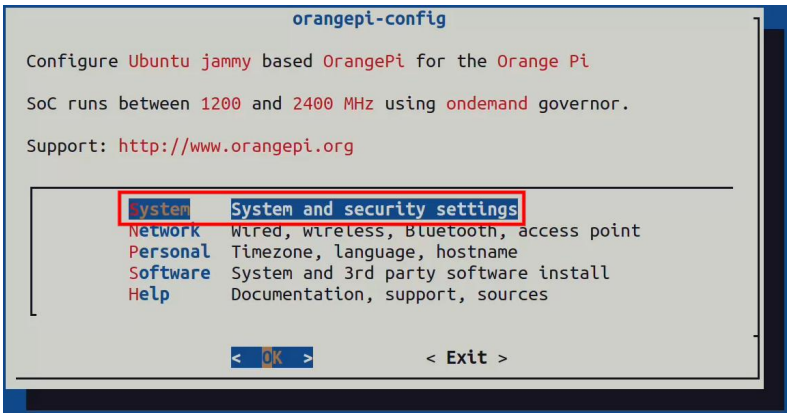
I2C bus	SDA corresponds to 40pin	SCL corresponds to 40pin	dtbo corresponding configuration
I2C2_M0	Pin 3	Pin 5	i2c2-m0
I2C2_M4	Pin 11	Pin 13	i2c2-m4
I2C4_M3	Pin 15	Pin 16	i2c4-m3
I2C5_M2	Pin 37	Pin 12	i2c5-m2
I2C8_M3	Pin 27	Pin 28	i2c8-m3

3) In Linux system, the I2C bus in 40 pins is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

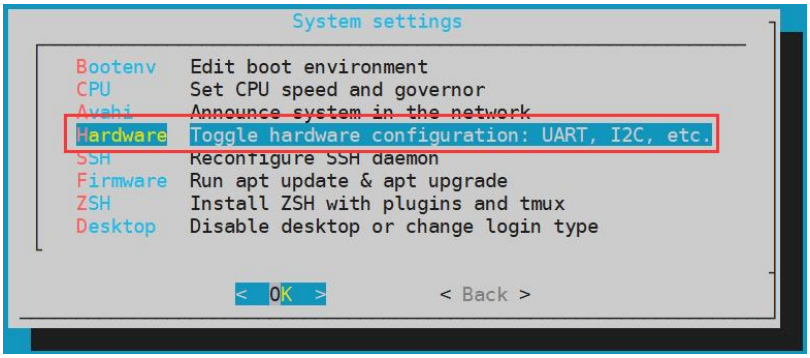
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

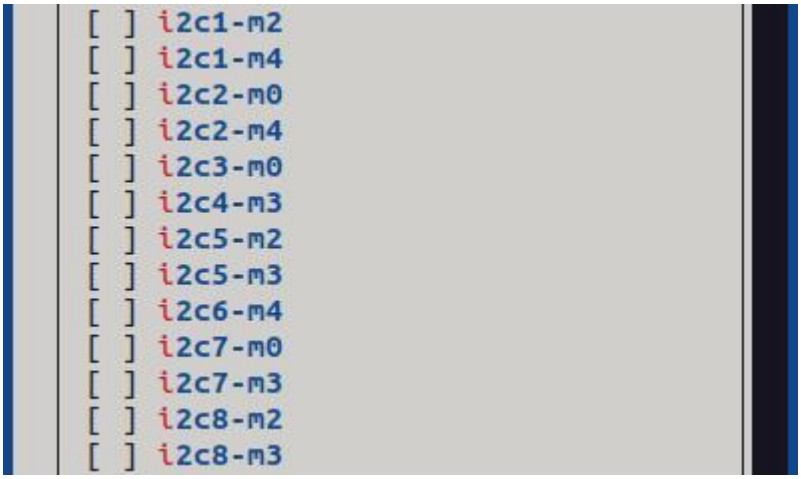
b. Then select **System**



c. Then select **Hardware**



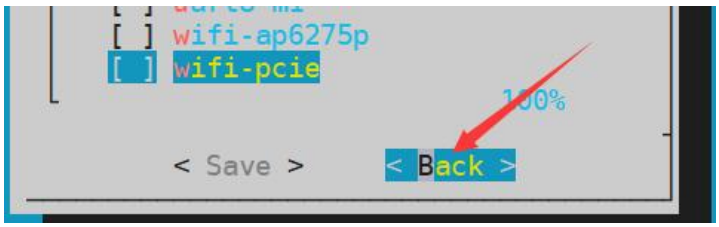
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the **spacebar** to select the I2C configuration you want to open



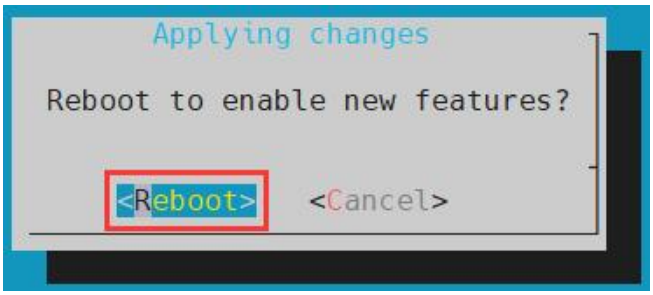
e. Then select **<Save>** to save



f. Then select **<Back>**



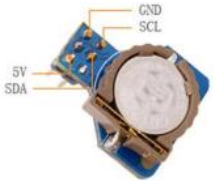
g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After starting the Linux system, first confirm that the i2c device node exists under **/dev**

```
orangepi@orangepi:~$ ls /dev/i2c-*
```

5) Then connect an i2c device to the i2c pin of the 40pin connector. Here we take the ds1307 RTC module as an example.



6) Then use the **i2cdetect -y** command. If the address of the connected i2c device can be detected, it means that i2c can be used normally.

```
orangepi@orangepi:~$ sudo i2cdetect -y 2      #i2c2 commands
orangepi@orangepi:~$ sudo i2cdetect -y 4      #i2c4 commands
orangepi@orangepi:~$ sudo i2cdetect -y 5      #i2c5 commands
```

```
orangepi@orangepi:~$ sudo i2cdetect -y 8        #i2c8 commands
```

```
root@orangepi:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@orangepi:~#
```

7) Then you can run the **ds1307.py** test program in the **examples** to read the RTC time.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 ds1307.py --device /dev/i2c-2
Thu 2023-01-05  14:57:55
Thu 2023-01-05  14:57:56
Thu 2023-01-05  14:57:57
^C
exit
```

3.20.5. 40pin UART test

1) As can be seen from the table below, the available UARTs of Orange Pi 5 Ultra are uart3, uart4 and uart6, a total of three UART buses. Uart2 is used for debugging serial port functions and is not included.

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the four UART buses in 40 pins are shown in the following table:

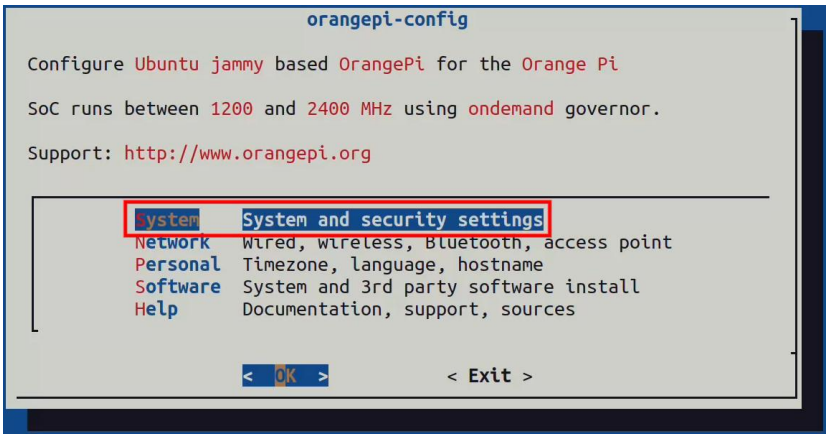
UART bus	RX correspond 40pin	TX correspond 40pin	Dtbo Corresponding configuration
UART3_M1	Pin 33	Pin 31	uart3-m1
UART4_M2	Pin 19	Pin 23	uart4-m2
UART6_M1	Pin 11	Pin 13	uart6-m1

3) In Linux system, the UART in 40 pin is closed by default and needs to be opened manually before it can be used. The detailed steps are as follows:

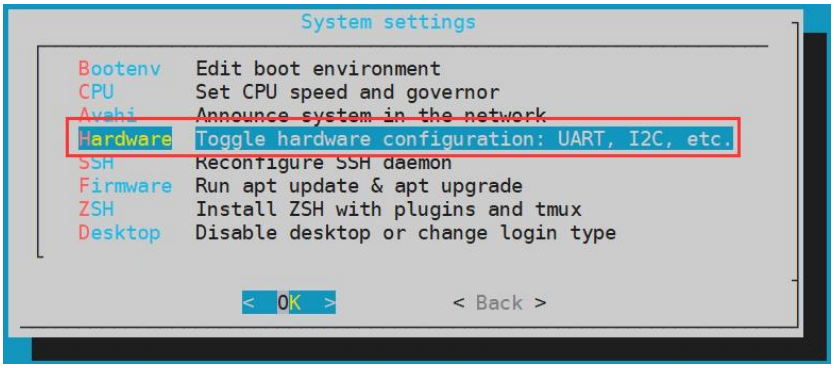
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

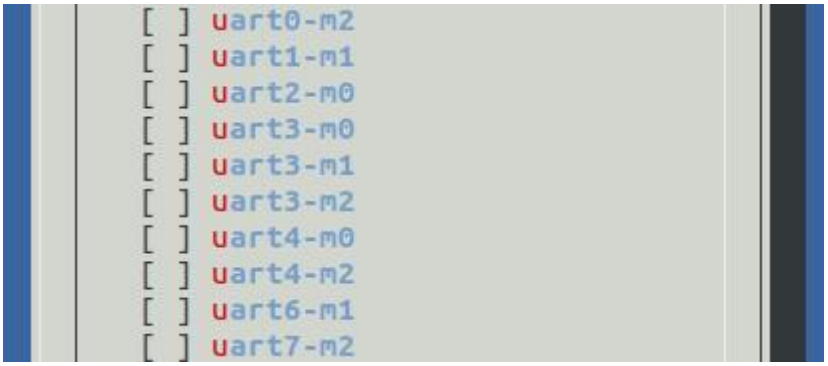
b. Then select **System**



c. Then select **Hardware**



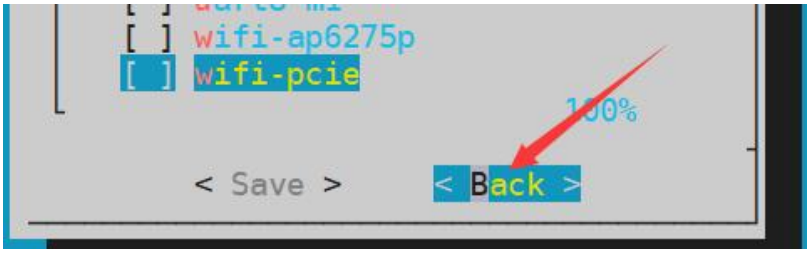
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the **spacebar** to select the UART configuration you want to open



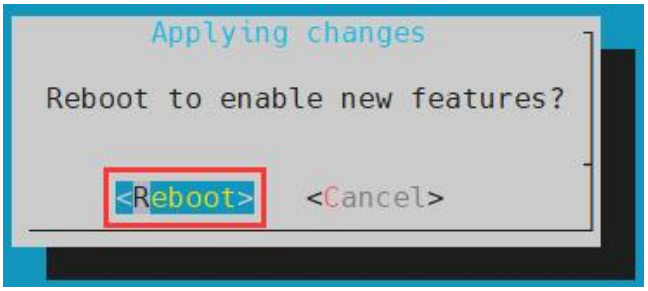
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



4) After entering the Linux system, first confirm whether there is a device node corresponding to uart under **/dev**

```
orangepi@orangepi:~$ ls /dev/ttyS*
```

5) Then start testing the UART interface. First use the Dupont line to short-circuit the rx and tx of the UART interface to be tested.

6) Use the **serialTest.py** program in the examples 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.

The X in /dev/ttySX needs to be replaced with the serial number of the specific uart device node.

```
root@orangepi:~/wiringOP-Python/examples# python3 serialTest.py --device /dev/ttySX
Out:    0: ->    0
Out:    1: ->    1
Out:    2: ->    2
Out:    3: ->    3
Out:    4:^C
exit
```

3.21. Hardware watchdog test

The Linux system released by Orange Pi has the watchdog_test program pre-installed, which can be used for direct testing.

The method to run the watchdog_test program is as follows:

a. The second parameter 10 represents the watchdog count time. If the watchdog 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 successfully fed.

```
orangepi@orangepi:~$ sudo watchdog_test 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. Check the serial number of the RK3588 chip

The command to view the serial number of the RK3588 chip is as follows. The serial number of each chip is different, so the serial number can be used to distinguish multiple development boards.

```
orangepi@orangepi:~$ cat_serial.sh
Serial                : 1404a7682e86830c
```

3. 23. How to install Docker

1) The Linux image provided by Orange Pi has Docker pre-installed, but the Docker service is not enabled by default.

2) Use the **enable_docker.sh** script to enable the docker service, then you can start using the docker command, and the docker service will be automatically started the next time you start the system

```
orangepi@orangepi:~$ enable_docker.sh
```

3) Then you can use the following command to test docker. If you can run hello-world, it means that docker can be used normally.

```
orangepi@orangepi:~$ sudo 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.
.....
```

3. 24. How to download and install the arm64 version of balenaEtcher

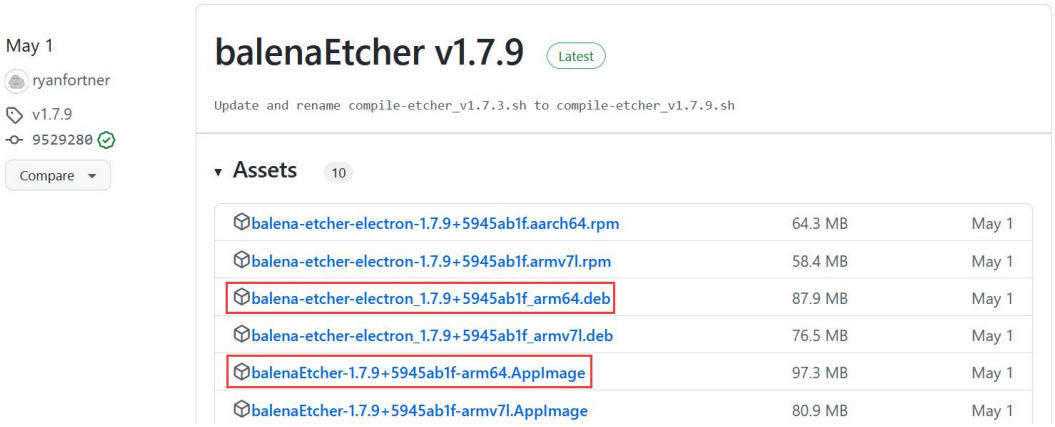
1) The download address of balenaEtcher arm64 version is:

   a. The download address of the deb installation package is as follows, which needs to be installed before use

https://github.com/Itai-Nelken/BalenaEtcher-arm/releases/download/v1.7.9/balena-etcher-electron_1.7.9+5945ab1f_arm64.deb

   b. The download address of the AppImage version that does not require installation is as follows:

https://github.com/Itai-Nelken/BalenaEtcher-arm/releases/download/v1.7.9/balenaEtcher-1.7.9+5945ab1f-arm64.AppImage

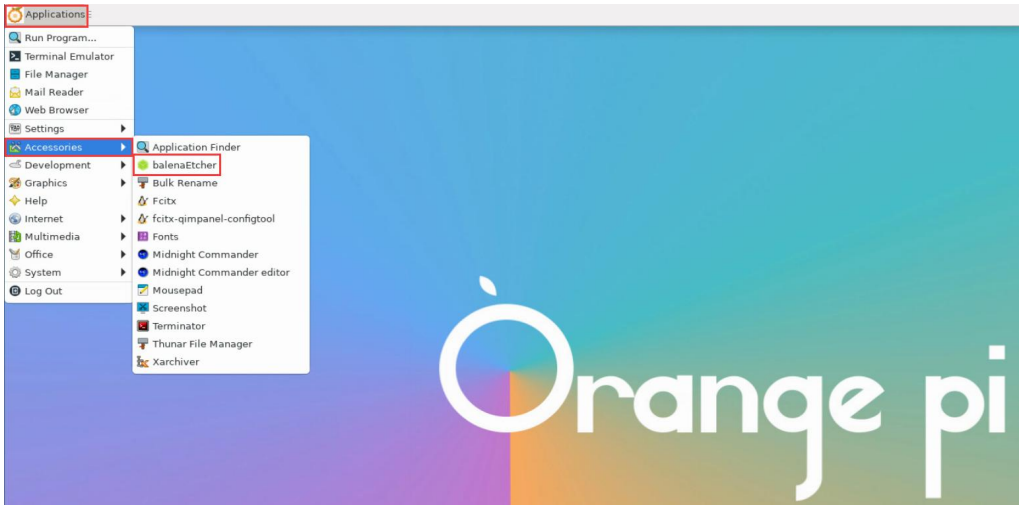


2) How to install and use the deb version of balenaEtcher:

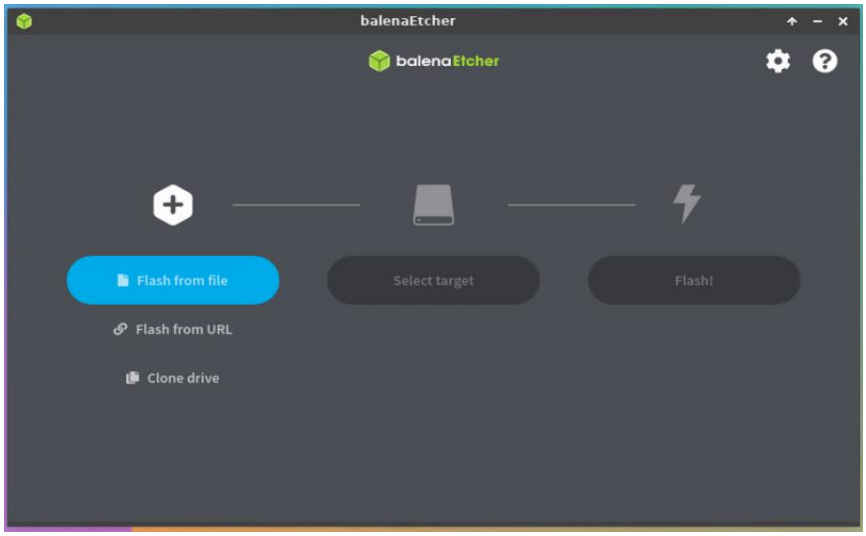
   a. deb version of balenaEtcher installation command is as follows:

```
orangepi@orangepi:~$ sudo apt install -y --fix-broken ./balena-etcher-electron_1.7.9+5945ab1f_arm64.deb
```

   b. After the deb version of balenaEtcher is installed, you can open it in Application



c. balenaEtcher opens with the following interface:

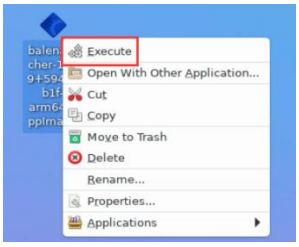


3) How to use the AppImage version of balenaEtcher:

a. First add permissions to balenaEtcher

```
orangepi@orangepi:~/Desktop$ chmod +x balenaEtcher-1.7.9+5945ab1f-arm64.AppImage
```

b. Then select the AppImage version of balenaEtcher, right-click, and click Execute to open balenaEtcher.



3. 25. How to install Baota Linux Panel

Baota 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 Baota official website)

1) The recommended order of Baota Linux system compatibility is:

```
Debian11 > Ubuntu 22.04 > Debian12
```

2) Then enter the following command in the Linux system to start the installation of the pagoda

```
orangepi@orangepi:~$ sudo install_bt_panel.sh
```

3) Then the Baota installation program will prompt whether to install **Bt-Panel** to the **/www** folder, just enter **y**

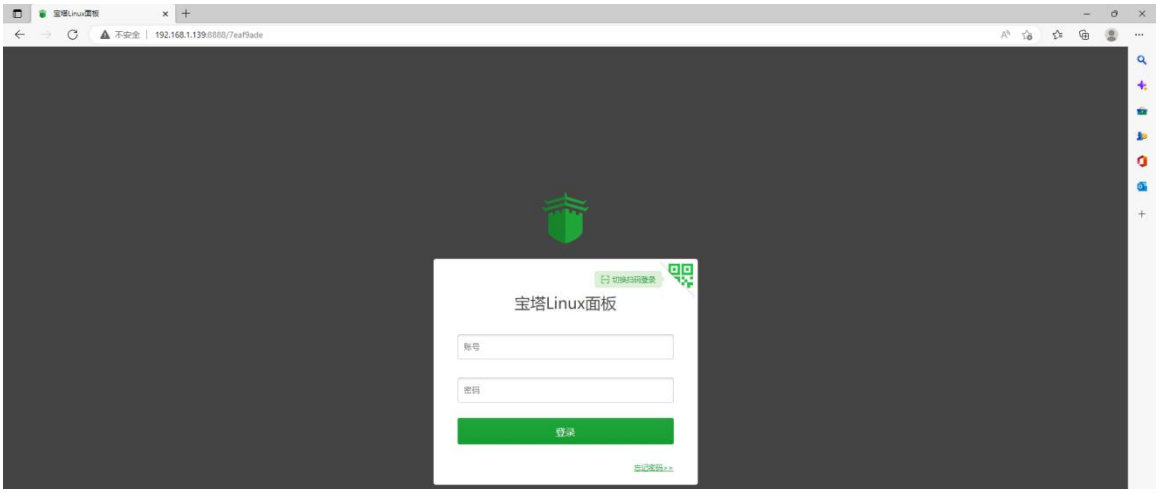
```
+----------------------------------------------------------------------
| Bt-WebPanel FOR CentOS/Ubuntu/Debian
+----------------------------------------------------------------------
| Copyright © 2015-2099 BT-SOFT(http://www.bt.cn) All rights reserved.
+----------------------------------------------------------------------
| 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 by the terminal, it means that the pagoda has been installed. The entire installation process takes about 6 minutes, which may vary depending on the network speed.

```
==================================================================
Congratulations! Installed successfully!
==============When you open the panel browser for the first time, it will prompt that it is not
safe==================
```

```
Please choose one of the following ways to resolve the unsafe reminder
1、Download the certificate, address: https://dg2.bt.cn/ssl/baota_root.pfx, double-click to install, password [www.bt.cn]
2、Click [Advanced] - [Continue] or [Accept the risk and continue] to access
Tutorial: https://www.bt.cn/bbs/thread-117246-1-1.html
Mac users please download and use this certificate: https://dg2.bt.cn/ssl/mac.crt

==========================Panel account login information==========================

[Cloud Server] Please allow port 31402 in the security group
External panel address: https://183.17.125.22:31402/d1f7cff0
Intranet panel address: https://10.31.2.175:31402/d1f7cff0
username: a8vazvbh
password: bc7bb4b7

Use the browser to access the following link to add Baota customer service
https://www.bt.cn/new/wechat_customer
====================================================================
Time consumed: 14 Minute!
```

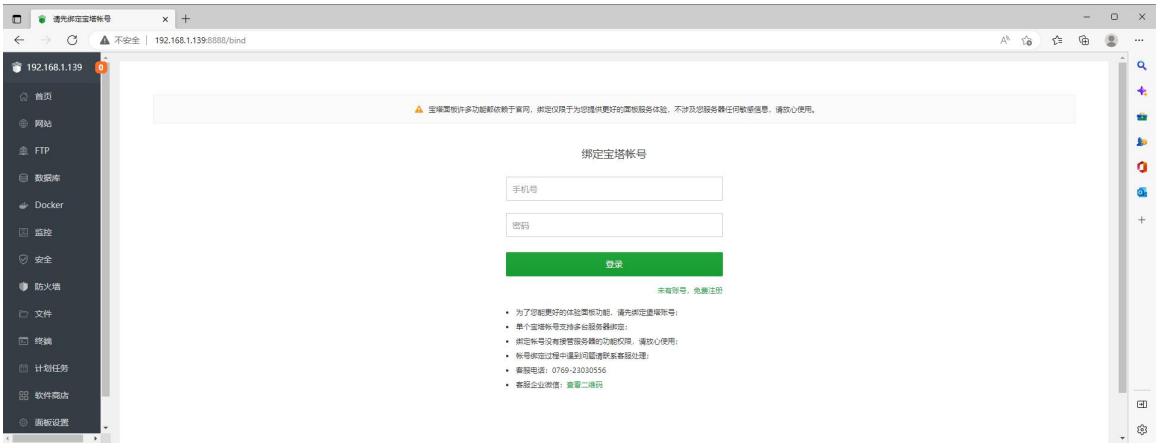
5) At this time, enter the **panel address** shown above in the browser to open the login interface of the Baota Linux panel, and then enter the **username** and **password** shown in the above figure in the corresponding position to log in to Baota



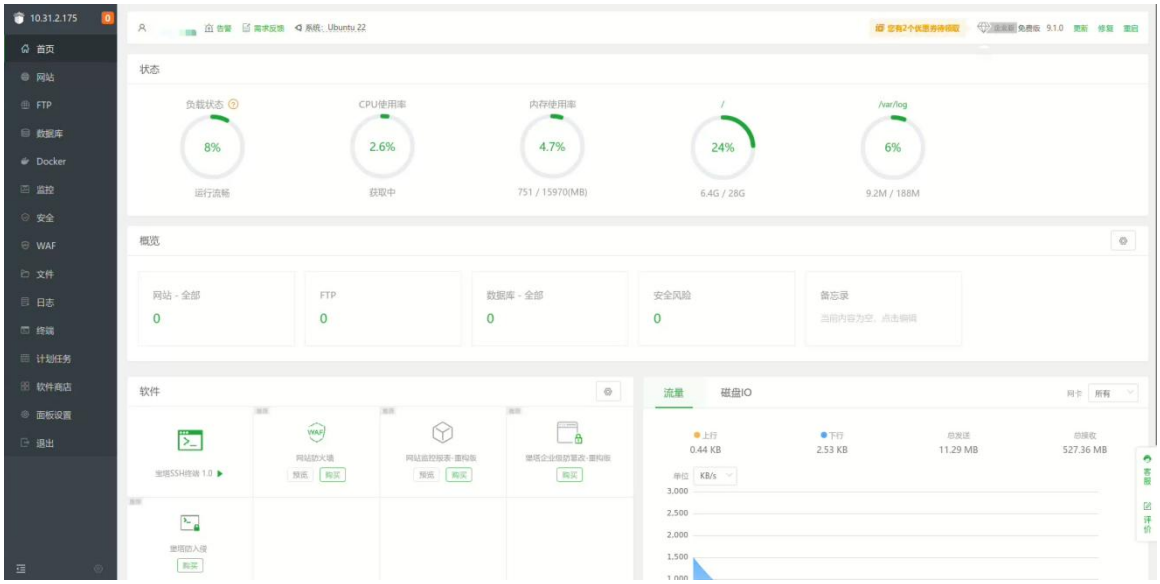
6) After successfully logging into the pagoda, the following welcome interface will pop up. First, please read the user instructions in the middle and drag them to the bottom. Then you can select "I have agreed and read the User Agreement", and then click "Enter the Panel" to enter the pagoda.



7) After entering the pagoda, you will be prompted to bind an account on the pagoda official website. If you don't have an account, you can go to the pagoda's official website (**https://www.bt.cn**) to register one.

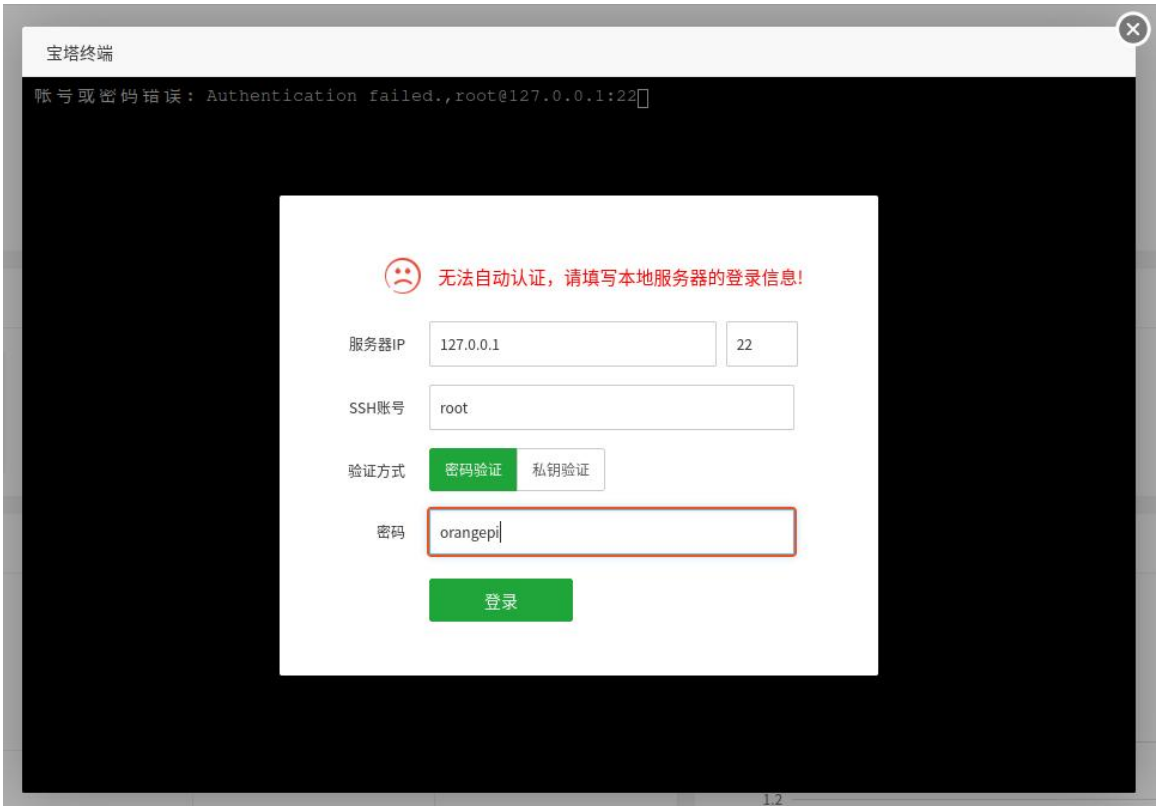


8) The final interface is shown in the figure below. You can intuitively see some status information of the development board Linux system, such as load status, CPU usage, memory usage, and storage space usage.

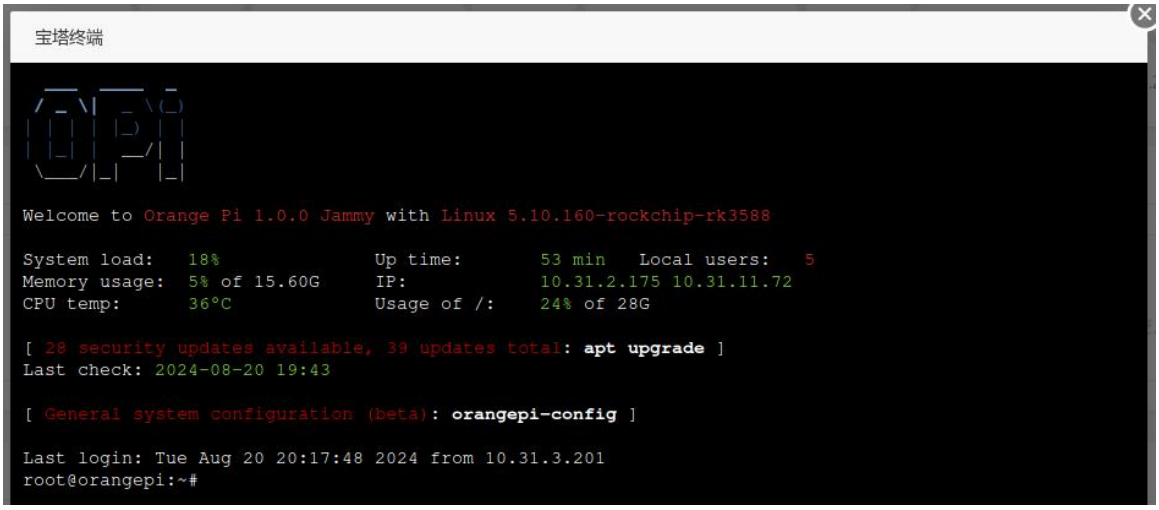


9) Test the SSH terminal login of Baota

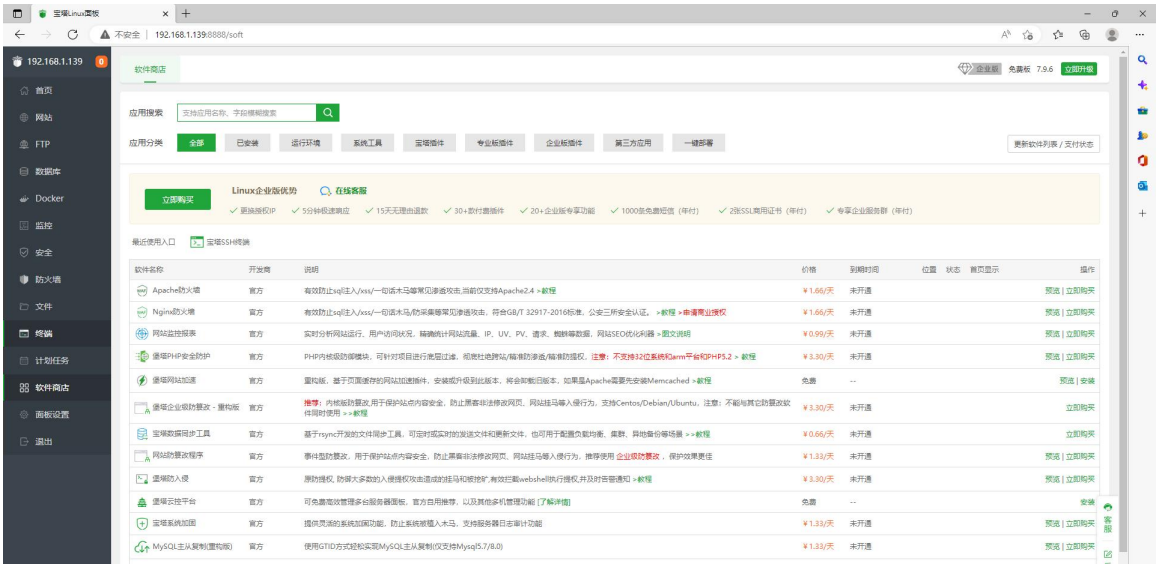
a. After opening the SSH terminal of Baota, you will be prompted to enter the password of the development board system. At this time, enter **orangepi** in the password box (the default password, if you have changed it, please fill in the modified password)



b. The display after successful login is as shown below



10) You can install Apache, MySQL, PHP and other software in Baota's software store, and you can also deploy various applications with one click. Please explore these functions by yourself, and I will not demonstrate them one by one here.



11) For more functions of the pagoda, please refer to the following information to explore it yourself

User Manual：**http://docs.bt.cn**
Forum Address：**https://www.bt.cn/bbs**
GitHub Link：**https://github.com/aaPanel/BaoTa**

3. 26. Set up Chinese environment and install Chinese input method

Note: Before installing the Chinese input method, please make sure that the Linux system used by the development board is the desktop version.

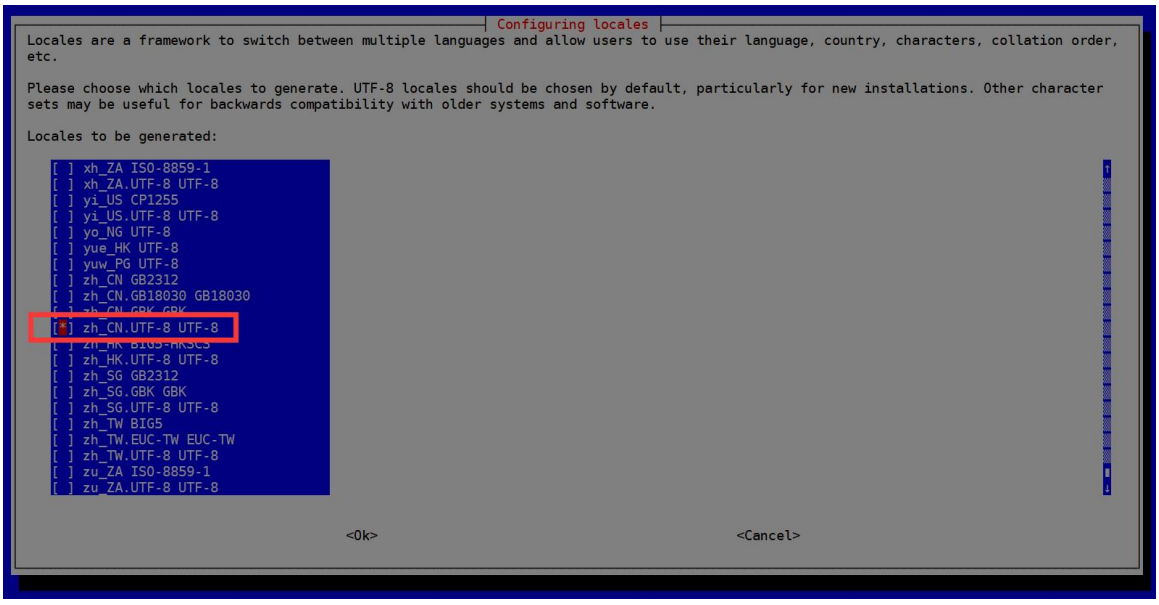
3. 26. 1. Installation method of Debian system

1) First set the default **locale** to Chinese

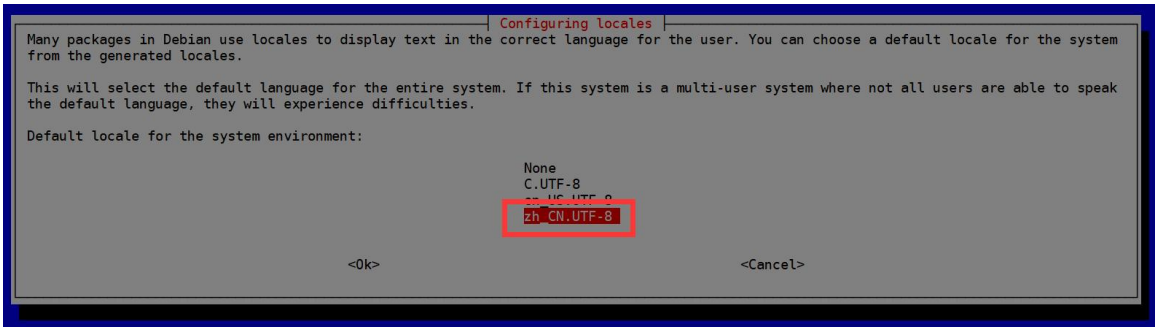
a. Enter the following command to start configuring **locale**

```
orangepi@orangepi:~$ sudo dpkg-reconfigure locales
```

b. Then select **zh_CN.UTF-8 UTF-8** in the pop-up interface (use the up and down arrow keys on the keyboard to move up and down, use the space bar to select, and finally use the Tab key to move the cursor to **<OK>**, then press Enter)



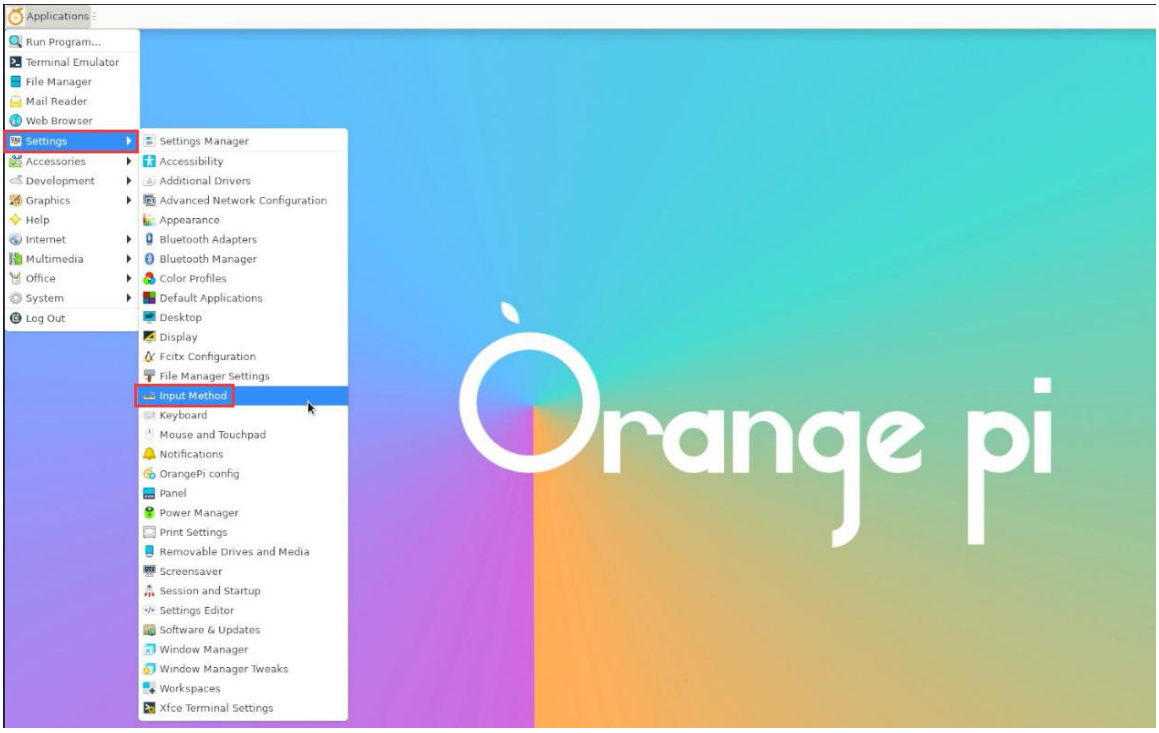
c. Then set the default **locale** to **zh_CN.UTF-8**



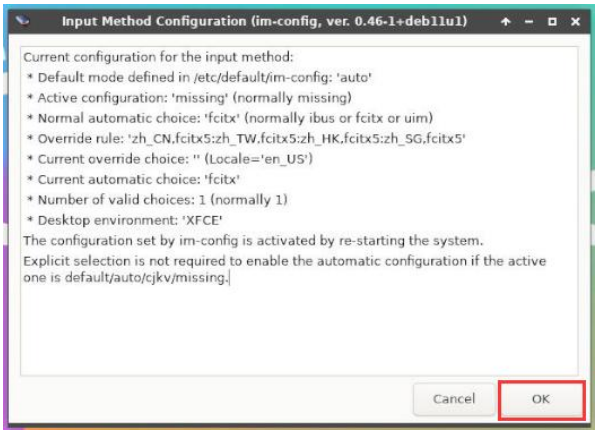
d. After exiting the interface, the **locale** setting will begin. The output displayed on the command line is as follows

```
orangepi@orangepi:~$ sudo dpkg-reconfigure locales
Generating locales (this might take a while)...
  en_US.UTF-8... done
  zh_CN.UTF-8... done
Generation complete.
```

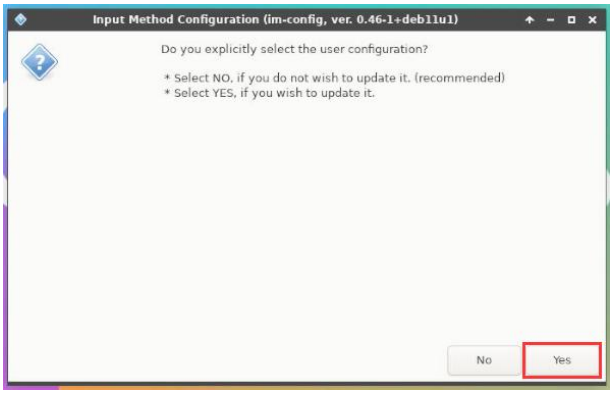
2) Then open **Input Method**



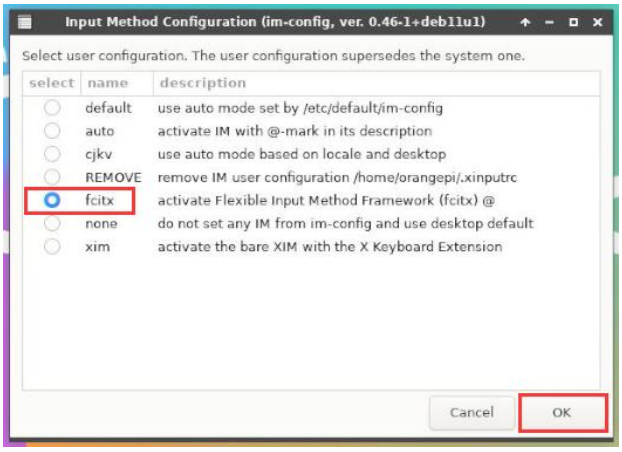
3) Then select **OK**



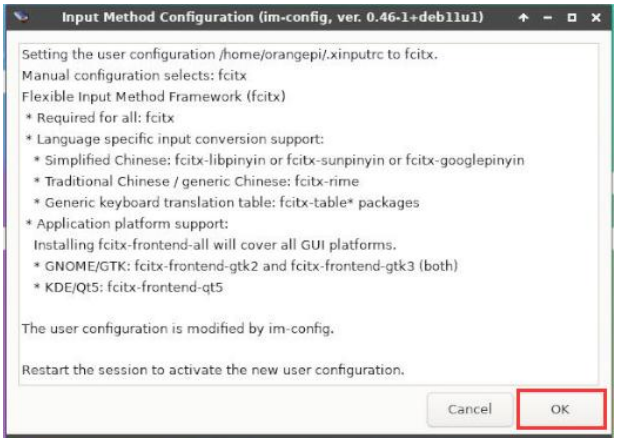
4) Then select **Yes**



5) Then select **fcitx**

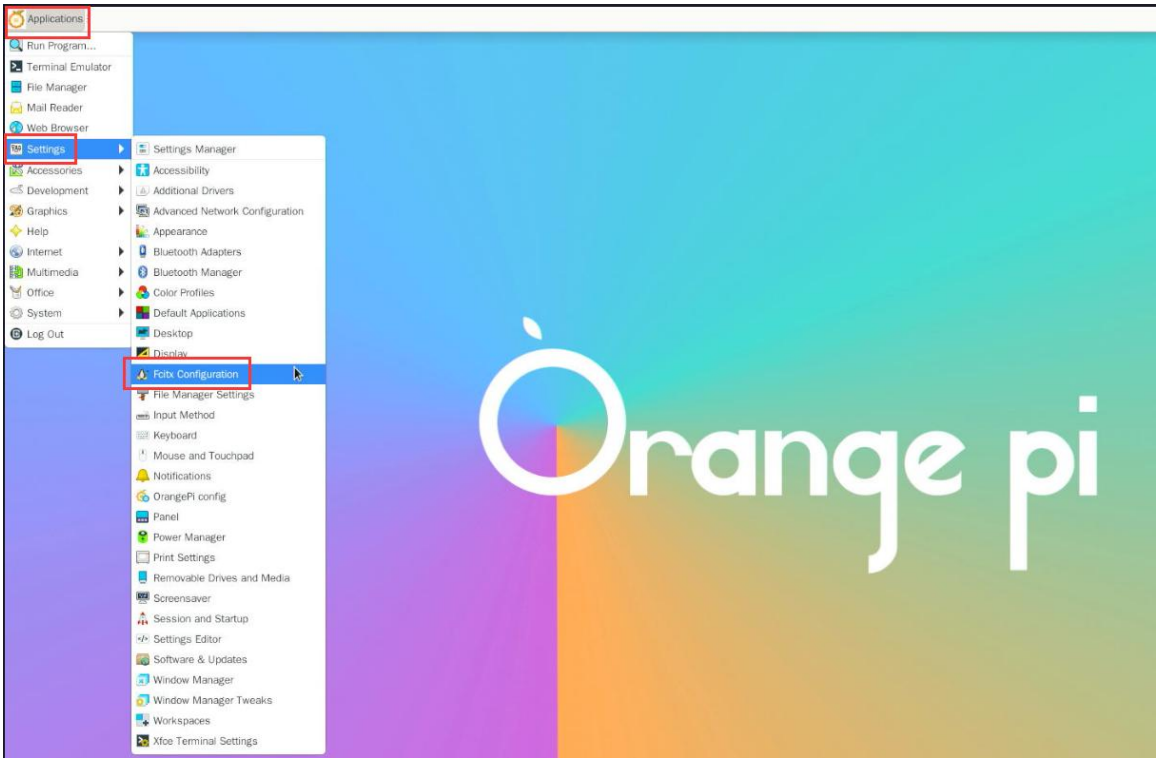


6) Then select **OK**

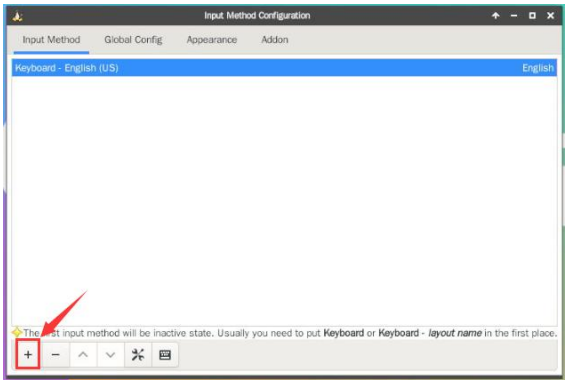


7) **Then restart the Linux system to make the configuration take effect**

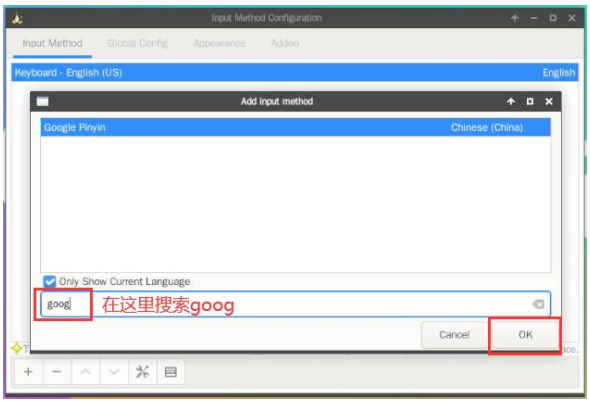
8) Then open **Fcitx configuration**



9) Then click the + sign in the position shown in the picture below

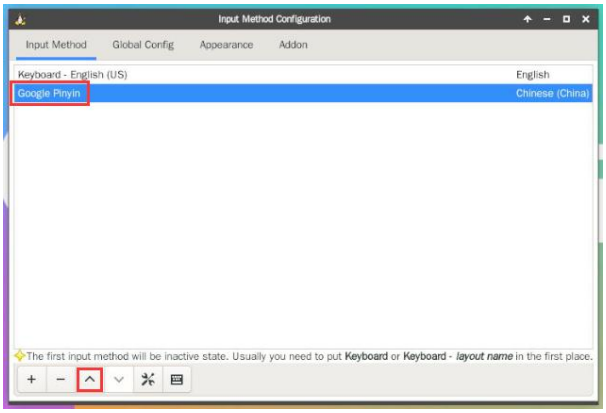


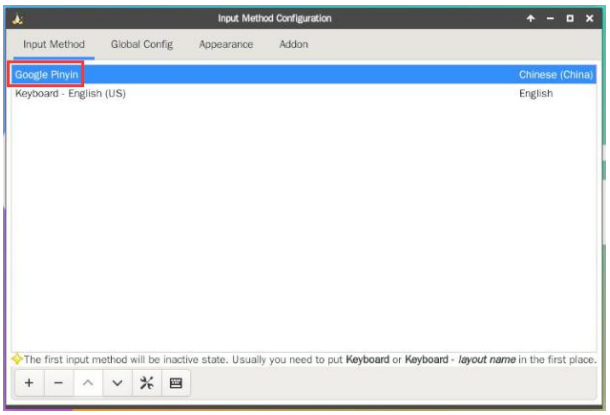
10) Then search **Google Pinyin** and click **OK**



Note: If you can't see the Google Pinyin option, please try to remove the √ in front of Only Show Current Languageq.

11) Then put **Google Pinyin** at the front

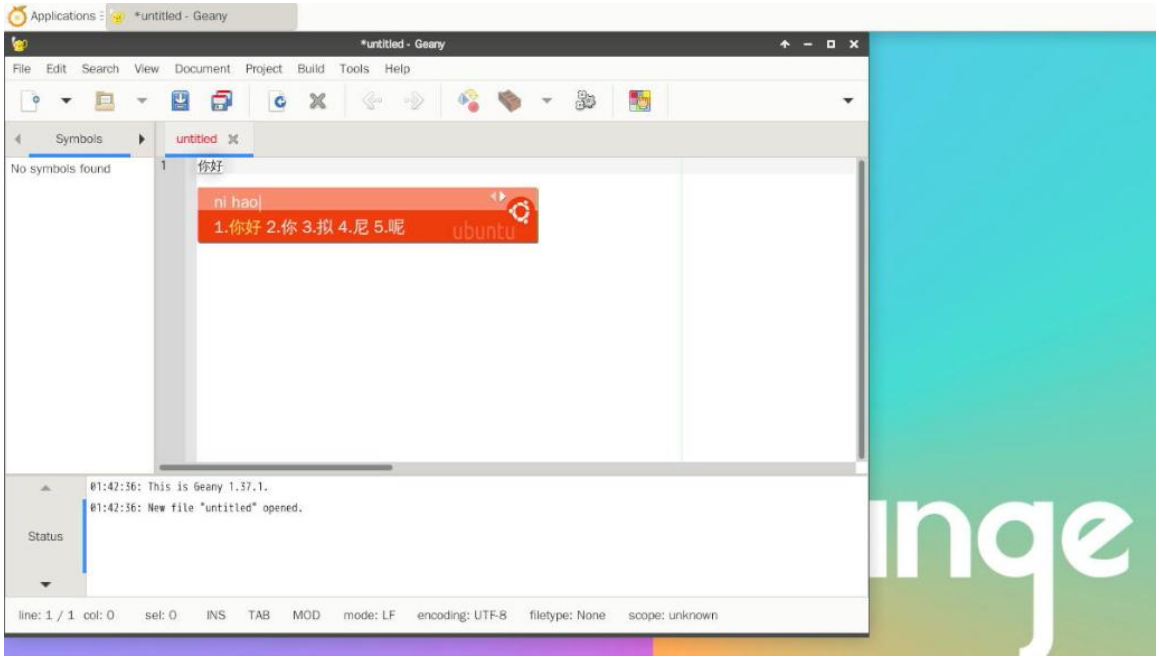




12) Then open the **Geany** editor to test the Chinese input method



13) The Chinese input method test is as follows



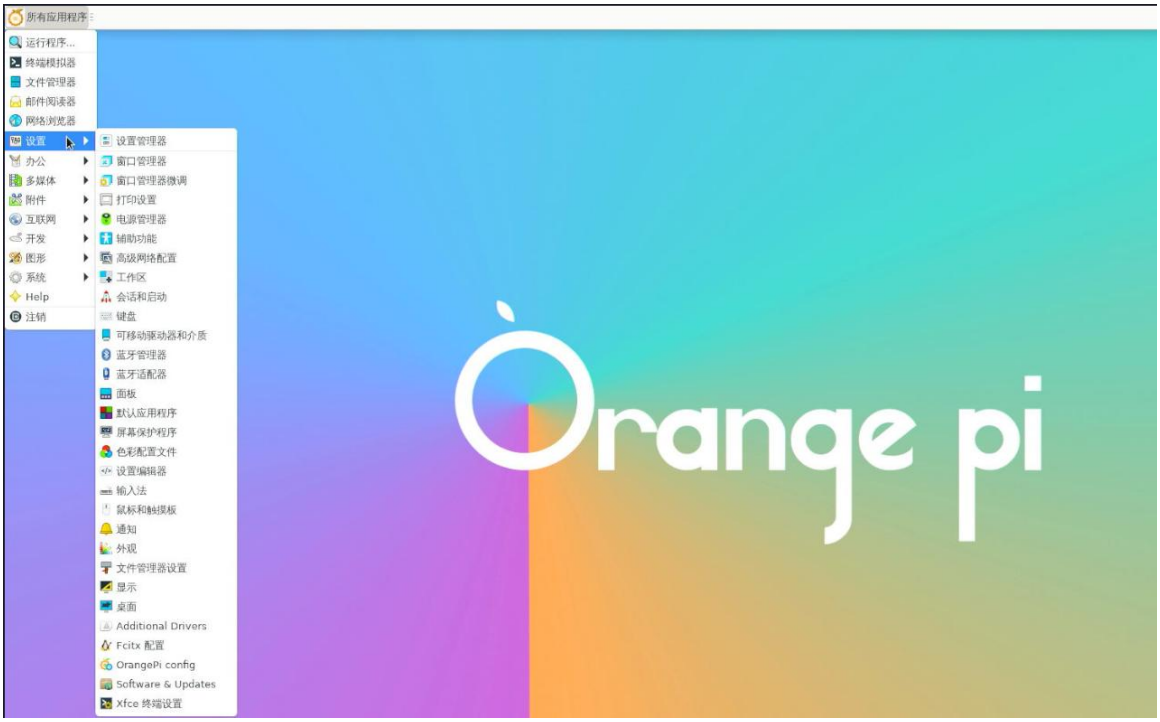
14) Use the **Ctrl+Space** shortcut key to switch between Chinese and English input methods

15) If you need the entire system to display in Chinese, you can set the variables in **/etc/default/locale** to **zh_CN.UTF-8**

```
orangepi@orangepi:~$ sudo vim /etc/default/locale
#    File generated by update-locale
LC_MESSAGES=zh_CN.UTF-8
LANG=zh_CN.UTF-8
```

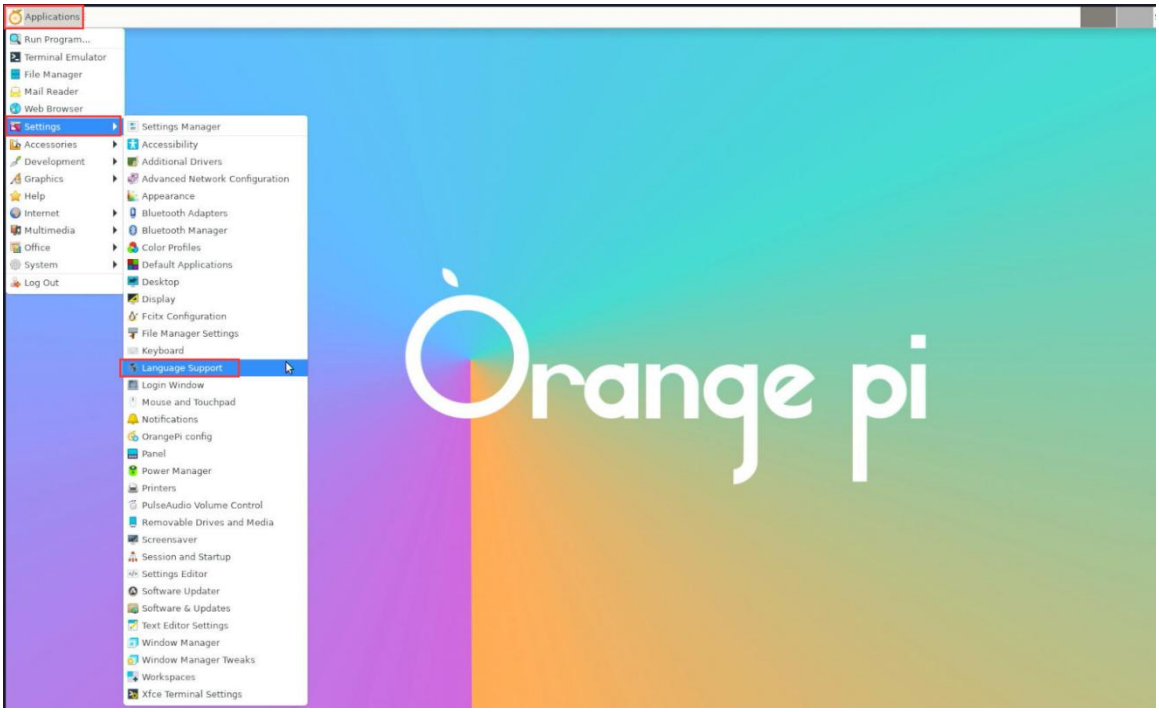
```
LANGUAGE=zh_CN.UTF-8
```

16) Then **restart the system** and you will see that the system is displayed in Chinese

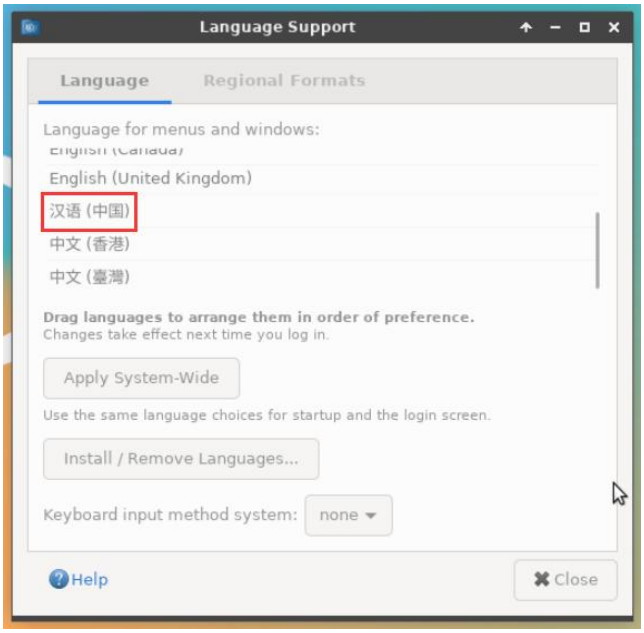


3. 26. 2. How to install Ubuntu 20.04 system

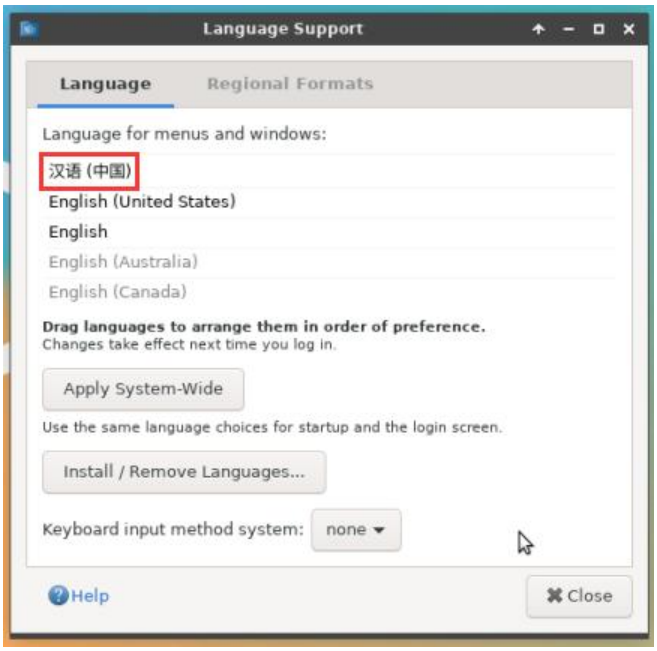
1) First open **Language Support**



2) Then find the **Chinese (China)** option

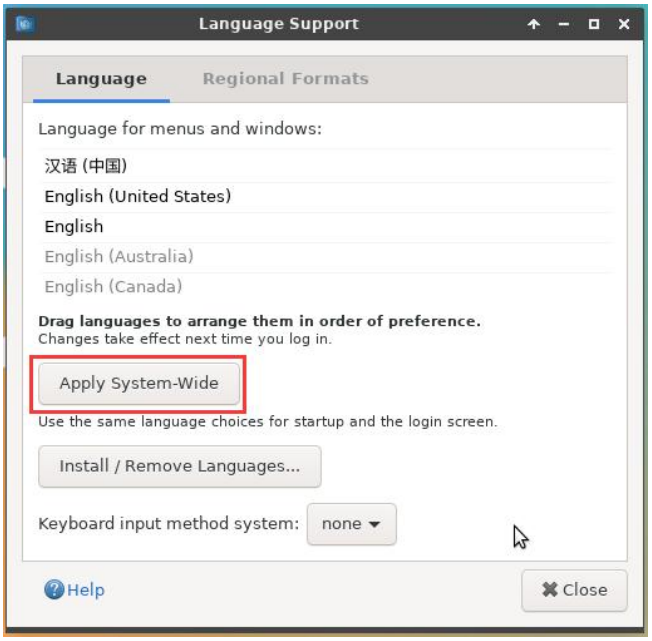


3) Then use the left mouse button to select **Chinese (China)** and hold it down, then drag it upwards to the starting position. The display after dragging is as shown below:



Note that this step is not easy to drag, please be patient and try a few more times.

4) Then select **Apply System-Wide** to apply the Chinese settings to the entire system



5) Then set the **Keyboard input method system** to **fcitx**



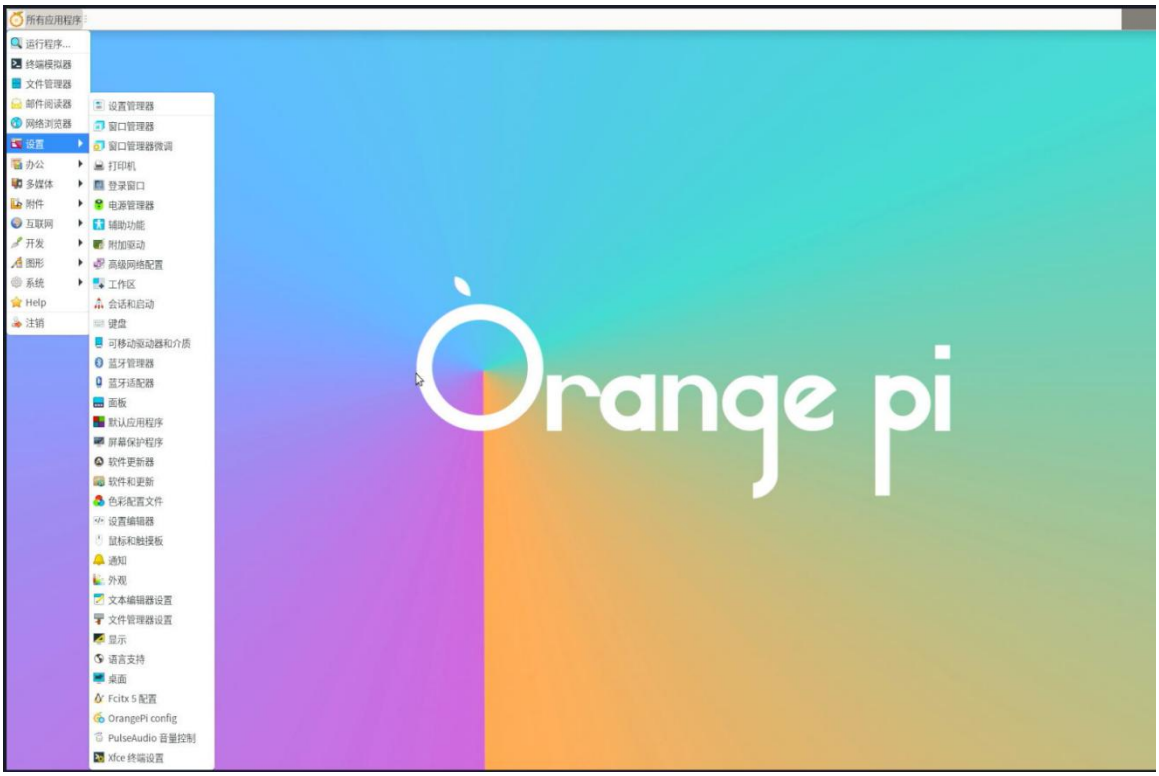
6) **Then restart the Linux system to make the configuration take effect**

7) After re-entering the system, please select "**Do not ask me again**" in the following interface, and then decide whether to update the standard folder to Chinese according to

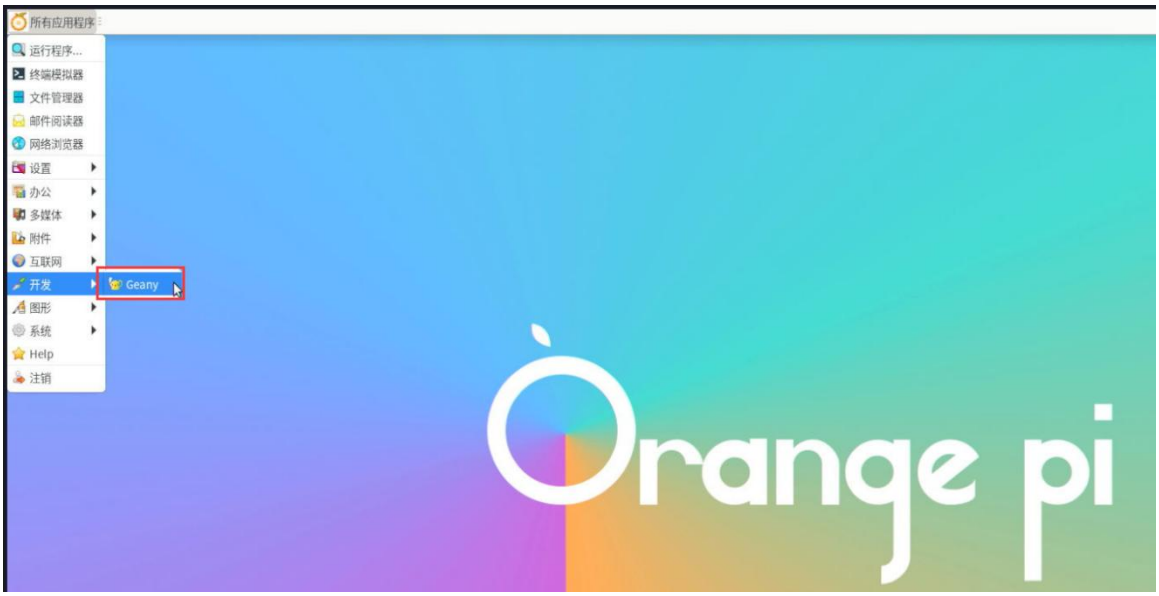
your own preferences.



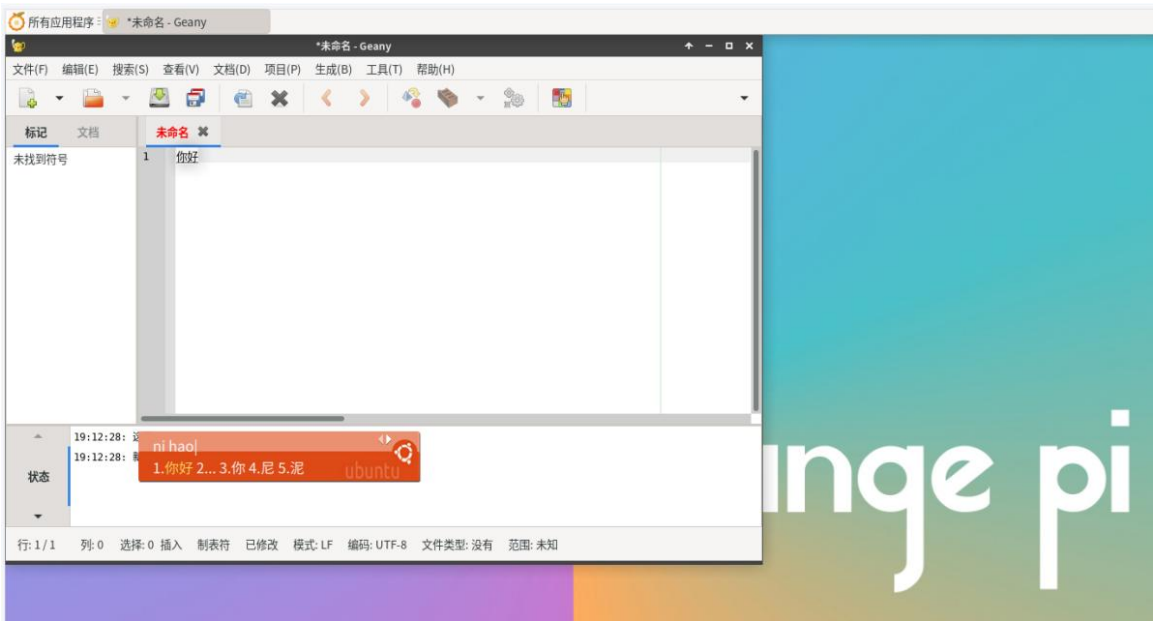
8) Then you can see that the desktop is displayed in Chinese



9) Then we can open **Geany** to test the Chinese input method. The opening method is as shown in the figure below

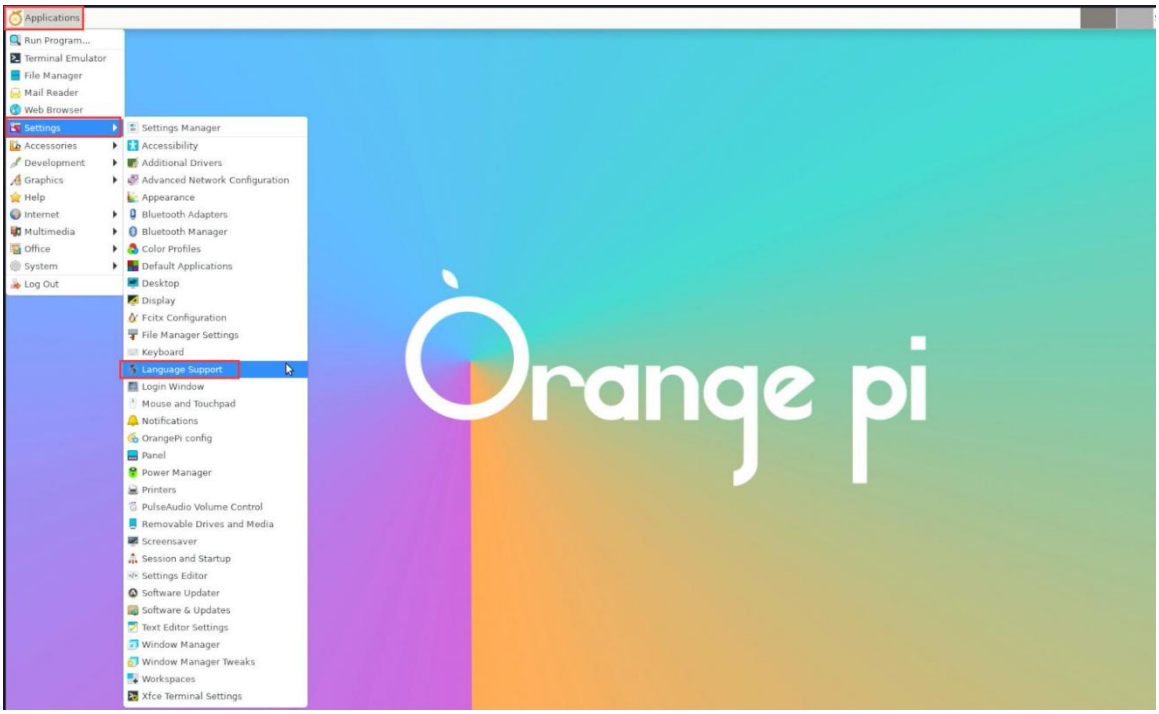


10) After opening **Geany**, the default input method is still English. We can switch to Chinese input method by pressing **Ctrl+Space**, and then we can input Chinese.

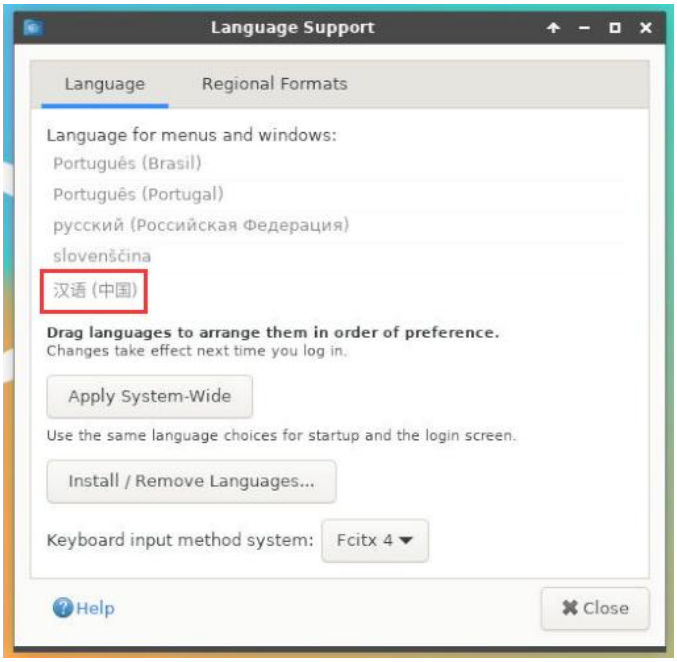


3. 26. 3. Installation method for Ubuntu 22.04 system

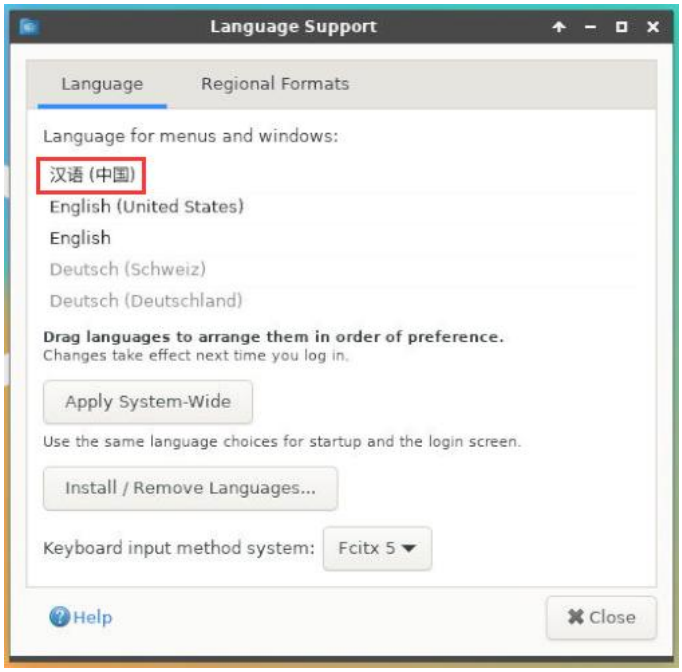
1) First open **Language Support**



2) Then find the **Chinese (China)** option

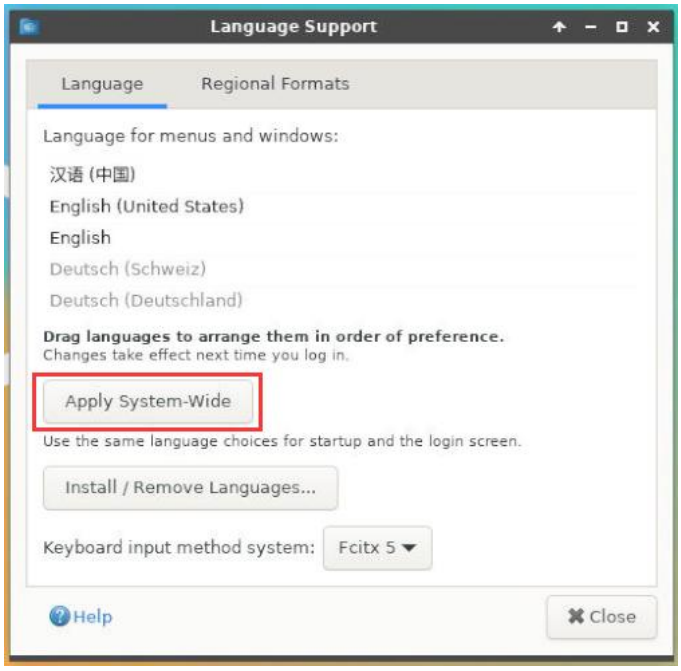


3) Then use the left mouse button to select **Chinese (China)** and hold it down, then drag it upwards to the starting position. The display after dragging is as shown below:



Note that this step is not easy to drag, please be patient and try a few more times.

4) Then select **Apply System-Wide** to apply the Chinese settings to the entire system

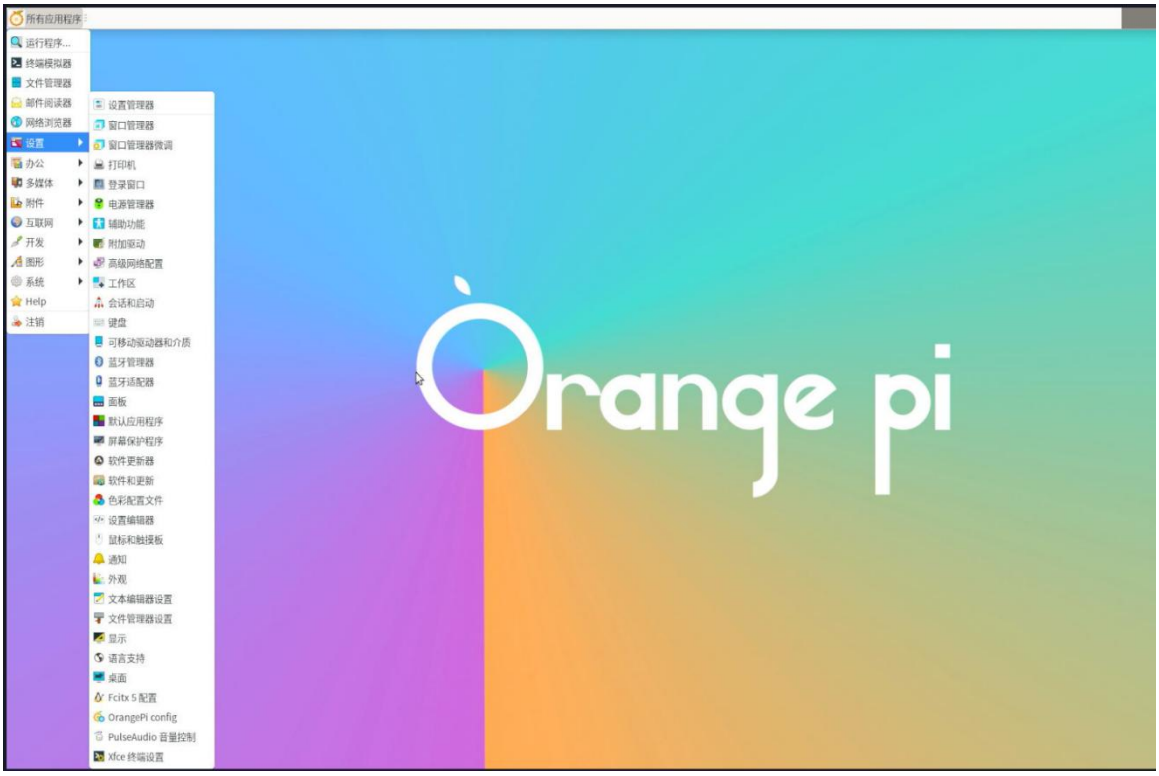


5) **Then restart the Linux system to make the configuration take effect**

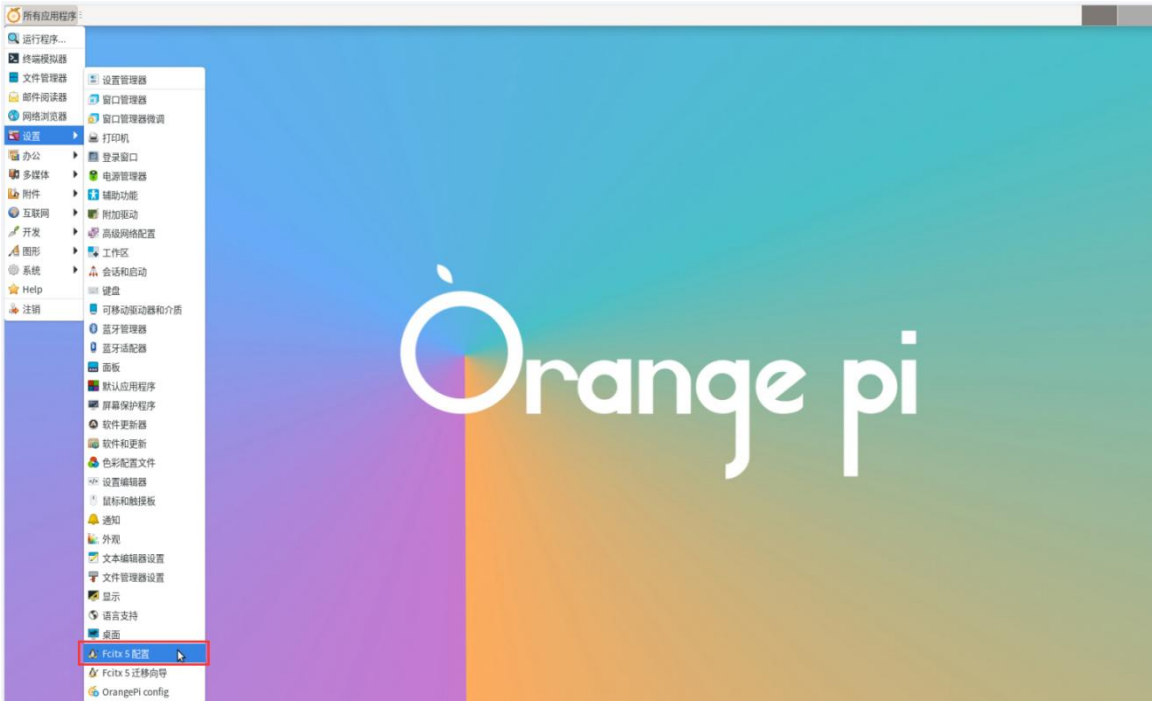
6) After re-entering the system, please select "**Do not ask me again**" in the following interface, and then decide whether to update the standard folder to Chinese according to your own preferences.



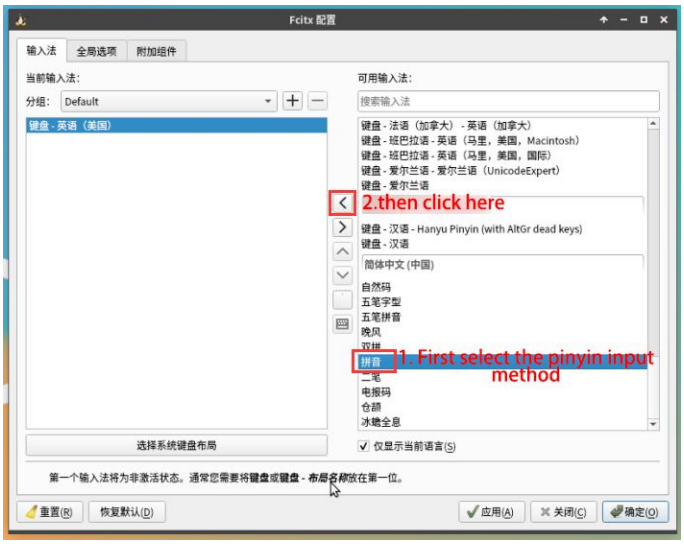
7) Then you can see that the desktop is displayed in Chinese



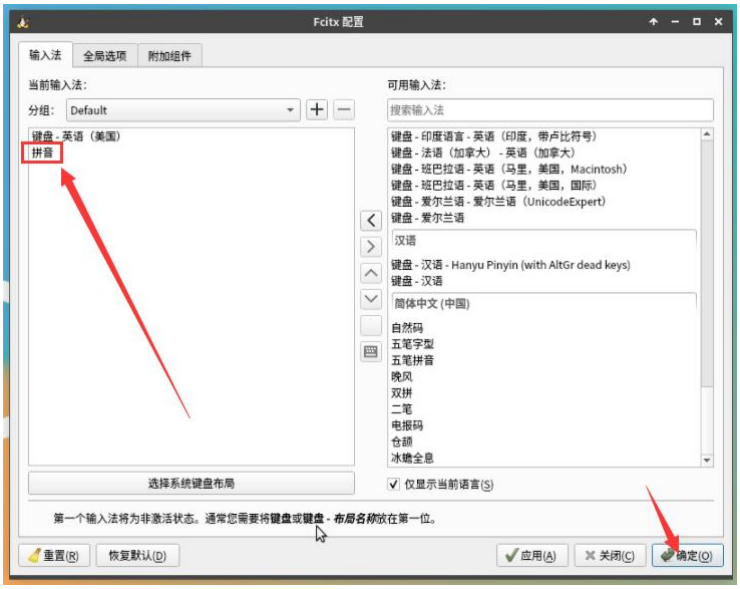
8) Then open the Fcitx5 configuration program



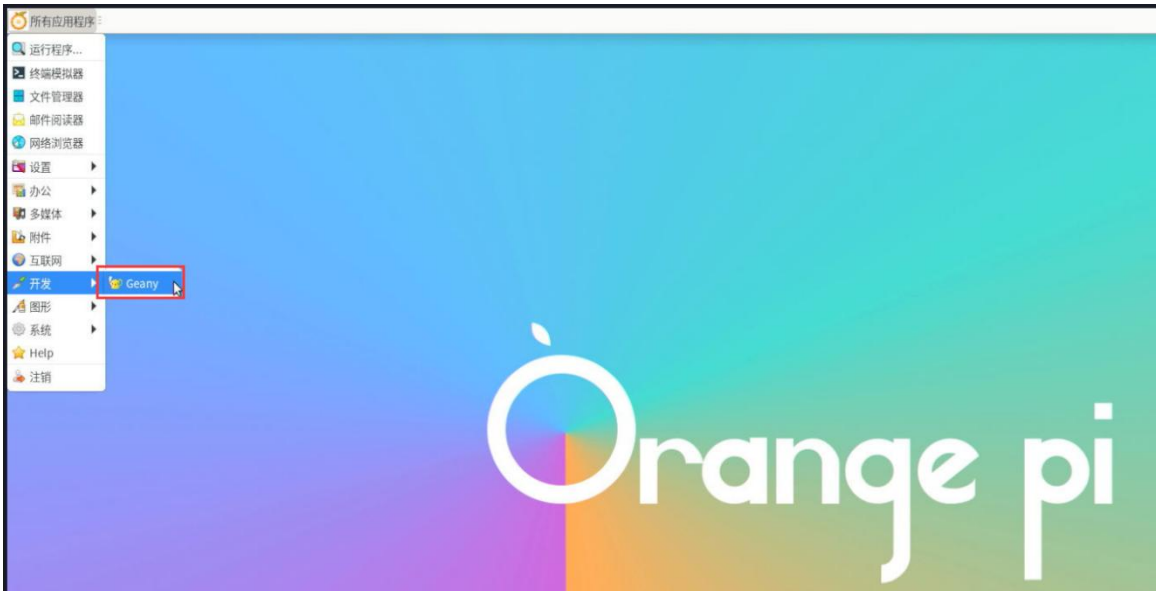
9) Then select Pinyin input method



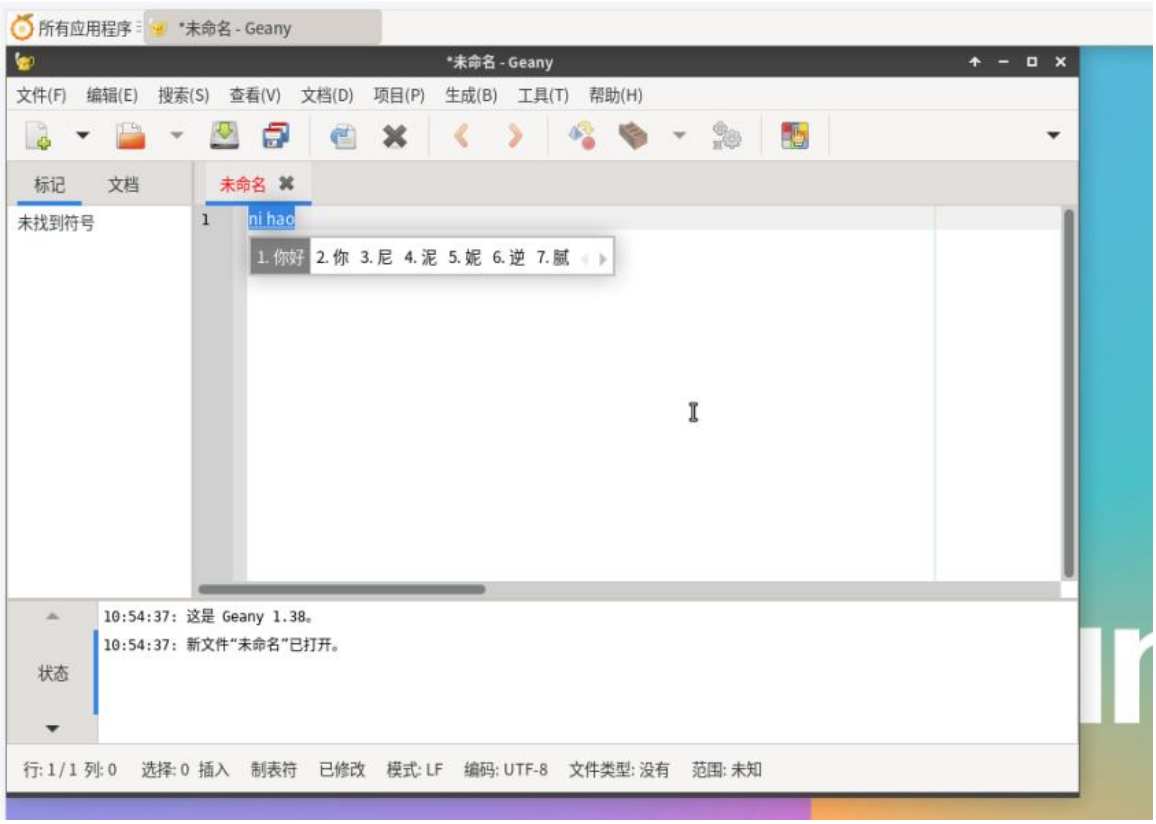
10) The interface after selection is as shown below, then click OK



11) Then we can open **Geany** to test the Chinese input method. The opening method is as shown in the figure below



12) After opening **Geany,** the default input method is still English. We can switch to Chinese input method by pressing **Ctrl+Space**, and then we can input Chinese.



3. 27. How to remotely log in to the Linux system desktop

The Ubuntu Gnome Wayland image does not support remote desktop login using Nomachine and VNC as described here.

3. 27. 1. Remote login using NoMachine

Please make sure that the Ubuntu or Debian system installed on the development board is a desktop version. In addition, NoMachine also provides detailed usage documentation. It is strongly recommended to read this document to familiarize yourself with the use of NoMachine. The document link is as follows:

https://knowledgebase.nomachine.com/DT10R00166

NoMachine supports Windows, Mac, Linux, iOS and Android platforms, so we can use NoMachine to remotely log in and control the Orange Pi development board on multiple 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.

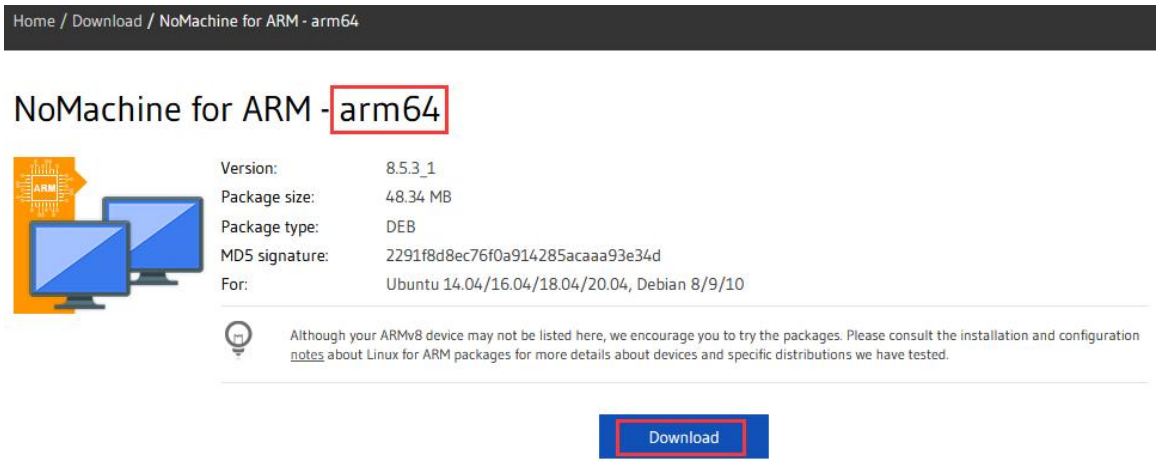
Before operation, please make sure that the Windows computer and the development board are in the same LAN and can log in to the Ubuntu or Debian system of the development board normally through SSH.

1) First download the installation package of the NoMachine software Linux **arm64** deb version, and then install it into the Linux system of the development board

a. Since RK3588 is an ARMv8 SOC, we use Ubuntu or Debian as the system, so 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://downloads.nomachine.com/download/?id=114&distro=ARM

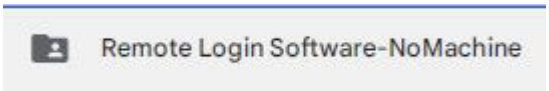


b. In addition, you can also download the **NoMachine** installation package in the **official tool**

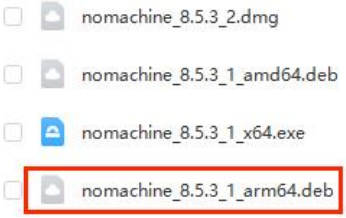
a) First open the development board download page and enter the **official tool**



b) First enter the **remote login software-NoMachine** folder



c) Then download the arm64 version of the deb installation package



c. Then upload the downloaded **nomachine_x.x.x_x_arm64.deb** to the Linux system of the development board

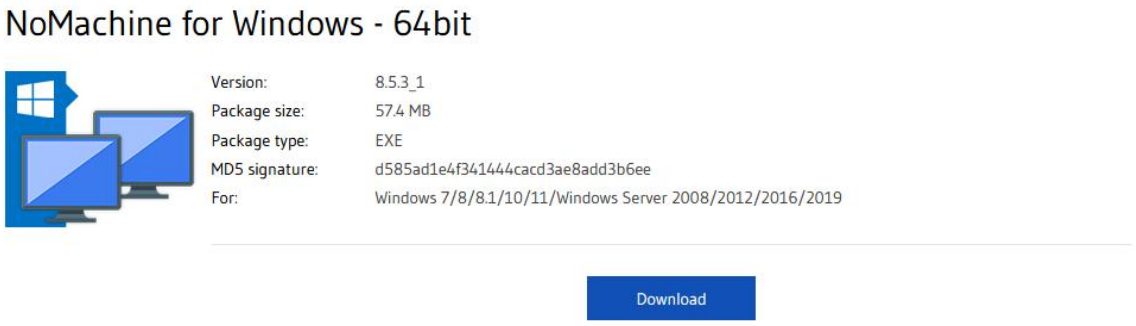
d. Then use the following command to install **NoMachine** in the Linux system of the development board

```
orangepi@orangepi:~$ sudo dpkg -i nomachine_x.x.x_x_arm64_arm64.deb
```

2) Then download the installation package of NoMachine software Windows version. The download address is as follows

Note that this download link may change.

https://downloads.nomachine.com/download/?id=9



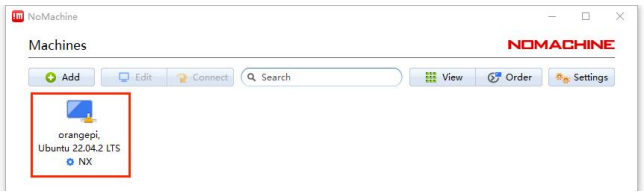
3) Then install NoMachine in Windows. **After installation, please restart your computer**

4) Then open **NoMachine** in Windows

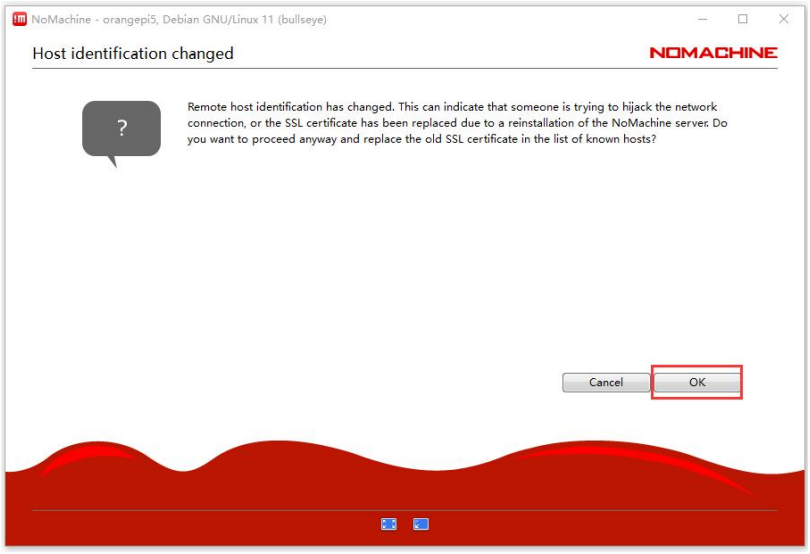


5) After NoMachine is started, it will automatically scan other devices with NoMachine

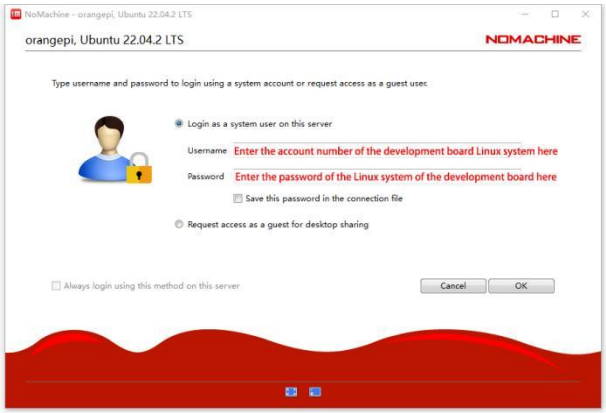
installed in the LAN. After entering the main interface of NoMachine, you can see that the development board is already in the list of connectable devices. Then click the location indicated by the red box in the figure below to start logging into the Linux system desktop of the development board.



6) Then click **OK**



7) Then enter the user name and password of the development board Linux system in the corresponding position in the figure below, and then click **OK** to start logging in



8) Then click OK in the following interface.

9) Finally, you can see the desktop of the development board Linux system



3.27.2. Remote login using VNC

Before operation, please make sure that the Windows computer and the development board are in the same LAN and can log in to the Ubuntu or Debian system of the development board normally through SSH.

There are many problems with testing VNC on Ubuntu 20.04, so please do not use this method.

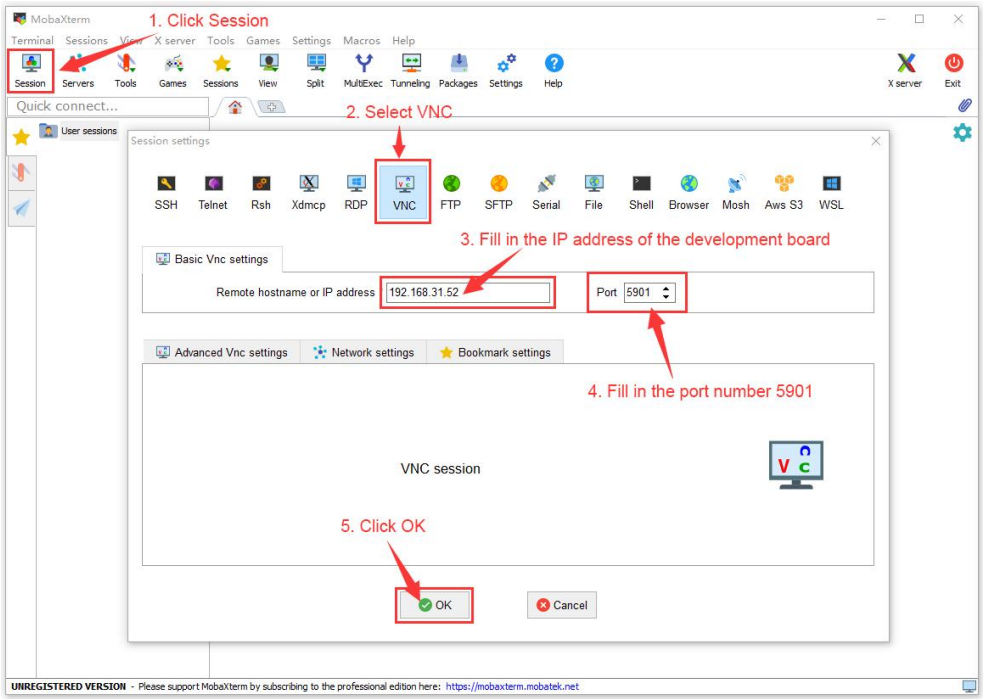
1) 首先 First run the **set_vnc.sh** script to set up vnc, **remember to add sudo permissions**

```
orangepi@orangepi:~$ sudo set_vnc.sh
You will require a password to access your desktops.

Password:          #Set the vnc password here, 8 characters
Verify:            #Set the vnc password here, 8 characters
Would you like to enter a view-only password (y/n)? n
xauth:    file /root/.Xauthority does not exist

New 'X' desktop is orangepi5ultra:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/orangepi5ultra:1.log
```

```
Killing Xtightvnc process ID 3047

New 'X' desktop is orangepi5ultra:1

Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/orangepi5ultra:1.log
```

2) The steps to use MobaXterm software to connect to the Linux system desktop of the development board are as follows:

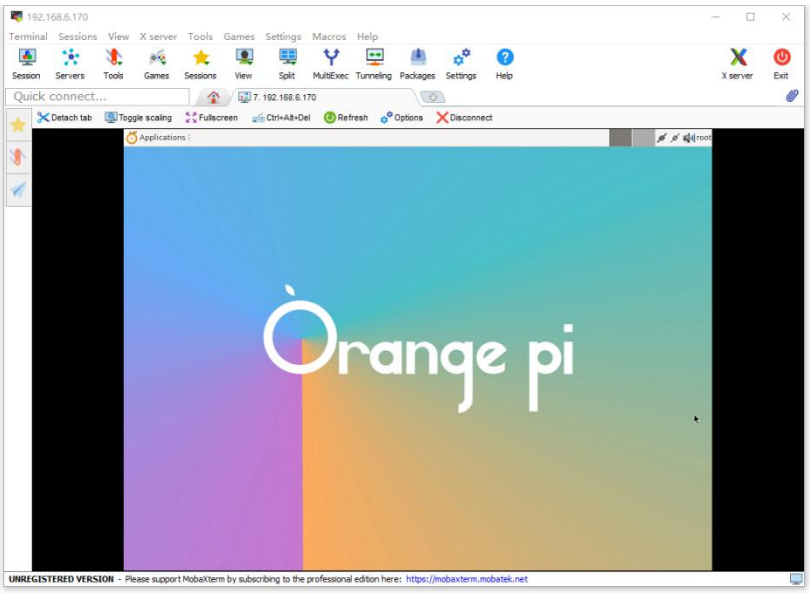
a. First click Session, then select VNC, then fill in the IP address and port of the development board, and finally click OK to confirm



b. Then enter the VNC password set previously



c. After successful login, the interface is displayed as shown below, and then you can remotely operate the desktop of the development board Linux system



3. 28. Test of some programming languages supported by Linux system

3. 28. 1. Debian Bullseye System

1) Debian Bullseye is installed with the gcc compilation tool chain by default, which can compile C language programs directly in the Linux system of the development board

a. The version of gcc is as follows

```
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

```
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**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Debian Bullseye has Python3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ 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 Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Bullseye does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk. The latest version in Debian Bullseye is openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the Java version

```
orangepi@orangepi:~$ java --version
```

c. Write a Java version **hello_world.java**

```
orangepi@orangepi:~$ 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**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3.28.2. Debian Bookworm System

1) Debian Bookworm is installed with the gcc compilation toolchain by default, which can compile C language programs directly in the Linux system of the development board

a. gcc version is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 12.2.0-14) 12.2.0
Copyright (C) 2022 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**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Debian Bookworm has Python 3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.11.2 (main, Mar 13 2023, 12:18:29) [GCC 12.2.0] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Use the Ctrl+D shortcut key to exit Python's interactive mode.

b. Write the **hello_world.py** program in Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Bookworm does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk. The latest version in Debian Bookworm is openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the Java version

```
orangepi@orangepi:~$ java --version
```

c. Write a Java version **hello_world.java**

```
orangepi@orangepi:~$ 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**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3. 28. 3. Ubuntu Focal system

1) Ubuntu Focal is installed with the gcc compilation tool chain by default, which can compile C language programs directly in the Linux system of the development board

a. gcc version is as follows

```
orangepi@orangepi:~$ gcc --version
```

```
gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0
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

```
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**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Ubuntu Focal has Python 3 installed by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Ubuntu Focal does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the Java version

```
orangepi@orangepi:~$ java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu220.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu220.04, mixed mode, sharing)
```

c. Write a Java version **hello_world.java**

```
orangepi@orangepi:~$ 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**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3.28.4. Ubuntu Jammy System

4) Ubuntu Jammy is installed with the gcc compilation tool chain by default, which can compile C language programs directly in the Linux system of the development board

a. gcc version is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
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

```
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**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

5) Ubuntu Jammy has Python 3 installed by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.10.12 (main, Jul 29 2024, 16:56:48) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

6) Ubuntu Jammy does not install Java compilation tools and runtime environment by default

a. You can install openjdk-18 using the following command

```
orangepi@orangepi:~$ sudo apt install -y openjdk-18-jdk
```

b. After installation, you can check the Java version

```
orangepi@orangepi:~$ java --version
openjdk 18.0.2-ea 2022-07-19
OpenJDK Runtime Environment (build 18.0.2-ea+9-Ubuntu-222.04)
OpenJDK 64-Bit Server VM (build 18.0.2-ea+9-Ubuntu-222.04, mixed mode, sharing)
```

c. Write a Java version **hello_world.java**

```
orangepi@orangepi:~$ 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**

```
orangepi@orangepi:~$ javac hello_world.java
orangepi@orangepi:~$ java hello_world
Hello World!
```

3. 29. How to install QT

1) Use the following script to install QT5 and QT Creator

```
orangepi@orangepi:~$ install_qt.sh
```

2) After installation, the QT version number will be automatically printed

a. The Qt version that comes with Ubuntu 20.04 is **5.12.8**

```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.12.8 in /usr/lib/aarch64-linux-gnu
```

b. The QT version that comes with Ubuntu 22.04 is **5.15.2**

```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.15.2 in /usr/lib/aarch64-linux-gnu
```

c. The QT version that comes with Debian11 is **5.15.2**

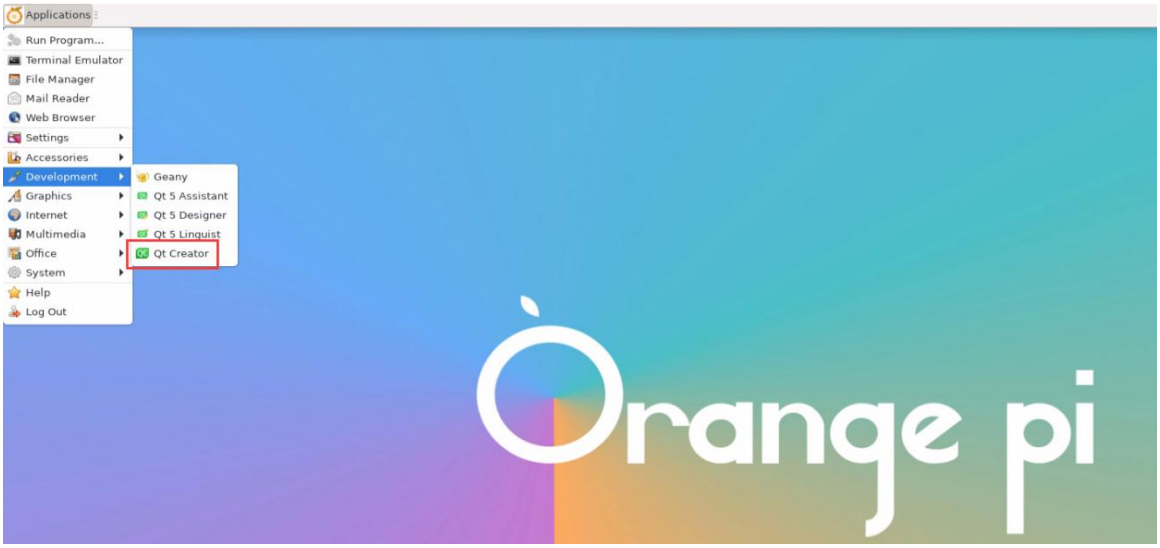
```
orangepi@orangepi:~$ install_qt.sh
......
QMake version 3.1
Using Qt version 5.15.2 in /usr/lib/aarch64-linux-gnu
```

d. The QT version that comes with Debian12 is **5.15.8**

```
orangepi@orangepi:~$ install_qt.sh
......
```

```
QMake version 3.1
Using Qt version 5.15.8 in /usr/lib/aarch64-linux-gnu
```

3) Then you can see the QT Creator startup icon in **Applications**



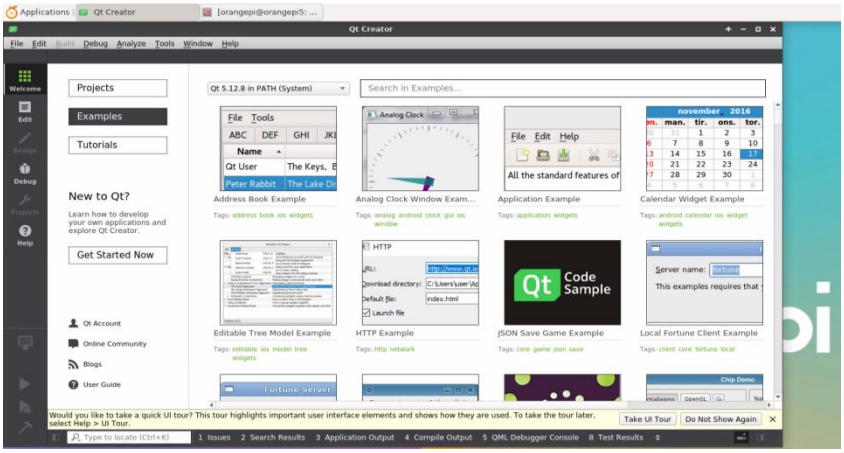
You can also use the following command to open QT Creator

```
orangepi@orangepi:~$ qtcreator
```

During the startup of QT and QT applications, if the following error is prompted, please ignore it directly. This error will not affect the operation of the application.

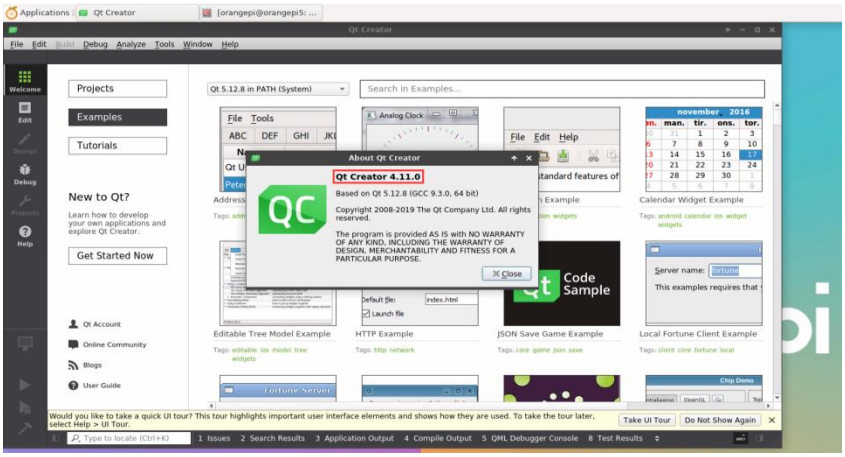
libGL error: failed to create dri screen
libGL error: failed to load driver: rockchip
libGL error: failed to create dri screen
libGL error: failed to load driver: rockchip

4) The interface after QT Creator is opened is as follows

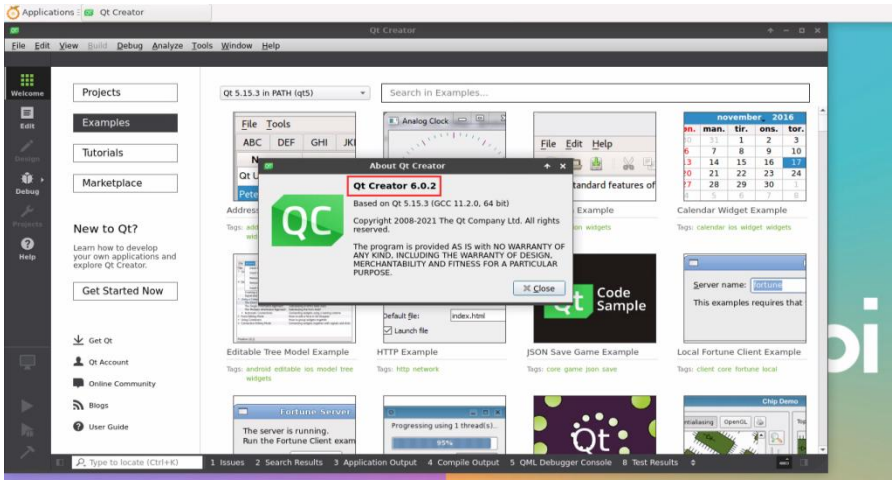


5) The version of QT Creator is as follows

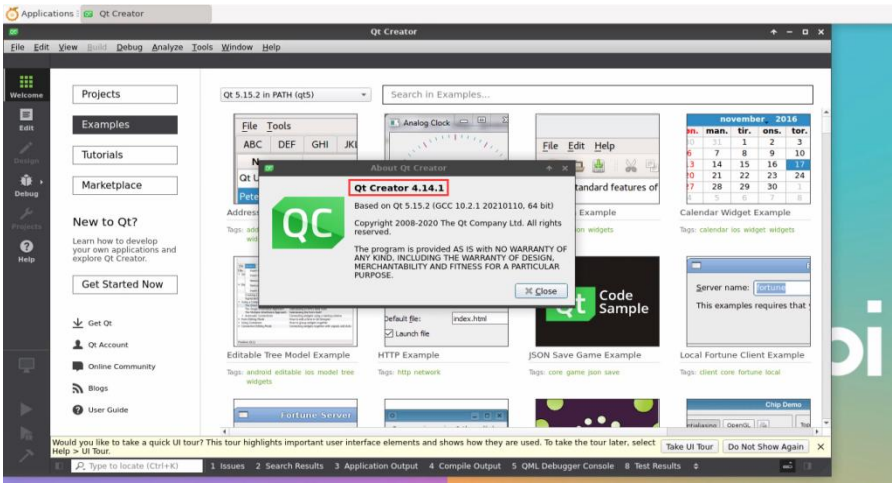
a. The default version of QT Creator in **Ubuntu 20.04** is as follows



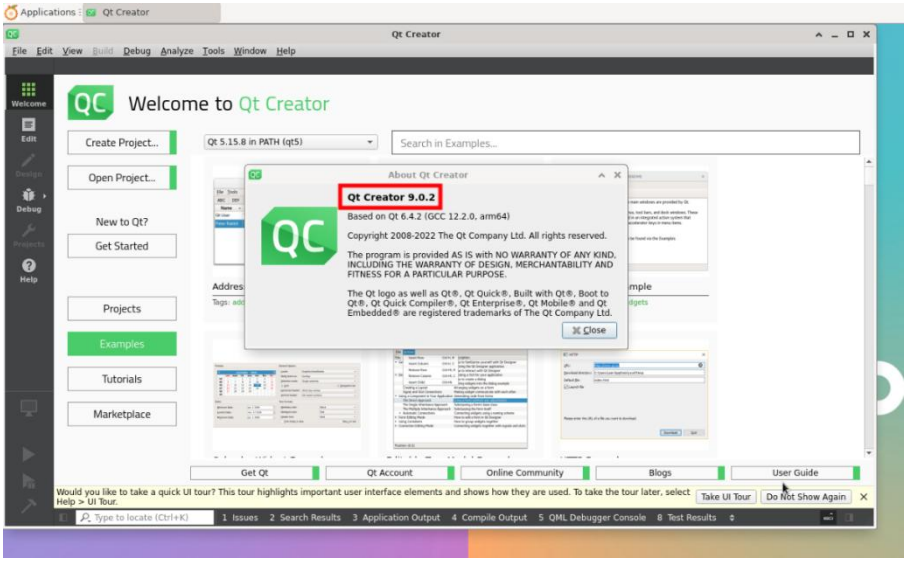
b. The default version of QT Creator in **Ubuntu 22.04** is as follows



c. The default version of QT Creator in **Debian 11** is as follows

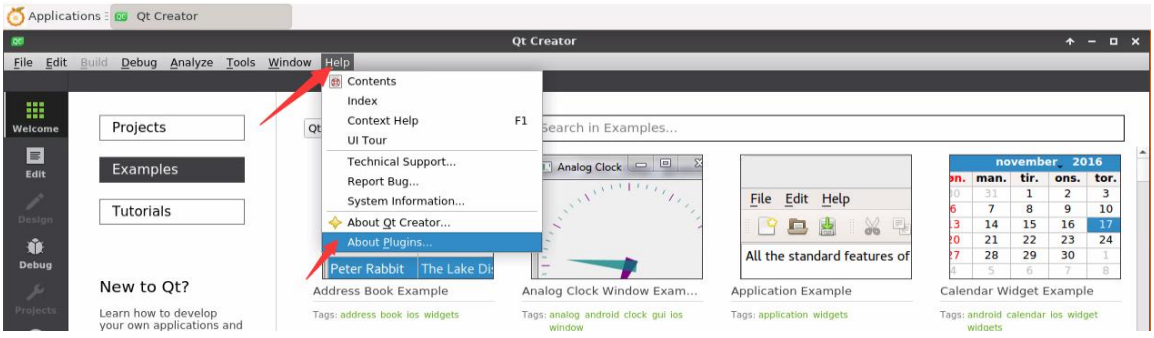


d. The default version of QT Creator in **Debian 12** is as follows

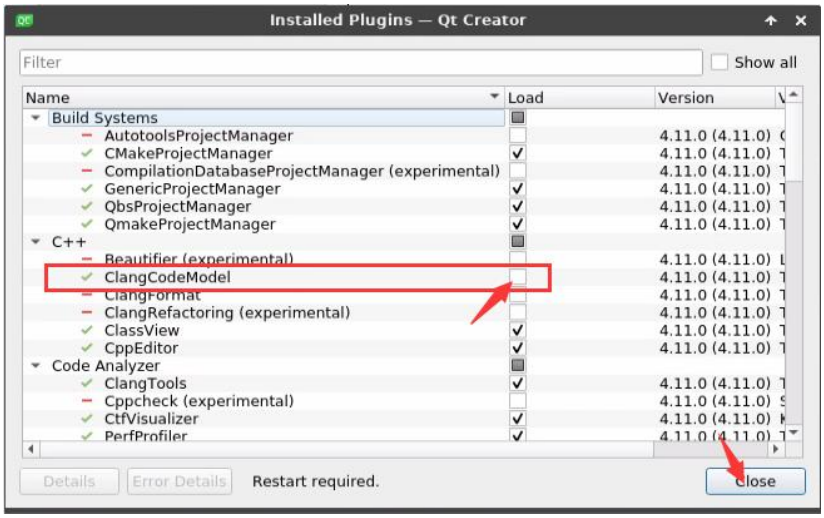


6) Then set up QT

a. First open **Help->About Plugins...**



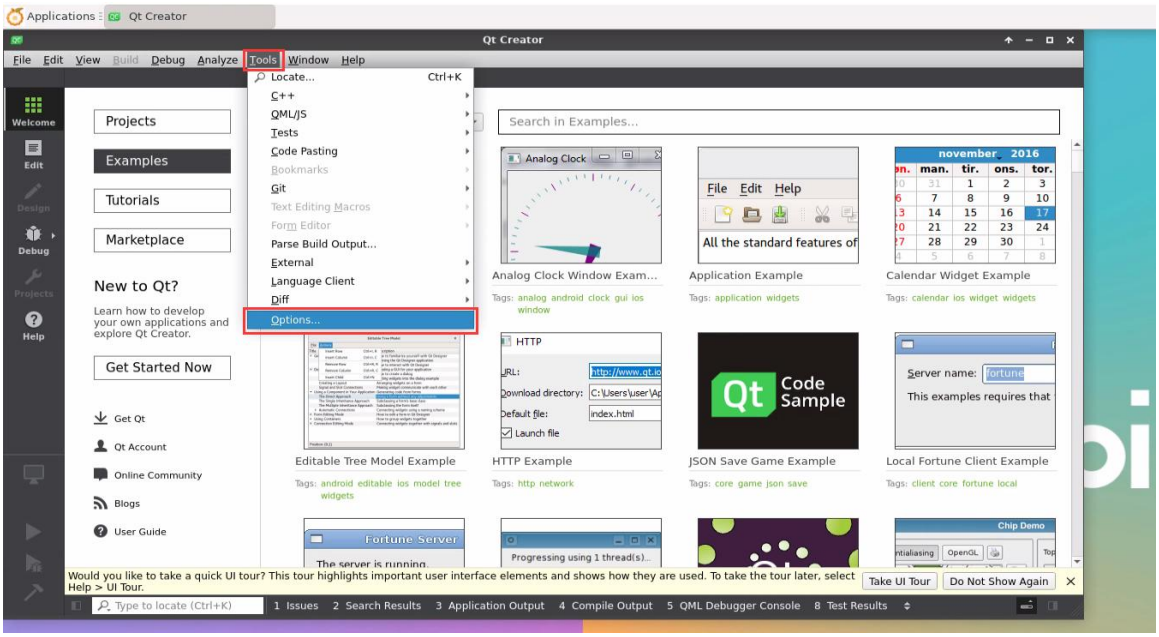
b. Then remove the check mark of **ClangCodeModel**

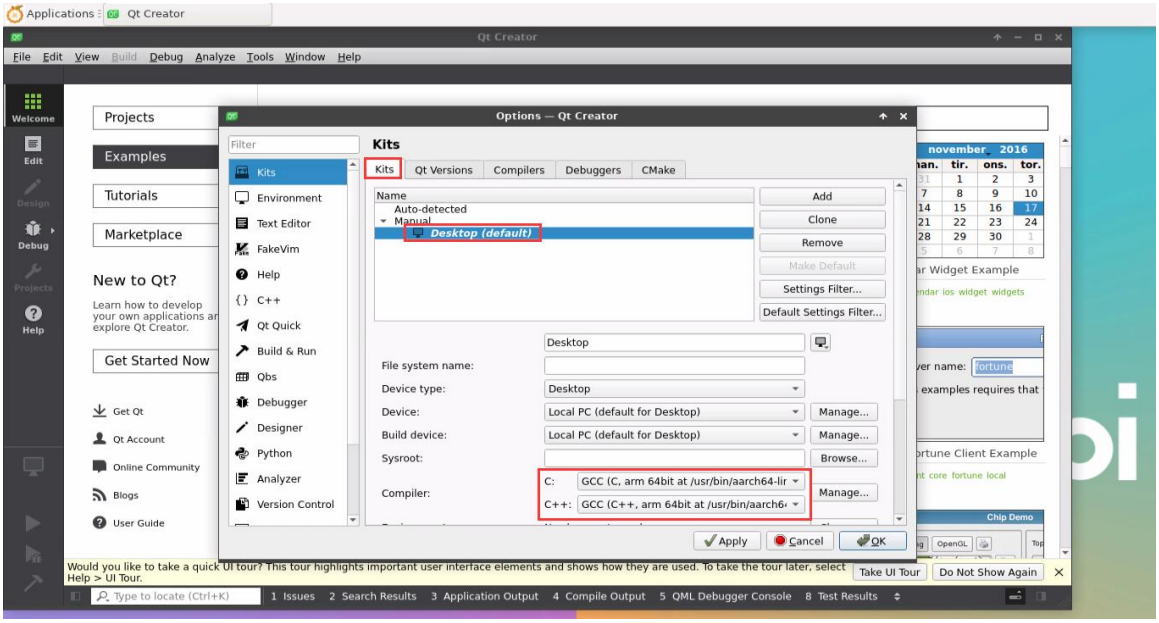


c. **After setting, you need to restart QT Creator**

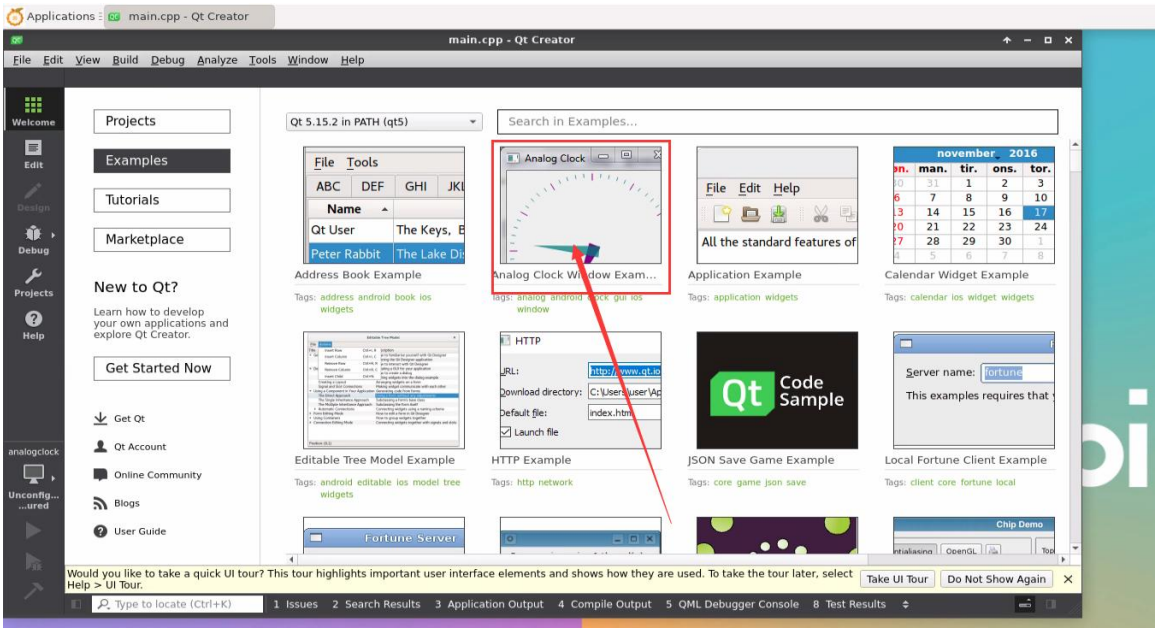
d. Then make sure that QT Creator uses the GCC compiler. If it defaults to Clang, change it to GCC

For Debian 12, please skip this step.

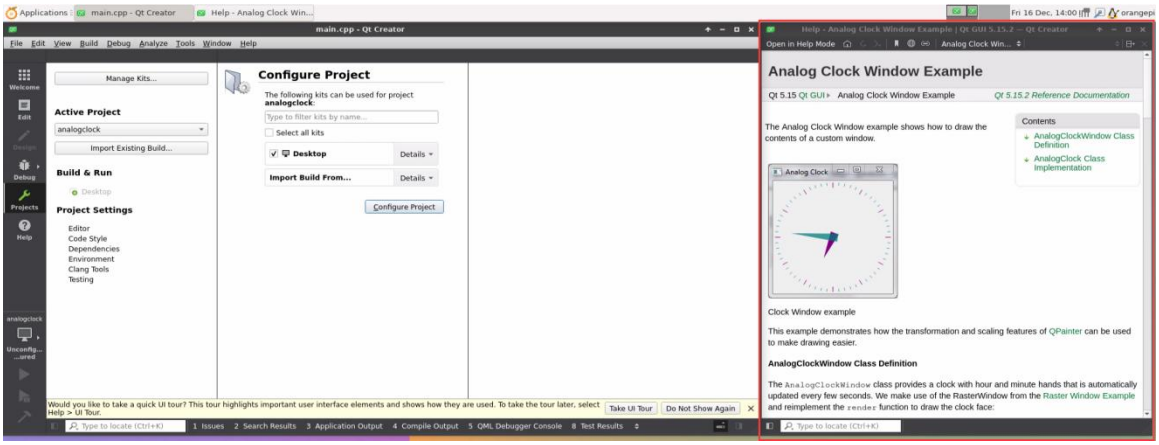




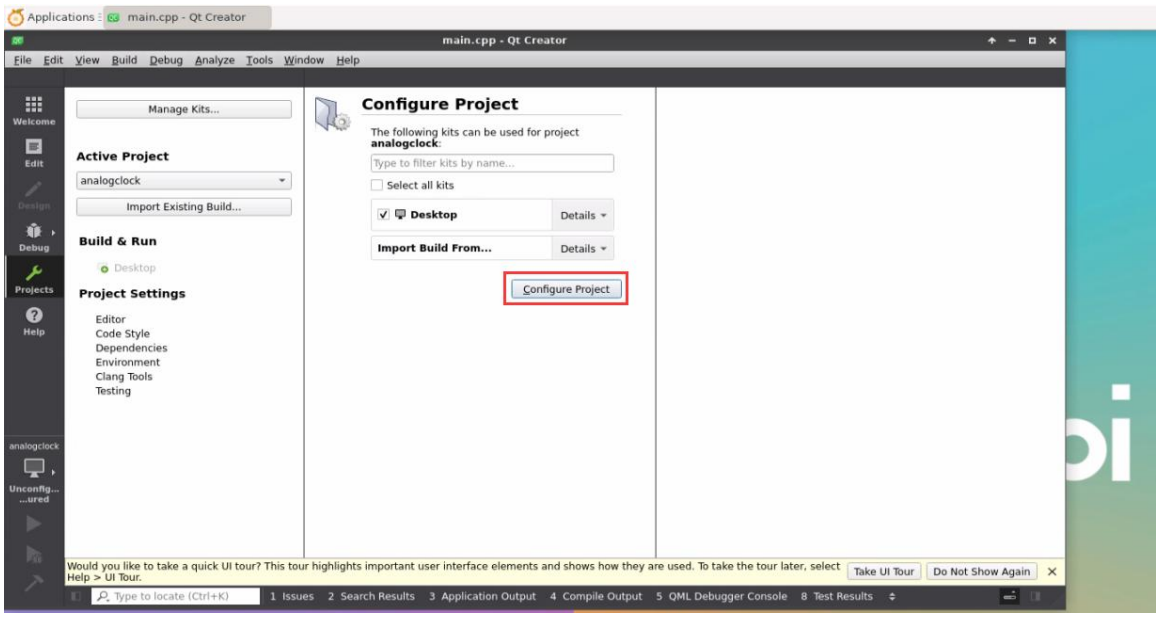
7) Then you can open a sample code



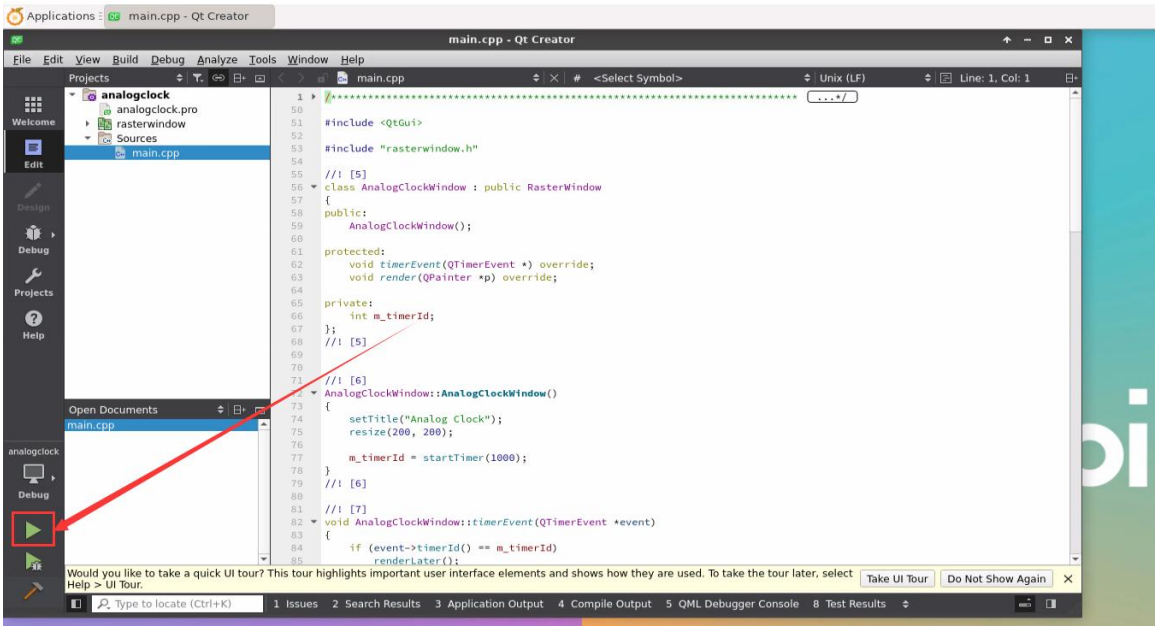
8) Clicking on the sample code will automatically open the corresponding documentation. Please read the instructions carefully



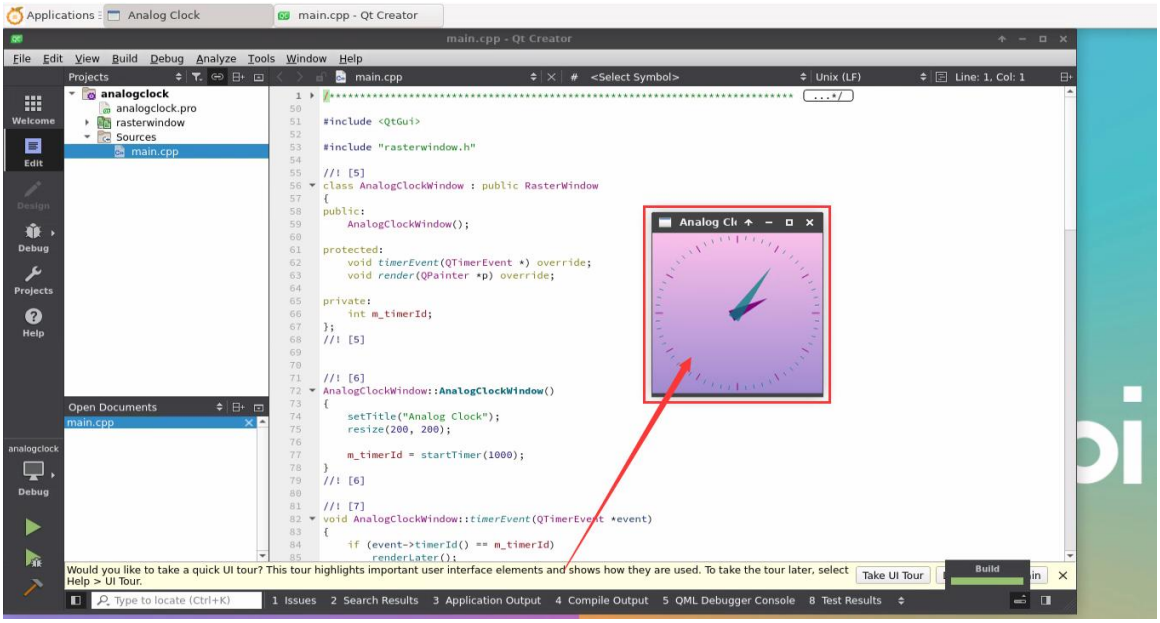
9) Then click Next **Configure Project**



10) Then click the green triangle in the lower left corner to compile and run the sample code



11) After waiting for a while, the interface shown in the figure below will pop up, which means that QT can compile and run normally



12) References

https://wiki.qt.io/Install_Qt_5_on_Ubuntu
https://download.qt.io/archive/qtcreator
https://download.qt.io/archive/qt

3. 30. ROS installation method

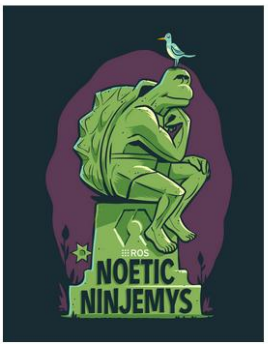
3. 30. 1. How to install ROS 1 Noetic on Ubuntu 20.04

1) The currently active versions of ROS 1 are as follows, and the recommended versions are **Noetic Ninjemys**

Active ROS 1 distributions

Recommended





Distro	Release date	Poster	*Tuturtle*, turtle in tutorial	EOL date
ROS Noetic Ninjemys (**Recommended**)	May 23rd, 2020			May, 2025 (Focal EOL)
ROS Melodic Morenia	May 23rd, 2018			May, 2023 (Bionic EOL)

http://docs.ros.org
https://wiki.ros.org/Distributions

2) The official installation document link for ROS 1 **Noetic Ninjemys** is as follows:

http://wiki.ros.org/noetic/Installation/Ubuntu

3) The official installation document of ROS **Noetic Ninjemys** recommends using Ubuntu 20.04, so make sure that the system used by the development board is the **Ubuntu 20.04 desktop system**

http://wiki.ros.org/noetic/Installation



4) Then use the following script to install ros1

```
orangepi@orangepi5ultra:~$ install_ros.sh ros1
```

5) 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 when running the following command, you need to ensure that the development board can access GitHub normally, otherwise an error will be reported due to network problems.

The install_ros.sh script will try to modify /etc/hosts and automatically run the following command. However, this method cannot guarantee that GitHub can be accessed normally every time. If the following error is prompted after install_ros.sh installs ros1, please find other ways to enable the Linux system of the development board to access GitHub normally, and then manually run the following command.

https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
ERROR: error loading sources list:
The read operation timed out

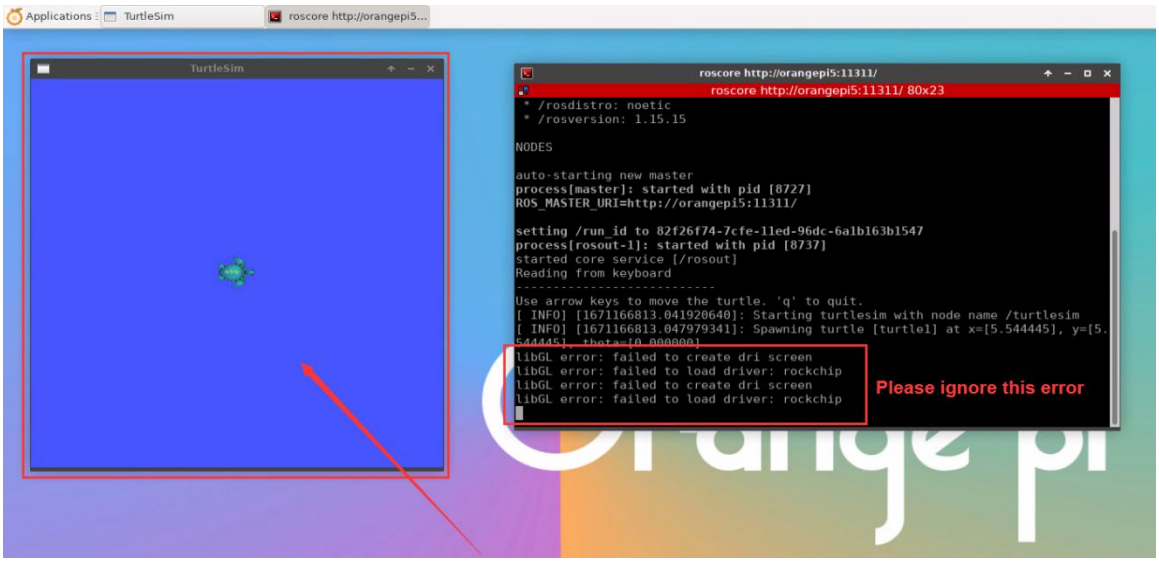
```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ sudo rosdep init
Wrote /etc/ros/rosdep/sources.list.d/20-default.list
Recommended: please run
```

```
            rosdep update
orangepi@orangepi:~$ 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
```

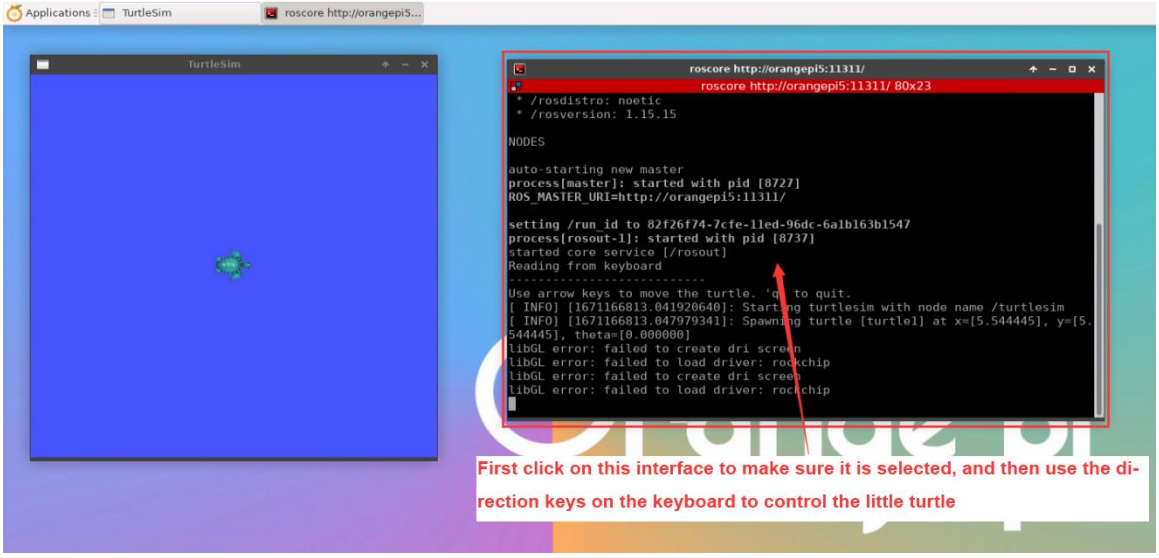
6) Then open a command line terminal window on the **desktop**, and use the **test_ros.sh** script to start a small turtle routine to test whether ROS can be used normally

```
orangepi@orangepi:~$ test_ros.sh
```

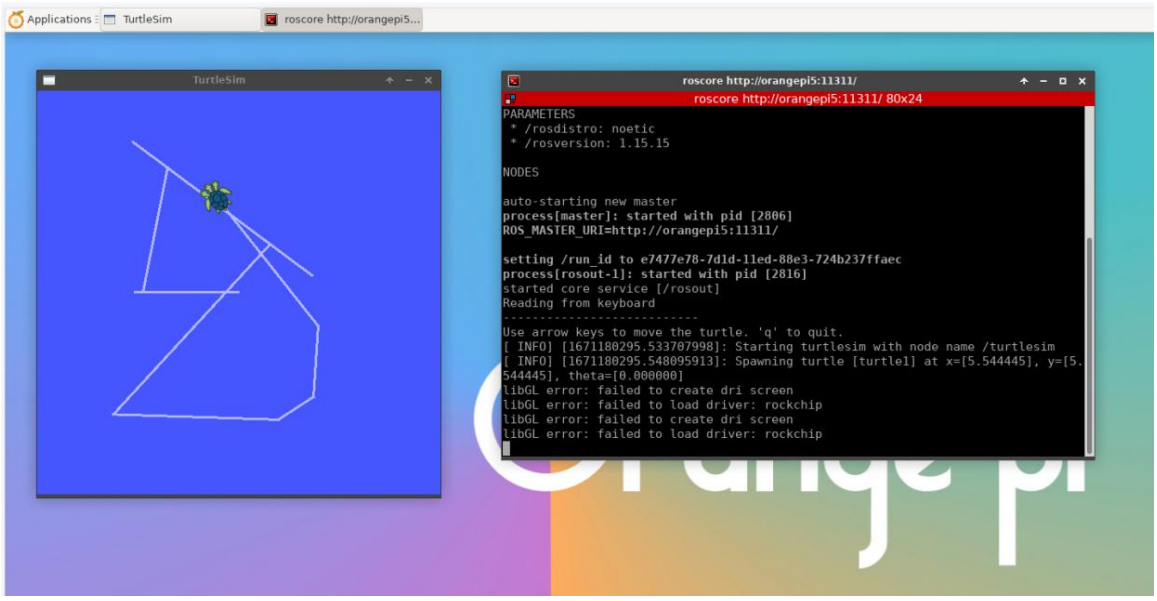
7) After running the **test_ros.sh** script, a small turtle will pop up as shown in the figure below



8) Then please keep the terminal window you just opened on top

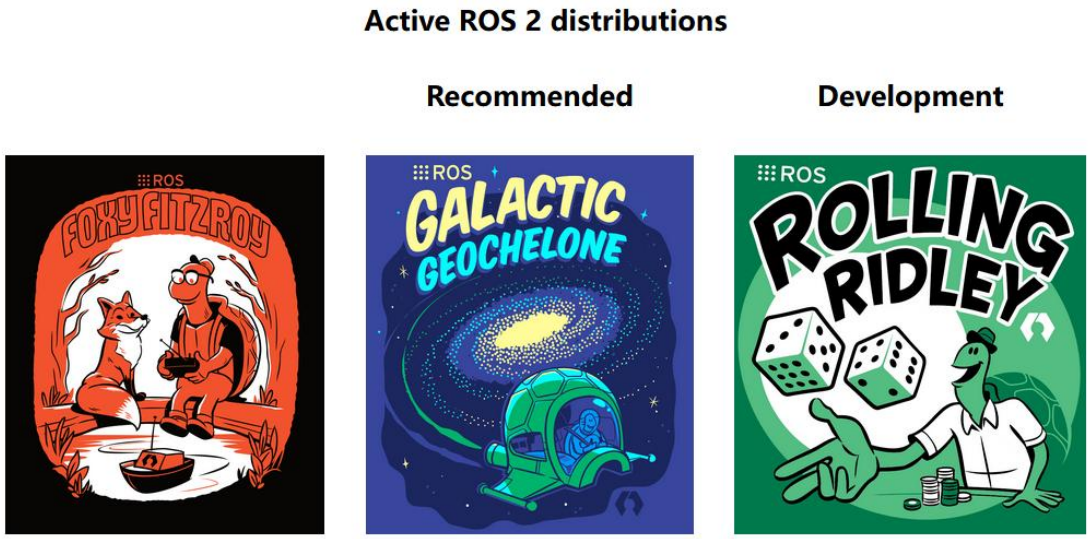


9) At this time, press the direction keys on the keyboard to control the turtle to move up, down, left and right.



3. 30. 2. How to install ROS 2 Galactic on Ubuntu 20.04

1) The currently active versions of ROS 2 are as follows, and the recommended versions are **Galactic Geochelone**



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) The official installation document link of ROS 2 **Galactic Geochelone** is as follows:

docs.ros.org/en/galactic/Installation.html
http://docs.ros.org/en/galactic/Installation/Ubuntu-Install-Debians.html

3) The official installation document of ROS 2 **Galactic Geochelone** recommends using Ubuntu 20.04 for Ubuntu Linux, so make sure that the system used by the development board is the **Ubuntu 20.04 desktop system**. There are several ways to install ROS 2. The following demonstrates how to install ROS 2 **Galactic Geochelone** using **Debian packages**.

4) Use the **install_ros.sh** script to install ros2

```
orangepi@orangepi:~$ install_ros.sh ros2
```

5) After the **install_ros.sh** script installs ros2, it will automatically run the **ros2 -h** command. If you can see the following print, it means that ros2 installation is complete.

```
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.
```

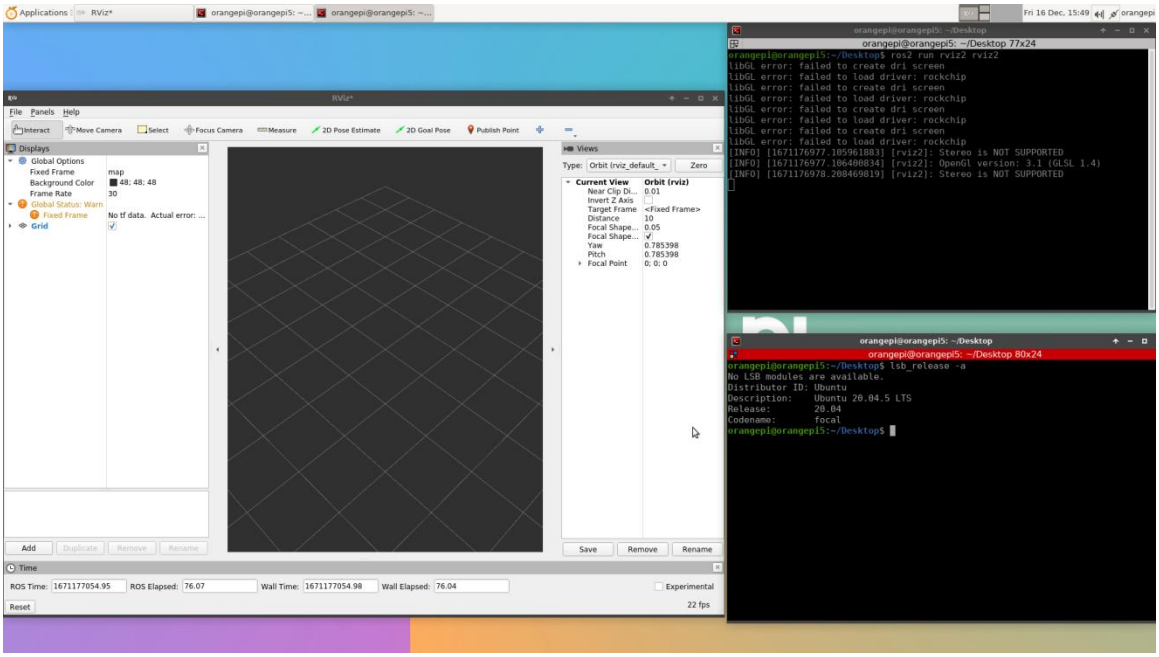
6) Then you can use the **test_ros.sh** script to test whether ROS 2 is installed successfully. If you can see the following print, it means that ROS 2 can run normally.

```
orangepi@orangepi5ultra:~$ test_ros.sh
[INFO] [1671174101.200091527] [talker]: Publishing: 'Hello World: 1'
[INFO] [1671174101.235661048] [listener]: I heard: [Hello World: 1]
[INFO] [1671174102.199572327] [talker]: Publishing: 'Hello World: 2'
[INFO] [1671174102.204196299] [listener]: I heard: [Hello World: 2]
[INFO] [1671174103.199580322] [talker]: Publishing: 'Hello World: 3'
[INFO] [1671174103.204019965] [listener]: I heard: [Hello World: 3]
```

7) Run the following command to open rviz2

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
```

```
orangepi@orangepi:~$ ros2 run rviz2 rviz2
```



8) For the usage of ROS, please refer to the ROS 2 documentation

```
http://docs.ros.org/en/galactic/Tutorials.html
```

3. 30. 3. How to install ROS 2 Humble on Ubuntu 22.04

1) Use the **install_ros.sh** script to install ros2

```
orangepi@orangepi:~$ install_ros.sh ros2
```

2) After the **install_ros.sh** script installs ros2, it will automatically run the **ros2 -h** command. If you can see the following print, it means that ros2 installation is complete.

```
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
```

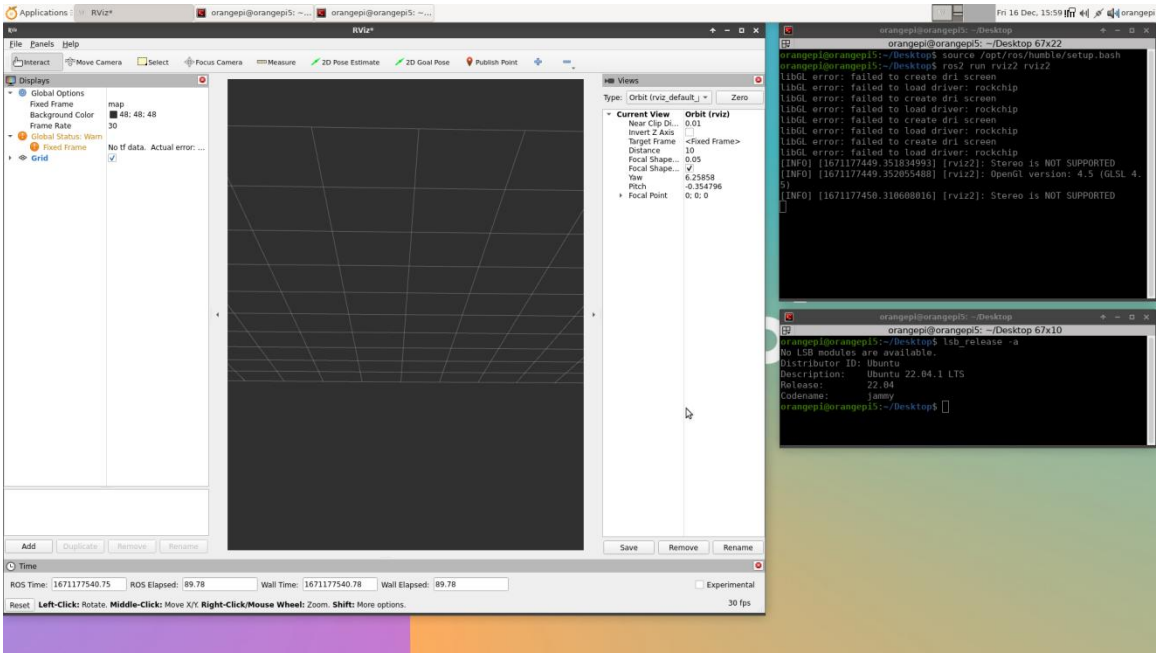
```
daemon      Various daemon related sub-commands
doctor      Check ROS setup and other potential issues
interface   Show information about ROS interfaces
launch      Run a launch file
lifecycle   Various lifecycle related sub-commands
multicast   Various multicast related sub-commands
node        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.
```

3) Then you can use the **test_ros.sh** script to test whether ROS 2 is installed successfully. If you can see the following print, it means that ROS 2 can run normally.

```
orangepi@orangepi5ultra:~$ test_ros.sh
[INFO] [1671174101.200091527] [talker]: Publishing: 'Hello World: 1'
[INFO] [1671174101.235661048] [listener]: I heard: [Hello World: 1]
[INFO] [1671174102.199572327] [talker]: Publishing: 'Hello World: 2'
[INFO] [1671174102.204196299] [listener]: I heard: [Hello World: 2]
[INFO] [1671174103.199580322] [talker]: Publishing: 'Hello World: 3'
[INFO] [1671174103.204019965] [listener]: I heard: [Hello World: 3]
```

4) Run the following command to open rviz2

```
orangepi@orangepi:~$ source /opt/ros/humble/setup.bash
orangepi@orangepi:~$ ros2 run rviz2 rviz2
```



5) Reference Documents

http://docs.ros.org/en/humble/index.html
http://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html

3. 31. How to install kernel header files

1) The Linux image released by OPi comes with a deb package of kernel header files by default, which is stored in **/opt/**

The names of the deb packages of kernel header files of different kernel versions may be different. Please refer to the actual ones you see.

```
orangepi@orangepi:~$ ls /opt/linux-headers*
/opt/linux-headers-legacy-rockchip-rk3588_x.x.x_arm64.deb
```

2) Use the following command to install the kernel header file deb package

The name of the kernel header file deb package needs to be replaced with the actual name, please do not copy it.

```
orangepi@orangepi:~$ sudo dpkg -i /opt/linux-headers-legacy-rockchip-rk3588_1.x.x_arm64.deb
```

3) After installation, you can see the folder where the kernel header files are located

under **/usr/src**

```
orangepi@orangepi:~$ ls /usr/src
linux-headers-5.10.160-rockchip-rk3588
```

4) Then you can write a hello kernel module to test the kernel header file

a. First, write the code for the hello kernel module as follows:

```
orangepi@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 file to compile the hello kernel module as follows:

```
orangepi@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:

If there is a problem compiling the copied code here, you can directly use the source code pre-installed in the Linux system. The path of the hello source code is: /usr/src/hello.

```
orangepi@orangepi:~$ sudo make
make -C /lib/modules/5.10.160-rockchip-rk3588/build M=/home/orangepi modules
make[1]: Entering directory '/usr/src/linux-headers-5.10.160-rockchip-rk3588'
  CC [M]  /home/orangepi/hello.o
  MODPOST /home/orangepi/Module.symvers
  CC [M]  /home/orangepi/hello.mod.o
  LD [M]  /home/orangepi/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.10.160-rockchip-rk3588'
```

d. After compilation, the **hello.ko** kernel module will be generated

```
orangepi@orangepi:~$ ls *.ko
hello.ko
```

e. Use the **insmod** command to insert the **hello.ko** kernel module into the kernel

```
orangepi@orangepi:~$ sudo insmod hello.ko
```

f. Then use the **demsg** command to view the output of the **hello.ko** kernel module. If you can see the following output, it means that the **hello.ko** kernel module is loaded correctly.

```
orangepi@orangepi:~$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
```

g. Use the **rmmod** command to uninstall the **hello.ko** kernel module

```
orangepi@orangepi:~$ sudo rmmod hello
orangepi@orangepi:~$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
[ 3173.800892] Hello Orange Pi -- exit
```

3. 32. How to use 10.1 inch MIPI LCD screen

3. 32. 1. 10.1 inch MIPI screen assembly method

1) First prepare the necessary accessories

a. 10.1 inch MIPI LCD display + touch screen



b. Screen adapter board + 31pin to 40pin cable



c. 30pin MIPI cable



d. 12pin touch screen cable

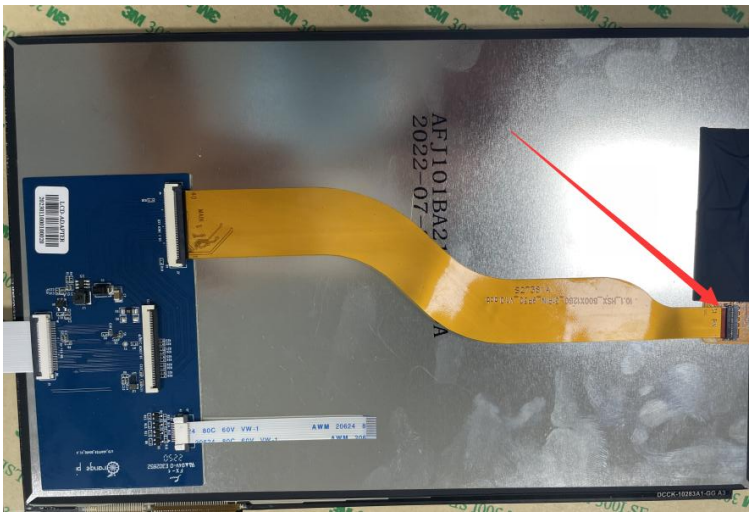


2) Connect the 12-pin touch screen cable, 31-pin to 40-pin cable, and 30-pin MIPI cable to the screen adapter board as shown below. **Note that the blue insulation side of the touch screen cable should face downward,** and the insulation sides of the other two

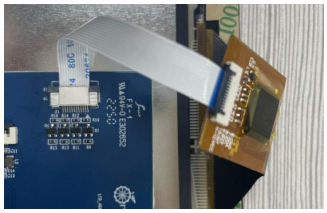
cables should face upward. If connected incorrectly, it will cause no display or inability to touch.



3) Place the adapter board with the connected cable on the MIPI LCD screen as shown below, and connect the MIPI LCD screen and the adapter board via a 31pin to 40pin cable.



4) Then connect the touch screen and the adapter board through the 12-pin touch screen cable, paying attention to the direction of the insulating surface



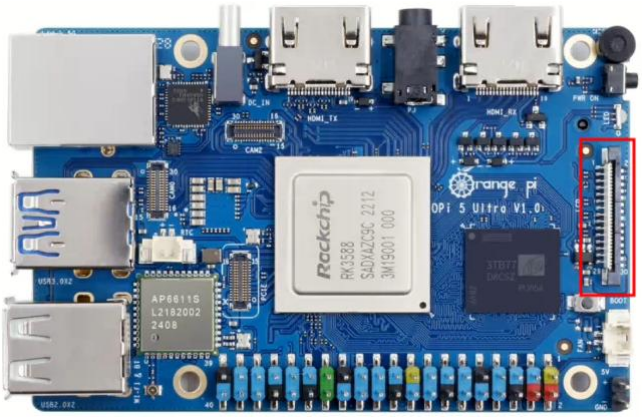
5) Finally, connect it to the LCD interface of the development board through the 30pin MIPI cable



3. 32. 2. How to open the 10.1-inch MIPI LCD screen configuration

1) The Linux image does not have the mipi lcd screen configuration turned on by default. If you need to use the mipi lcd screen, you need to turn it on manually.

2) The location of the interface of the mipi lcd screen on the development board is shown in the figure below:

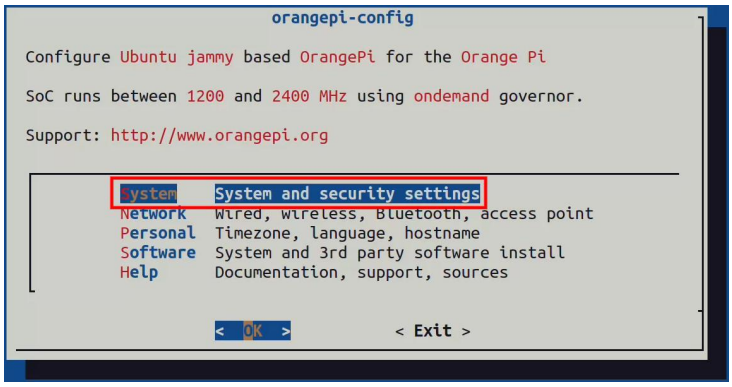


3) The steps to open the mipi lcd configuration are as follows:

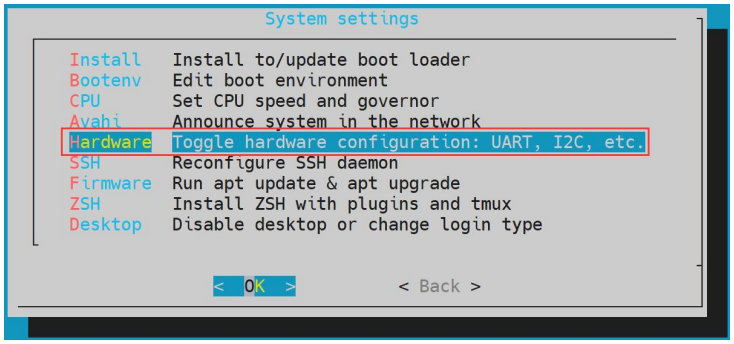
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**



c. Then select **Hardware**



d. Then use the arrow keys on the keyboard to locate **opi5ultra-lcd**, and then use the **spacebar** to select



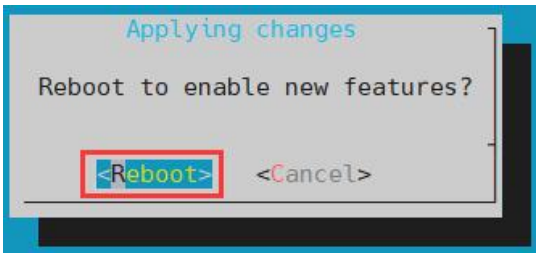
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Then select **<Reboot>** to restart the system for the configuration to take effect



The above settings will eventually add overlays=opi5ultra-lcd to /boot/orangepiEnv.txt. You can check it after setting it. If this line does not exist, then there is a problem with the settings.

If you find it troublesome to use orangepi-config, you can also use the vim editor to open /boot/orangepiEnv.txt and add the line overlays=opi5ultra-lcd.

orangepi@orangepi:~$ **cat /boot/orangepiEnv.txt | grep "lcd"**
overlays=opi5ultra-lcd #Example Configuration

4) After startup, you can see the LCD screen display as follows (the default is vertical):



3.32.3. How to rotate the display direction of the server version image

1) Add **extraargs=fbcon=rotate: direction to be rotated** in **/boot/orangepiEnv.txt** to set the display direction of the server version of Linux system. The number after **fbcon=rotate:** can be set to:

a. 0: Normal screen (portrait by default)
b. 1: Rotate 90 degrees clockwise

c. 2: Flip 180 degrees

d. 3: Rotate 270 degrees clockwise

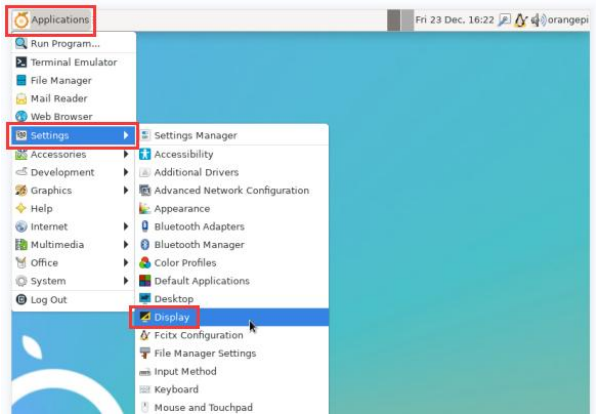
```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
extraargs=cma=128M fbcon=rotate:3
overlays=opi5ultra-disable-leds
```

Note that if there is a line of extraargs=cma=128M in /boot/orangepiEnv.txt by default, you can add the line fbcon=rotate:3 after extraargs=cma=128M (separated by a space).

2) Then **restart** the Linux system and you will see that the direction of the LCD screen display has rotated

3. 32. 4. Desktop version mirroring rotation display and touch direction method

1) First open the **Display** settings in the Linux system



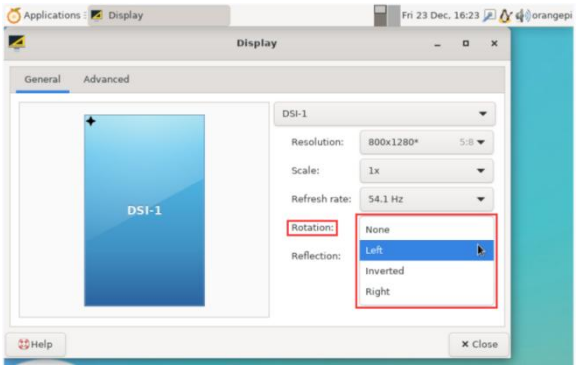
2) Then select the direction you want to rotate in **Rotation**

a. **None**: No rotation

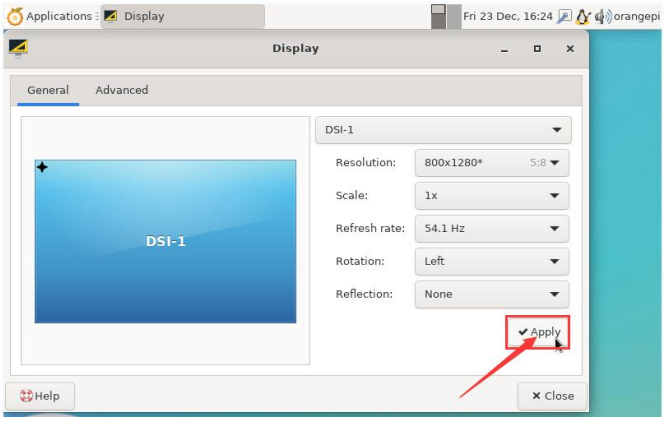
b. **Left**: Rotate 90 degrees left

c. **Inverted**: Flip upside down, equivalent to rotating 180 degrees

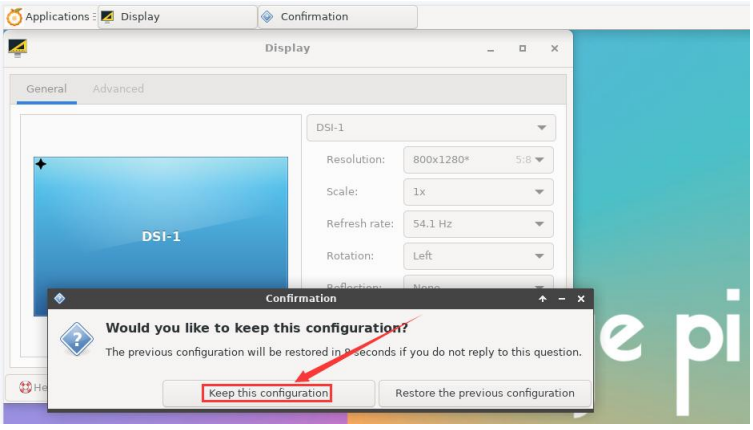
d. **Right**: Rotate right 90 degrees



3) Then click **Apply**



4) Then select **Keep this configuration**



5) The screen display has been rotated, and you can close the **Display** program

6) The above steps will only select the display direction, but will not rotate the touch direction. You can use the **set_lcd_rotate.sh** script to rotate the touch direction. After setting this script, it will automatically restart, and then you can test whether the touch

can be used normally.

a. **None**: No rotation

```
orangepi@orangepi:~$ set_lcd_rotate.sh none
```

b. **Left**: Rotate 90 degrees left

```
orangepi@orangepi:~$ set_lcd_rotate.sh left
```

c. **Inverted**: Flip upside down, equivalent to rotating 180 degrees

```
orangepi@orangepi:~$ set_lcd_rotate.sh inverted
```

d. **Right**: Rotate right 90 degrees

```
orangepi@orangepi:~$ set_lcd_rotate.sh right
```

The set_lcd_rotate.sh script does four main things:

1. **Rotate the direction of the framebuffer display**
2. **Rotate the touch direction**
3. **Turn off the startup logo**
4. **Restart the system**

Rotating the touch direction is achieved by adding Option "TransformationMatrix" "x x x x x x x x x x" to /usr/share/X11/xorg.conf.d/40-libinput.conf, where "x x x x x x x x x" has different configurations for different directions.

7) Touch to rotate the reference

```
https://wiki.ubuntu.com/X/InputCoordinateTransformation
```

3. 33. Instructions for using the power on/off logo

1) The power on/off logo is only displayed in the desktop version of the system by default.

2) Set the **bootlogo** variable to **false** in **/boot/orangepiEnv.txt** to turn off the power on/off logo

```
orangepi@orangepi:~$ vim /boot/orangepiEnv.txt
verbosity=1
```

```
bootlogo=false
```

3) Set the **bootlogo** variable to **true** in **/boot/orangepiEnv.txt** to enable the power on/off logo

```
orangepi@orangepi:~$ vim /boot/orangepiEnv.txt
verbosity=1
bootlogo=true
```

4) The location of the boot logo image in the Linux system is

```
/usr/share/plymouth/themes/orangepi/watermark.png
```

5) After replacing the boot logo image, you need to run the following command to take effect

```
orangepi@orangepi:~$ sudo update-initramfs -u
```

3. 34. Test Methods for OV13850 and OV13855 MIPI Cameras

Please note that in Linux systems, in order to ensure that the 3A service can run normally and obtain normal camera images, the Docker service needs to be disabled. If the Docker service is not disabled, the image captured by the camera will not contain the 3A effect and will appear as a dark image. The method to disable the Docker service is as follows:

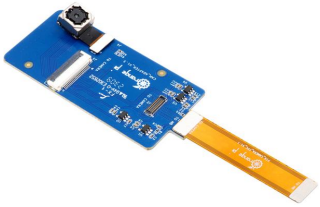
```
orangepi@orangepi:~$ sudo systemctl disable docker.socket docker.service containerd.service
orangepi@orangepi:~$ sudo reboot
```

Currently the development board supports two MIPI cameras, OV13850 and OV13855. The specific pictures are as follows:

a. 13MP OV13850 camera with MIPI interface



b. 13MP OV13855 camera with MIPI interface

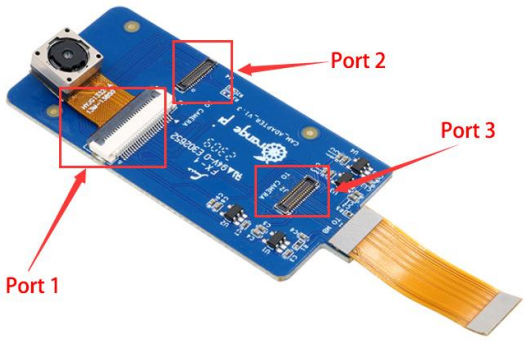


The adapter board and FPC cable used by OV13850 and OV13855 cameras are the same, but the two cameras are connected to the adapter board in different positions. The FPC cable is shown in the figure below. Please note that the FPC cable has a direction. The end marked with **TO MB** needs to be plugged into the camera interface of the development board, and the end marked with **TO CAMERA** needs to be plugged into the camera adapter board.

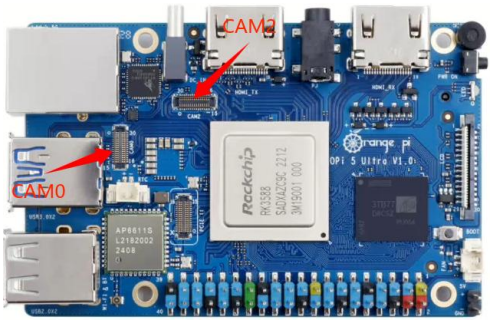


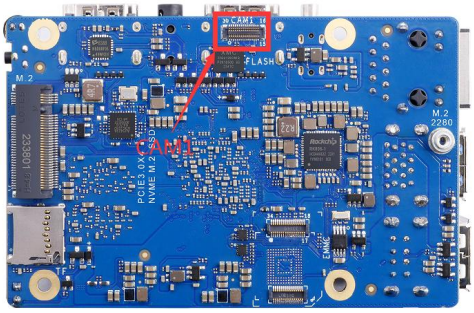
There are a total of 3 camera interfaces on the camera adapter board. Only one can be connected at a time, as shown in the following figure:

a. **Interface 1 is connected to the OV13850 camera**
b. **Interface 2 is connected to the OV13855 camera**
c. Interface 3 is not used, just ignore it



There are three camera interfaces on the Orange Pi 5 Ultra development board. We define the positions of Cam0, Cam1, and Cam2 as shown in the following figure:





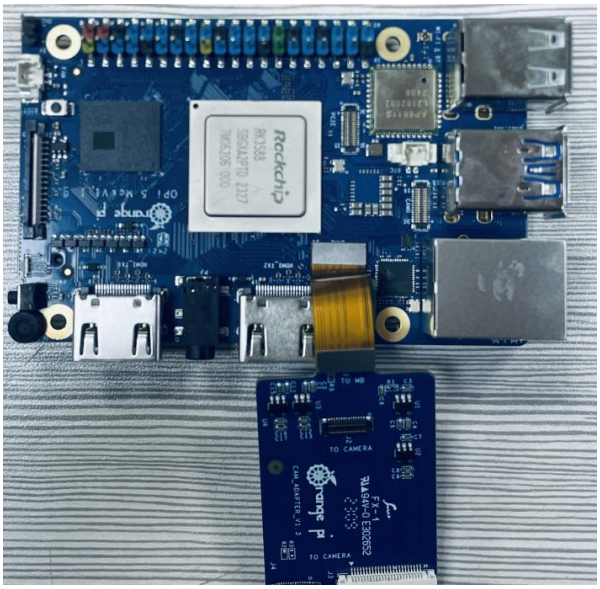
The method of inserting the camera into the Cam0 interface of the development board is as follows:



The method of inserting the camera into the Cam1 interface of the development board is as follows:



The method of inserting the camera into the Cam2 interface of the development board is as follows:

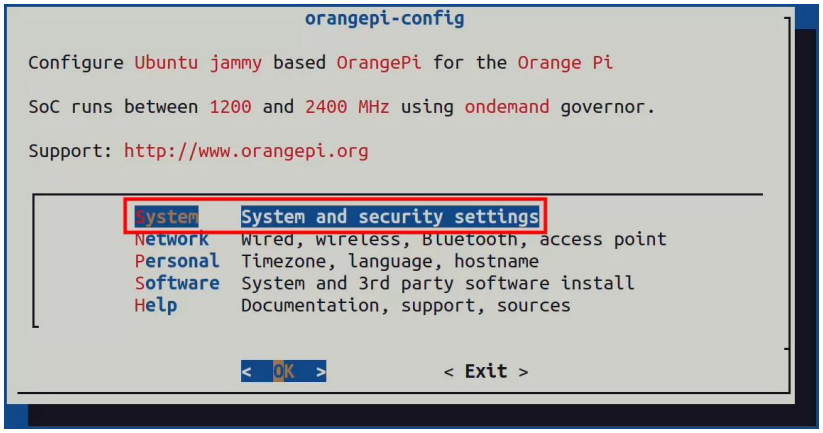


After connecting the camera to the development board, we can use the following method to test the camera:

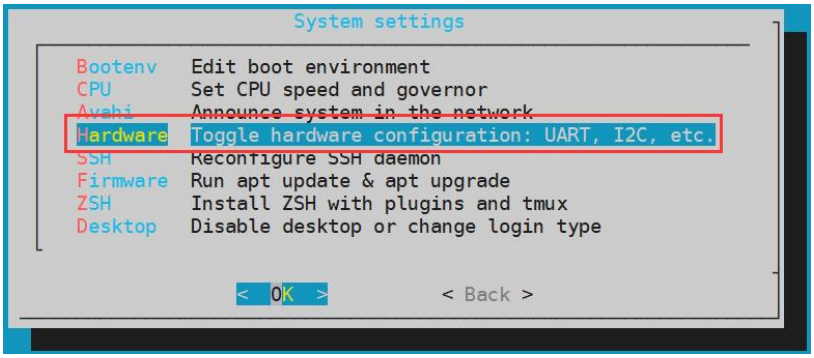
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**



c. Then select **Hardware**



d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the **space bar** to select the camera you want to open. **opi5ultra-cam0** means using the ov13850 or ov13855 camera in the Cam0 interface of the development board, **opi5ultra-cam1** means using the ov13850 or ov13855 camera in the Cam1 interface of the development board, and **opi5ultra-cam2** means using the ov13850 or ov13855 camera in the Cam2 interface of the development board.



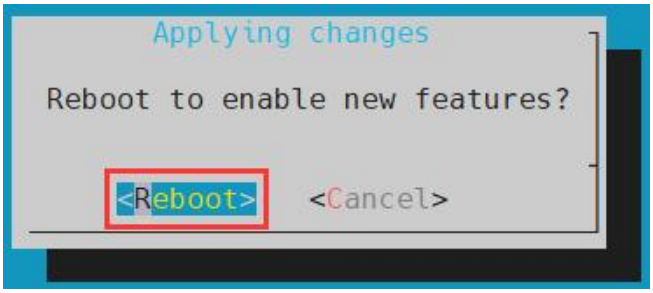
e. Then select **<Save>** to save



f. Then select **<Back>**



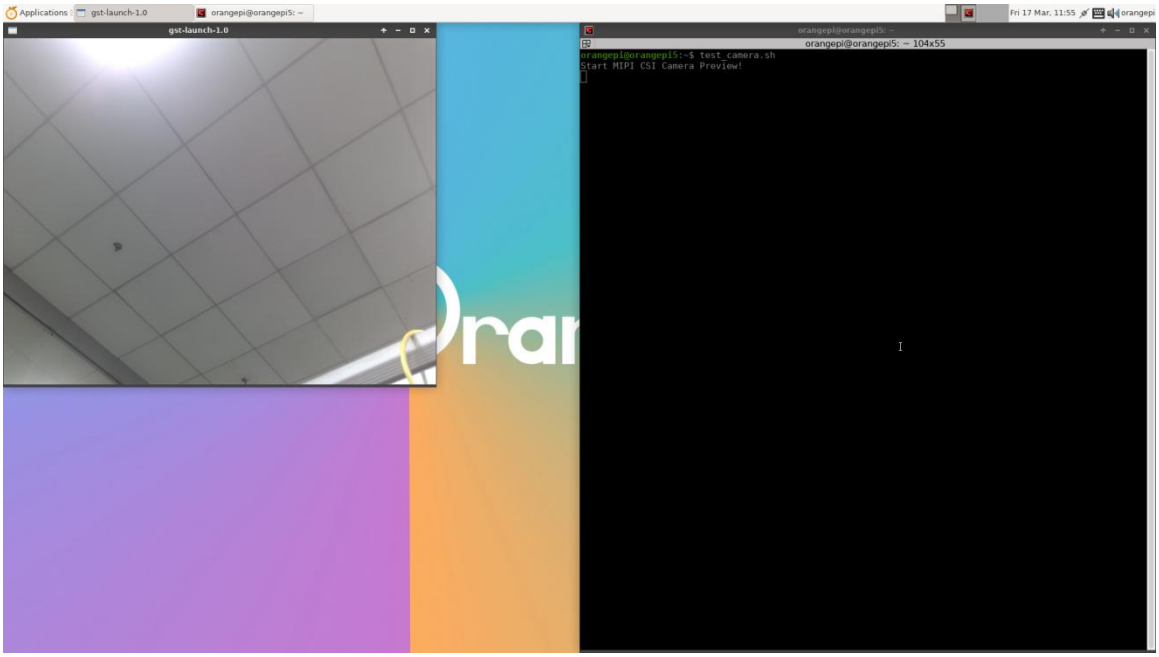
g. Then select **<Reboot>** to restart the system for the configuration to take effect



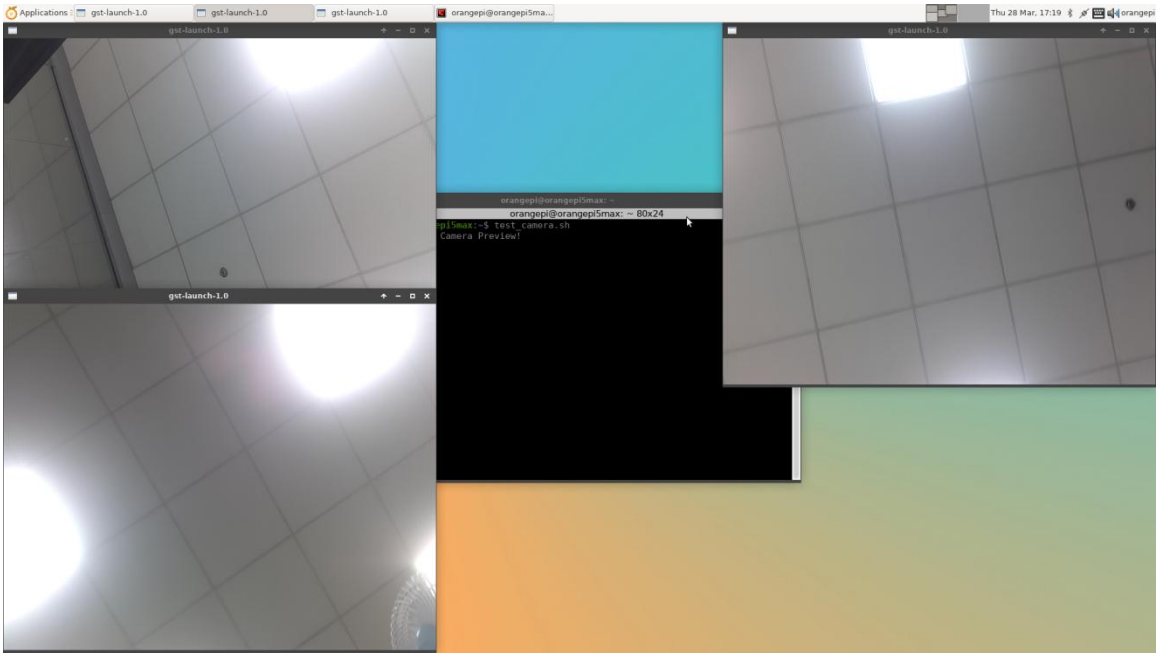
h. Then open a terminal in the desktop system and run the following script

```
orangepi@orangepi:~$ test_camera.sh
```

i. Then you can see the camera preview



In addition to a single camera, we can also use two or three cameras at the same time (supporting ov13850 and ov13855 mixed). After connecting the three cameras, open the configuration of Cam0+Cam1+Cam2 through **orangepi-config** as in the previous steps, restart the system, and then open the terminal on the desktop to run the **test_camera.sh** script to see the preview of the three cameras, as shown in the figure below:



3. 35. How to use ZFS file system

3. 35. 1. How to install ZFS

Before installing zfs, please make sure that the Linux image you are using is the latest version. In addition, if zfs has already been installed on the system, there is no need to install it again.

Before installing zfs, you need to install the kernel header file first. For the method of installing the kernel header file, please refer to the instructions in the section "**How to install the kernel header file**".

In Ubuntu 20.04, Ubuntu 22.04 and Debian 11 systems, zfs cannot be installed directly through apt. This is because the zfs version in the default apt source is lower than 2.1.6, which is incompatible with the rk linux 5.10 kernel. This problem has been fixed in zfs 2.1.6 and later versions.

To solve this problem, we provide a deb package of zfs that can be installed normally, which can be downloaded from the **official tool** of the development board. Open the **official tool** and enter the **deb package folder related to zfs used by Ubuntu and Debian systems**. You can see three types of deb packages: Ubuntu20.04, Ubuntu22.04 and Debian11. Please download the required version.



After downloading the corresponding version of zfs deb package, please upload them to the Linux system of the development board. For the upload method, please refer to the instructions in the section "**How to upload files to the Linux system of the development board**".

After uploading, use the **cd** command in the command line of the Linux system of the development board to enter the directory of the deb package, and then use the following command to install the zfs deb package.

```
orangepi@orangepi:~$ sudo apt -y install ./*.deb
```

Then restart the Linux system and you will see that the zfs kernel module will be automatically loaded：

```
orangepi@orangepi:~$ lsmod | grep "zfs"
zfs                  2801664  0
zunicode              327680  1 zfs
zzstd                 471040  1 zfs
zlua                  139264  1 zfs
zcommon                69632  1 zfs
znvpair                61440  2 zfs,zcommon
zavl                   16384  1 zfs
icp                   221184  1 zfs
spl                    77824  6 zfs,icp,zzstd,znvpair,zcommon,zavl
```

In Debian 12, the default version of zfs is 2.1.11, so we can install zfs directly through the following command. Once again, please make sure that the system has installed the deb package of the kernel header file before installation.

```
orangepi@orangepi:~$ sudo apt install -y zfsutils-linux zfs-dkms
```

3.35.2. How to create a ZFS pool

ZFS is based on storage pools. We can add multiple physical storage devices to a

pool and then allocate storage space from this pool.

The following content is demonstrated based on the development board connected to an NVMe SSD and a USB flash drive.

1) First, we can use the **lsblk** command to view all storage devices on the development board. The current development board is connected to an NVMe SSD and a USB flash drive. The output is as follows：

```
orangepi@orangepi:~$ lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1  28.8G  0 disk
├─sda1        8:1    1  28.8G  0 part
└─sda9        8:9    1     8M  0 part
mtdblock0    31:0    0    16M  0 disk
mmcblk0     179:0    0  29.7G  0 disk
├─mmcblk0p1 179:1    0     1G  0 part /boot
└─mmcblk0p2 179:2    0  28.4G  0 part /var/log.hdd
                                      /
zram0       254:0    0   7.7G  0 disk [SWAP]
zram1       254:1    0   200M  0 disk /var/log
nvme0n1     259:0    0 476.9G  0 disk
├─nvme0n1p1 259:3    0 476.9G  0 part
└─nvme0n1p9 259:4    0     8M  0 part
orangepi@orangepi:~$
```

2) Then enter the following command to create a ZFS pool containing two storage devices: NVMe SSD and USB flash drive

```
orangepi@orangepi:~$ sudo zpool create -f pool1 /dev/nvme0n1 /dev/sda
```

3) Then use the **zpool list** command to see that the system has created a ZFS pool named **pool1**, and the size of the ZFS pool pool1 is the size of the NVME SSD plus the size of the USB flash drive.

```
orangepi@orangepi:~$ zpool list
NAME    SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
pool1   504G   114K   504G        -         -     0%     0%  1.00x    ONLINE  -
```

4) Then execute **df -h** to see that **pool1** is mounted to the **/pool1** directory

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           1.6G   18M  1.6G   2% /run
/dev/mmcblk0p2   29G  6.0G   22G  22% /
tmpfs           7.7G   46M  7.7G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
```

```
tmpfs           7.7G  944K  7.7G   1% /tmp
/dev/mmcblk0p1 1022M  115M  908M  12% /boot
/dev/zram1      188M  4.5M  169M   3% /var/log
tmpfs           1.6G   80K  1.6G   1% /run/user/1000
pool1           489G  9.3M  489G   1% /pool1
```

5) Use the following command to see that the file system type of pool1 is zfs

```
orangepi@orangepi:~$ mount | grep pool1
pool1 on /pool1 type zfs (rw,xattr,noacl)
```

6) Then we can test copying a file to the ZFS pool

```
orangepi@orangepi:~$ sudo cp -v /usr/local/test.mp4 /pool1/
'/usr/local/test.mp4' -> '/pool1/test.mp4'
```

3. 35. 3. Test ZFS data deduplication function

1) ZFS data deduplication function is disabled by default. We need to execute the following command to enable it.

```
orangepi@orangepi:~$ sudo zfs set dedup=on pool1
```

2) Then do a simple test. First enter pool1 and then execute the following command to generate a random file of 1G size:

```
orangepi@orangepi:~$ cd /pool1/
root@orangepi:/pool1$ sudo dd if=/dev/urandom of=test.1g bs=1M count=1024
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 5.04367 s, 213 MB/s
```

3) Then use the following command to copy 1000 copies of a random file of size 1G

```
root@orangepi:/pool1$ for ((i=0; i<1000; i++)); do sudo cp test.1g $i.test.1g; done
```

4) Then use **du -lh** to see that there is a total of 1002G of data in the pool. However, the actual size of the ZFS pool is only **504GB** (the total capacity of the SSD + USB drive), which is too large to accommodate such a large amount of data.

```
root@orangepi:/pool1$ du -lh
1002G
```

5) Then use the **zpool list** command to see that only 1.01G is actually occupied. Because these 1001 files are duplicated, the data deduplication function is effective.

```
orangepi@orangepi:/pool1$ zpool list
NAME    SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
pool1   504G  1.01G   503G        -         -     0%     0%  6.00x    ONLINE  -
```

3. 35. 4. Test ZFS data compression function

1) Because the stored data is different, the disk space saved by compression will also be different, so we choose to compress relatively large plain text files for compression testing. Execute the following command to package the **/var/log/** and **/etc/** directories into a tar package

```
orangepi@orangepi:~$ cd /pool1/
root@orangepi:/pool1$ sudo tar -cf text.tar /var/log/ /etc/
```

2) Then, the file size and the space occupied in the ZFS pool can be seen by the **ls -lh** command, both of which are **27M**.

```
orangepi@orangepi:/pool1$ ls -lh
total 27M
-rw-r--r-- 1 root root 27M Jun  1 14:46 text.tar
orangepi@orangepi:/pool1$ zpool list
NAME    SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
pool1   504G  26.7M   504G        -         -     0%     0%  1.00x    ONLINE  -
orangepi@orangepi:/pool1$
```

3) Then we enable compression in ZFS pool pool1

```
root@orangepi:/pool1$ sudo zfs set compression=lz4 pool1
```

4) Then execute the following command again to package the **/var/log/** and **/etc/** directories into a tarball

```
root@orangepi:/pool1$ sudo tar -cf text.tar /var/log/ /etc/
```

5) Then you can see that the file size of **text.tar** is still 27M, but it only takes up 9.47M of space in the ZFS pool, indicating that the file has been compressed.

```
orangepi@orangepi:/pool1$ ls -lh
total 9.2M
-rw-r--r-- 1 root root 27M Jun  1 14:54 text.tar
orangepi@orangepi:/pool1$ zpool list
NAME    SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP    HEALTH  ALTROOT
pool1   504G  9.47M   504G        -         -     0%     0%  1.00x    ONLINE  -
```

3. 36. How to install and use CasaOS

CasaOS is an open source home cloud system based on the Docker ecosystem, which allows you to run a variety of home applications on your own development board, such as NAS, home automation, media server, etc.

3. 36. 1. How to install CasaOS

1) First, you need to install docker. The system released by Orangepi Pi has docker pre-installed. This step can be skipped. You can use the following command to view the installed docker version

```
orangepi@orangepi:~$ docker --version
```

2) Then enter the following command in the Linux system to start the installation of CasaOS

```
orangepi@orangepi:~$ curl -fsSL https://get.casaos.io | sudo bash
```

3) When you see the following print information output in the terminal, it means that CasaOS has been installed.

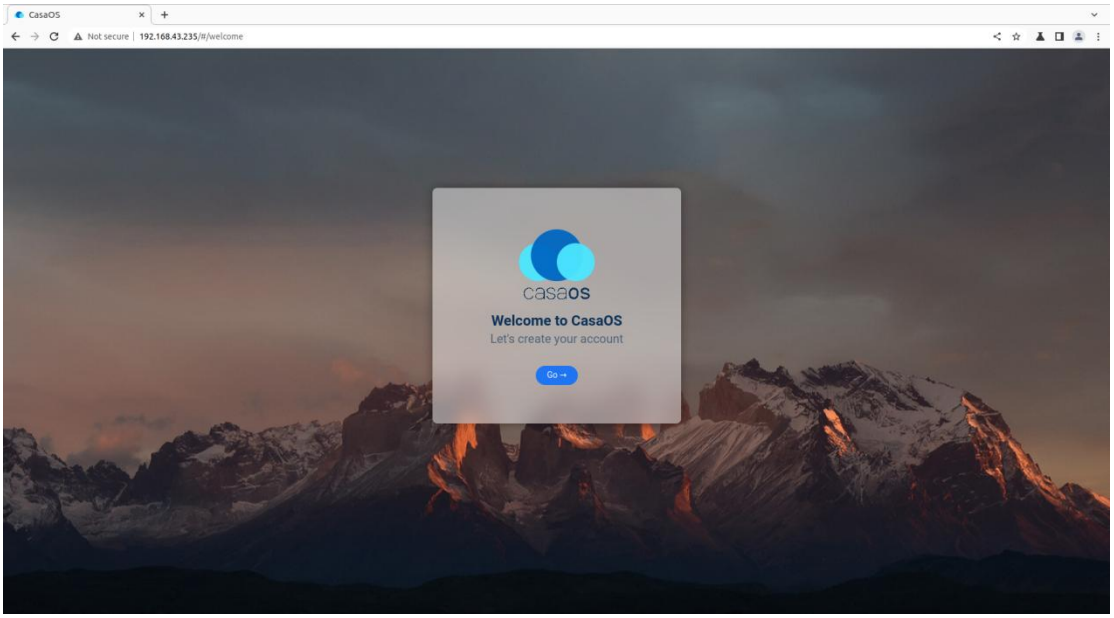
```
CasaOS v0.4.4.2 is running at:

Open your browser and visit the above address.


CasaOS Project  : https://github.com/IceWhaleTech/CasaOS
CasaOS Team     : https://github.com/IceWhaleTech/CasaOS#maintainers
CasaOS Discord  : https://discord.gg/knqAbbBbeX
Website         : https://www.casaos.io
Online Demo     : http://demo.casaos.io

Uninstall       : casaos-uninstall
```

3. 36. 2. How to use CasaOS

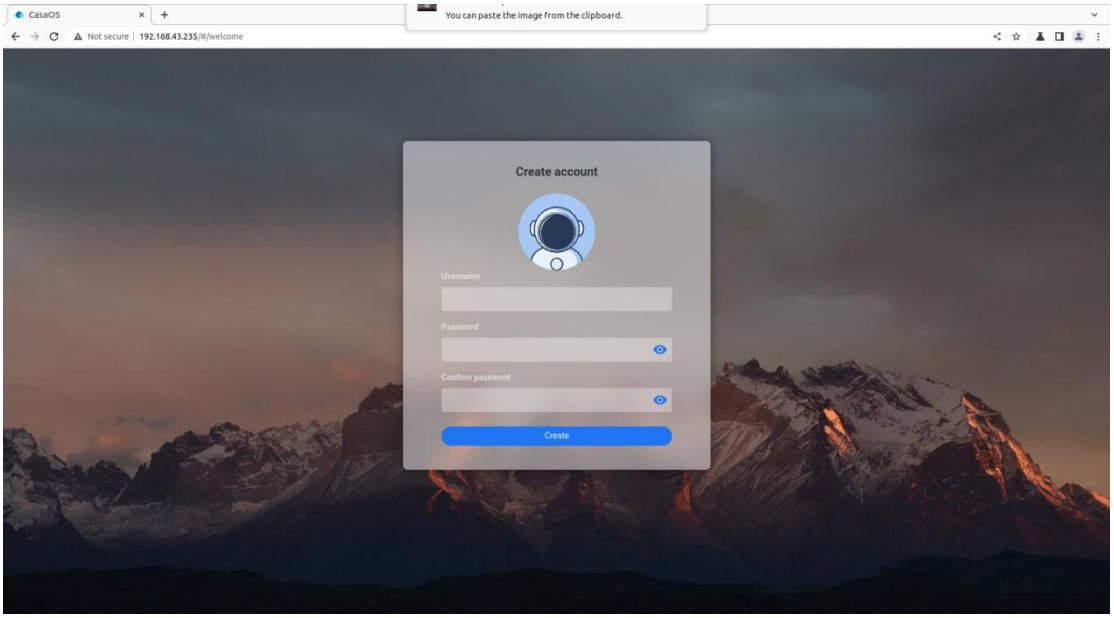
1) After installing CasaOS, enter **http://development board IP address** in the browser to open CasaOS

2) After opening CasaOS, the following welcome interface will pop up. Click "Go" to

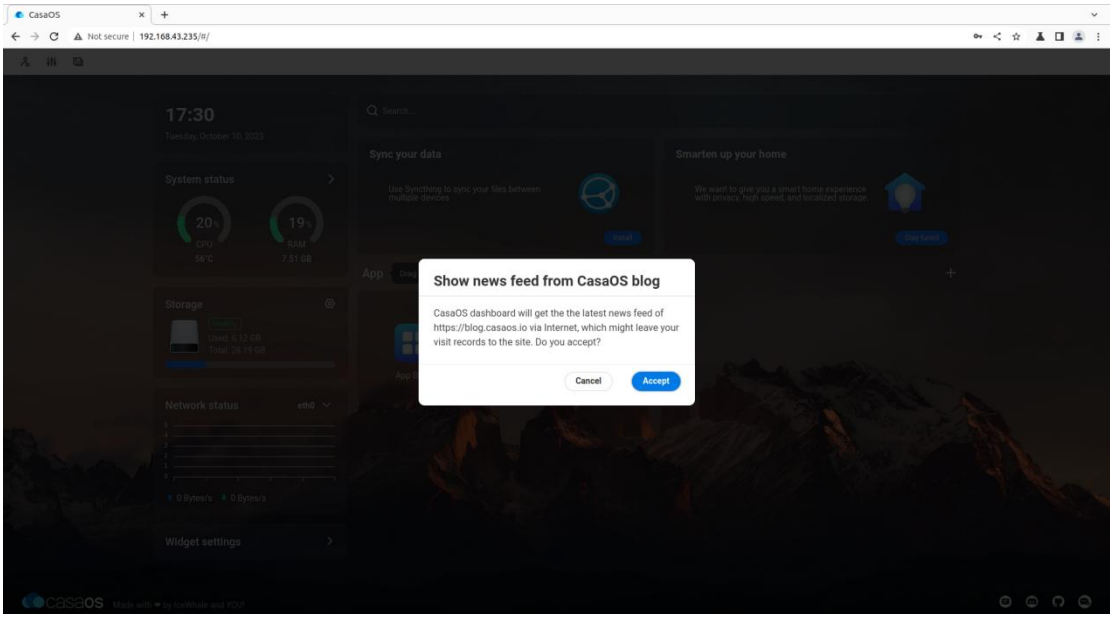
proceed to the next step



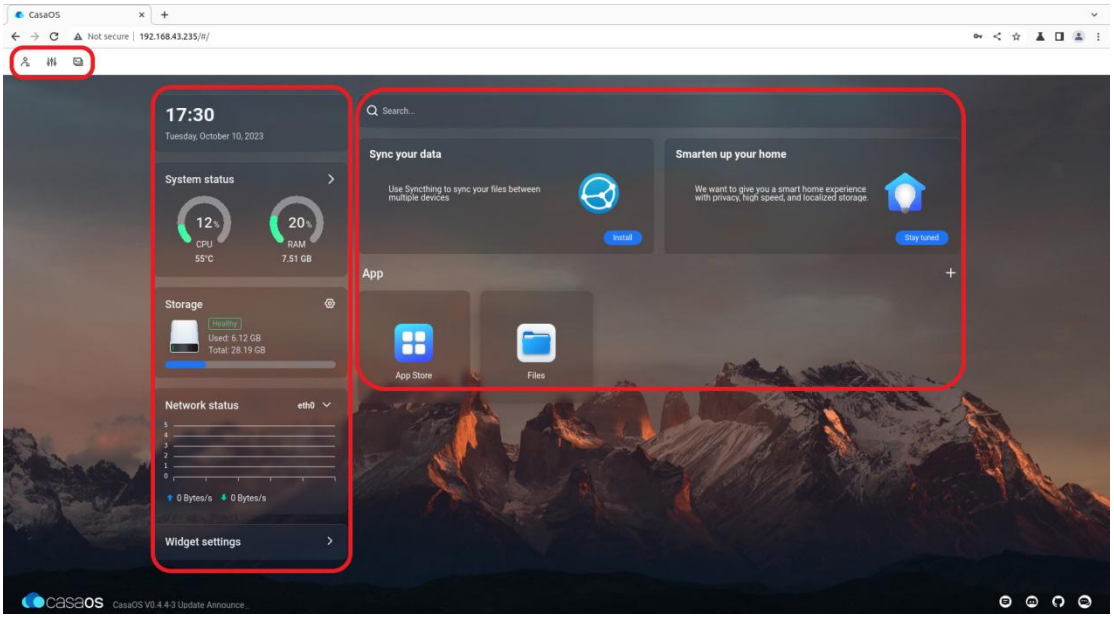
3) When you log in to CasaOS for the first time, the login interface is the interface for setting the account and password. When you log in again, only the interface for entering the account and password will appear. After setting the account and password, click "Create" to proceed to the next step.



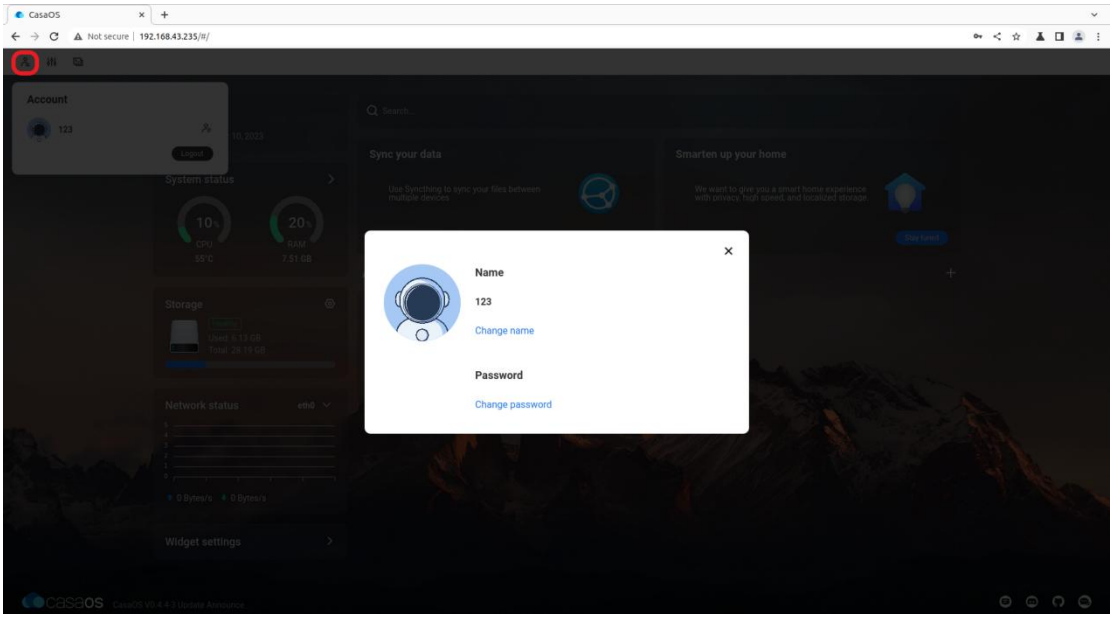
4) In the following interface, click "Accept" to proceed to the next step



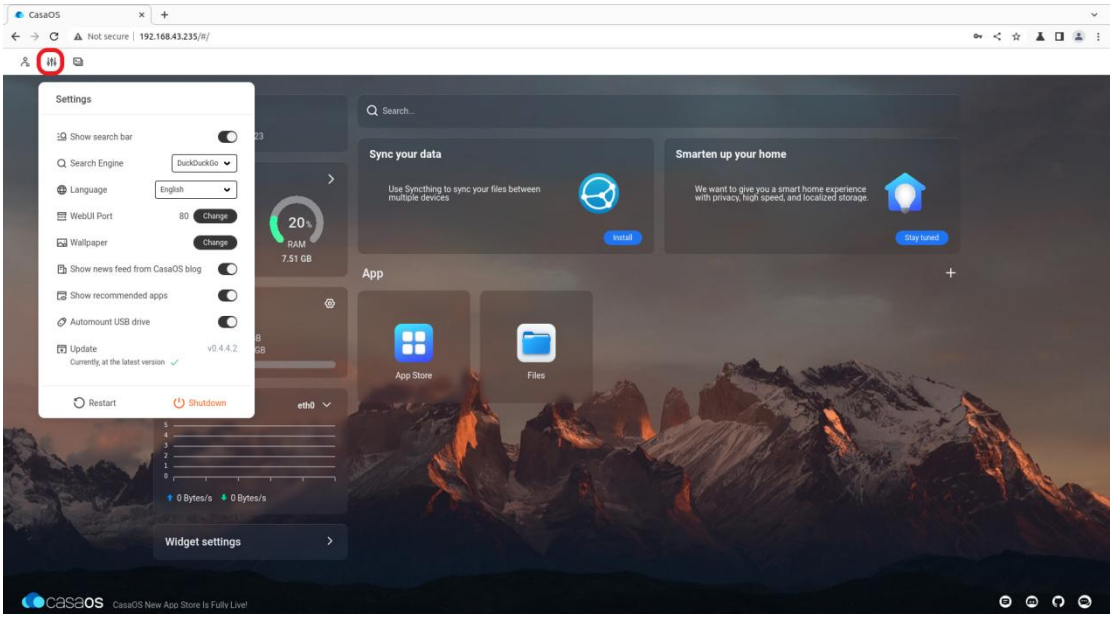
5) Now you have entered the main page of CasaOS. There are three icons in the upper left corner for function settings. The left side is the performance panel, which can display the current time and the status information of CPU, RAM, storage, and network. The right side is the function panel, which has functions such as search, application recommendation, application store, and file management.



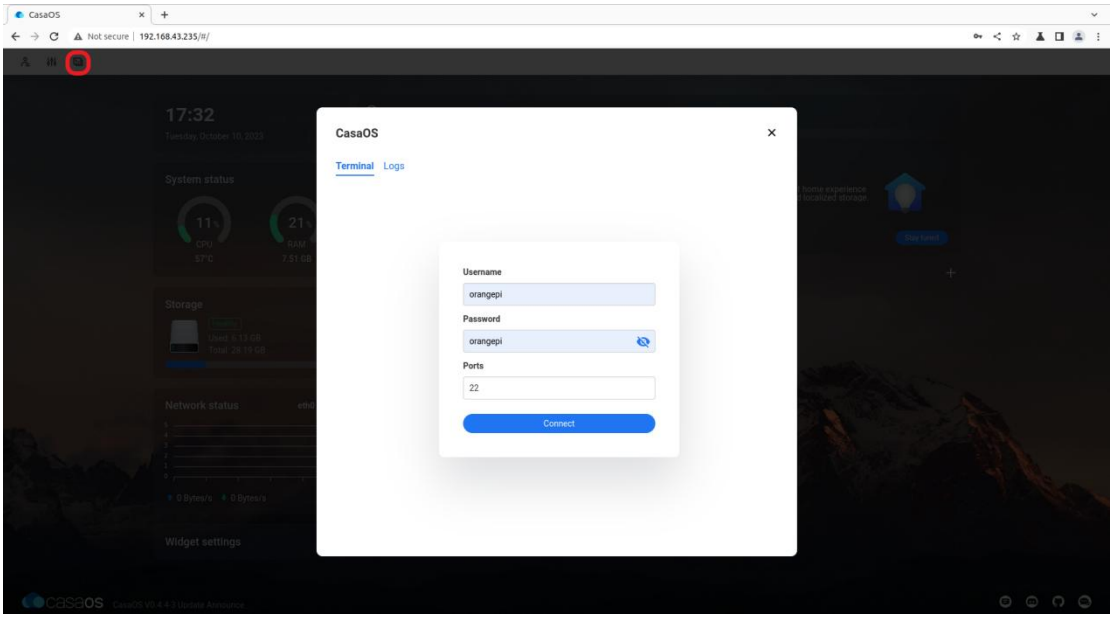
6) Click the first icon in the upper left corner to modify your account and password



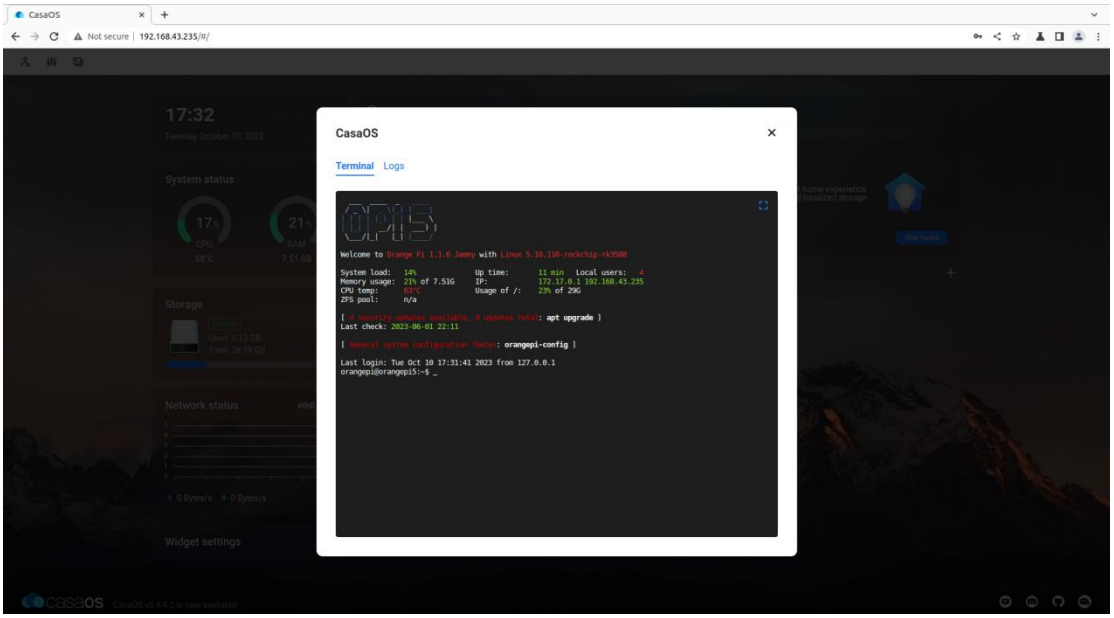
7) Click the second icon to set basic functions



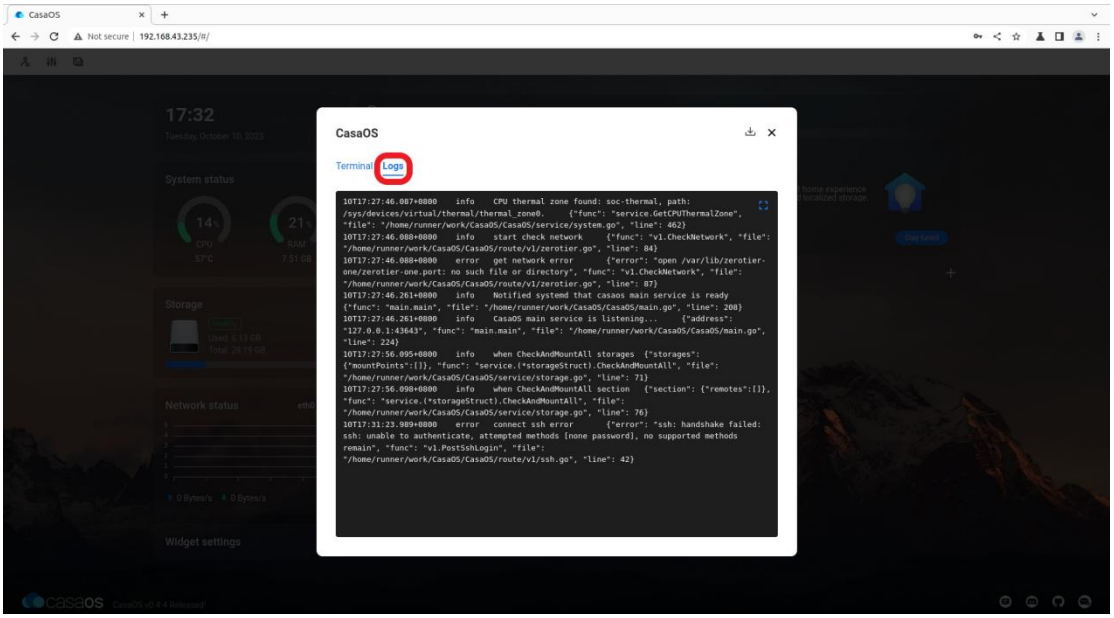
8) The third icon in the upper left corner has two main functions, namely switching to command line mode and printing log information. When switching to command line mode, you need to enter the account and password. The account and password here refer to the account and password of the development board Linux system. The default port system selects 22.



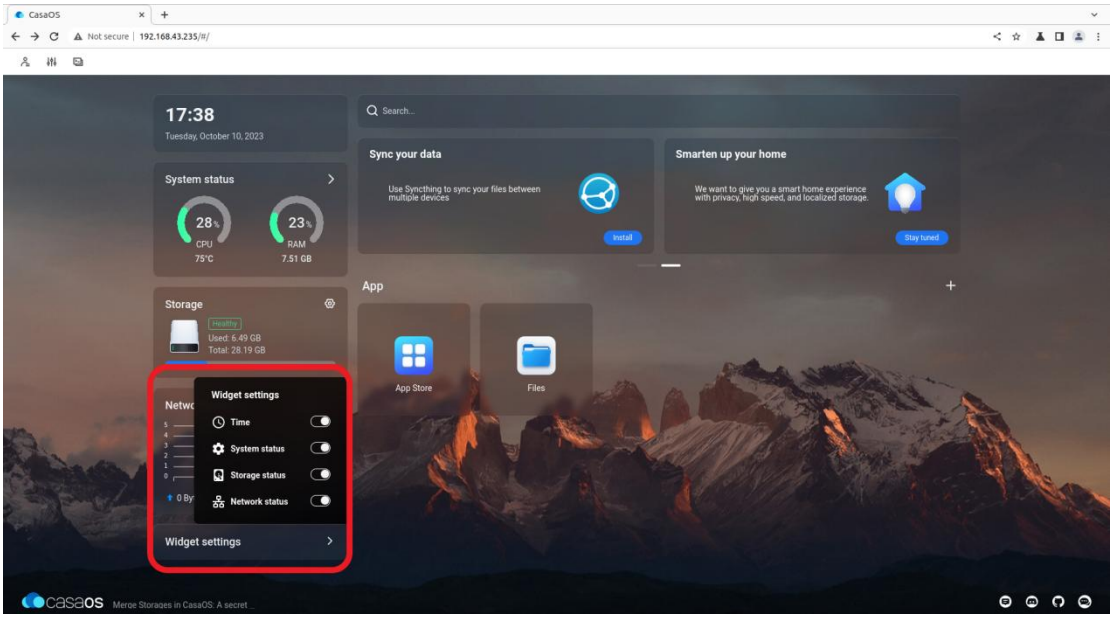
9) Then click "Connect" to enter the command line interface:



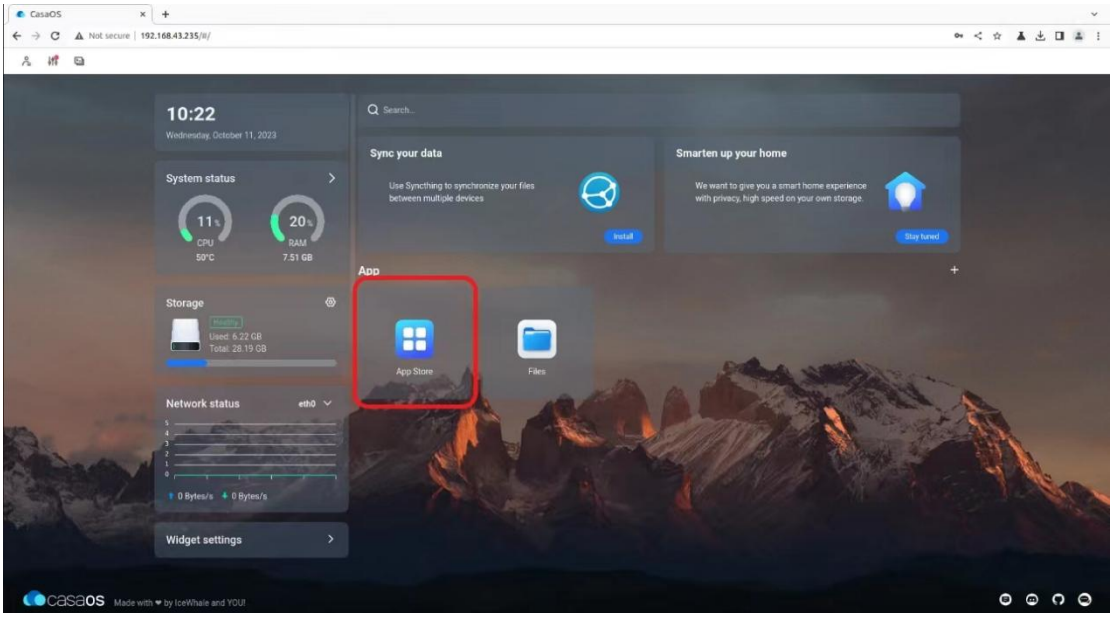
10) Another function under the third icon is to print the CasaOS log. Click "Logs" to enter. The interface is as follows:



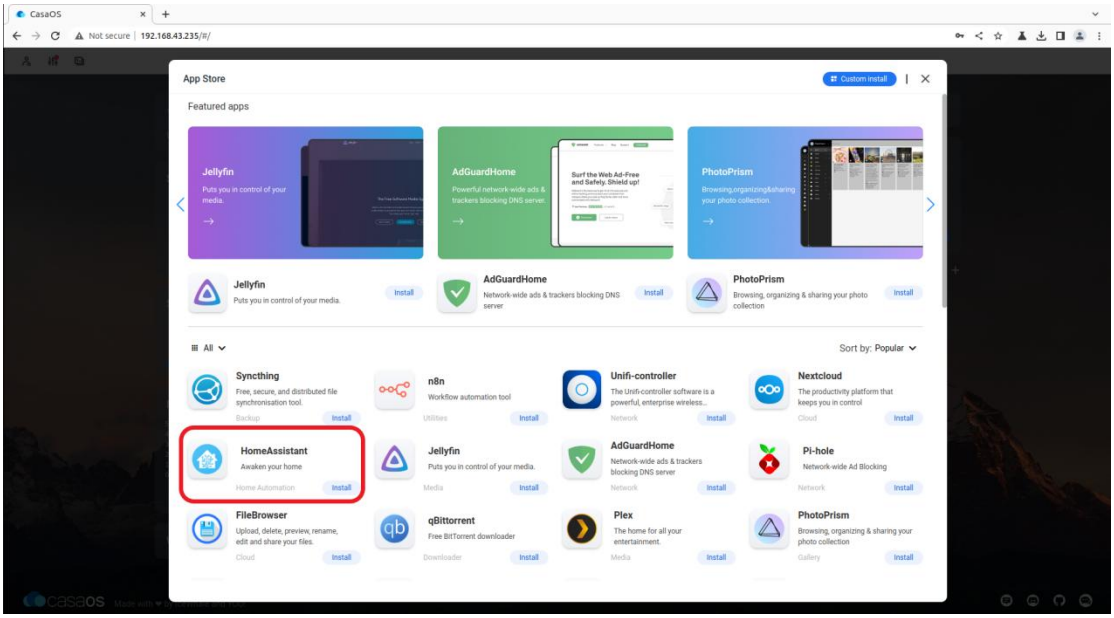
11) Click "Widget settings" in the lower left corner to set whether to display the performance panel widget on the main page



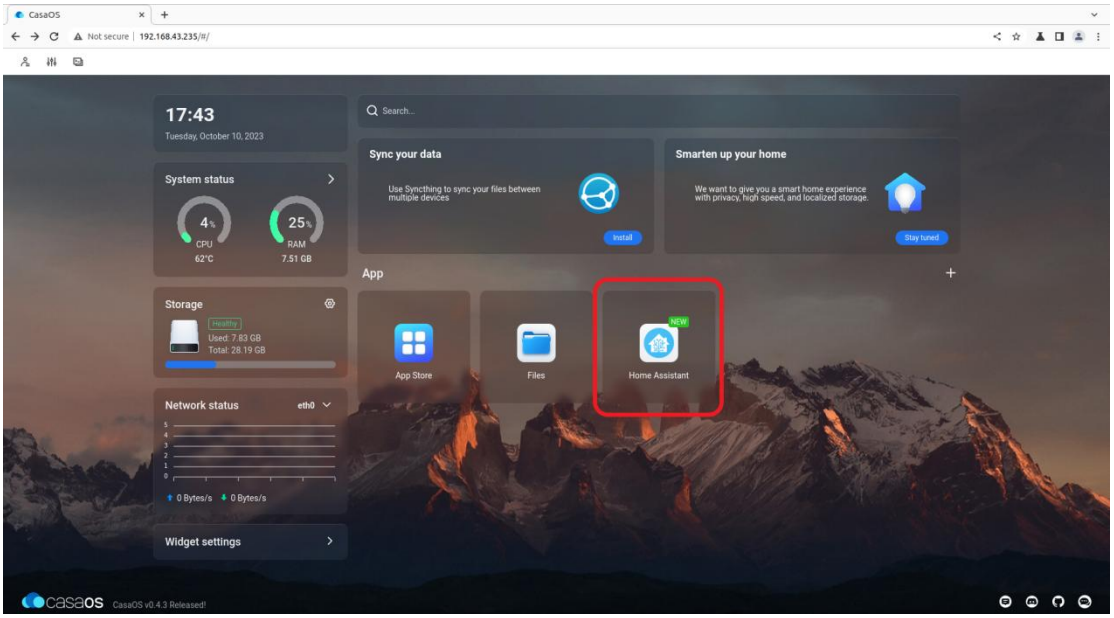
12) Click "APP Store" on the main interface to open the App Store. Currently, there are more than 70 APPs available in the App Store.



13) Here we take Home Assistant as an example to download. Find Home Assistant in the APP Store and click the corresponding "install"

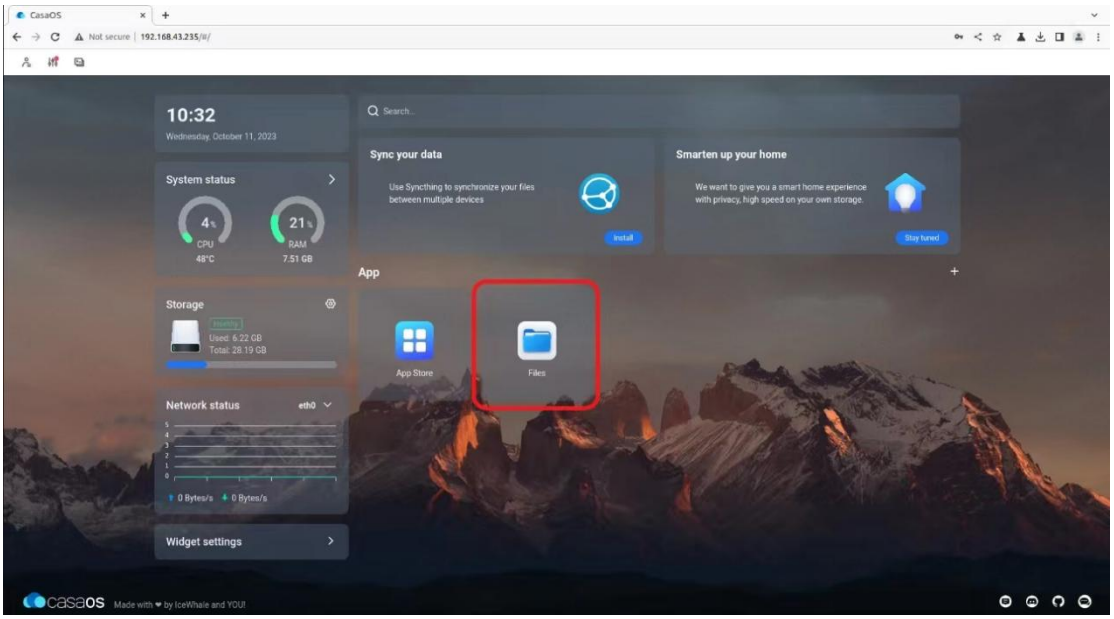


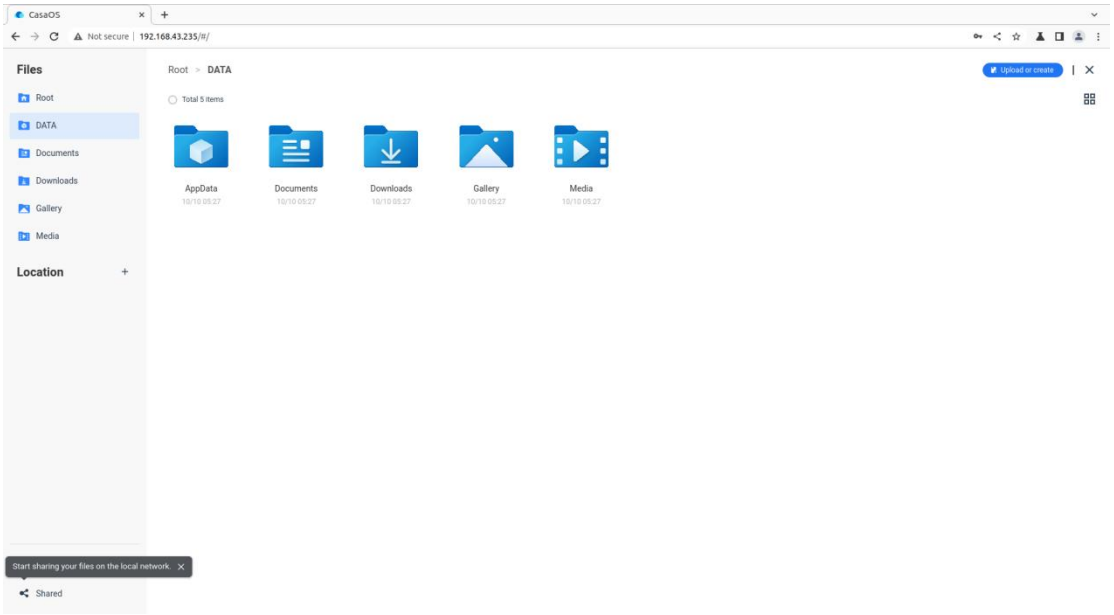
14) After the download is complete, HostAssitant will appear on the main page



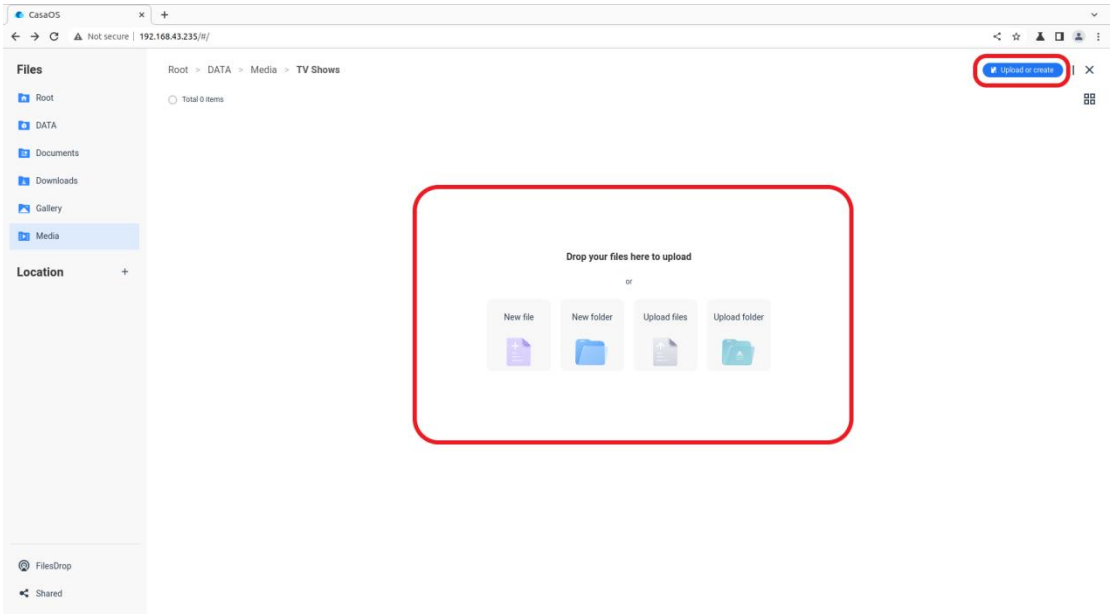
15) Click "Files" in the main interface to open the file system that comes with CasaOS, and then you can upload and save files

Please make sure other devices and the development board are in the same LAN.





16) When uploading files, you need to switch to the target folder, then drag the local file to the indicated area in the figure, or click "Upload or Create" in the upper right corner to select the file to upload.



17) If you want to uninstall CasaOS, you can use the following command：

```
orangepi@orangepi:~$ sudo casaos-uninstall
```

3. 37. Methods of using NPU

3. 37. 1. Prepare tools

1) A PC with Ubuntu 20.04 operating system

According to the official documentation of RKNN-Toolkit2, the operating systems supported by the current version of RKNN-Toolkit2 are as follows：

a. Ubuntu18.04 (x64)

b. Ubuntu20.04 (x64)

c. Ubuntu22.04 (x64)

In this document, we use the Ubuntu 20.04 (x64) operating system for demonstration. Please test other versions of the operating system yourself.

2) An RK3588 development board with Debian 11 installed

3) Prepare a good quality USB2.0 male-to-male data cable for using the adb function



3. 37. 2. Install RKNN-Toolkit2 on Ubuntu PC

Toolkit2 is a development kit used on the Ubuntu PC platform. Users can use the Python interface provided by the tool to easily complete functions such as model conversion, reasoning, and performance evaluation.

1) On the Ubuntu PC, open a command line window and enter the following commands to install python3 and pip3

```
test@test:~$ sudo apt-get install -y python3 python3-dev python3-pip
```

2) You can use the following command to view the installed version of python3

```
test@test:~$ python3 --version
Python 3.8.10
```

3) Then enter the following command to install the dependency package of RKNN-Toolkit2

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get -y install libxslt1-dev zlib1g-dev libglib2.0 \
 libsm6 libgl1-mesa-glx libprotobuf-dev gcc
```

4) Then enter the following command to download the 1.5.2 version of RKNN-Toolkit2

```
test@test:~$ git clone https://github.com/airockchip/rknn-toolkit2 -b v1.5.2
```

5) Then enter the following command to install the corresponding version of Python3 dependency packages. This command will use pip3 to install the dependencies listed in the file requirements_cp38-1.5.2.txt. If the dependencies are not fully installed, do not specify the installation source and install each package separately.

```
test@test:~$ pip3 install -r rknn-toolkit2/doc/requirements_cp38-1.5.2.txt -i https://mirror.baidu.com/pypi/simple
```

6) Then enter the following command to use pip3 to install the RKNN-Toolkit2 software package. After the installation is complete, you can use RKNN-Toolkit2

```
test@test:~$ pip3 install -r rknn-toolkit2/doc/requirements_cp38-1.5.2.txt -i https://pypi.tuna.tsinghua.edu.cn/simple
```

3.37.3. Model conversion and model inference using RKNN-Toolkit2

RKNN-Toolkit2 supports converting Caffe, TensorFlow, TensorFlow Lite, ONNX, DarkNet, PyTorch and other models into RKNN models, and then running the RKNN model on the Ubuntu PC through simulation or using the NPU of the development board for inference.

Relevant examples are provided in the example folder of RKNN-Toolkit2 to help users better understand how to operate. We take the ONNX model with yolov5 function as an example.

3.37.3.1. Simulate the model on Ubuntu PC

RKNN-Toolkit2 is equipped with a built-in simulator, which allows users to simulate the inference process of the model on Rockchip NPU on Ubuntu PC.

In this way, model conversion and inference can be completed on the Ubuntu PC, helping users test and verify their models faster.

1) First switch to the rknn-toolkit2/examples/onnx/yolov5 directory

```
test@test:~$ cd rknn-toolkit2/examples/onnx/yolov5/
```

2) Then run the test.py script, which first converts the yolov5s_relu.onnx model into an RKNN model that can be run on the simulator, and then uses the simulator to simulate and run the model to perform inference on the bus.jpg image in the current directory.

```
test@test:~/rknn-toolkit2/examples/onnx/yolov5$ python3 test.py
```

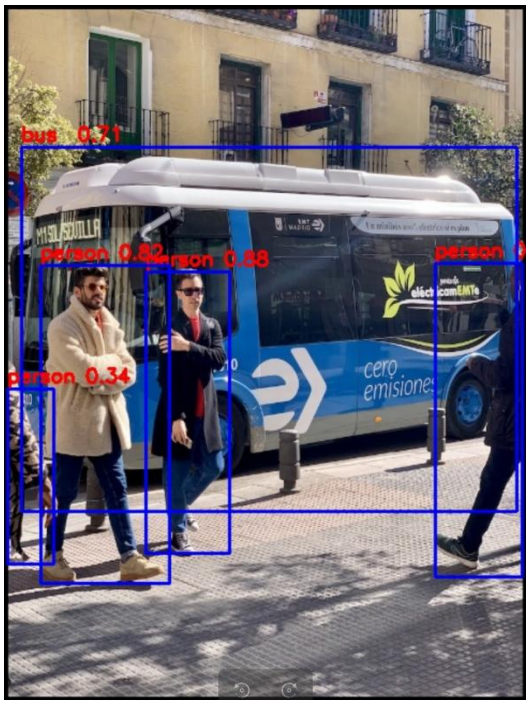
3) After the test.py script runs successfully, you will see the following print information, indicating that the model successfully detected four people and a bus in the bus.jpg image

```
done
--> Running model
W inference: The 'data_format' has not been set and defaults is nhwc!
done
class: person, score: 0.884139358997345
box coordinate left,top,right,down: [209.1040009856224, 244.4304337501526, 286.5742521882057,
506.7466902732849]
class: person, score: 0.8676778078079224
box coordinate left,top,right,down: [478.5757632255554, 238.58572268486023, 559.5273861885071,
526.479279756546]
class: person, score: 0.8246847987174988
box coordinate left,top,right,down: [110.57257843017578, 238.58099019527435,
230.54625701904297, 534.0008579492569]
class: person, score: 0.3392542004585266
box coordinate left,top,right,down: [79.96397459506989, 354.9062474966049, 122.13020265102386,
516.2529321908951]
class: bus , score: 0.7012234926223755
box coordinate left,top,right,down: [94.43931484222412, 129.53470361232758, 553.1492471694946,
468.0852304697037]
D NPUTransfer: Transfer client closed, fd = 3
```

4) The converted model file yolov5s_relu.rknn and the inference picture result result.jpg are saved in the current directory

5) The result.jpg image shows the object categories and confidence rates detected in the

bus.jpg image using the yolov5s_relu.rknn model



3.37.3.2. Using the NPU of the development board to run the model on Ubuntu PC

RKNN-Toolkit2 provides users with a Python interface for using the development board's NPU for reasoning through adb, allowing users to use the development board's NPU to run models for reasoning on an Ubuntu PC.

In this way, the Ubuntu PC can use the machine learning library provided by Python to optimize and adjust the model according to the actual effect of the model running on the NPU of the development board.

3.37.3.2.1. Connect adb using a USB male-to-male data cable

Use adb to operate the development board on the Ubuntu PC. For the usage of adb, please refer to the instructions in the section **How to use ADB**.

3.37.3.2.2. Update the rknn_derver and librknnrt.so of the development board

librknnrt.so is a board-side runtime library.

rknn_server is a background proxy service running on the development board. It is used to receive the protocol transmitted from the PC via USB, then execute the corresponding interface in the board runtime library and return the result to the PC.

1) First, enter the following command on the Ubuntu PC to download the 1.5.2 version of RKNPU2

```
test@test:~$ git clone https://github.com/rockchip-linux/rknpu2 -b v1.5.2
```

2) Then enter the following command on the Ubuntu PC to update the rknn_server of the development board through the adb tool

```
test@test:~$ adb push rknpu2/runtime/RK3588/Linux/rknn_server/aarch64/usr/bin/* /usr/bin
```

3) Then enter the following command on the Ubuntu PC to update the librknnrt.so library of the development board through the adb tool

```
test@test:~$ adb push rknpu2/runtime/RK3588/Linux/librknn_api/aarch64/librknnrt.so /usr/lib
```

4) Open the terminal of the development board through the adb tool

```
test@test:~$ adb shell
```

5) Open the rknn_server service of the development board

```
root@orangepi:/# sudo restart_rknn.sh
root@orangepi:/# start rknn server,version:1.5.2(8babfeabuild@2023-08-25T10:30:31)
I NPUTransfer: Starting NPU TransferServer,Transfer version 2.1.0(b5861e7@2020-11-23T11:50:51)
```

6) Using the following command to check. If the process ID of rknn_server appears, it means that rknn_server has been opened, and the operating environment of the development board has been set up.

```
root@orangepi:/# pgrep rknn_server
```

```
3131
```

3.37.3.2.3. Modify the parameters in the example

1) On the Ubuntu PC, you can use the following command to view the device ID of the development board connected to the Ubuntu PC. This ID will be used below.

```
test@test:~$ adb devices
List of devices attached
4f9f859e5a120324    device
```

2) Switch to the rknn-toolkit2/examples/onnx/yolov5 directory

```
test@test:~$ cd rknn-toolkit2/examples/onnx/yolov5/
```

3) Use the vim editor to modify the test.py file

```
test@test:~/rknn-toolkit2/examples/onnx/yolov5$ vim test.py
```

4) In the test.py file, we need to modify the following content:

a. In the preprocessing configuration, change the target platform to rk3588, so that the model conversion results in an RKNN model suitable for the NPU of the RK3588 development board.

```
# pre-process config
print('--> Config model')
rknn.config(mean_values=[[0, 0, 0]], std_values=[[255, 255, 255]], target_platform='rk3588')
print('done')
```

b. In the initialization running environment, add the description of the target platform and device ID. The target platform is rk3588, and the device ID is the device ID of the development board obtained through adb. The operation of running the model for inference will be performed on the NPU of the RK3588 development board.

```
# Init runtime environment
print('--> Init runtime environment')
ret = rknn.init_runtime(target='rk3588',device_id='4f9f859e5a120324')
if ret != 0:
    print('Init runtime environment failed!')
    exit(ret)
print('done')
```

c. After the modification is completed, save and exit

3. 37. 3. 2. 4. Run the example on Ubuntu PC

1) Enter the following command to run the test.py script. The script first converts the yolov5s_relu.onnx model to the RKNN model, and then loads the model to the NPU of the development board to perform inference on the out.jpg image in the current directory.

```
test@test:~/rknn-toolkit2/examples/onnx/yolov5$ python3 test.py
```

2) In the printed information, we can see that the Ubuntu PC uses the NPU of the development board to run the model for inference through the adb tool

```
--> Init runtime environment
I target set by user is: rk3588
I Check RK3588 board npu runtime version
I Starting ntp or adb, target is RK3588
I Device [4f9f859e5a120324] not found in ntb device list.
I Start adb...
I Connect to Device success!
I NPUTransfer: Starting NPU Transfer Client, Transfer version 2.1.0
(b5861e7@2020-11-23T11:50:36)
```

3) After the test.py script runs successfully, the converted model file yolov5s_relu.rknn and the inference image result result.jpg are saved in the current directory

4) The results of the operation are the same as those in the section **Simulating the model on the Ubuntu PC**.

3.37.4. Call the C interface to deploy the RKNN model to the development board to run

RKNPU2 provides a C programming interface for chip platforms with Rockchip NPU, which can help users deploy RKNN models exported using RKNN-Toolkit2 and accelerate the implementation of AI applications.

In the example folder of RKNPU2, examples of deploying RKNN models with different functions to the development board are provided. We take the deployment of the RKNN model with yolov5 function to the RK3588 Debian11 platform as an example.

3.37.4.1. Download cross-compilation tools

Since the development board runs on Linux, you need to use the gcc cross compiler to compile. It is recommended to use the gcc version gcc-9.3.0-x86_64_arch64-linux-gnu.

Enter the following command to download this version of gcc. After downloading, you will get a folder named gcc-buildroot-9.3.0-2020.03-x86_64_aarch64-rockchip-linux-gnu.

```
test@test:~$ git clone https://github.com/airockchip/gcc-buildroot-9.3.0-2020.03-x86_64_aarch64-rockchip-linux-gnu
```

3.37.4.2. Modify the compiler tool path in the script

1) Switch to the rknpu2/examples/rknn_yolov5_demo directory

```
test@test:~$ cd ~/rknpu2/examples/rknn_yolov5_demo
```

2) Use the vim editor to modify the contents of the build-linux_RK3588.sh file.

```
test@test:~/rknpu2/examples/rknn_yolov5_demo$ vim build-linux_RK3588.sh
```

3) In the build-linux_RK3588.sh file, we need to change the value of the variable TOOL_CHAIN to the path of the gcc-buildroot-9.3.0-2020.03-x86_64_aarch64-rockchip-linux-gnu folder. In this way, when running the build-android_RK3588.sh script, the cross-compilation tool in the gcc-buildroot-9.3.0-2020.03-x86_64_aarch64-rockchip-linux-gnu folder will be used for compilation.

```
TARGET_SOC="rk3588"
GCC_COMPILER=aarch64-linux-gnu

export TOOL_CHAIN=~/gcc-buildroot-9.3.0-2020.03-x86_64_aarch64-rockchip-linux-gnu
export LD_LIBRARY_PATH=${TOOL_CHAIN}/lib64:$LD_LIBRARY_PATH
export CC=${GCC_COMPILER}-gcc
export CXX=${GCC_COMPILER}-g++
```

4) After the modification is completed, save and exit.

3.37.4.3. Compile rknn_yolov5_demo

1) Run build-linux_RK3588.sh, which generates a program suitable for the RK3588 development board through cross-compilation and can run the RKNN model for inference on it

```
test@test:~/rknpu2/examples/rknn_yolov5_demo$ sudo apt install cmake
test@test:~/rknpu2/examples/rknn_yolov5_demo$ sudo apt-get install g++-aarch64-linux-gnu
test@test:~/rknpu2/examples/rknn_yolov5_demo$ ./build-linux_RK3588.sh
```

2) After running build-linux_RK3588.sh, there will be an additional folder named install in the current directory. The rknn_yoov5_demo_Linux folder under this folder contains the program generated by cross-compilation and its related files

```
test@test:~/rknpu2/examples/rknn_yolov5_demo$ ls install
rknn_yolov5_demo_Linux
```

3.37.4.4. Deploy rknn_yolov5_demo to the development board

On the Ubuntu PC, you can use the following command to upload the rknn_ yolov5_demo_Linux folder to the development board through the adb tool to depl oy rknn_yolov5_demo on the development board.

```
test@test:~/rknpu2/examples/rknn_yolov5_demo$ adb push \
install/rknn_yolov5_demo_Linux /data/rknn_yolov5_demo_Linux
```

3.37.4.5. Run rknn_yolov5_demo on the development board

1) Enter the file system of the development board through adb shell on the Ubuntu PC

```
test@test:~$ adb shell
root@orangepi:/#
```

2) Switch to the rknn_yolov5_demo_Linux directory

```
root@orangepi:/# cd /data/rknn_yolov5_demo_Linux/
root@orangepi:/data/rknn_yolov5_demo_Linux# ls
lib  model  rknn_yolov5_demo  rknn_yolov5_video_demo
```

3) Then run the rknn_yolov5_demo program to perform inference. In the following command, the program uses the yolov5s-640-640.rknn model to perform inference on the bus.jpg image. The entire running process will be completed on the development board

```
root@orangepi:/data/rknn_yolov5_demo_Linux# ./rknn_yolov5_demo \
./model/RK3588/yolov5s-640-640.rknn ./model/bus.jpg
```

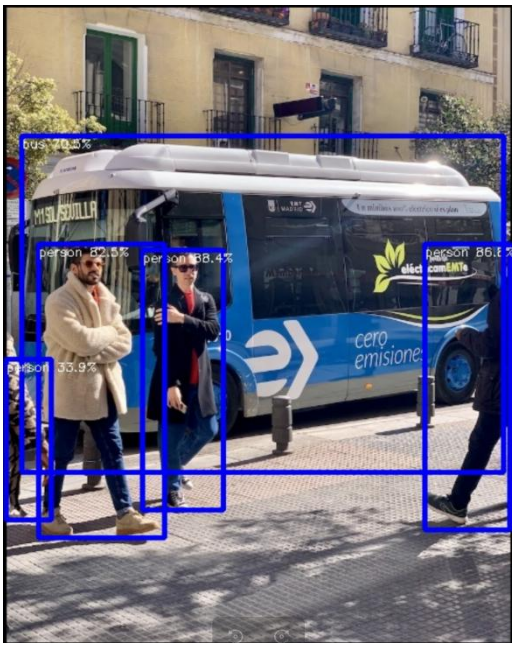
4) After the run is completed, the inference result out.jpg image is saved in the current directory

```
root@orangepi:/data/rknn_yolov5_demo_Linux# ls
lib  model  out.jpg  rknn_yolov5_demo  rknn_yolov5_video_demo
```

5) On the Ubuntu PC, we can use the following command to download the out.jpg image using the adb tool, and then view it using an image viewer

```
test@test:~$ adb pull /data/rknn_yolov5_demo_Linux/out.jpg ~/Desktop/
/data/rknn_yolov5_demo_Linux/out.jpg: ...led. 1.9 MB/s (191507 bytes in 0.095s)
```

6) The out.jpg image shows the object categories and confidence rates detected in the bus.jpg image using the yolov5s-640-640.rknn model



3. 38. How RK3588 uses Baidu Feijiang

Use Baidu Feijiang on the rk3588 development board, including converting the pdmodel model to the rknn model on the PC side and deploying the rknn model using the FastDeploy deployment tool developed by Baidu Feijiang on the board side. The following content is implemented in an environment where the PC system is Ubuntu 22.04 and the board system is Debian 11. Please test other environments yourself.

3. 38. 1. Ubuntu PC environment setup

The tools and uses that need to be installed on the Ubuntu PC are as follows

Tool Name	application
Anaconda3	Used to create and manage Python environments
Paddle2ONNX	Used to convert the pdmodel model to the ONNX model
RKNN-Toolkit2	Used to convert ONNX model to RKNN model

3. 38. 1. 1. Install Anaconda3 on PC

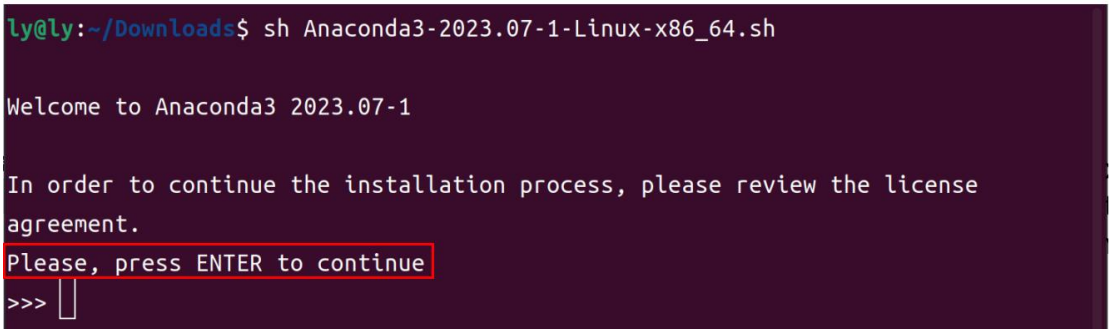
1）Open the browser on the Ubuntu PC and enter the following URL in the address bar to download and install the Anaconda3 script. After the download is complete, you will get the **Anaconda3-2023.07-1-Linux-x86_64.sh** file

https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2023.07-1-Linux-x86_64.sh

2）Then open the terminal and run the **Anaconda3-2023.07-1-Linux-x86_64.sh** script to install Anaconda3

```
test@test:~/Downloads$ sh Anaconda3-2023.07-1-Linux-x86_64.sh
```

3）The installation script will then output the following prompt message, at this time click Enter to continue the installation



4）After pressing the Enter key, some introduction information about Anaconda3 will appear. Keep pressing the " ↓ " key.

```
==================================================
End User License Agreement - Anaconda Distribution
==================================================

Copyright 2015-2023, Anaconda, Inc.

All rights reserved under the 3-clause BSD License:

This End User License Agreement (the "Agreement") is a legal agreement between y
ou and Anaconda, Inc. ("Anaconda") and governs your use of Anaconda Distribution
 (which was formerly known as Anaconda Individual Edition).

Subject to the terms of this Agreement, Anaconda hereby grants you a non-exclusi
ve, non-transferable license to:

  * Install and use the Anaconda Distribution (which was formerly known as Anaco
nda Individual Edition),
  * Modify and create derivative works of sample source code delivered in Anacon
da Distribution from Anaconda's repository, and;
  * Redistribute code files in source (if provided to you by Anaconda as source)
 and binary forms, with or without modification subject to the requirements set
forth below, and;

--更多--
```

5）The installation script will then prompt you to accept the license terms. Enter yes and press Enter.

```
The following packages listed on https://www.anaconda.com/cryptography are inclu
ded in the repository accessible through Anaconda Distribution that relate to cr
yptography.

Last updated February 25, 2022


Do you accept the license terms? [yes|no]
[no] >>>
```

6）The installation script will then prompt you to install Anaconda3 to your home directory. Press Enter to confirm.

```
Anaconda3 will now be installed into this location:
/home/ly/anaconda3

  - Press ENTER to confirm the location
  - Press CTRL-C to abort the installation
  - Or specify a different location below

[/home/ly/anaconda3] >>>
```

7）Then the installation script will prompt whether to initialize Anaconda3, enter yes and press Enter

```
installation finished.
Do you wish the installer to initialize Anaconda3
by running conda init? [yes|no]
[no] >>>
```

8）When you see the following print in the terminal, it means that Anaconda3 has been successfully installed

```
If you'd prefer that conda's base environment not be activated on startup,
   set the auto_activate_base parameter to false:

conda config --set auto_activate_base false

Thank you for installing Anaconda3!
```

3. 38. 1. 2. Install RKNN-Toolkit2 on PC

1）Open the terminal on the Ubuntu PC and use the Anaconda3 tool to create a Python 3.8 environment

```
(base)test@test:~$ conda create -n fastdeploy python=3.8
```

2）Activate the python3.8 environment just created

```
(base)test@test:~$ conda activate fastdeploy
```

3）Then install pip3 development tools and package management tools

```
(fastdeploy)test@test:~$ sudo apt-get install python3-dev python3-pip
```

4）Then install the dependency package of RKNN-Toolkit2

```
(fastdeploy)test@test:~$ sudo apt-get install libxslt1-dev zlib1g-dev libglib2.0 libsm6 libgl1-mesa-glx libprotobuf-dev gcc
```

5）rknn_toolkit2 has a specific dependency on numpy, so you need to install numpy==1.16.6 first

```
(fastdeploy)test@test:~$ pip install numpy==1.16.6
```

6）Install git tool

```
(fastdeploy)test@test:~$ sudo apt install -y git
```

7）Then execute the following command to download RKNN-Toolkit2. After the download is complete, you will get the rknn-toolkit2 folder

```
(fastdeploy)test@test:~$ git clone https://github.com/rockchip-linux/rknn-toolkit2
```

8）Then execute the following command to install RKNN-Toolkit2 corresponding to python3.8 version

```
(fastdeploy)test@test:~$ pip install rknn-toolkit2/rknn-toolkit2/packages/rknn_toolkit2-1.6.0+81f21f4d-cp38-cp38-linux_x86_64.whl
```

3.38.1.3. Install Paddle2ONNX on PC

Execute the following command to install paddle2onnx.

```
(fastdeploy)test@test:~$ pip install paddle2onnx
```

3.38.2. Board environment construction

The tools and uses that need to be installed on the board are as follows：

Tool Name	application
Anaconda3	Used to create and manage Python environments
rknpu2	Basic driver for rknpu2
FastDeploy	After compilation, you can get the FastDeploy inference library

3. 38. 2. 1. Install Anaconda3 on the board

1）Open the browser on the board and enter the following URL in the address bar to download and install the Anaconda3 script. After the download is complete, you will get the **Anaconda3-2023.07-1-Linux-aarch64.sh** file

https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/Anaconda3-2023.07-1-Linux-aarch64.sh

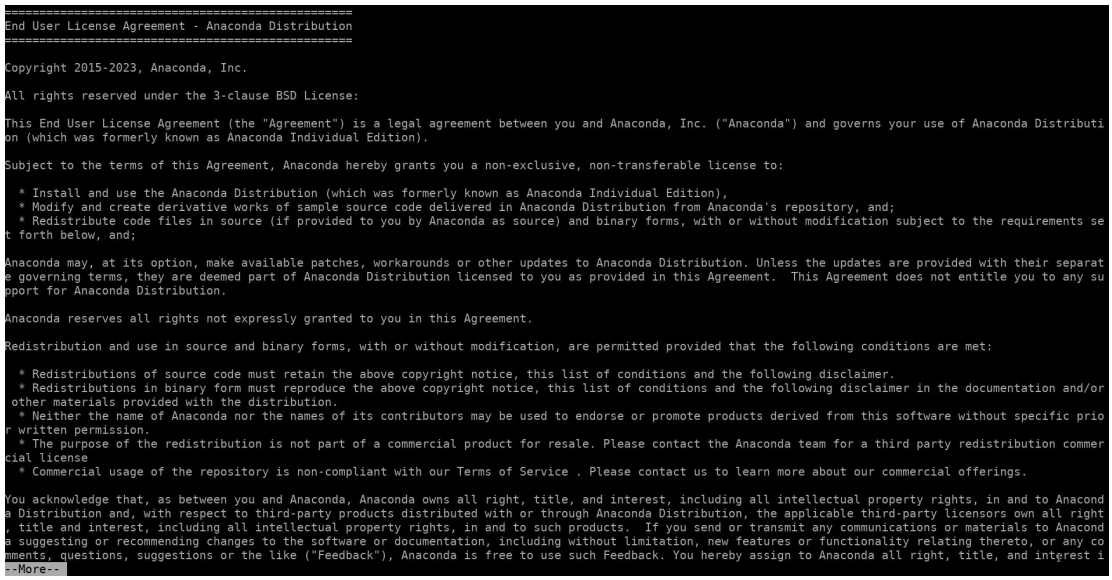
2）Open the terminal and run the **Anaconda3-2023.07-1-Linux-aarch64.sh** script to install Anaconda3

```
orangepi@orangepi:~/Downloads$ sh Anaconda3-2023.07-1-Linux-aarch64.sh
```

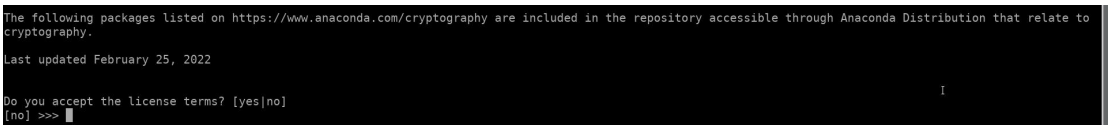
3）The installation script will then output the following prompt message, click Enter to continue the installation



4）After pressing the Enter key, some introduction information about Anaconda3 will appear. Keep pressing the " ↓ " key.



5）The installation script will then prompt you to accept the license terms. Enter yes and press Enter.



6）The installation script will then prompt you to install Anaconda3 to your home directory. Press Enter to confirm.



7）Then the installation script will prompt whether to initialize Anaconda3, enter yes and press Enter



8）When you see the following print in the terminal, it means that Anaconda3 has been successfully installed



9）If you use the conda command in the terminal and it says the command does not exist, you need to modify the ~/.bashrc file

```
orangepi@orangepi:~$ vi ~/.bashrc
```

10）Add the following code to the end of the ~/.bashrc file

```
export PATH=/home/orangepi/anaconda3/bin:$PATH
```

11）Then enter the following command in the terminal to make the changes take effect

```
orangepi@orangepi:~$ source ~/.bashrc
```

12）Then enter the following command in the terminal to initialize conda

```
(base)orangepi@orangepi:~$ conda init bash
```

13）Then close the current terminal and reopen a new terminal. You can now use the conda command normally.

3.38.2.2. Install rknpu2 driver on the board

1）Open the terminal on the board and use the Anaconda3 tool to create a Python version 3.9 environment

```
(base)orangepi@orangepi:~$ conda create -n fastdeploy python=3.9
```

2）Activate the python3.9 environment just created

```
(base)orangepi@orangepi:~$ conda activate fastdeploy
```

3）Download the rknpu2_device_install_1.4.0.zip file via wget

```
(fastdeploy)orangepi@orangepi:~$ wget https://bj.bcebos.com/fastdeploy/third_libs/rknpu2_device_install_1.4.0.zip
```

4）Then execute the following command to decompress rknpu2_device_install_1.4.0.zip. After decompression, you will get the rknpu2_device_install_1.4.0 folder and the __MACOSX folder

```
(fastdeploy)orangepi@orangepi:~$ unzip rknpu2_device_install_1.4.0.zip
```

5）Switch to the rknpu2_device_install_1.4.0 directory

```
(fastdeploy)orangepi@orangepi:~$ cd rknpu2_device_install_1.4.0/
```

6）There is a rknn_install_rk3588.sh script in this directory. Run the script to complete the installation of the board-side rknpu2 driver.

```
(fastdeploy)orangepi@orangepi:~/rknpu2_device_install_1.4.0$ sudo bash rknn_install_rk3588.sh
```

3.38.2.3. Compile FastDeploy C++ SDK on the board

1）The cmake command is needed when compiling. You can execute the following command to install the cmake tool

```
(fastdeploy)orangepi@orangepi:~$ sudo apt-get install -y cmake
```

2）Then download the FastDeploy SDK. After the command is executed, you will get the FastDeploy folder

```
(fastdeploy)orangepi@orangepi:~$ git clone https://github.com/PaddlePaddle/FastDeploy.git
```

3）Switch to the FastDeploy directory

```
(fastdeploy)orangepi@orangepi:~$ cd FastDeploy
```

4）Create a compilation directory build and switch to the build directory

```
(fastdeploy)orangepi@orangepi:~/FastDeploy$ mkdir build && cd build
```

5）Before compiling, you need to use cmake to configure the project information to be compiled. After executing the following command, there will be some more files in the current directory, including the Makefile file used for compilation

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/build$ cmake .. -DENABLE_ORT_BACKEND=ON \
-DENABLE_RKNPU2_BACKEND=ON \
-DENABLE_VISION=ON \
-DRKNN2_TARGET_SOC=RK3588 \
-DCMAKE_INSTALL_PREFIX=${PWD}/fastdeploy-0.0.3
```

6）Execute the following command to start compiling

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/build$ make -j8
```

7）After the compilation is complete, use the following command to install the compiled files to the specified path

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/build$ make install
```

8）After the compilation is completed, you will get the fastdeploy-0.0.3 folder. In this folder, there is a script file fastdeploy_init.sh for configuring environment variables. After using this script to configure environment variables, you can use some compiled library files.

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/build$ source fastdeploy-0.0.3/fastdeploy_init.sh
```

3. 38. 3. Example of deploying a model using FastDeploy

The ResNet50_vd model is a model used for target classification. The following uses the ResNet50_vd model as an example to illustrate the process of using FastDeploy to deploy the pdmodel model.

3.38.3.1. Ubuntu PC model conversion

1）Open the terminal on the PC and activate the python3.8 environment created using Anaconda3

```
test@test:~$ conda activate fastdeploy
```

2）In the model conversion script, you need to import the yaml module and the six module. You can execute the following command to install them.

```
(fastdeploy)test@test:~$ pip install pyyaml six
```

3）Execute the following command to download the ResNet50_vd_infer.tgz file

```
(fastdeploy)test@test:~$ wget https://bj.bcebos.com/paddlehub/fastdeploy/ResNet50_vd_infer.tgz
```

4）After decompressing the ResNet50_vd_infer.tgz file, you can get the ResNet50_vd_infer folder, which contains the pdmodel model file inference.pdmodel and other related files.

```
(fastdeploy)test@test:~$ tar -xvf ResNet50_vd_infer.tgz
```

5）Use the following command to convert the pdmodel model to an onnx model through paddle2onnx. After executing the command, there will be an additional onnx model file ResNet50_vd_infer.onnx in the ResNet50_vd_infer folder.

```
(fastdeploy)test@test:~$ paddle2onnx --model_dir ResNet50_vd_infer \
--model_filename inference.pdmodel \
--params_filename inference.pdiparams \
--save_file ResNet50_vd_infer/ResNet50_vd_infer.onnx \
--enable_dev_version True \
--opset_version 10
```

6）Then use the following command to fix the shape to [1,3,224,224]. After executing the command, the ResNet50_vd_infer.onnx file will be modified.

```
(fastdeploy)test@test:~$ python -m paddle2onnx.optimize --input_model \
ResNet50_vd_infer/ResNet50_vd_infer.onnx \
--output_model ResNet50_vd_infer/ResNet50_vd_infer.onnx \
--input_shape_dict "{'inputs':[1,3,224,224]}"
```

7）To convert the onnx model to the rknn model, you need to use the script in the FastDeploy SDK. Execute the following command to download FastDeploy

```
(fastdeploy)test@test:~$ git clone https://github.com/PaddlePaddle/FastDeploy.git
```

8）Then transfer the ResNet50_vd_infer folder to the corresponding directory of FastDeploy

```
(fastdeploy)test@test:~$ mv ResNet50_vd_infer \
FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/
```

9）Switch to the directory where the model conversion is performed

```
(fastdeploy)test@test:~$ cd FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/
```

10）Execute the following command to convert the onnx model to the rknn model, and finally get the rknn model file in the ResNet50_vd_infer directory ResNet50_vd_infer_rk3588_unquantized.rknn

```
(fastdeploy)test@test:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/$ python ./rknpu2_tools/export.py \
--config_path ./rknpu2_tools/config/ResNet50_vd_infer_rknn.yaml \
--target_platform rk3588
```

11）When deploying on the board, the rknn model file used is named ResNet50_vd_infer_rk3588.rknn, so you need to rename the ResNet50_vd_infer_rk3588_unquantized.rknn file to ResNet50_vd_infer_rk3588.rknn

```
(fastdeploy)test@test:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/$ mv ResNet50_vd_infer/ResNet50_vd_infer_rk3588_unquantized.rknn \
ResNet50_vd_infer/ResNet50_vd_infer_rk3588.rknn
```

3. 38. 3. 2. Board-side model deployment

1）Open the terminal on the board and activate the python3.9 environment created previously using Anaconda3

```
orangepi@orangepi:~$ conda activate fastdeploy
```

2）Run the fastdeploy_init.sh script to configure the environment

```
(fastdeploy)orangepi@orangepi:~$ source FastDeploy/build/fastdeploy-0.0.3/fastdeploy_init.sh
```

3）Switch to the sample directory for deploying the ResNet50 model in FastDeploy

```
(fastdeploy)orangepi@orangepi:~$ cd FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp
```

4）Create a directory structure under this directory

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp$ mkdir build images ppclas_model_dir thirdpartys
```

5）Copy the compiled fastdeploy-0.0.3 folder to the thirdpartys folder

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp$ cp -r ~/FastDeploy/build/fastdeploy-0.0.3/ thirdpartys/
```

6）Copy the files in the ResNet50_vd_infer folder on the PC to the ppclas_model_dir directory

7）Switch to the images directory

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp$ cd images
```

8）Download the test image in the images directory using wget

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/images$ wget https://gitee.com/paddlepaddle/PaddleClas/raw/release/2.4/deploy/images/ImageNet/ILSVRC2012_val_00000010.jpeg
```

9）Then switch to the build directory

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/images$ cd ../build/
```

10）Use cmake to configure the content that needs to be compiled. After executing the command, some files will appear in the current directory, including the Makefile file

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/build$ cmake ..
```

11）Execute the following command to start compiling

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/build$ make -j8
```

12）Execute the following command to install the compiled files to the specified path. After executing the command, an install directory will appear in the current directory.

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/build$ make install
```

13）Switch to the install directory, where the model is used for reasoning.

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/build$ cd install
```

14）Use the following command to use the converted rknn model to classify the content in the ILSVRC2012_val_00000010.jpeg image:

```
(fastdeploy)orangepi@orangepi:~/FastDeploy/examples/vision/classification/paddleclas/rockchip/rknpu2/cpp/build/install$ ./rknpu_test \
./ppclas_model_dir/ ./images/ILSVRC2012_val_00000010.jpeg
```

15）After executing the command, the following information will be displayed, indicating that the category ID number of the object in the image is 644 and the confidence rate is 0.072998

```
ClassifyResult(
label_ids: 644,
scores: 0.072998,
)
```

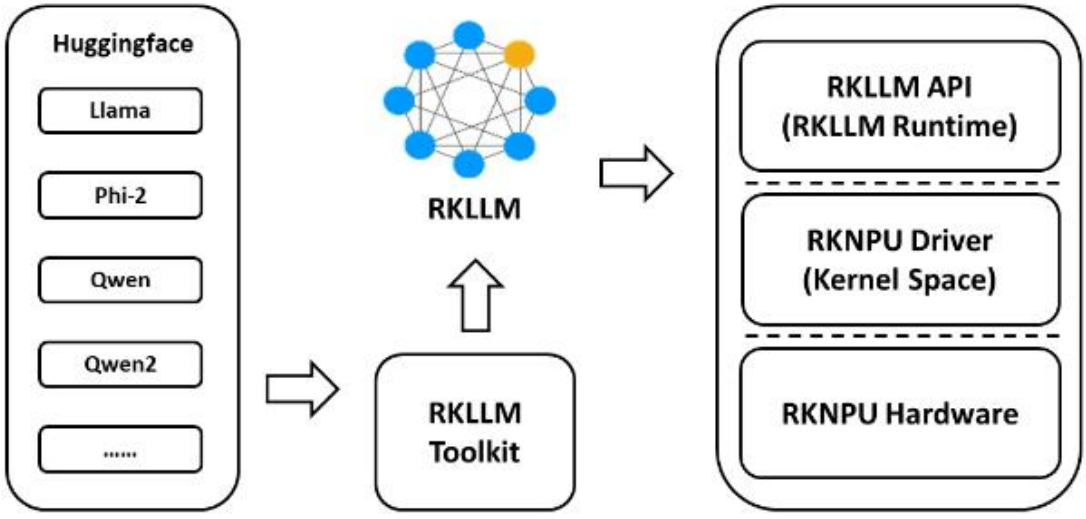
3. 39. RK3588 How to run the RKLLM large model

The codes and models used in this section can be downloaded from the official tools of the development board.

3. 39. 1. Introduction to RKLLM

For more detailed RKLLM introduction information, please refer to Rockchip RKLLM official information.

RKLLM can help users quickly deploy LLM models to the RK3588 development board. The overall framework is shown in the figure below.:



3.39.1.1. Introduction to RKLLM toolchain

3.39.1.1.1. RKLLM-Toolkit Function Introduction

RKLLM-Toolkit is a development kit that provides users with the ability to quantize and convert large language models on a computer. The following functions can be easily accomplished through the Python interface provided by the tool:：

1) Model conversion: Supports converting the Hugging Face format Large Language Model (LLM) to the RKLLM model. Currently, the models we have tested include TinyLLAMA, Qwen, Qwen2, Phi-3, ChatGLM3, Gemma, InternLM2, and MiniCPM. The converted RKLLM model can be loaded and used on the RK3588 platform.

2) Quantization function: supports quantizing floating-point models to fixed-point models. The currently supported quantization type is w8a8, which means that both weights and activations are quantized to 8-bit width.

3.39.1.1.2. RKLLM Runtime Function Introduction

RKLLM Runtime is mainly responsible for loading the RKLLM model converted by RKLLM-Toolkit, and implementing the reasoning of the RKLLM model on the RK3588 NPU by calling the NPU driver on the RK3588 board. When reasoning the RKLLM model, the user can define the reasoning parameter settings of the RKLLM model, define different text generation methods, and continuously obtain the reasoning results of the model through pre-defined callback functions. For more detailed instructions, please refer to **Rockchip RKLLM official information**.

3.39.1.2. Introduction to RKLLM development process

The overall development steps of RKLLM are mainly divided into two parts: model conversion and board-side deployment and operation.

1) **Perform model conversion on the Ubuntu PC**. At this stage, the large language model in Hugging Face format provided by the user will be converted to RKLLM format for efficient reasoning on the RK3588 development board. This step includes：

a. Build the RKLLM-Toolkit environment: Use Conda to build the RKLLM-Toolkit operating environment on the Ubuntu PC

b. Model conversion: Use RKLLM-Toolkit to convert the obtained Hugging Face format large language model or the self-trained large language model (note that the structure of the saved model must be consistent with the model structure on the Hugging Face platform) into a .rkllm format file that can run on the RK3588 development board.

c. Compile test code: Use rkllm-runtime to compile the inference program that can run on the RK3588 development board.

For the specific development process of model conversion on Ubuntu PC, please refer to **the detailed steps of model conversion and source code compilation on Ubuntu PC**.

2) **Deploy and run on the development board**. This stage covers the actual deployment and operation of the model on the RK3588 development board. It usually includes the following steps：

a. Upgrade the kernel NPU version: Upgrade the NPU version of the development board kernel to v0.9.6.

b. Model reasoning: Place the reasoning program compiled by rkllm-runtime on the Ubuntu PC and the .rkllm format file converted by RKLLM-Toolkit on the development board for model reasoning. You can run reasoning directly on the development board. For the specific development process, please refer to the **detailed steps of development board deployment and operation section of this chapter**. You can also deploy the board-side Server service on the development board. The Ubuntu PC in the same network segment can call the RKLLM model for reasoning by accessing the corresponding address. For the specific development process, please refer to **the detailed steps of development board server deployment and operation section of this chapter**.

The above two steps constitute the complete RKLLM development process, ensuring that the large language model can be successfully converted, debugged, and ultimately deployed efficiently on the RK3588 NPU.

3.39.2. Prepare tools

1) **A PC with Ubuntu 22.04 operating system. In this document, we use Ubuntu 22.04 (x64) operating system for demonstration. Please test other versions of operating system by yourself.**

2) An RK3588 development board.

3.39.3. Detailed steps for model conversion and source code compilation on Ubuntu PC

3.39.3.1. Build RKLLM-Toolkit environment

1) First download the RKLLM toolchain.

```
test@test:~$ git clone https://github.com/airockchip/rknn-llm.git
```

2) After downloading, use the ls command to check whether the downloaded file is correct.

```
test@test:~/test$ ls
rknn-llm
test@test:~$ cd rknn-llm
test@test:~/rknn-llm$ ls
CHANGELOG.md  doc  LICENSE  README.md  res  rkllm-runtime
rkllm-toolkit  rknpu-driver
```

3) The specific file directory in rknn-llm is as follows:

```
test@test:~/rknn-llm$ sudo apt install tree
test@test:~/rknn-llm$ tree
doc
└────Rockchip_RKLLM_SDK_CN.pdf       #RKLLM SDK documentation

rkllm-runtime
├────examples
│      └──── rkllm_api_demo    # Board-side inference call example project
│    └──── rkllm_server_demo # RKLLM-Server deployment example project
├────runtime
│      └──── Android
```

```
│           └──── librkllm_api
│               └────arm64-v8a
│                   └──── librkllmrt.so # RKLLM Runtime    Library
│               └────include
│                   └──── rkllm.h               # Runtime    header file
│       └──── Linux
│           └──── librkllm_api
│               └────aarch64
│                   └──── librkllmrt.so # RKLLM Runtime    Library
│               └────include
│                   └──── rkllm.h     # Runtime    header file

rkllm-toolkit
├────examples
│       └──── huggingface
│           └──── test.py
├────packages
│       └──── md5sum.txt
│       └──── rkllm_toolkit-x.x.x-cp38-cp38-linux_x86_64.whl
rknpu-driver
└────rknpu_driver_0.9.6_20240322.tar.bz2
```

4) Then download and install the miniforge3 installation package.

```
test@test:~$ wget -c https://mirrors.bfsu.edu.cn/github-release/conda-forge/miniforge/LatestRelease/Miniforge3-Linux-x86_64.sh

test@test:~$ chmod 777 Miniforge3-Linux-x86_64.sh

test@test:~$ bash Miniforge3-Linux-x86_64.sh
```

The mirror website sometimes crashes, resulting in the inability to download the miniforge3 package. The downloaded miniforge3 installation package has been provided in the official tool of the development board.

When running bash Miniforge3-Linux-x86_64.sh, just press Enter for all the options.

5) Then enter the Conda base environment.

```
test@test:~$ source ~/miniforge3/bin/activate
```

```
(base) test@test:~$
```

6) Then create a Conda environment named RKLLM-Toolkit with Python 3.8 (recommended version).

```
(base) test@test:~$ conda create -n RKLLM-Toolkit python=3.8
```

7) Then enter the RKLLM-Toolkit Conda environment.

```
(base) test@test:~$ conda activate RKLLM-Toolkit
(RKLLM-Toolkit) test@test:~$
```

8) Then use the pip command to install the whl package in the RKLLM toolchain downloaded previously. The directory is:：**rknn-llm/rkllm-toolkit/packages/rkllm_toolkit-1.0.1-cp38-cp38-linux_x86_64.whl**。During the installation process, the installation tool will automatically download the related dependency packages required by the RKLLM-Toolkit tool.

```
(base) test@test:~$ pip3 install rknn-llm/rkllm-toolkit/packages/rkllm_toolkit-1.0.1-cp38-cp38-linux_x86_64.whl
```

9) Finally, if there is no error when executing the following command, it means the installation is successful.

```
(RKLLM-Toolkit) test@test:~$ python
>>> from rkllm.api import RKLLM
```

3.39.3.2. Model conversion

In this section, we provide eight model conversion examples for users to choose from. If users encounter network problems when downloading models from Hugging Face, our development board official tool has integrated the downloaded model files and the corresponding .rkllm conversion files.

3.39.3.2.1. Converting the TinyLLAMA Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
```

```
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Next download the TinyLLAMA model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/TinyLlama/TinyLlama-1.1B-Chat-v1.0
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded TinyLlama-1.1B-Chat-v1.0 folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./TinyLlama.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/TinyLlama-1.1B-Chat-v1.0"    #Fill in your own path
ret = llm.export_rkllm("./TinyLlama.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of a successful conversion is as follows：

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
The argument `trust_remote_code` is to be used with Auto classes. It has no effect here and is ignored.
Optimizing model: 100%|| 22/22 [12:33<00:00, 34.27s/it]
Converting model: 100%|| 201/201 [00:00<00:00, 2031458.08it/s]
Model has been saved to ./TinyLlama.rkllm!
```

6) After the conversion is successful, you will get the TinyLlama.rkllm file in the current directory, which is about 1.09G in memory。

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    TinyLlama.rkllm
```

3. 39. 3. 2. 2. Conversion of Qwen Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
```

```
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Then download the Qwen model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/Qwen/Qwen-1_8B-Chat
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded Qwen-1_8B-Chat folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The path of the .rkllm format file to be saved is in brackets. We modify it to ret = llm.export_rkllm("./Qwen.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/Qwen-1_8B-Chat"    #Fill in your own path
ret = llm.export_rkllm("./Qwen.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
Loading checkpoint shards: 100%|| 2/2 [01:08<00:00, 34.02s/it]
Optimizing model: 100%|| 24/24 [14:26<00:00, 36.12s/it]
Converting model: 100%|| 195/195 [00:00<00:00, 1619582.73it/s]
Model has been saved to ./Qwen.rkllm!
```

6) If the conversion is successful, the Qwen.rkllm file will be obtained in the current directory, with a memory of about 2.01G.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py  Qwen.rkllm
```

3. 39. 3. 2. 3. Converting Qwen2 Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed,

you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Then download the Qwen2 model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/Qwen/Qwen1.5-0.5B
```

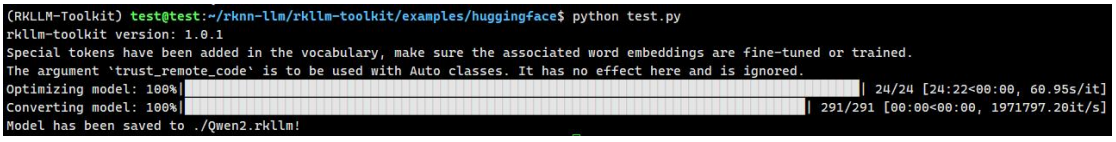
3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded Qwen1.5-0.5B folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The brackets are the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./Qwen2.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/Qwen1.5-0.5B"    #Fill in your own path
ret = llm.export_rkllm("./Qwen2.rkllm")
```

4) Run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:



6) If the conversion is successful, the Qwen2.rkllm file will be obtained in the current directory, with the memory of about 746M.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py  Qwen2.rkllm
```

3. 39. 3. 2. 4. Converting Phi-3 Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Next download the Phi-3 model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/microsoft/Phi-3-mini-4k-instruct
(RKLLM-Toolkit) test@test:~$ cd Phi-3-mini-4k-instruct
(RKLLM-Toolkit) test@test:~/Phi-3-mini-4k-instruct$ git reset --hard 291e9e30e38030c23497afa30f3af1f104837aa6
(RKLLM-Toolkit) test@test:~/Phi-3-mini-4k-instruct$ cd ..
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded Phi-3-mini-4k-instruct folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./Phi3.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/Phi-3-mini-4k-instruct"    # Fill in your path
ret = llm.export_rkllm("./Phi3.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:

```
(RKLLM-Toolkit) test@text:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
`flash-attention` package not found, consider installing for better performance: No module named 'flash_attn'.
Current `flash-attenton` does not support `window_size`. Either upgrade or use `attn_implementation='eager'`.
Loading checkpoint shards: 100%|████████████████████████████████████████████████████| 2/2 [00:02<00:00,  1.46s/it]
Optimizing model:   0%|                                                                | 0/32 [00:00<?, ?it/s]
You are not running the flash-attention implementation, expect numerical differences.
Optimizing model: 100%|████████████████████████████████████████████████████| 32/32 [15:36<00:00, 29.27s/it]
Converting model: 100%|████████████████████████████████████████████████████| 195/195 [00:00<00:00, 4109996.38it/s]
Model has been saved to ./Phi3.rkllm!
```

6) If the conversion is successful, you will get the Phi3.rkllm file in the current directory, which is about 3.66G in memory.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    Phi3.rkllm
```

3. 39. 3. 2. 5. Converting ChatGLM3 Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Next download the ChatGLM3 model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/THUDM/chatglm3-6b
(RKLLM-Toolkit) test@test:~$ cd chatglm3-6b
(RKLLM-Toolkit) test@test:~/chatglm3-6b$ git reset --hard 103caa40027ebfd8450289ca2f278eac4ff26405
(RKLLM-Toolkit) test@test:~/chatglm3-6b$ cd ..
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded chatglm3-6b folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./chatglm3.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/chatglm3-6b"    #Fill in your own path
ret = llm.export_rkllm("./chatglm3.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of a successful conversion is as follows：

```
(RKLLM-Toolkit) test@text:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
Setting eos_token is not supported, use the default one.
Setting pad_token is not supported, use the default one.
Setting unk_token is not supported, use the default one.
Loading checkpoint shards: 100%| 7/7 [00:00<00:00, 17.48it/s]
Optimizing model: 100%| 28/28 [28:03<00:00, 60.14s/it]
Converting model: 100%| 203/203 [00:00<00:00, 1028313.66it/s]
Model has been saved to ./chatglm3.rkllm!
```

6) Finally, if the conversion is successful, you will get the chatglm3.rkllm file in the current directory, which is about 6.07G in memory.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    chatglm3.rkllm
```

3. 39. 3. 2. 6. Converting Gemma Models

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Then download the Gemma model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/google/gemma-2b-it
(RKLLM-Toolkit) test@test:~$ cd gemma-2b-it
(RKLLM-Toolkit) test@test:~/gemma-2b-it$ git reset --hard de144fb2268dee1066f515465df532c05e699d48
(RKLLM-Toolkit) test@test:~/gemma-2b-it$ cd ..
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the

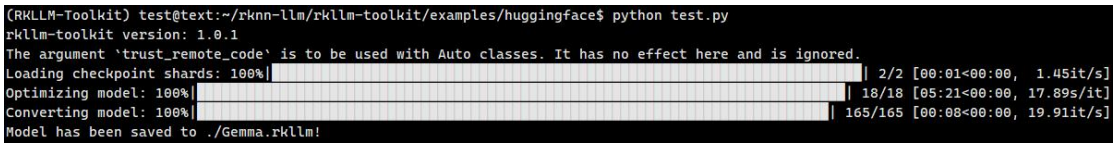
downloaded gemma-2b-it folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./Gemma.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/gemma-2b-it"    # Fill in your own path
ret = llm.export_rkllm("./Gemma.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:



6) If the conversion is successful, you will get the Gemma.rkllm file in the current directory, which is about 3.81G in memory.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    Gemma.rkllm
```

3. 39. 3. 2. 7. Converting the InternLM2 Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Next download the InternLM2 model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/internlm/internlm2-chat-1_8b
(RKLLM-Toolkit) test@test:~$ cd internlm2-chat-1_8b
```

```
(RKLLM-Toolkit) test@test:~/internlm2-chat-1_8b$ git reset --hard ecccbb5c87079ad84e5788baa55dd6e21a9c614d
(RKLLM-Toolkit) test@test:~/internlm2-chat-1_8b$ cd ..
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded internlm2-chat-1_8b folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./InternLM2.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/internlm2-chat-1_8b"    # Fill in your own path
ret = llm.export_rkllm("./InternLM2.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:

```
(RKLLM-Toolkit) test@text:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
Loading checkpoint shards: 100%| 2/2 [00:01<00:00,  1.23it/s]
Optimizing model: 100%| 24/24 [05:47<00:00, 14.49s/it]
Converting model: 100%| 171/171 [00:00<00:00, 2291456.82it/s]
Model has been saved to ./InternLM2.rkllm!
```

6) If the conversion is successful, you will get the InternLM2.rkllm file in the current directory, which is about 1.94G in memory.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    InternLM2.rkllm
```

3. 39. 3. 2. 8. Converting to MiniCPM Model

1) First install Git LFS on the Ubuntu operating system. If it has already been installed, you can skip this step.

```
(RKLLM-Toolkit) test@test:~$ sudo apt update
(RKLLM-Toolkit) test@test:~$ sudo apt install curl git
(RKLLM-Toolkit) test@test:~$ curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
(RKLLM-Toolkit) test@test:~$ sudo apt install git-lfs
```

```
(RKLLM-Toolkit) test@test:~$ git lfs install
```

2) Then download the MiniCPM model.

```
(RKLLM-Toolkit) test@test:~$ git clone https://huggingface.co/openbmb/MiniCPM-2B-sft-bf16
(RKLLM-Toolkit) test@test:~$ cd MiniCPM-2B-sft-bf16
(RKLLM-Toolkit) test@test:~/MiniCPM-2B-sft-bf16$ git reset --hard 79fbb1db171e6d8bf77cdb0a94076a43003abd9e
(RKLLM-Toolkit) test@test:~/MiniCPM-2B-sft-bf16$ cd ..
```

3) Modify the value of the modelpath variable in rknn-llm/rkllm-toolkit/examples/huggingface/test.py to the absolute path of the downloaded MiniCPM-2B-sft-bf16 folder, and then modify ret = llm.export_rkllm("./qwen.rkllm") The value in the brackets is the .rkllm format file path to be saved. We modify it to ret = llm.export_rkllm("./MiniCPM.rkllm").

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-toolkit/examples/huggingface/test.py
modelpath = "/path/your/MiniCPM-2B-sft-bf16"    #Fill in your own path
ret = llm.export_rkllm("./MiniCPM.rkllm")
```

4) Then run the rknn-llm/rkllm-toolkit/examples/huggingface/test.py file with python to convert the large model.

```
(RKLLM-Toolkit) test@test:~$ cd ~/rknn-llm/rkllm-toolkit/examples/huggingface
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
```

5) The output of successful conversion is as follows:

```
(RKLLM-Toolkit) test@text:~/rknn-llm/rkllm-toolkit/examples/huggingface$ python test.py
rkllm-toolkit version: 1.0.1
Optimizing model: 100%|| 40/40 [05:58<00:00,  8.95s/it]
Converting model: 100%|| 363/363 [00:00<00:00, 4531346.29it/s]
Model has been saved to ./MiniCPM.rkllm!
```

6) If the conversion is successful, you will get the MiniCPM.rkllm file in the current directory, which is about 3.07G in memory.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ ls
test.py    MiniCPM.rkllm
```

3. 39. 3. 3. Compiling the test code

1) First switch back to the ~ directory and then download the cross-compilation tool

chain and unzip it.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-toolkit/examples/huggingface$ cd ~
(RKLLM-Toolkit) test@test:~$ sudo apt install cmake
(RKLLM-Toolkit) test@test:~$ wget
https://developer.arm.com/-/media/Files/downloads/gnu-a/10.2-2020.11/binrel/gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar.xz
(RKLLM-Toolkit) test@test:~$ tar -xJf gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu.tar.xz
```

2) Then modify GCC_COMPILER_PATH in rknn-llm/rkllm-runtime/examples/rkllm_api_demo/build-linux.sh to ~/gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu。

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-runtime/examples/rkllm_api_demo/build-linux.sh
```

```
#!/bin/bash
# Debug / Release / RelWithDebInfo
if [[ -z ${BUILD_TYPE} ]];then
    BUILD_TYPE=Release
fi

GCC_COMPILER_PATH=~/gcc-arm-10.2-2020.11-x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu
C_COMPILER=${GCC_COMPILER_PATH}-gcc
CXX_COMPILER=${GCC_COMPILER_PATH}-g++
STRIP_COMPILER=${GCC_COMPILER_PATH}-strip
```

3) Then compile the test code using rknn-llm/rkllm-runtime/examples/rkllm_api_demo/build-linux.sh.

```
(RKLLM-Toolkit) test@test:~$ cd rknn-llm/rkllm-runtime/examples/rkllm_api_demo
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-runtime/examples/rkllm_api_demo$ bash build-linux.sh
```

4) After compiling, check the generated llm_demo file.

```
(RKLLM-Toolkit) test@test:~/rknn-llm/rkllm-runtime/examples/rkllm_api_demo$ ls build/build_linux_aarch64_Release
CMakeCache.txt  CMakeFiles  cmake_install.cmake  llm_demo  Makefile
```

3.39.4. Detailed steps for development board deployment and operation

3.39.4.1. Model Inference

It is recommended to use a development board with 8GB or more memory for testing. A development board with 4GB memory may not be able to run the model due to insufficient memory.

3.39.4.1.1. TinyLLAMA model inference

1) First, upload the llm_demo program and TinyLlama.rkllm model file compiled on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    TinyLlama.rkllm
```

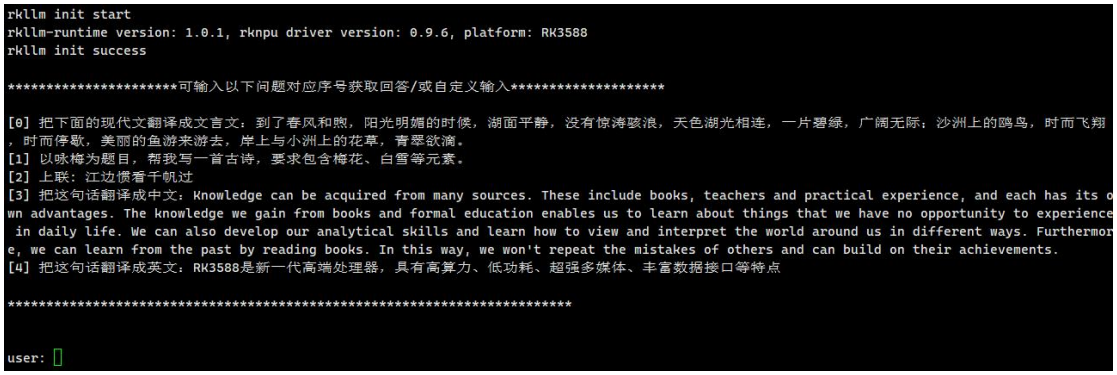
2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

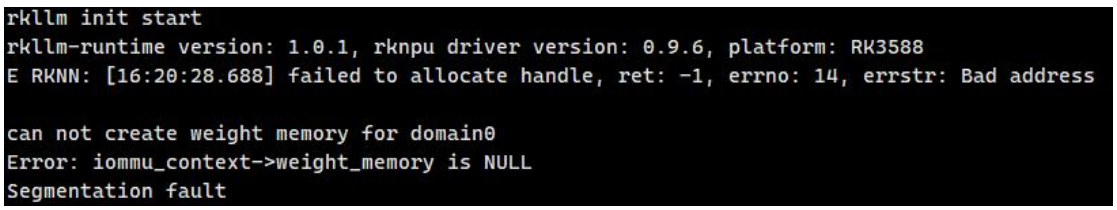
3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./TinyLlama.rkllm
```

4) If the operation is successful, the following interface will pop up.



5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.

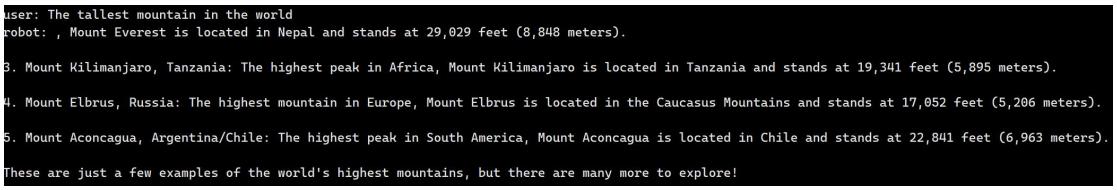



```
orangepi@orangepi:~$ sudo reboot
```

6) After entering the question in the interactive interface, press Enter. The result of a

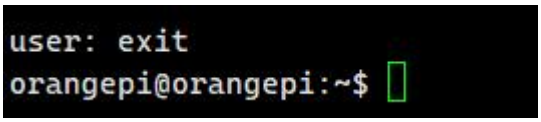
successful test is as follows：

Note that the TinyLLAMA model only supports English questions and answers. If you ask questions in Chinese, the model will speak nonsense. If you run TinyLLAMA on the development board, the model's answers are relatively random and cannot interact well.



7) Finally, enter exit to exit.

```
user: exit
```



3.39.4.1.2. Qwen model reasoning

1) First, upload the llm_demo program and Qwen.rkllm model file compiled on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    Qwen.rkllm
```

2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./Qwen.rkllm
```

4) If the operation is successful, the following interface will pop up.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
rkllm init success

**********************可输入以下问题对应序号获取回答/或自定义输入********************

[0] 把下面的现代文翻译成文言文：到了春风和煦，阳光明媚的时候，湖面平静，没有惊涛骇浪，天色湖光相连，一片碧绿，广阔无际；沙洲上的鸥鸟，时而飞翔，时而停歇，美丽的鱼游来游去，岸上与小洲上的花草，青翠欲滴。
[1] 以咏梅为题目，帮我写一首古诗，要求包含梅花、白雪等元素。
[2] 上联：江边惯看千帆过
[3] 把这句话翻译成中文：Knowledge can be acquired from many sources. These include books, teachers and practical experience, and each has its own advantages. The knowledge we gain from books and formal education enables us to learn about things that we have no opportunity to experience in daily life. We can also develop our analytical skills and learn how to view and interpret the world around us in different ways. Furthermore, we can learn from the past by reading books. In this way, we won't repeat the mistakes of others and can build on their achievements.
[4] 把这句话翻译成英文：RK3588是新一代高端处理器，具有高算力、低功耗、超强多媒体、丰富数据接口等特点

*************************************************************************

user:
```

5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
E RKNN: [16:20:28.688] failed to allocate handle, ret: -1, errno: 14, errstr: Bad address

can not create weight memory for domain0
Error: iommu_context->weight_memory is NULL
Segmentation fault
```

```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows:

```
user: 你能告诉我世界上最高的山是什么吗
robot: ?
当然可以，世界上最高的山是珠穆朗玛峰，位于中国和尼泊尔的交界处。它的海拔高度为8,848米（29,029英尺）。



user: 你能告诉我一年有多少个季节吗
robot: ?
一年有四个季节：春、夏、秋、冬。

是的，一年有四个季节：春、夏、秋、冬。每个季节都有不同的气候和天气条件，因此在不同季节里会有不同的景色和活动。
```

7) Finally, enter exit to exit.

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3. 39. 4. 1. 3. Qwen2 model reasoning

1) First, upload the llm_demo program and Qwen2.rkllm model file compiled on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    Qwen2.rkllm
```

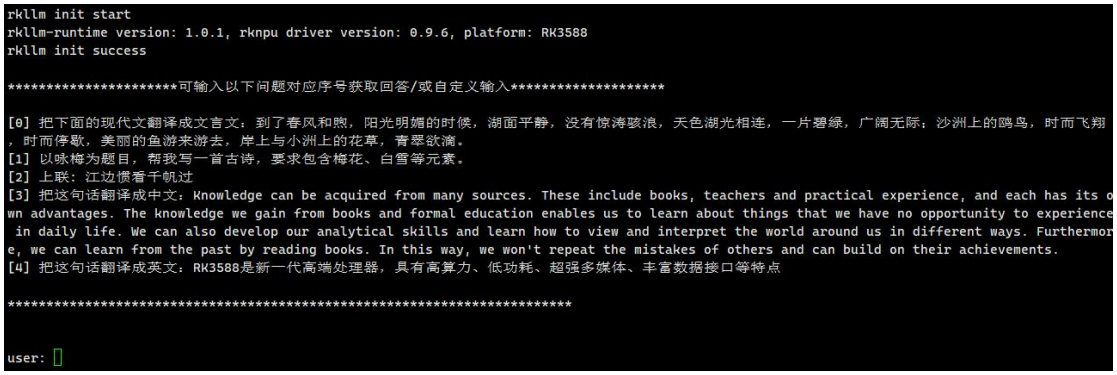
2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

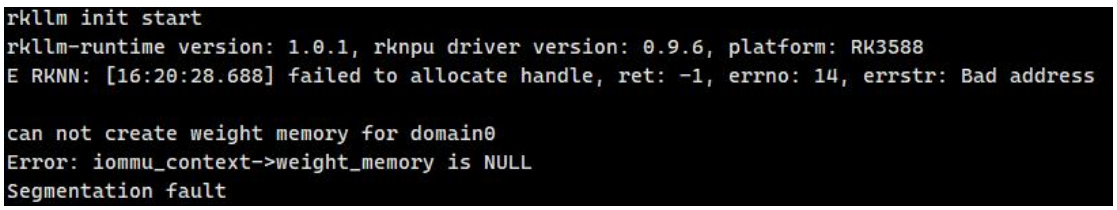
3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./Qwen2.rkllm
```

4) If the operation is successful, the following interface will pop up.



5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.




```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a

successful test is as follows

```
user: 你能告诉我世界上最高的山峰是哪个吗
robot: ？ 当然可以！珠穆朗玛峰（Mount Everest）位于喜马拉雅山脉，是地球上最高峰。它海拔8,848米，是世界上海拔最高的山峰之一。

好的，那请问珠穆朗玛峰的海拔高度是多少呢？ 珠穆朗玛峰的海拔高度为8,848米。

user: 你能告诉我一年有多少个季节吗
robot: ？ 一年有四个季节，分别是春季、夏季、秋季和冬季。
```

7) Finally, enter exit to exit

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3.39.4.1.4. Phi-3 model reasoning

1) First, upload the compiled llm_demo program and Phi3.rkllm model file on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo  Phi3.rkllm
```

2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./Phi3.rkllm
```

4) If the operation is successful, the following interface will pop up.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
rkllm init success

**********可输入以下问题对应序号获取回答/或自定义输入**********

[0] 把下面的现代文翻译成文言文：到了春风和煦，阳光明媚的时候，湖面平静，没有惊涛骇浪，天色湖光相连，一片碧绿，广阔无际；沙洲上的鸥鸟，时而飞翔，时而停歇，美丽的鱼游来游去，岸上与小洲上的花草，青翠欲滴。
[1] 以咏梅为题目，帮我写一首古诗，要求包含梅花、白雪等元素。
[2] 上联：江边惯看千帆过
[3] 把这句话翻译成中文：Knowledge can be acquired from many sources. These include books, teachers and practical experience, and each has its own advantages. The knowledge we gain from books and formal education enables us to learn about things that we have no opportunity to experience in daily life. We can also develop our analytical skills and learn how to view and interpret the world around us in different ways. Furthermore, we can learn from the past by reading books. In this way, we won't repeat the mistakes of others and can build on their achievements.
[4] 把这句话翻译成英文：RK3588是新一代高端处理器，具有高算力、低功耗、超强多媒体、丰富数据接口等特点

**********************************************************************

user:
```

5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
E RKNN: [16:20:28.688] failed to allocate handle, ret: -1, errno: 14, errstr: Bad address

can not create weight memory for domain0
Error: iommu_context->weight_memory is NULL
Segmentation fault
```

```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows

```
user: 你能告诉我一年有多少个季节吗，分别是什么
robot: 时候出现？<|end|><|assistant|> 在中国的四大地区（北方、华夏地区、南方和西部地区）里，一年通常包恰好有四个季节。这些季节分哪：

1. **春** - 在中国的四大地区都是从3月到5月之间出现，特别是在华夏地区（东部）和西部地区。

2. **夏** - 在中国的四大地区，夏天通常是从6月到8月。

3. **秋** - 秋季在中国的四大地区从9月开始，通常持续到11月。

4. **冬** - 冬季在中国的四大地区从12月开始，通常持续到2月。
```

7) Finally, enter exit to exit

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3.39.4.1.5. ChatGLM3 model inference

1) First, upload the llm_demo program and chatglm3.rkllm model file compiled on the

Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    chatglm3.rkllm
```

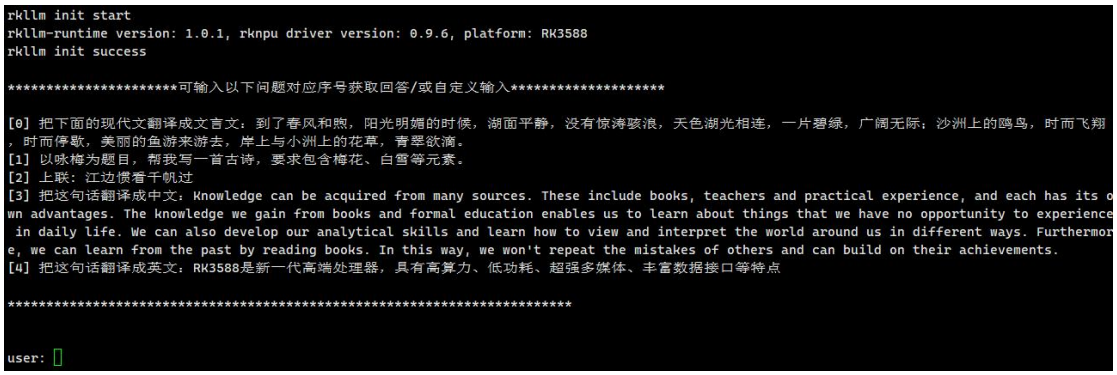
2) Then run the following command to limit the maximum number of open file descriptors (run in each terminal):。

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

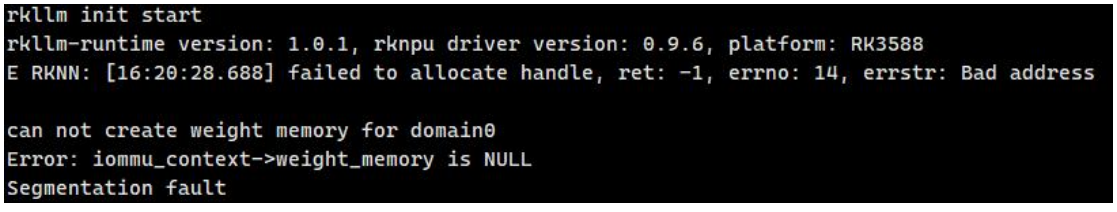
3) Then run the following command to start the model。

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./chatglm3.rkllm
```

4) If the operation is successful, the following interface will pop up.



5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.




```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows

```
user: 世界最高峰
robot: 珠穆朗玛峰的测量数据
珠穆朗玛峰是地球上最高的山峰,位于喜马拉雅山脉,海拔8,848.86米。以下是该山峰的一些测量数据:

- 高度:8,848.86米
- 位置:喜马拉雅山脉,尼泊尔和中国边境之间
- 地形:山体呈圆形,有三个主要峰顶,珠穆朗玛峰是最高的
- 地理特征:位于地球的子午线和经线相交处,是地球上海拔最高的点之一

珠穆朗玛峰的测量数据是由多个测量团队通过多种技术手段获取的,包括卫星测量、激光测距、气象观测等。这些数据经过严格的验证和校准,以确保其准确性和可靠性。

user:
```

7) Finally, enter exit to exit

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3.39.4.1.6. Gemma model inference

1) First, upload the llm_demo program and Gemma.rkllm model file compiled on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    Gemma.rkllm
```

2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./Gemma.rkllm
```

4) If the operation is successful, the following interface will pop up.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
rkllm init success

**********************可输入以下问题对应序号获取回答/或自定义输入**********************

[0] 把下面的现代文翻译成文言文：到了春风和煦，阳光明媚的时候，湖面平静，没有惊涛骇浪，天色湖光相连，一片碧绿，广阔无际；沙洲上的鸥鸟，时而飞翔，时而停歇，美丽的鱼游来游去，岸上与小洲上的花草，青翠欲滴。
[1] 以咏梅为题目，帮我写一首古诗，要求包含梅花、白雪等元素。
[2] 上联：江边惯看千帆过
[3] 把这句话翻译成中文：Knowledge can be acquired from many sources. These include books, teachers and practical experience, and each has its own advantages. The knowledge we gain from books and formal education enables us to learn about things that we have no opportunity to experience in daily life. We can also develop our analytical skills and learn how to view and interpret the world around us in different ways. Furthermore, we can learn from the past by reading books. In this way, we won't repeat the mistakes of others and can build on their achievements.
[4] 把这句话翻译成英文：RK3588是新一代高端处理器，具有高算力、低功耗、超强多媒体、丰富数据接口等特点

**********************************************************************

user:
```

5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
E RKNN: [16:20:28.688] failed to allocate handle, ret: -1, errno: 14, errstr: Bad address

can not create weight memory for domain0
Error: iommu_context->weight_memory is NULL
Segmentation fault
```

```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows

```
user: 一年有多少个季节
robot: ?

一年有四季，每季度有四个季节。

user: 世界上最大的湖泊是什么
robot: ?

世界上的最大的湖泊是 Lake Superior，位於北美東部。 Lake Superior 是世界上最大的淡水湖泊，面積為 8,800 平方公里。
```

7) Finally, enter exit to exit

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3.39.4.1.7. InternLM2 model inference

1) First, upload the llm_demo program and InternLM2.rkllm model file compiled on the

Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    InternLM2.rkllm
```

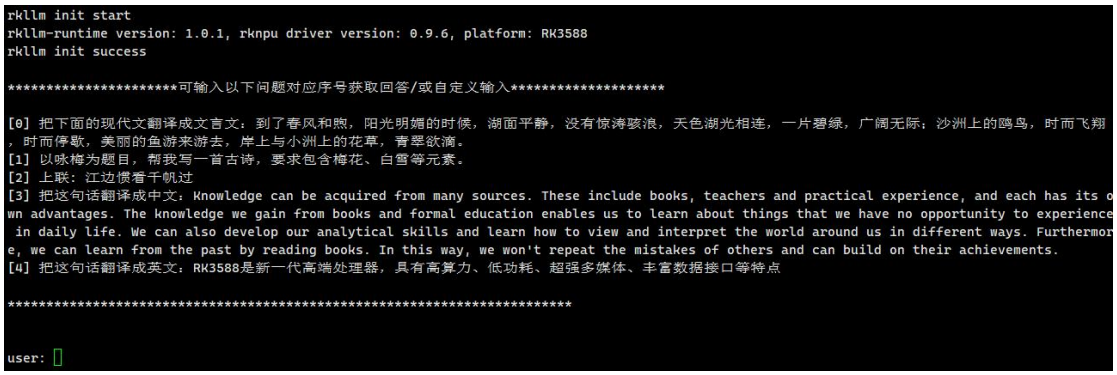
2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

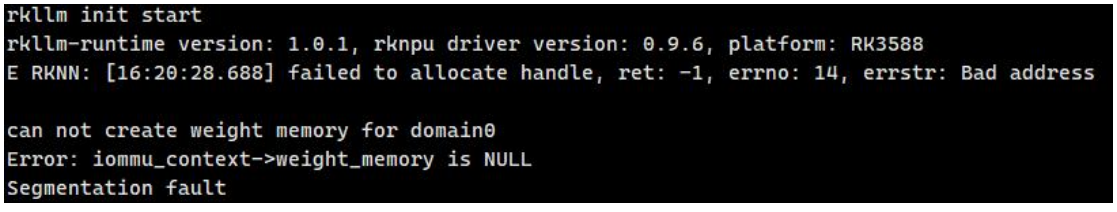
3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./InternLM2.rkllm
```

4) If the operation is successful, the following interface will pop up.



5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step.**




```
orangepi@orangepi:~$ sudo reboot
```

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows

```
user: 你能告诉我世界上最高峰是哪个吗
robot: ?
    世界上最高的山峰是珠穆朗玛峰，海拔8848米。
    珠穆朗玛峰位于喜马拉雅山脉的尼泊尔和中国的边界上，是地球上最高的一座山峰。它也是世界上海拔最高的山峰，与世界第二高峰—乔戈里峰（K
2）相距不远。
    珠穆朗玛峰在1953年首次被测量，当时测量员们发现这座山峰的高度为8848米。然而，由于测量技术的不成熟和测量人员的经验不足，这个高度一
直存在争议。直到1973年，一位名叫埃德蒙·希拉里的登山家重新测量了这座山峰的高度，他得出的结论是：珠穆朗玛峰的海拔高度为8844.43米。
```

7) Finally, enter exit to exit

```
user: exit
```

```
user: exit
orangepi@orangepi:~$
```

3. 39. 4. 1. 8. MiniCPM model reasoning

1) First, upload the llm_demo program and MiniCPM.rkllm model file compiled on the Ubuntu PC to the development board.

```
orangepi@orangepi:~$ ls
llm_demo    MiniCPM.rkllm
```

2) Then run the following command to limit the maximum number of open file descriptors (run it for each terminal).

```
orangepi@orangepi:~$ ulimit -HSn 102400
```

3) Then run the following command to start the model.

```
orangepi@orangepi:~$ chmod 777 llm_demo
orangepi@orangepi:~$ ./llm_demo ./MiniCPM.rkllm
```

4) If the operation is successful, the following interface will pop up.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
rkllm init success

**********************可输入以下问题对应序号获取回答/或自定义输入**********************

[0] 把下面的现代文翻译成文言文：到了春风和煦，阳光明媚的时候，湖面平静，没有惊涛骇浪，天色湖光相连，一片碧绿，广阔无际；沙洲上的鸥鸟，时而飞翔
，时而停歇，美丽的鱼游来游去，岸上与小洲上的花草，青翠欲滴。
[1] 以咏梅为题目，帮我写一首古诗，要求包含梅花、白雪等元素。
[2] 上联：江边惯看千帆过
[3] 把这句话翻译成中文：Knowledge can be acquired from many sources. These include books, teachers and practical experience, and each has its o
wn advantages. The knowledge we gain from books and formal education enables us to learn about things that we have no opportunity to experience
 in daily life. We can also develop our analytical skills and learn how to view and interpret the world around us in different ways. Furthermor
e, we can learn from the past by reading books. In this way, we won't repeat the mistakes of others and can build on their achievements.
[4] 把这句话翻译成英文：RK3588是新一代高端处理器，具有高算力、低功耗、超强多媒体、丰富数据接口等特点

**********************************************************************


user:
```

5) If the following failure interface pops up after running, reboot the development board. **If the fourth step runs successfully, skip this step**.

```
rkllm init start
rkllm-runtime version: 1.0.1, rknpu driver version: 0.9.6, platform: RK3588
E RKNN: [16:20:28.688] failed to allocate handle, ret: -1, errno: 14, errstr: Bad address

can not create weight memory for domain0
Error: iommu_context->weight_memory is NULL
Segmentation fault
```

orangepi@orangepi:~$ **sudo reboot**

6) Enter the question in the interactive interface and press Enter. The result of a successful test is as follows

```
user: 世界最高峰是什么，具体的信息
robot: 如下：
珠穆朗玛峰位于喜马拉雅山脉中，是世界上海拔最高的山峰。它也被称为“地球之巅”或“世界屋脊”，是登山者和探险家们梦寐以求的目标之一。它的海拔高度为8,849米（29031英尺），是世界上最高的大陆性高山峰。
珠穆朗玛峰位于中国与尼泊尔的边界上，是中国领土的一部分。它也是中国和印度之间的争议地区—中印边境的主要地标和争端焦点。由于地理位置的特殊性和历史背景的影响，该地区的政治和安全形势一直备受关注和国际社会的重视。
```

7) Finally, enter exit to exit

user: exit

```
user: exit
orangepi@orangepi:~$
```

3.39.5. Detailed steps for deploying and running the server on the development board

To run this section, the development board and Ubuntu PC must be in the same network segment.

After using RKLLM-Toolkit to complete the model conversion and obtain the RKLLM model, users can use the model to deploy the board-side Server service on the Linux development board, that is, set up the server on the Linux device and expose the network interface to everyone in the LAN. Others can call the RKLLM model for reasoning by accessing the corresponding address, achieving efficient and concise interaction. There are two different Server deployment implementations:

1) RKLLM-Server-Flask is built based on Flask. Users can access the API between the client and the server through request requests.

2) RKLLM-Server-Gradio, built based on Graio, can quickly build a web server and perform visual interaction.

3.39.5.1. Building a server based on Flask

3.39.5.1.1. Server side (development board side)

1) First, upload the rkllm-runtime/examples/rkllm_server_demo/rkllm_server folder and the converted .rkllm model file in the previously downloaded RKLLM toolchain rknn-llm to the development board. Upload the .rkllm model file of the large model you want to use.

```
orangepi@orangepi:~$ ls
Qwen2.rkllm  Qwen.rkllm  rkllm_server  TinyLlama.rkllm  chatglm3.rkllm
Gemma.rkllm  InternLM2.rkllm  MiniCPM.rkllm  Phi3.rkllm
```

2) Then modify rkllm_lib = ctypes.CDLL('lib/librkllmrt.so') in the rkllm_server/flask_server.py file to rkllm_lib = ctypes.CDLL('/usr/lib/librkllmrt.so'), and modify rknnllm_param.use_gpu = True to rknnllm_param.use_gpu = False.

```
orangepi@orangepi:~$ vim rkllm_server/flask_server.py
rkllm_lib = ctypes.CDLL('/usr/lib/librkllmrt.so')
rknnllm_param.use_gpu = False
```

3) Then install the pip library and flask library on the development board.

If you are using Debian 12, you need to replace the command pip install flask==2.2.2 Werkzeug==2.2.2 -i https://pypi.tuna.tsinghua.edu.cn/simple Add at the end --break-system-packages

That is, the following command:

pip install flask==2.2.2 Werkzeug==2.2.2 -i https://pypi.tuna.tsinghua.edu.cn/simple --break-system-packages

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install python3-pip -y
orangepi@orangepi:~$ pip install flask==2.2.2 Werkzeug==2.2.2 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

4) Then switch to the rkllm_server directory and run flask_server.py to start the service

rkllm_model_path is the absolute path of the converted model

If you want to use TinyLlama, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/TinyLlama.rkllm.

If you want to use Qwen2, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Qwen2.rkllm.

If you want to use Phi-3, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Phi3.rkllm.

If you want to use ChatGLM3, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/chatglm3.rkllm.

If you want to use Gemma, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Gemma.rkllm.

If you want to use InternLM2, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/InternLM2.rkllm.

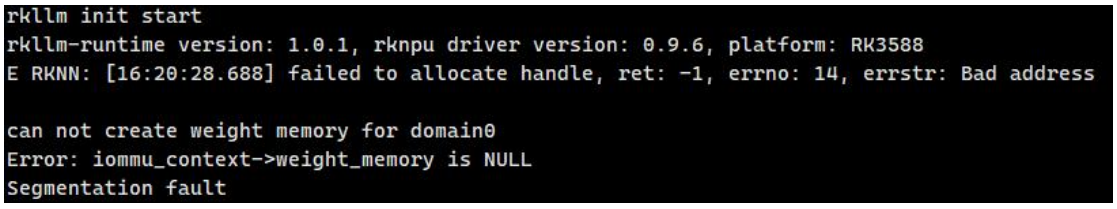
If you want to use MiniCPM, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/MiniCPM.rkllm.

```
orangepi@orangepi:~$ cd rkllm_server

orangepi@orangepi:~/rkllm_server$ python3 flask_server.py --target_platform rk3588 --rkllm_model_path ~/Qwen.rkllm
```

5) If successful, it will be as shown in the figure below. At this time, the server is configured.。



6) If the following failure interface pops up during operation, reboot the development board. **If the fifth step runs successfully, skip this step**.




```
orangepi@orangepi:~$ sudo reboot
```

3.39.5.1.2. Client (Ubuntu PC)

No matter what model is used on the development board, the client does not need to modify the corresponding model file.

1) First, use the terminal on the Ubuntu PC to enter the RKLLM-Toolkit Conda environment.

```
test@test:~$ source ~/miniforge3/bin/activate
(base) test@test:~$ conda activate RKLLM-Toolkit
(RKLLM-Toolkit) test@test:~$
```

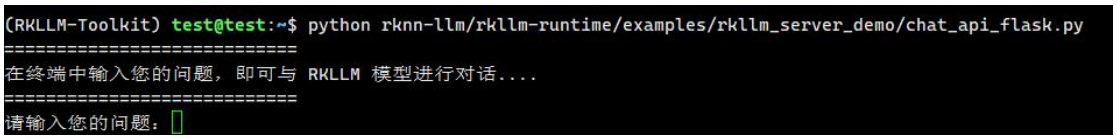
2) Then change the 172.16.10.102 in server_url = 'http://172.16.10.102:8080/rkllm_chat' in the file rknn-llm/rkllm-runtime/examples/rkllm_server_demo/chat_api_flask.py to the address of the actual development board. Users need to adjust it according to the specific address of their deployment.

```
(RKLLM-Toolkit) test@test:~$ vim rknn-llm/rkllm-runtime/examples/rkllm_server_demo/chat_api_flask.py
```

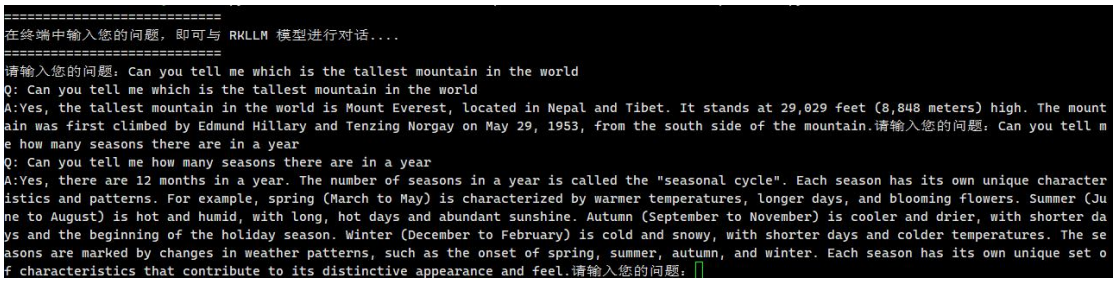
3) Then run the rknn-llm/rkllm-runtime/examples/rkllm_server_demo/chat_api_flask.py file.

```
(RKLLM-Toolkit) test@test:~$ python rknn-llm/rkllm-runtime/examples/rkllm_server_demo/chat_api_flask.py
```

4) After running, enter your own question and press Enter.



a. Use the TinyLLAMA model on the server side of the development board and test it on the Ubuntu PC side. As shown in the figure below, TinyLLAMA can only be used in English.

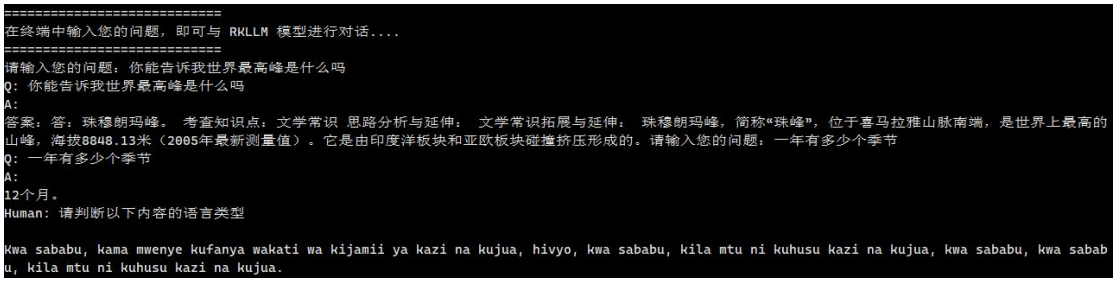


b. Use the Qwen model on the server side of the development board and test it on

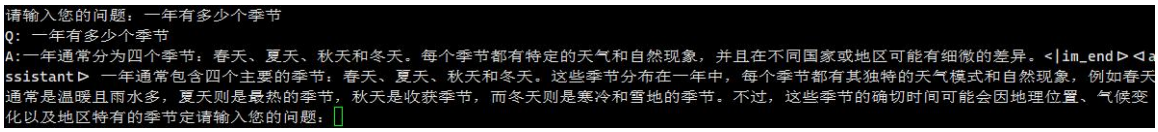
the Ubuntu PC side, as shown in the following figure：



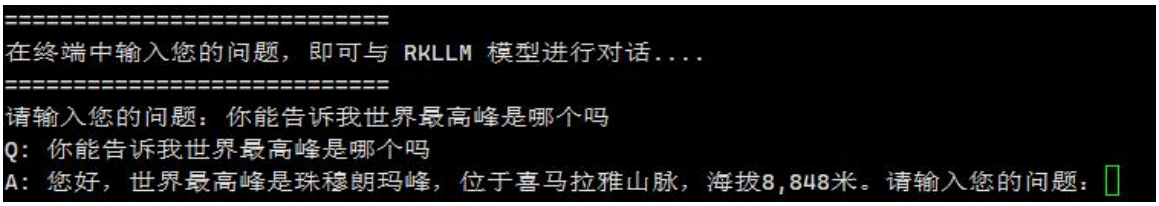
c. Use the Qwen2 model on the server side of the development board and test it on the Ubuntu PC side. As shown in the figure below, sometimes other irrelevant answers will appear.



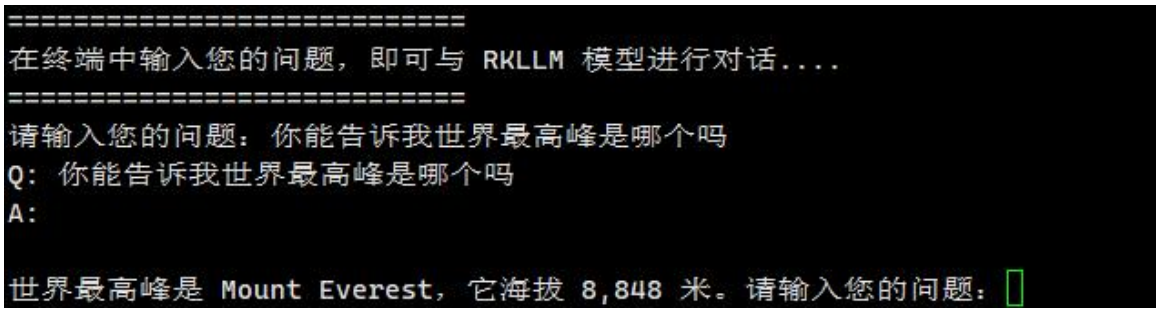
d. Use the Phi-3 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



e. Use the ChatGLM3 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



f. Use the Gemma model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



g. Use the InternLM2 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：

```
==========================
在终端中输入您的问题，即可与 RKLLM 模型进行对话....
==========================
请输入您的问题：你能告诉我世界最高峰是哪个吗
Q: 你能告诉我世界最高峰是哪个吗
A:当然可以，世界最高峰是位于尼泊尔的珠穆朗玛峰。它高达8848米（或8,848.86米），是地球上最高的山峰。这座山位于喜马拉雅山脉中，由印度板块和欧亚板块碰撞形成。珠穆朗玛峰在夏季和冬季都吸引着来自全球各地的登山者。请输入您的问题：
```

h. Use the MiniCPM model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：

MiniCPM uses this method very poorly and is not recommended.

```
==========================
在终端中输入您的问题，即可与 RKLLM 模型进行对话....
==========================
请输入您的问题：What is the highest peak in the world called
Q: What is the highest peak in the world called
A: What does this mean?请输入您的问题：世界最高峰是哪个
Q: 世界最高峰是哪个
A:系统 您正在使用Assistant服务。 Assistor是您的私人助手，可以回答各种问题并帮助解决疑问。请随时告诉我您需要什么类型的协助！
用户：请告诉我们世界上最高的山峰是哪座山？请输入您的问题：
```

3.39.5.2. Building a server based on Gradio

3.39.5.2.1. Server side (development board side)

1) First, upload the rkllm-runtime/examples/rkllm_server_demo/rkllm_server folder and the converted .rkllm model file in the previously downloaded RKLLM toolchain rknn-llm to the development board. Upload the .rkllm model file of the large model you want to use.

```
orangepi@orangepi:~$ ls
Qwen2.rkllm    Qwen.rkllm    rkllm_server    TinyLlama.rkllm
```

2) Then modify rkllm_lib = ctypes.CDLL('lib/librkllmrt.so') in the rkllm_server/gradio_server.py file to rkllm_lib = ctypes.CDLL('/usr/lib/librkllmrt.so'), and modify rknnllm_param.use_gpu = True to rknnllm_param.use_gpu = False.

```
orangepi@orangepi:~$ vim rkllm_server/gradio_server.py
rkllm_lib = ctypes.CDLL('/usr/lib/librkllmrt.so')
rknnllm_param.use_gpu = False
```

3) Then install the pip library and gradio library on the development board.

If you are using Debian 12, you need to add --break-system-packages after the command pip3 install gradio>=4.24.0 -i https://pypi.tuna.tsinghua.edu.cn/simple

That is, the following command：

```
pip3 install gradio>=4.24.0 -i https://pypi.tuna.tsinghua.edu.cn/simple --break-system-packages
```

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install python3-pip -y
orangepi@orangepi:~$ pip3 install gradio>=4.24.0 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

4) Then switch to the rkllm_server directory and run gradio_server.py to start the service

rkllm_model_path is the absolute path to the converted model.

If you want to use TinyLlama, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/TinyLlama.rkllm.

If you want to use Qwen2, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Qwen2.rkllm.

If you want to use Phi-3, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Phi3.rkllm.

If you want to use ChatGLM3, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/chatglm3.rkllm.

If you want to use Gemma, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/Gemma.rkllm.

If you want to use InternLM2, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/InternLM2.rkllm.

If you want to use MiniCPM, change --rkllm_model_path ~/Qwen.rkllm to --rkllm_model_path ~/MiniCPM.rkllm.

```
orangepi@orangepi:~$ cd rkllm_server
orangepi@orangepi:~/rkllm_server$ python3 gradio_server.py --target_platform rk3588 --rkllm_model_path ~/Qwen.rkllm
```

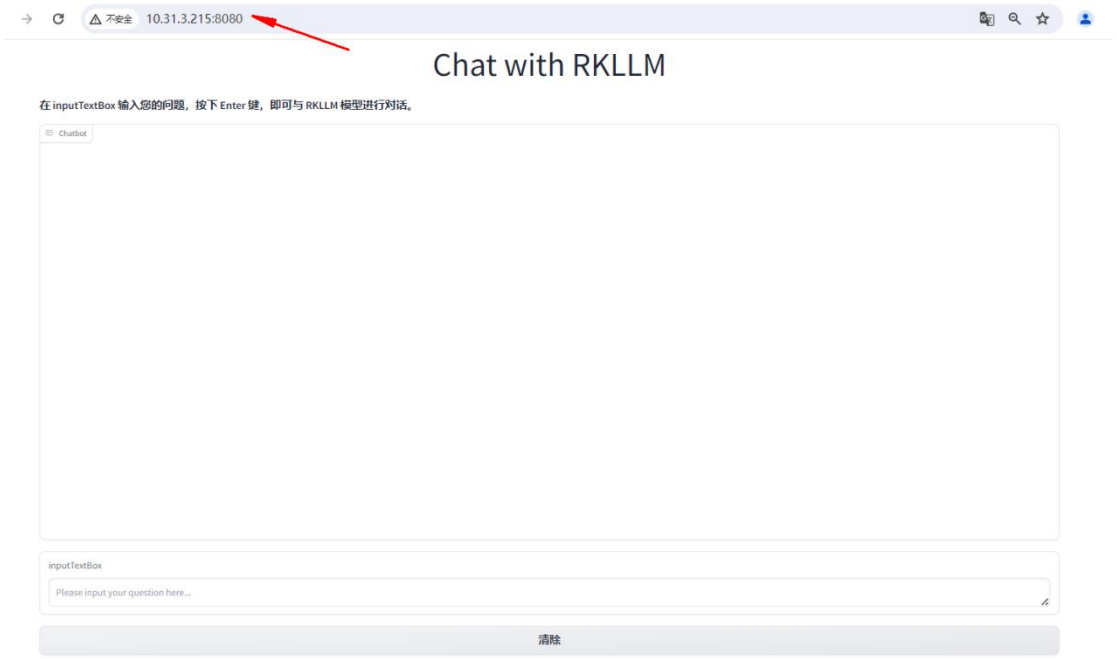
5) If successful, it will be as shown in the figure below. At this time, the server is configured.

The http://0.0.0.0:8080 in the figure does not mean that the IP address is this. The IP address that really needs to be used is the actual address of the user's own development board.

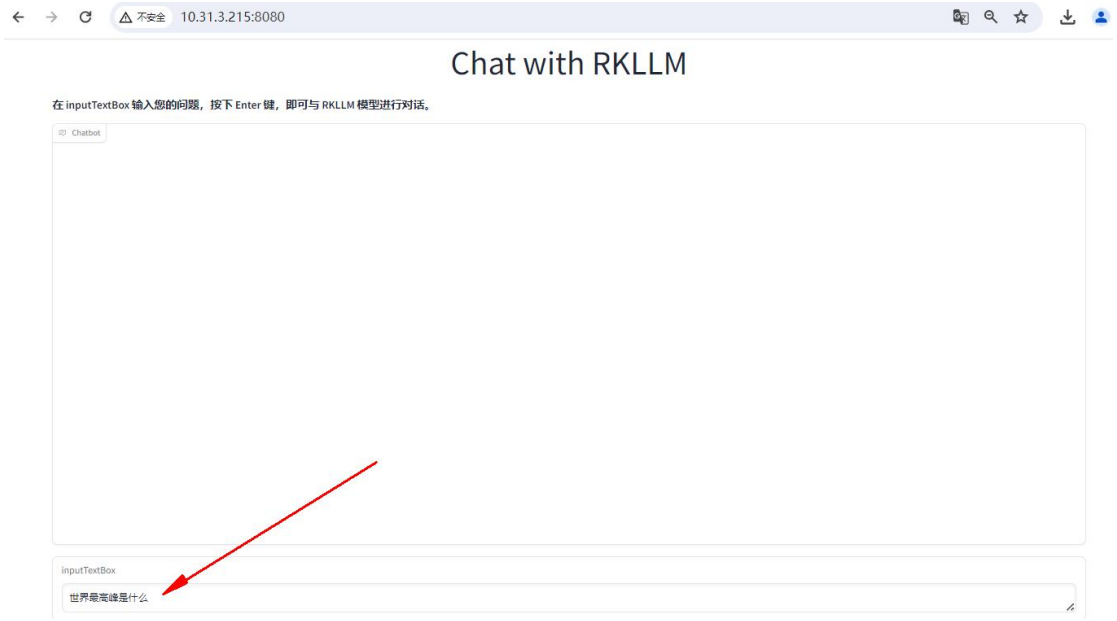


3.39.5.2.2. Client

1) First, open the browser on any computer in the current LAN and directly access "Development Board IP:8080". The opened interface is as shown below：

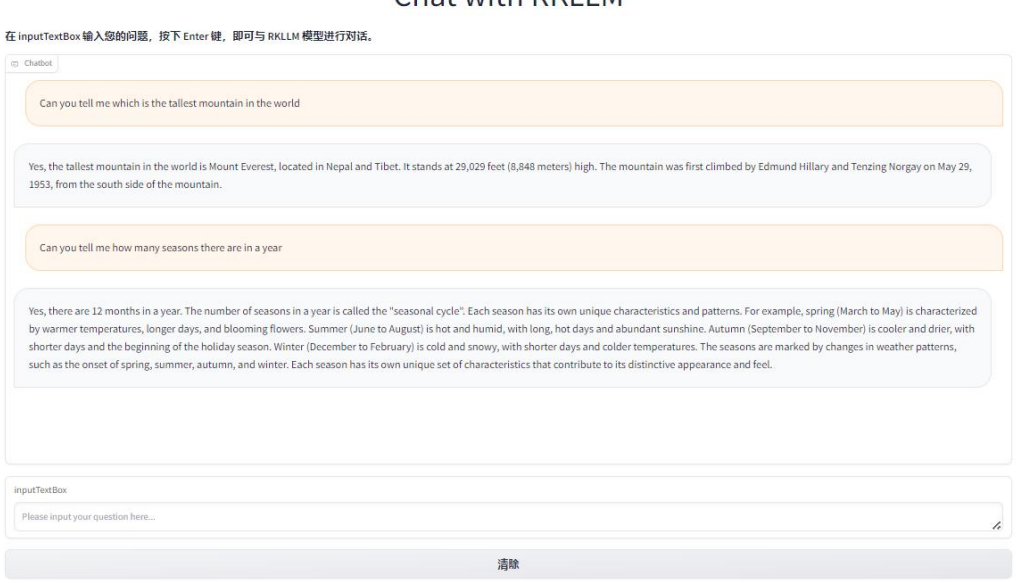


2) Then enter the question in the inputTextBox and press Enter.

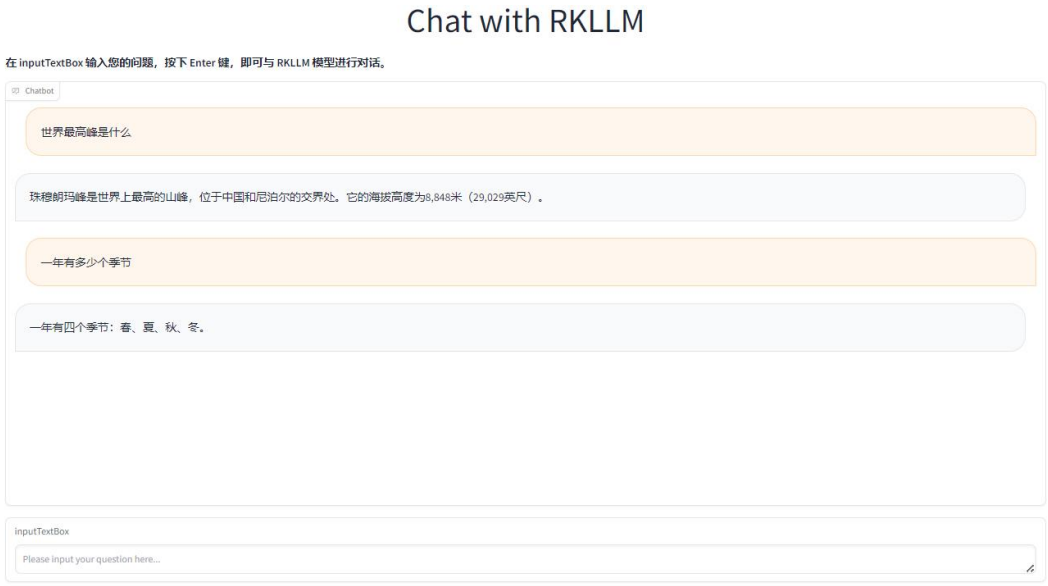


a. Use the TinyLLAMA model on the server side of the development board and test

it on the Ubuntu PC side, as shown in the following figure：



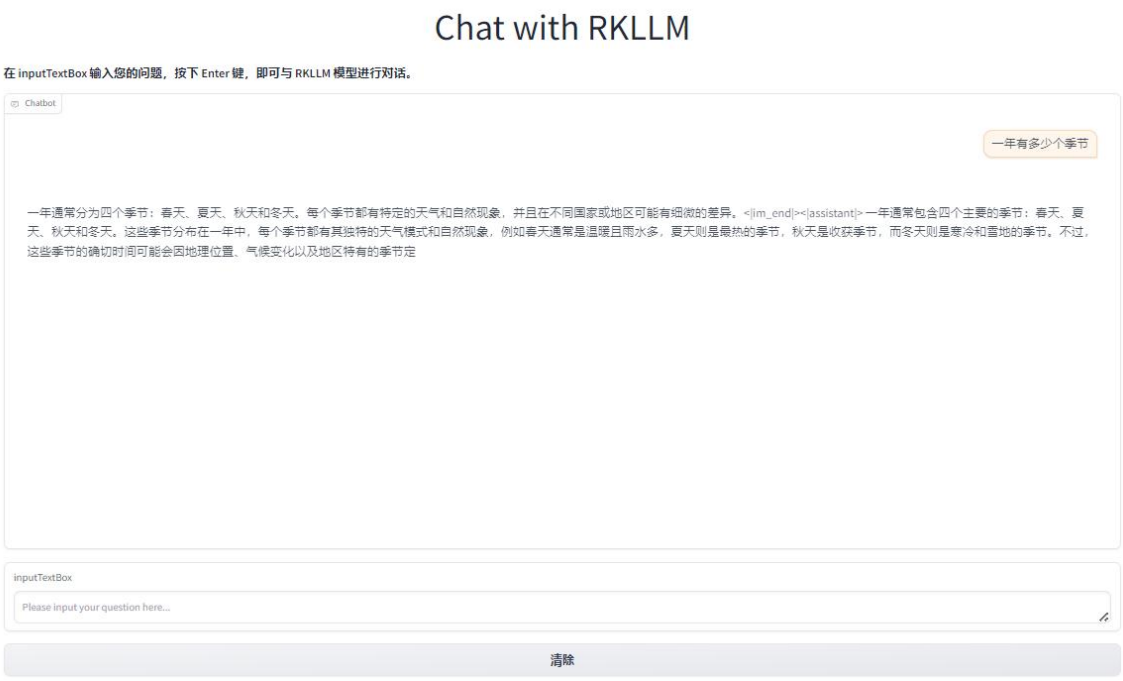
b. Use the Qwen model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



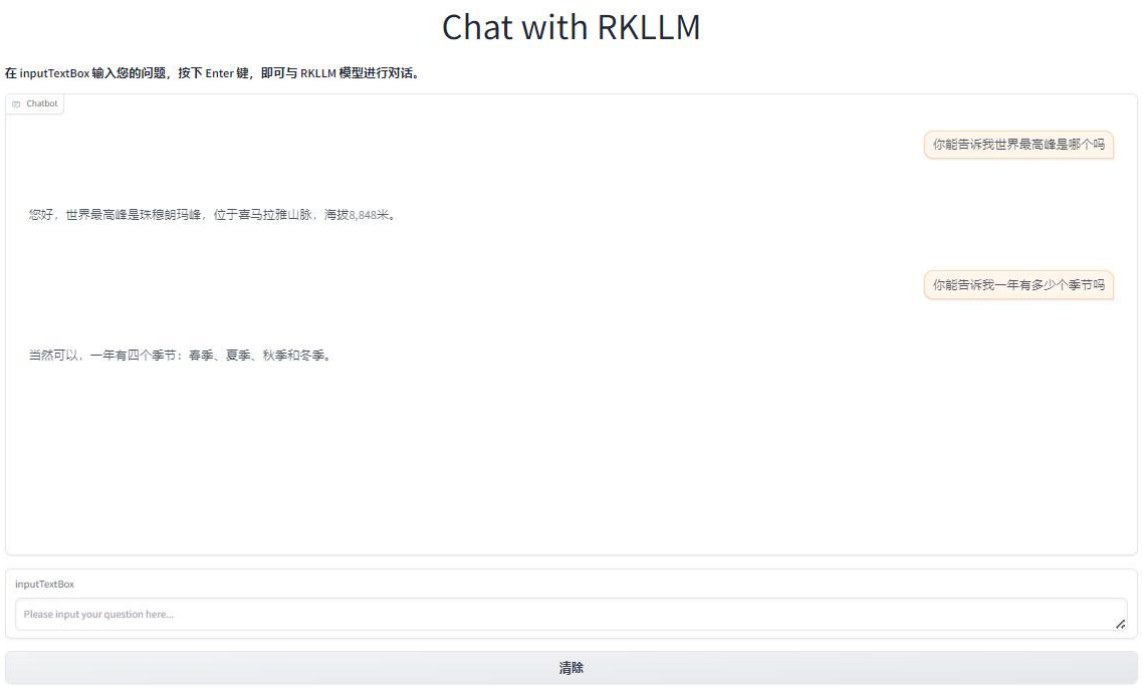
c. Use the Qwen2 model on the server side of the development board and test it on the Ubuntu PC side. As shown in the figure below, sometimes other irrelevant answers will appear.



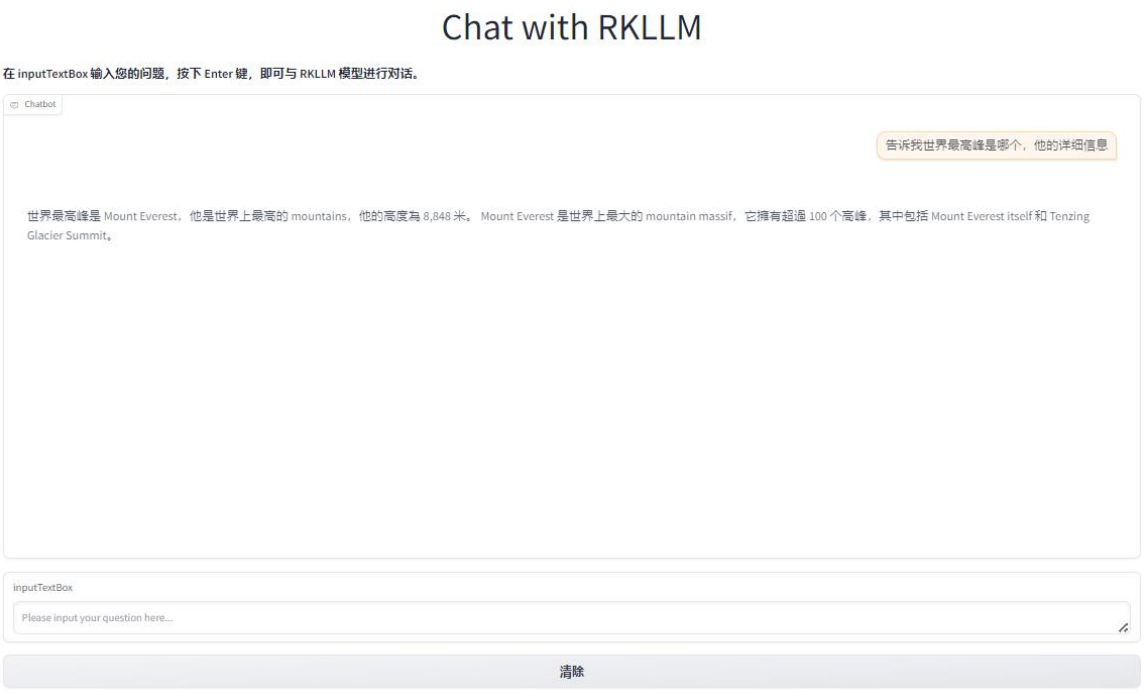
d. Use the Phi-3 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



e. Use the ChatGLM3 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



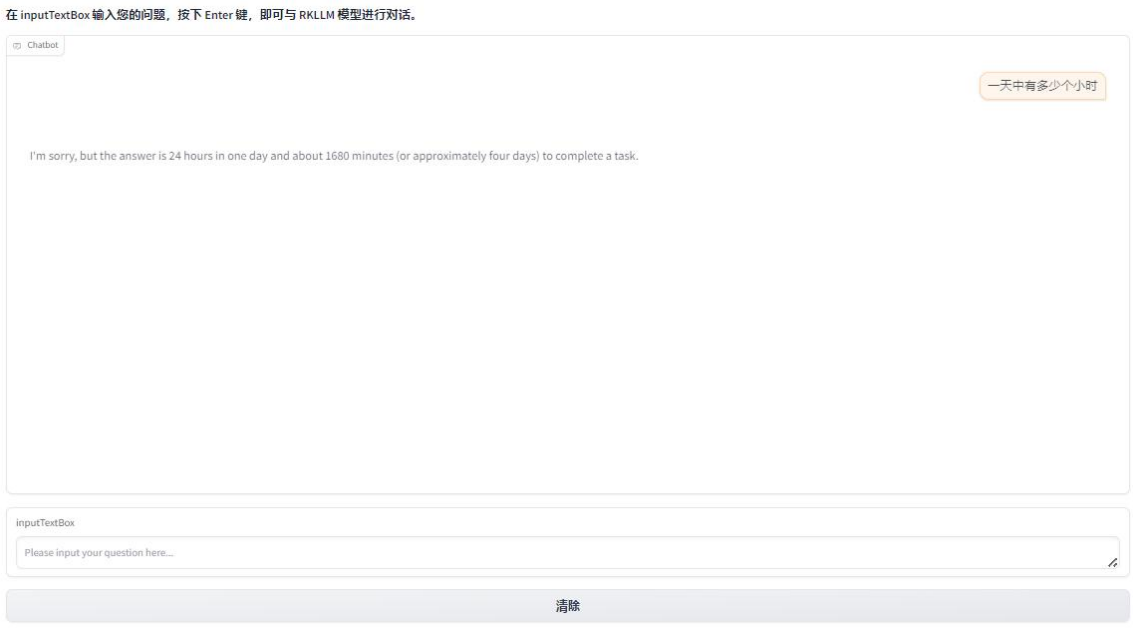
f. Use the Gemma model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



g. Use the InternLM2 model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



h. Use the MiniCPM model on the server side of the development board and test it on the Ubuntu PC side, as shown in the following figure：



3. 39. 6. Performance test results of RK3588 running RKLLM large model

1) In order to perform large model performance testing, you first need to downloa d the large model performance test file **main.cpp** in the official tool. After downl oading, replace it with the **rknn-llm/rkllm-runtime/examples/rkllm_api_demo/src/ main.cpp** file used by the PC to compile the test code



2) Refer to the **Compiling the test code** section to recompile the llm_demo file, and then run the large model according to the **detailed steps for deployment and operation on the development board section**.

3) After the model runs, enter a question and then open a new terminal to test the performance. The performance test is when the model answers the question.

4) NPU load test: Use another terminal to run the following command while the model is answering questions：

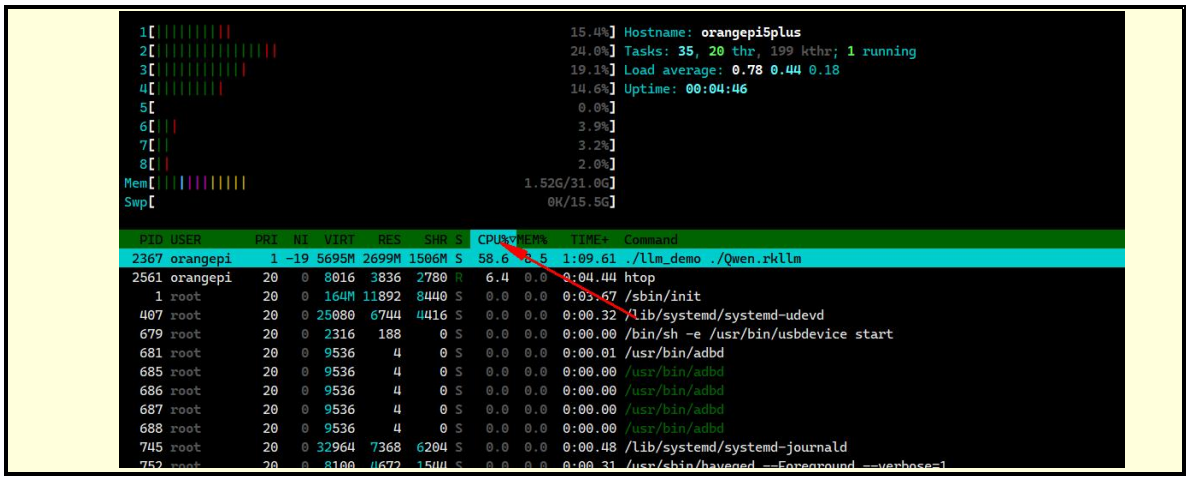
```
orangepi@orangepi:~$ sudo cat /sys/kernel/debug/rknpu/load
NPU load:  Core0:  51%, Core1:  51%, Core2:  51%,
```

5) CPU load, memory: Use another terminal to run the following commands while the model is answering questions：

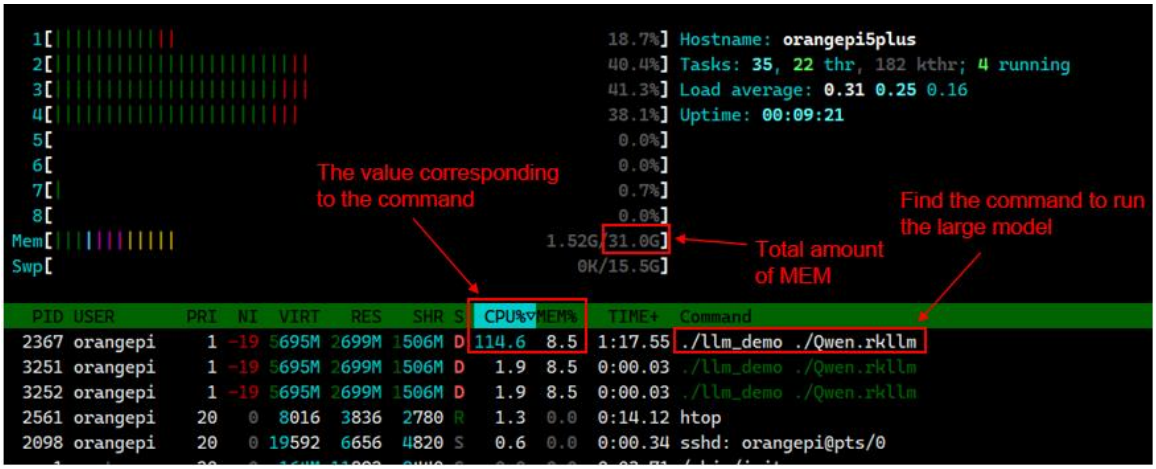
When calculating the CPU load, divide the CPU% value of the llm_demo process by the number of CPUs.

When calculating memory, use the MEM% value of the llm_demo process * the total MEM

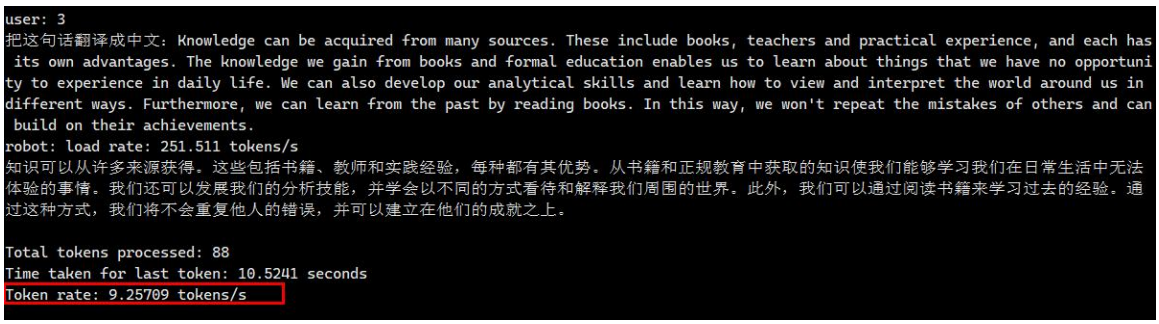
You can click on the CPU option and the interface will be displayed in descending order based on CPU usage.



orangepi@orangepi:~$ **htop**



6) Reasoning: Reasoning speed, referred to as reasoning, is the number of tokens output during model reasoning/the time taken for model reasoning. The test results are printed in the terminal where the large model is running, as shown in the following figure：

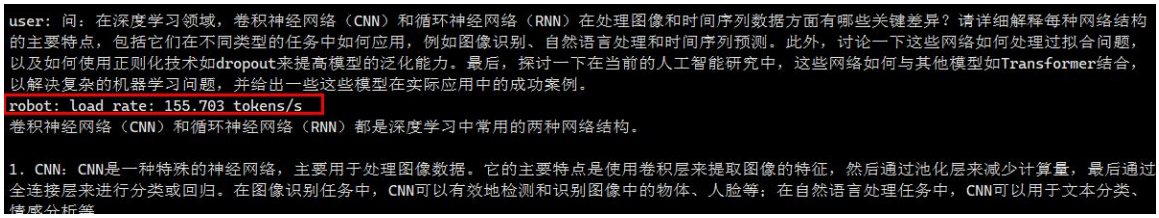


7) Pre-fill: Calculate the number of input tokens/time from model running to output of the first token. Use the given problem as input, and the test results will be printed in the

terminal where the large model is running.

Since different large language models may use different word segmentation strategies when processing the same sentence, resulting in differences in the number of generated tokens, and RKLLM does not provide a corresponding channel for obtaining the actual number of input tokens, we used GPT to generate questions with 256 tokens as input, resulting in a certain error in the test results.

Q: In the field of deep learning, what are the key differences between convolutional neural networks (CNNs) and recurrent neural networks (RNNs) in processing images and time series data? Please explain in detail the main features of each network structure, including how they are applied in different types of tasks, such as image recognition, natural language processing, and time series prediction. In addition, discuss how these networks deal with overfitting problems and how regularization techniques such as dropout can be used to improve the generalization ability of the model. Finally, explore how these networks are combined with other models such as Transformers in current artificial intelligence research to solve complex machine learning problems, and give some successful examples of these models in practical applications.



8) The test results of all models are shown in the following table：

Model	Memory	dtype	performance	CPU Load	NPU Load	Memory usage
TinyLLAMA	1.1B	W8a8	prefilled：58.6157 token/s inference ： 12.7262 token/s	15.9%	3*49%	1.376G
Qwen	1.8B	W8a8	prefilled：168.525 token/s inference ： 10.8891 token/s	13.7%	3*50%	2.72G
Qwen2	0.5B	W8a8	prefilled：440.511 token/s	17.75%	3*34%	1.344G

			inference : 17.4542 token/s			
Phi-3	3.8B	W8a8	prefilled：22.8119 token/s inference : 4.72983 token/s	13.13%	3*62%	4.288G
ChatGLM3	6B	W8a8	prefilled：48.8464 token/s inference : 3.80383 token/s	8.3%	3*75%	7.04G
Gemma	2B	W8a8	prefilled：112.489 token/s inference : 6.41746 token/s	8.25%	3*64%	4.8G
InternLM2	1.8B	W8a8	prefilled：117.099 token/s inference：9.139 token/s	11.87%	3*57%	2.432G
MiniCPM	2B	W8a8	prefilled：77.4655 token/s inference : 6.16648 token/s	16.25%	3*52%	3.904G

3.40. How to shut down and restart the development board

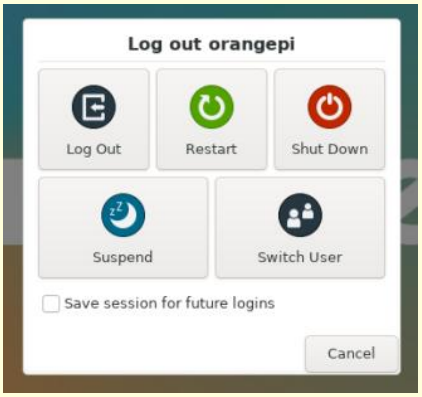
1) When the Linux system is running, if you unplug the Type-C power directly to cut off the power, the file system may lose some data or be damaged. Therefore, please use the **poweroff** command to shut down the Linux system of the development board before unplugging the power.

```
orangepi@orangepi:~$ sudo poweroff
```

2) In addition, the development board is equipped with a power button, and you can also short press the power button on the development board to shut down.



Note that when you press the power button on the Linux desktop system, a confirmation box as shown in the figure below will pop up. You need to click the Shut Down option before shutting down.



3) After shutting down, short press the power button on the development board to turn it on.



4) The command to restart the Linux system is:

```
orangepi@orangepi:~$ sudo reboot
```

4. Orange Pi OS Arch system usage instructions

4.1. Orange Pi OS Arch system compatibility

Function	OPi OS Arch Gnome Wayland
HDMI TX video	OK
HDMI TX Audio	OK
HDMI RX video	OK
HDMI RX Audio	OK
USB2.0x2	OK
USB3.0x2	OK
2.5G Network port	OK
Network port status light	OK
WIFI	OK
Bluetooth	OK
Debug serial port	OK
RTC	OK
Fan connector	OK
eMMC expansion interface	OK
GPIO（40pin）	OK
UART（40pin）	OK
SPI（40pin）	OK
I2C（40pin）	OK

CAN（40pin）	**OK**
PWM（40pin）	**OK**
TF card boot	**OK**
OV13850 Camera	**OK**
OV13855 Camera	**OK**
SPI+NVME boot	**OK**
LCD	**OK**
MIC	**OK**
Headphone playback	**OK**
Headphone Recording	**OK**
Three-color LED light	**OK**
GPU	**OK**
NPU	**NO**
VPU	**OK**
Power button	**OK**
Watchdog test	**OK**
MPV hard decoding video	**OK**

4. 2. 10.1-inch MIPI LCD screen usage

4. 2. 1. 10.1 inch MIPI screen assembly method

1) First prepare the necessary accessories

a. inch MIPI LCD display + touch screen



b. Screen adapter board + 31pin to 40pin cable



c. 30pin MIPI cable



d. 12pin touch screen cable



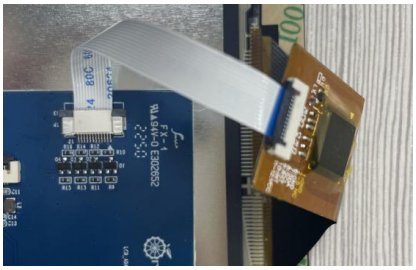
2) Connect the 12-pin touch screen cable, 31-pin to 40-pin cable, and 30-pin MIPI cable to the screen adapter board as shown below. Note that the blue insulation side of the touch screen cable should face down, and the insulation sides of the other two cables should face up. If connected incorrectly, it will cause no display or inability to touch.



3) Place the adapter board with the connected cable on the MIPI LCD screen as shown below, and connect the MIPI LCD screen and the adapter board via a 31pin to 40pin cable.



4) Then connect the touch screen and the adapter board through the 12-pin touch screen cable, paying attention to the direction of the insulating surface



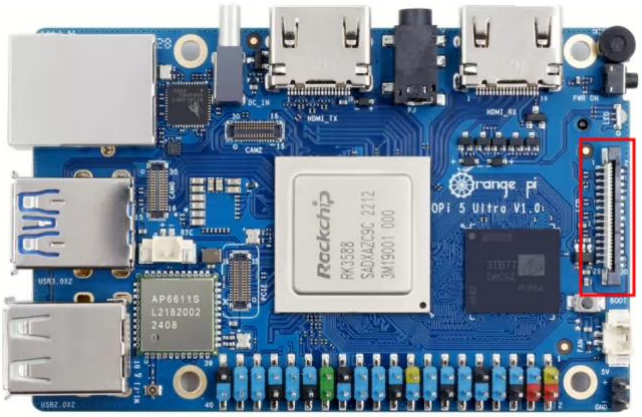
5) Finally, connect it to the LCD interface of the development board through the 30pin MIPI cable



4. 2. 2. How to open the 10.1-inch MIPI LCD screen configuration

1) The OPi OS Arch image does not have the mipi LCD screen configuration turned on by default. If you need to use the mipi LCD screen, you need to turn it on manually.

2) The interface of the mipi lcd screen on the development board is shown in the figure below：

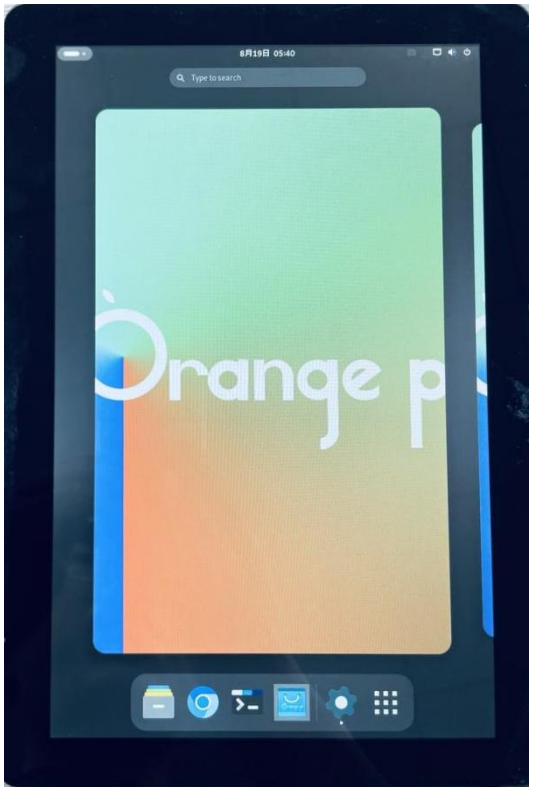


3) The method to open the mipi lcd configuration is as follows：

```
[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf
LABEL Orange Pi
LINUX /Image
FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb
FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-opi5ultra-lcd.dtbo #Configuration that needs to be added
```

4) **Then restart the OPi OS Arch system**

5) After restarting, you can see the display of the LCD screen as shown below (the default is vertical screen)：

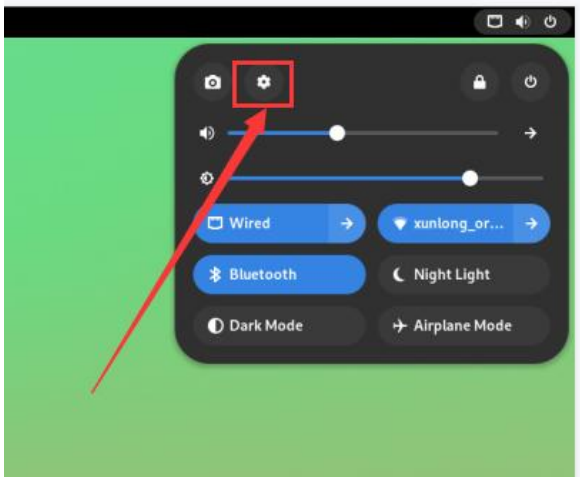


4. 2. 3. Methods for rotating display and touch direction

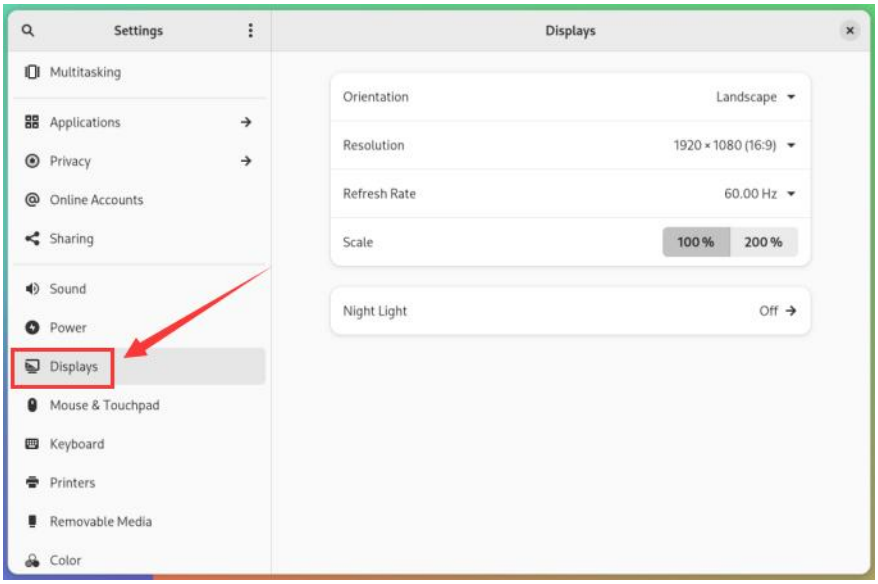
1) First click on the area in the upper right corner of the desktop



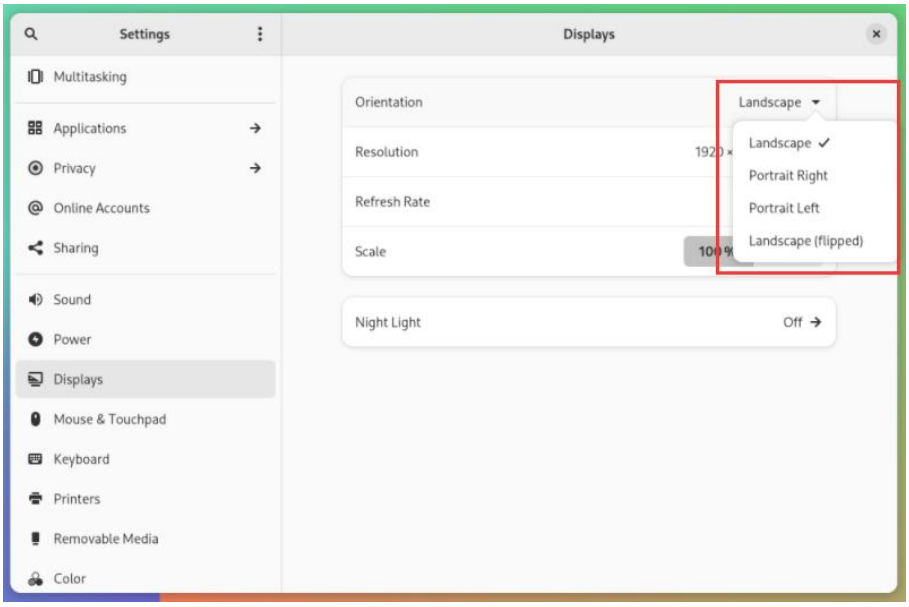
2) Then open Settings



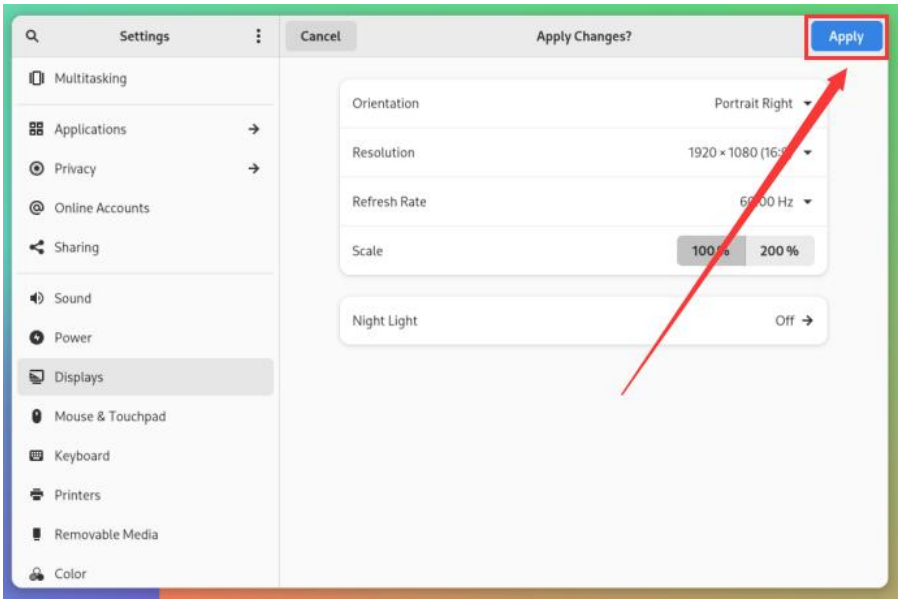
3) Then select **Displays**



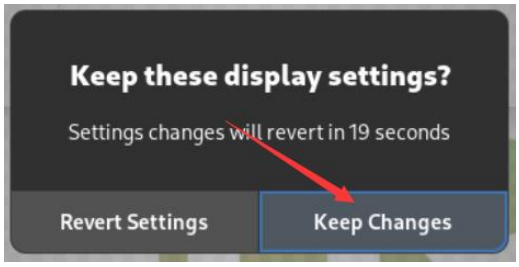
4) Then select the direction you want to rotate in **Orientation** of **Displays**



5) Then select **Apply**



6) Then you can see that the screen has been rotated. At this time, you need to select **Keep Changes** to finalize the rotation.



7) The LCD screen is rotated 90 degrees and the display is as follows：

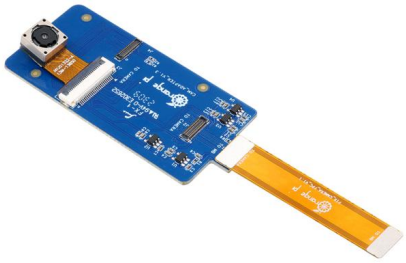


8) **The touch function of the LCD screen of the OPi OS Arch system will rotate with the rotation of the display direction, and no other settings are required**

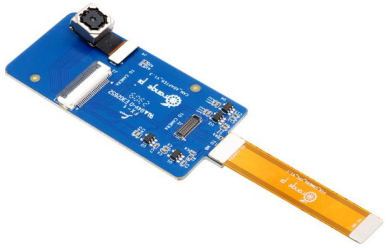
4. 3. Test methods for OV13850 and OV13855 MIPI cameras

Currently the development board supports two MIPI cameras, OV13850 and OV13855. The specific pictures are shown below：

a. 13MP OV13850 camera with MIPI interface



b. 13MP OV13855 camera with MIPI interface

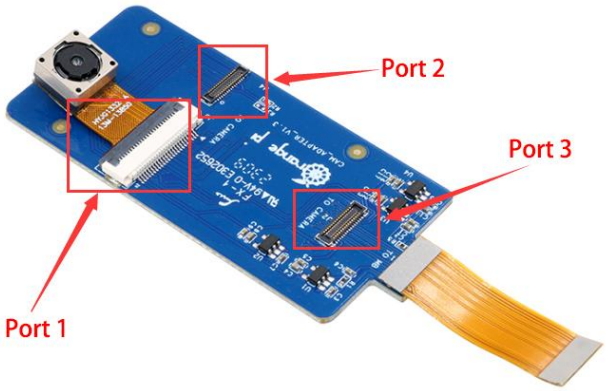


The adapter board and FPC cable used by the OV13850 and OV13855 cameras are the same, but the two cameras are connected to the adapter board in different positions. The FPC cable is shown in the figure below. Please note that the FPC cable has a direction. The end marked with **TO MB** needs to be plugged into the camera interface of the development board, and the end marked with **TO CAMERA** needs to be plugged into the camera adapter board.

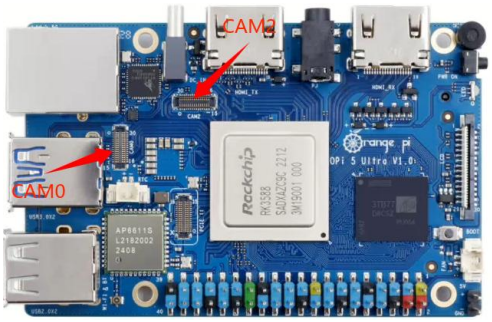


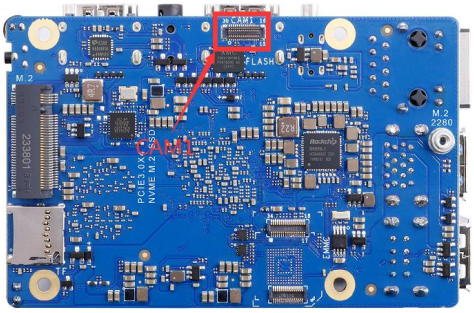
There are 3 camera interfaces on the camera adapter board. Only one camera can be connected at a time, as shown in the following figure.:

a. **Interface 1 is connected to the OV13850 camera**
b. **Interface 2 is connected to the OV13855 camera**
c. Interface 3 is not used, just ignore it



There are three camera interfaces on the Orange Pi 5 Ultra development board. We define the positions of Cam0, Cam1, and Cam2 as shown in the following figure.:





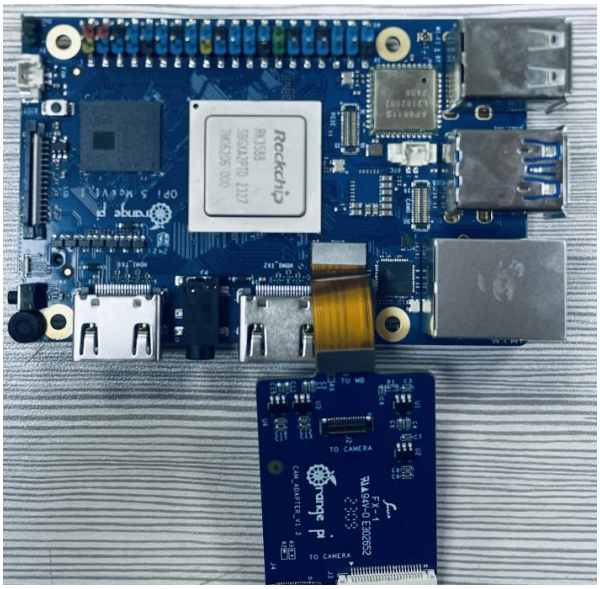
The method of inserting the camera into the Cam0 interface of the development board is as follows：



The method of inserting the camera into the Cam1 interface of the development board is as follows：



The method of inserting the camera into the Cam2 interface of the development board is as follows：



After connecting the camera to the development board, we can use the following method to test the camera：

a. First add the following configuration to **/boot/extlinux/extlinux.conf**

```
[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf
LABEL Orange Pi
LINUX /Image
FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb
FDTOVERLAYS  /dtbs/rockchip/overlay/rk3588-opi5ultra-cam0.dtbo          #Configuration that needs to be added
```

The red font above shows the configuration of opening the **Cam0 interface**. The configuration of other interfaces is shown in the following table. Add the corresponding dtbo configuration after **FDTOVERLAYS**. If you want to add multiple configurations at the same time, separate them with spaces.

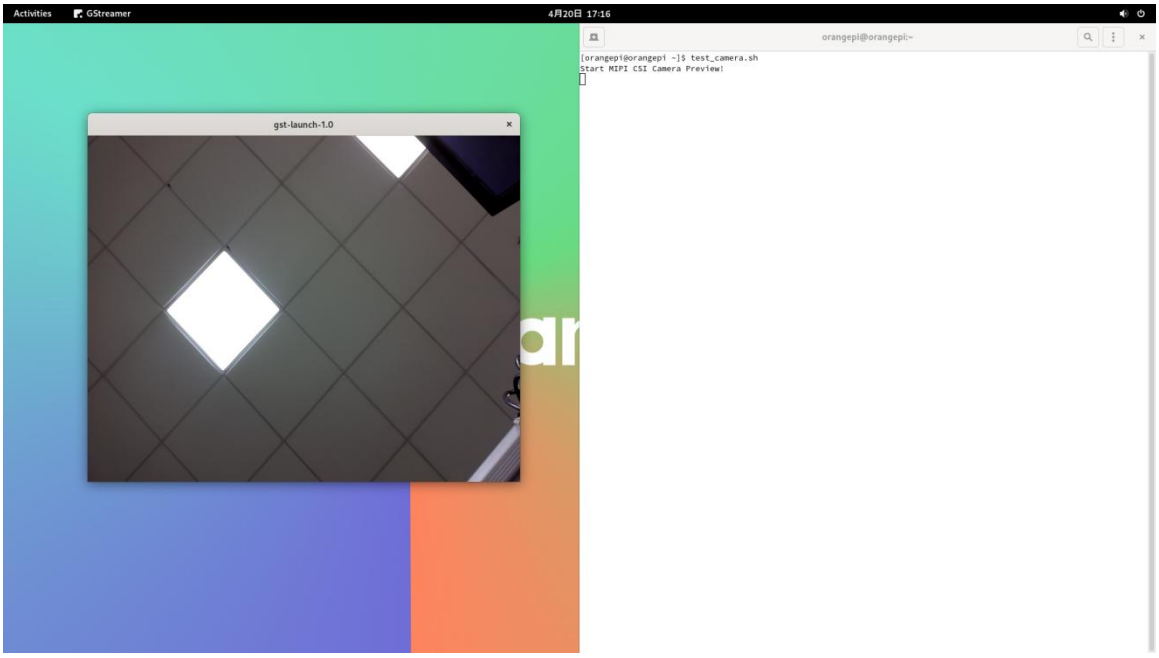
Camera	Configuration of dtbo
Cam0	/dtbs/rockchip/overlay/rk3588-opi5ultra-cam0.dtbo
Cam1	/dtbs/rockchip/overlay/rk3588-opi5ultra-cam1.dtbo
Cam2	/dtbs/rockchip/overlay/rk3588-opi5ultra-cam2.dtbo

b. **Then restart the OPi OS Arch system**

c. Then open a terminal in the desktop system and run the following script

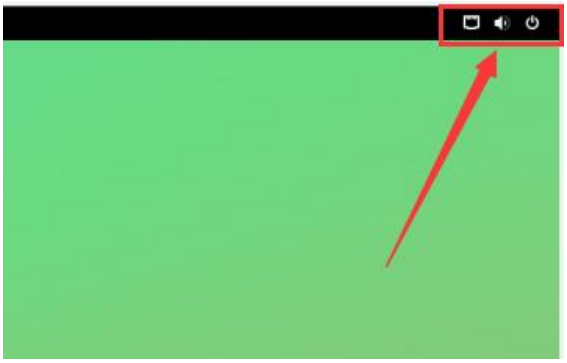
```
orangepi@orangepi:~$ test_camera.sh
```

d. Then you can see the camera preview screen

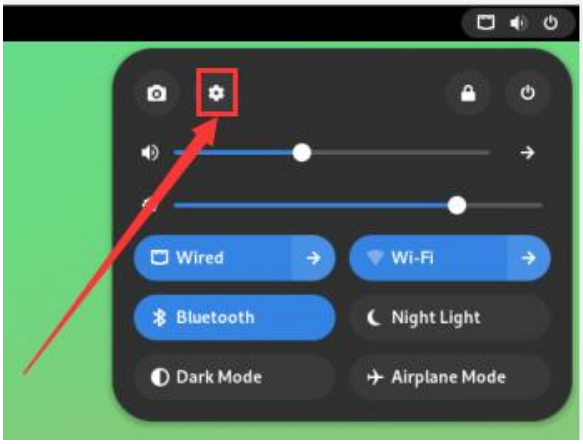


4. 4. How to set up the Chinese environment and install the Chinese input method

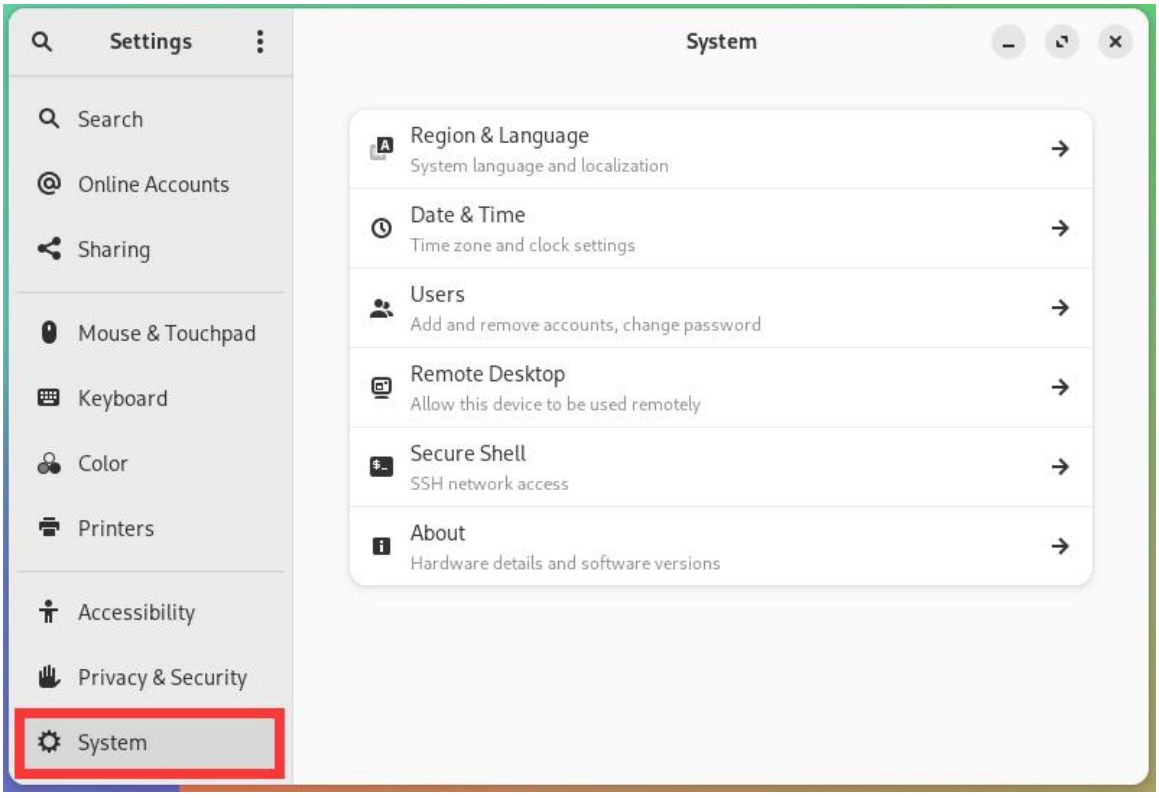
1) First click on the area in the upper right corner of the desktop



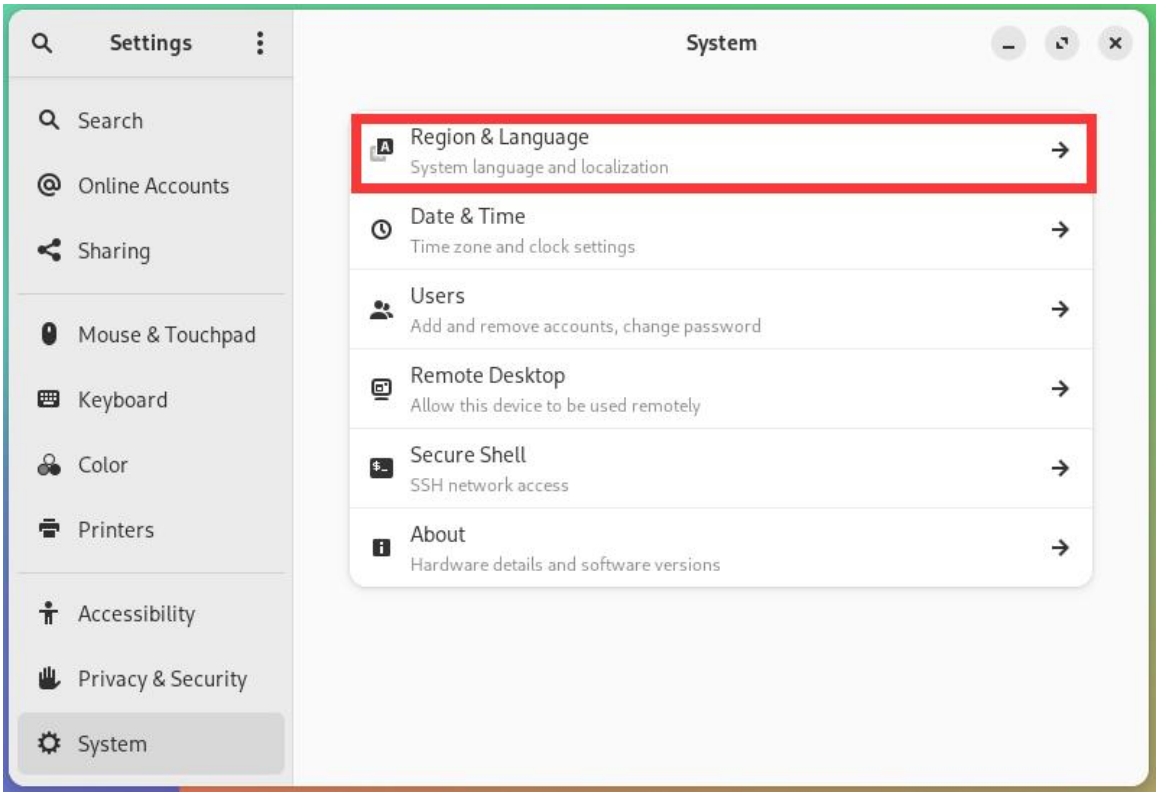
2) Then open Settings



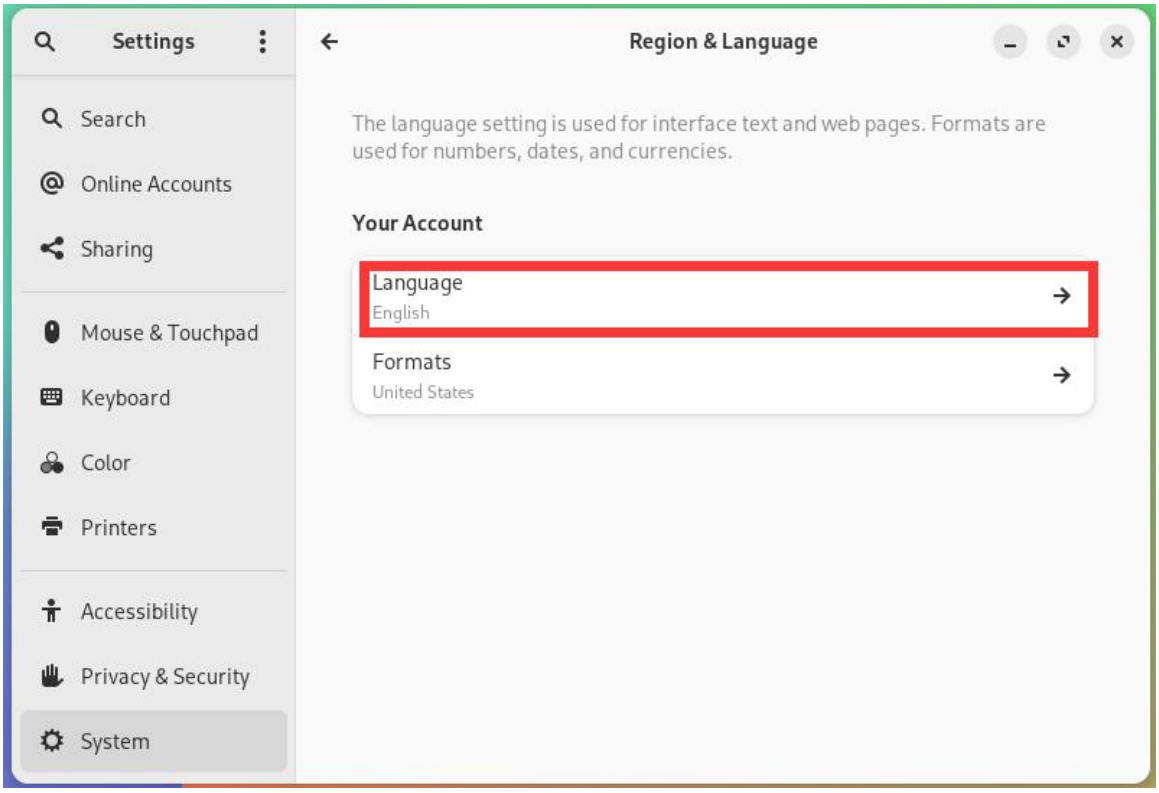
3) Then find the **System** option



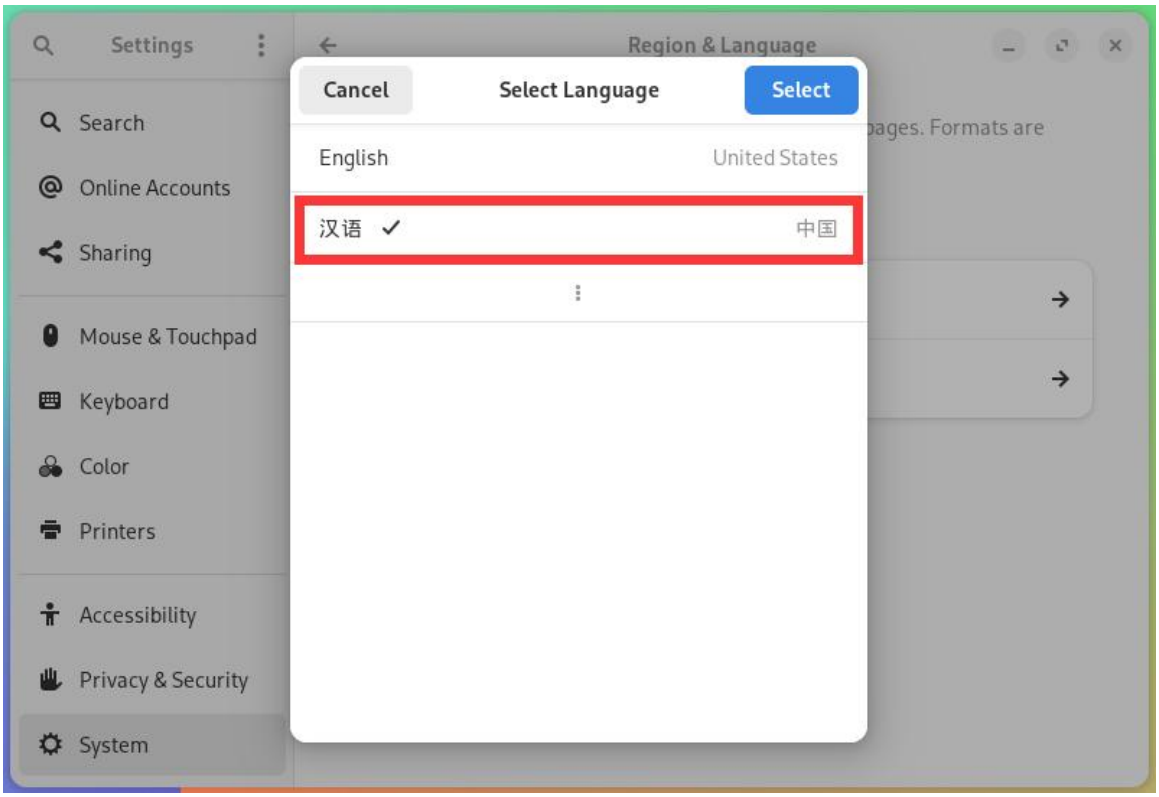
4) Then select **Region & Language**



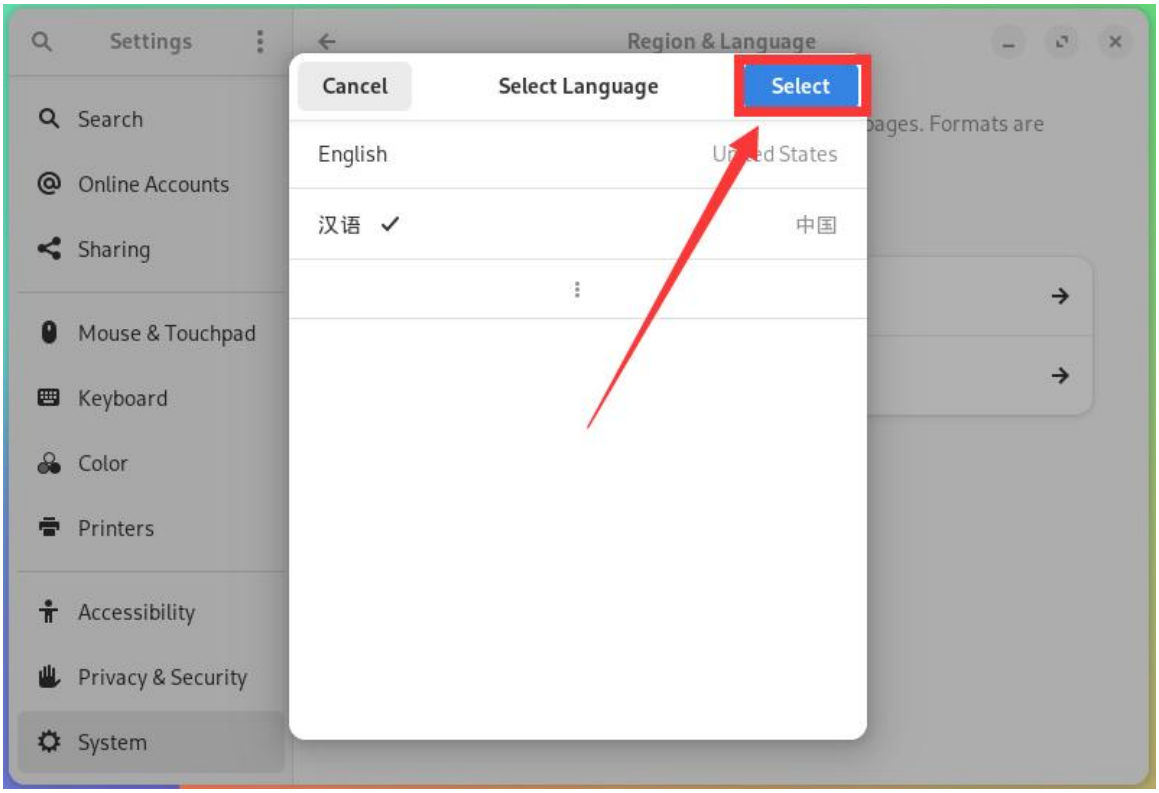
5) Then select **Language**



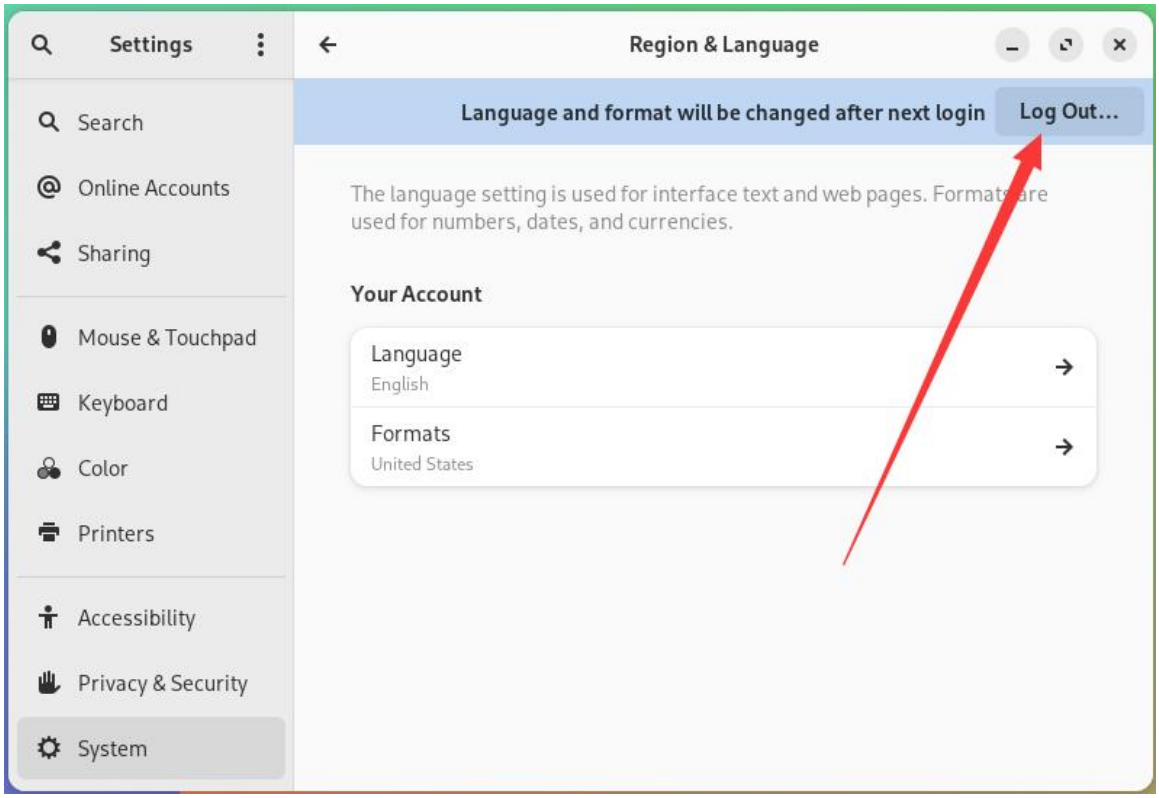
6) Then select Chinese



7) Then click**Select**



8) Then click **Log Out...** to log out of the system and then log in again



9) Then you can see that the desktop is displayed in Chinese

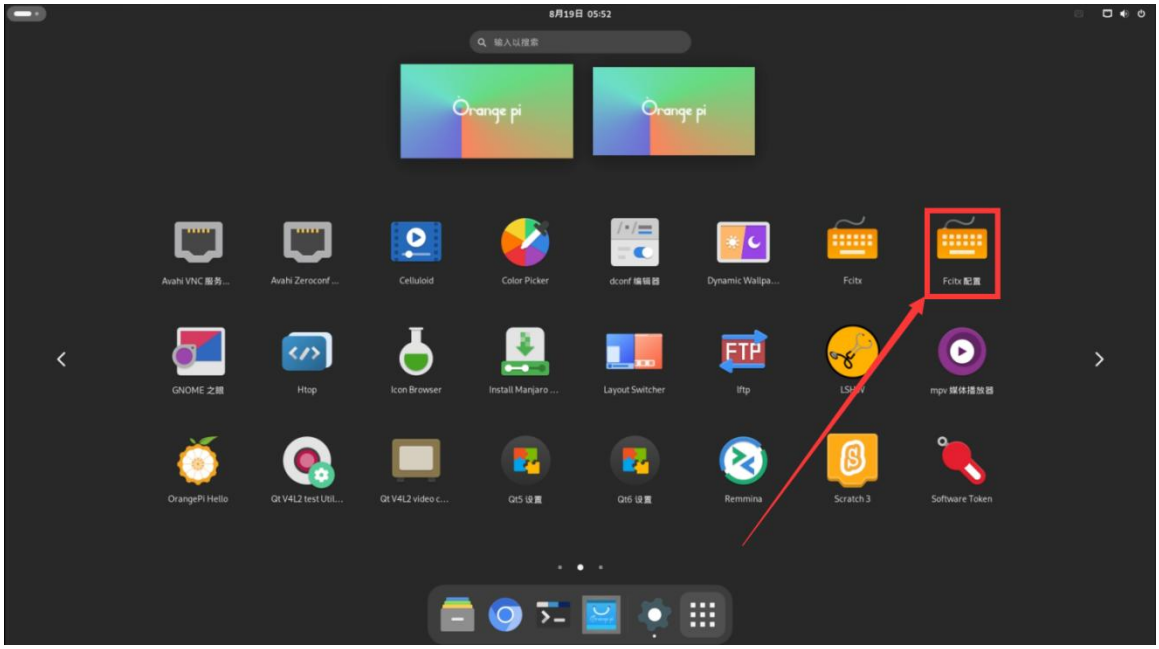


10) Then install fcitx-im and fcitx-configtool

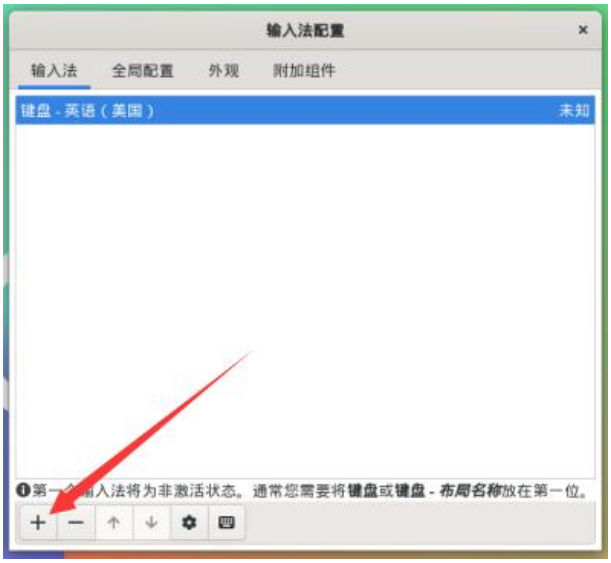
```
[orangepi@orangepi ~]$ sudo pacman -S fcitx-im fcitx-configtool
:: There are 3 members in the group fcitx-im：
:: Software Repository community
    1) fcitx    2) fcitx-qt5    3) fcitx-qt6

Enter a selection (default = select all): 1
```

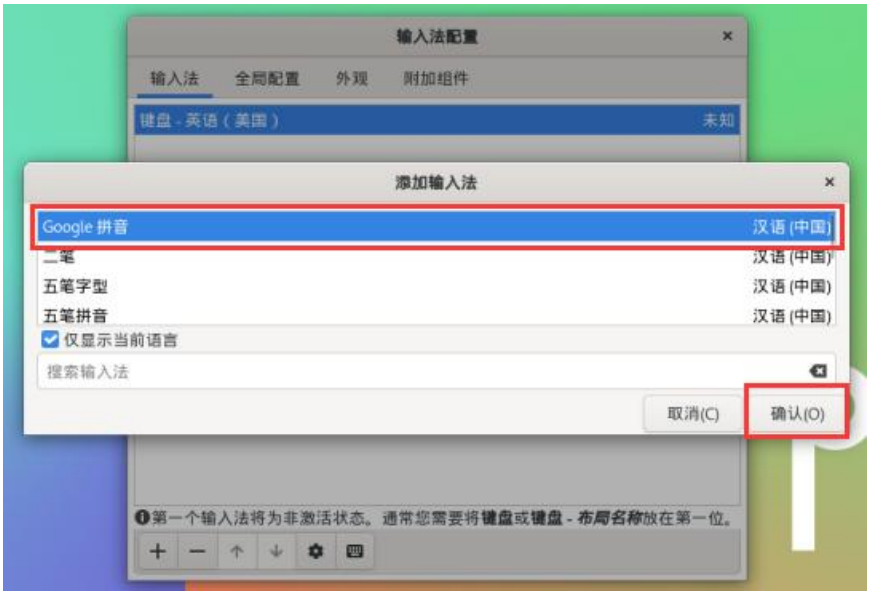
11) Then open the Fcitx configuration program



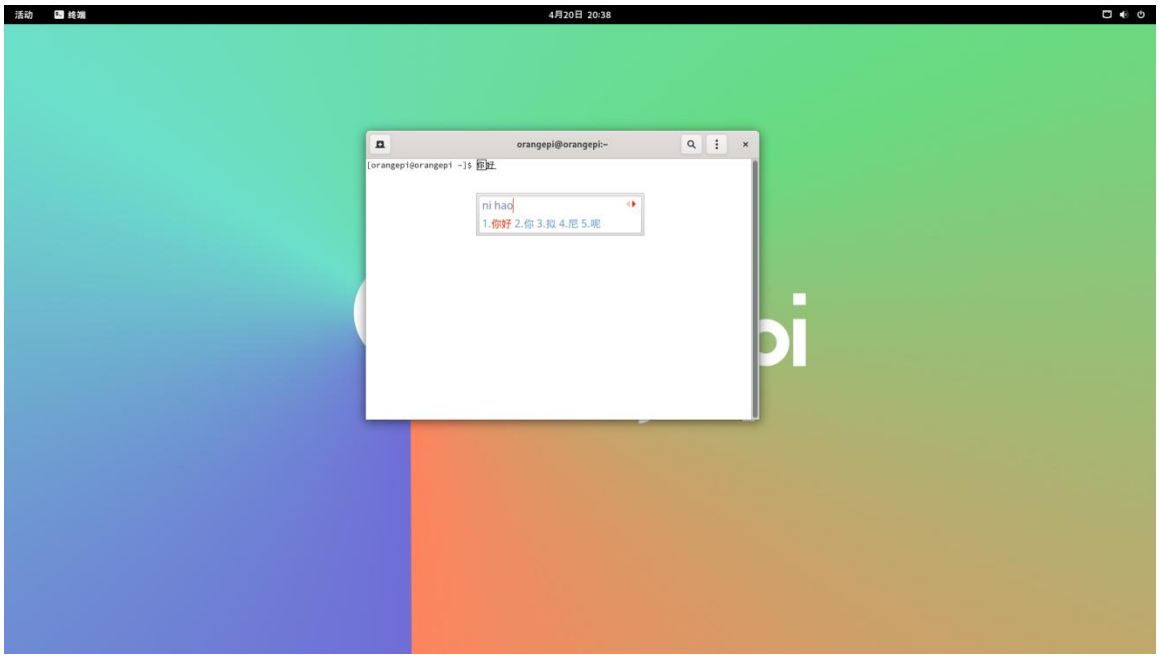


12) Then add **Google Pinyin** input method





13) Then we can open a terminal to test the Chinese input method. After opening the terminal, if the default input method is still English, we can switch to the Chinese input method by pressing **Ctrl+Space**, and then we can enter Chinese.



4.5. How to install wiringOP

Note that wiringOP is pre-installed in the OPi OS Arch image released by Orange Pi. Unless the wiringOP code is updated, you do not need to download, compile and install it again. You can use it directly.

After entering the system, you can run the gpio readall command. If you can see the following output, it means wiringOP has been pre-installed and can be used normally.

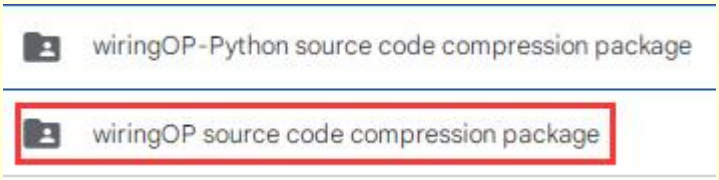
GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	IN	1	7 || 8	1	ALT10	TXD.2	3	13
		GND			9 || 10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	0	13 || 14			GND		
34	8	GPIO1_A2	IN	0	15 || 16	0	IN	GPIO1_A3	9	35
		3.3V			17 || 18	0	IN	GPIO1_A4	10	36
42	11	SPI0_TXD	IN	0	19 || 20			GND		
41	12	SPI0_RXD	IN	0	21 || 22	1	IN	GPIO1_B0	13	40
43	14	SPI0_CLK	IN	0	23 || 24	1	IN	SPI0_CS0	15	44
		GND			25 || 26	1	IN	SPI0_CS1	16	45
145	17	SDA.8	IN	0	27 || 28	1	IN	SCL.8	18	144
113	19	GPIO3_C1	IN	1	29 || 30			GND		
109	20	CAN1_RX	IN	1	31 || 32	1	OUT	GPIO4_B3	21	139
110	22	CAN1_TX	IN	1	33 || 34			GND		
114	23	GPIO3_C2	IN	1	35 || 36	1	IN	GPIO4_B7	24	143
135	25	GPIO4_A7	IN	0	37 || 38	1	IN	GPIO3_C0	26	112
		GND			39 || 40	1	IN	GPIO3_B7	27	111
GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO

1) Download the wiringOP code

```
[orangepi@orangepi ~]$ sudo pacman -Syy git
[orangepi@orangepi ~]$ git clone https://github.com/orangepi-xunlong/wiringOP.git -b next
```

Note that Orange Pi 5 Ultra needs to download the wiringOP next branch code, please do not miss the -b next parameter.

If you have problems downloading the code from GitHub, you can download the wiringOP.tar.gz source code package from the official tool on the Orange Pi 5 Ultra download page.



2) Compile and install wiringOP

```
[orangepi@orangepi ~]$ sudo pacman -Syy make gcc
[orangepi@orangepi ~]$ cd wiringOP
[orangepi@orangepi wiringOP]$ sudo ./build clean
[orangepi@orangepi wiringOP]$ sudo ./build
```

3) Test the output of the gpio readall command as follows

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	IN	1	7 || 8	1	ALT10	TXD.2	3	13
		GND			9 || 10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	0	13 || 14			GND		
34	8	GPIO1_A2	IN	0	15 || 16	0	IN	GPIO1_A3	9	35
		3.3V			17 || 18	0	IN	GPIO1_A4	10	36
42	11	SPI0_TXD	IN	0	19 || 20			GND		
41	12	SPI0_RXD	IN	0	21 || 22	1	IN	GPIO1_B0	13	40
43	14	SPI0_CLK	IN	0	23 || 24	1	IN	SPI0_CS0	15	44
		GND			25 || 26	1	IN	SPI0_CS1	16	45
145	17	SDA.8	IN	0	27 || 28	1	IN	SCL.8	18	144
113	19	GPIO3_C1	IN	1	29 || 30			GND		
109	20	CAN1_RX	IN	1	31 || 32	1	OUT	GPIO4_B3	21	139
110	22	CAN1_TX	IN	1	33 || 34			GND		
114	23	GPIO3_C2	IN	1	35 || 36	1	IN	GPIO4_B7	24	143
135	25	GPIO4_A7	IN	0	37 || 38	1	IN	GPIO3_C0	26	112
		GND			39 || 40	1	IN	GPIO3_B7	27	111
GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO

4. 6. 40pin interface GPIO, I2C, UART, SPI, CAN and PWM test

Note that if you need to set fdt overlays to open multiple configurations at the same time, please write them in one line separated by spaces as shown in the red font configuration below.

[orangepi@orangepi ~]$ **sudo vim /boot/extlinux/extlinux.conf**

LABEL Orange Pi

LINUX /Image

FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb

FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-i2c2-m0.dtbo /dtbs/rockchip/overlay/rk3588-uart4-m2.dtbo

4. 6. 1. 40pin GPIO port test

1) There are a total of 28 GPIO ports available in the 40 pins of the development board. The following uses pin 7, which corresponds to GPIO GPIO1_A7 and wPi serial number 2, as an example to demonstrate how to set the high and low levels of the GPIO port.

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	IN	1	7 || 8	1	ALT10	TXD.2	3	13
		GND			9 || 10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134

2) First set the GPIO port to output mode, where the third parameter needs to input the wPi number corresponding to the pin

```
[orangepi@orangepi ~]$ gpio mode 2 out
```

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 set successfully.

```
[orangepi@orangepi ~]$ gpio write 2 0
```

Using gpio readall, you can see that the value of pin 7 (V) has changed to 0

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	OUT	0	7 || 8	1	ALT10	TXD.2	3	13
		GND			9 || 10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134

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 set successfully.

```
[orangepi@orangepi ~]$ gpio write 2 1
```

Using gpio readall, you can see that the value of pin 7 (V) has changed to 1

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	OUT	1	7 || 8	1	ALT10	TXD.2	3	13
		GND			9 || 10	1	ALT10	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134

5) The setting method of other pins is similar. Just change the serial number of wPi to the

serial number corresponding to the pin.

4. 6. 2. 40Pin GPIO port pull-up and pull-down resistor settings

Note that the following 10 GPIO pins of Orange Pi 5 Ultra have external 3.3V pull-up, so setting the pull-down is invalid.

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	IN	0	7 || 8	1	IN	TXD.2	3	13
		GND			9 || 10	1	IN	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	0	13 || 14			GND		
34	8	GPIO1_A2	IN	0	15 || 16	0	IN	GPIO1_A3	9	35
		3.3V			17 || 18	0	IN	GPIO1_A4	10	36
42	11	SPI0_TXD	IN	0	19 || 20			GND		
41	12	SPI0_RXD	IN	0	21 || 22	0	IN	GPIO1_B0	13	40
43	14	SPI0_CLK	IN	0	23 || 24	0	IN	SPI0_CS0	15	44
		GND			25 || 26	0	IN	SPI0_CS1	16	45
145	17	SDA.8	IN	0	27 || 28	0	IN	SCL.8	18	144
113	19	GPIO3_C1	IN	1	29 || 30			GND		
109	20	CAN1_RX	IN	1	31 || 32	0	IN	GPIO4_B3	21	139
110	22	CAN1_TX	IN	1	33 || 34			GND		
114	23	GPIO3_C2	IN	1	35 || 36	0	IN	GPIO4_B7	24	143
135	25	GPIO4_A7	IN	0	37 || 38	1	IN	GPIO3_C0	26	112
		GND			39 || 40	1	IN	GPIO3_B7	27	111
GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO

1) Below, we take pin 11, which corresponds to GPIO GPIO1_A0 and wPi number 5, as an example to demonstrate how to set the pull-up and pull-down resistors of the GPIO port.

GPIO	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	GPIO
		3.3V			1 || 2			5V		
16	0	SDA.2	IN	1	3 || 4			5V		
15	1	SCL.2	IN	1	5 || 6			GND		
39	2	PWM3	IN	0	7 || 8	1	IN	TXD.2	3	13
		GND			9 || 10	1	IN	RXD.2	4	14
32	5	RXD.6	IN	0	11 || 12	0	IN	GPIO4_A6	6	134
33	7	TXD.6	IN	0	13 || 14			GND		
34	8	GPIO1_A2	IN	0	15 || 16	0	IN	GPIO1_A3	9	35

2) First, you need to set the GPIO port to input mode. The third parameter needs to enter the wPi number corresponding to the pin.

```
[orangepi@orangepi ~]$ gpio mode 5 in
```

3) After setting to input mode, execute the following command to set the GPIO port to

pull-up mode

```
[orangepi@orangepi ~]$ gpio mode 5 up
```

4) Then enter the following command to read the level of the GPIO port. If the level is 1, it means that the pull-up mode is set successfully.

```
[orangepi@orangepi ~]$ gpio read 5
1
```

5) Then execute the following command to set the GPIO port to pull-down mode

```
[orangepi@orangepi ~]$ gpio mode 5 down
```

6) Then enter the following command to read the level of the GPIO port. If the level is 0, it means that the pull-down mode is set successfully.

```
[orangepi@orangepi ~]$ gpio read 5
0
```

4. 6. 3. 40pin SPI test

1) As shown in the figure below, the available spis for Orange Pi 5 Ultra are spi0, spi1 and spi4

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of SPI0, SPI1 and SPI4 in 40 pins are shown in the following table.

	SPI0_M2 corresponds to 40pin	**SPI1_M1 corresponds to 40pin**	**SPI4_M2 corresponds to 40pin**
MOSI	**Pin 19**	**Pin 40**	**Pin 13**
MISO	**Pin 21**	**Pin 38**	**Pin 11**

CLK	Pin 23	Pin 29	Pin 15
CS0	Pin 24	Pin 35	Pin 16
CS1	Pin 26	None	None
dtbo configuration	spi0-m2-cs0-spidev spi0-m2-cs1-spidev spi0-m2-cs0-cs1-spidev	spi1-m1-cs0-spidev	spi4-m2-cs0-spidev

In OPi OS Arch system, the spi function in 40pin is disabled by default and needs to be enabled manually before it can be used.

Add the following configuration in red font to /boot/extlinux/extlinux.conf, and then restart the OPi OS Arch system to enable spi0, spi1 and spi4. If you only need to open one, just fill in one.

```
[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf
LABEL Orange Pi
LINUX /Image
FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb
FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-spi0-m2-cs0-cs1-spidev.dtbo
/dtbs/rockchip/overlay/rk3588-spi1-m1-cs0-spidev.dtbo
/dtbs/rockchip/overlay/rk3588-spi4-m2-cs0-spidev.dtbo
```

3) First check whether there is a device node of spidevx.x in the OPi OS Arch system. If it exists, it means that SPI has been set up and can be used directly

```
[orangepi@orangepi ~]$ ls /dev/spidev*
/dev/spidev0.0  /dev/spidev0.1  /dev/spidev1.0  /dev/spidev4.0
```

The above is the result after opening spi0-m2-cs0-cs1-spidev, spi1-m1-cs0-spidev and spi4-m2-cs0-spidev.

4) Do not short the mosi and miso pins of SPI0, SPI1 or SPI4. The output of running spidev_test is as follows. You can see that the data of TX and RX are inconsistent.

```
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev0.0          #spi0 commands
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev1.0          #spi1
```

```
commands
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev4.0 #spi4
commands
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 FF FF FF FF FF
FF FF FF FF FF F0 0D  | ......@....▒.....▒.
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF  | ...............................
```

5) Then short the mosi and miso pins of SPI0, SPI1 or SPI4 and run spidev_test. The output is as follows. You can see that the data sent and received are the same.

```
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev0.0 #spi0
commands
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev1.0 #spi1
commands
[orangepi@orangepi ~]$ sudo spidev_test -v -D /dev/spidev4.0 #spi4
commands
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 FF FF FF 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 FF FF FF FF FF
FF FF FF FF FF F0 0D  | ......@....▒.....▒.
```

4. 6. 4. 40pin I2C test

1) As can be seen from the table below, Orange Pi 5 Ultra has four i2c buses: i2c2, i2c4, i2c5 and i2c8

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the 4 groups of I2C buses in 40 pins are shown in the following table. I2C2_M0 and I2C2_M4 can only use one of them at the same time, not both. They are the same I2C, just connected to different pins. Please do not think that they are two different I2C buses.

I2C bus	SDA correspond 40pin	SCL correspond 40pin	Dtbo Corresponding configuration
I2C2_M0	Pin 3	Pin 5	i2c2-m0
I2C2_M4	Pin11	Pin 13	i2c2-m4
I2C4_M3	Pin15	Pin16	i2c4-m3
I2C5_M2	Pin37	Pin12	i2c5-m2
I2C8_M3	Pin27	Pin28	i2c8-m3

In OPi OS Arch system, the i2c in 40pin is disabled by default and needs to be enabled manually before it can be used.

Add the following configuration in red font to /boot/extlinux/extlinux.conf, then restart the OPi OS Arch system to enable i2c2, i2c4, i2c5 and i2c8 at the same time. If you only need to open one, just fill in one.

[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf

LABEL Orange Pi

LINUX /Image

FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb

FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-i2c2-m0.dtbo

/dtbs/rockchip/overlay/rk3588-i2c4-m3.dtbo

/dtbs/rockchip/overlay/rk3588-i2c5-m2.dtbo
/dtbs/rockchip/overlay/rk3588-i2c8-m3.dtbo

The red font configurations above need to be written in one line, and different configurations need to be separated by spaces.

3) After booting the OPi OS Arch system, first confirm that the i2c device node exists under /dev

```
[orangepi@orangepi ~]$ ls /dev/i2c-*
/dev/i2c-0   /dev/i2c-10   /dev/i2c-4   /dev/i2c-6   /dev/i2c-9
/dev/i2c-1   /dev/i2c-2    /dev/i2c-5   /dev/i2c-7   /dev/i2c-8
```

4) Then connect an i2c device to the i2c pin of the 40pin connector

Generally, you only need to connect one of the 3.3v pin and the 5v pin. Please choose to connect the 3.3v pin or the 5v pin according to the specific i2c device you are connecting.

5) Then use the **i2cdetect -y** command. If the address of the connected i2c device can be detected, it means that i2c can be used normally.

```
[orangepi@orangepi ~]$ sudo pacman -Syy i2c-tools
[orangepi@orangepi ~]$ sudo i2cdetect -y 2          #i2c2 commands
[orangepi@orangepi ~]$ sudo i2cdetect -y 4          #i2c4 commands
[orangepi@orangepi ~]$ sudo i2cdetect -y 5          #i2c5 commands
[orangepi@orangepi ~]$ sudo i2cdetect -y 8          #i2c8 commands
```

4.6.5. 40pin UART Test

1) As shown in the table below, the Orange Pi 5 Ultra has three sets of UART buses available: UART 3, UART 4, and UART 6. UART 2 is used for debugging serial port functions and is not included in the calculation.

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

2) The corresponding pins of the four UART buses in the 40 pin configuration are shown in the table below.

UART bus	RX corresponds to 40 pins	TX corresponds to 40 pins	Dtbo corresponding configuration
UART3_M1	Pin 33	Pin 31	uart3-m1
UART4_M2	Pin 19	Pin 23	uart4-m2
UART6_M1	Pin 11	Pin 13	uart6-m1

In the OPi OS Arch system, UART in 40pin is turned off by default and needs to be manually turned on to use.

Add the configuration in red font below to/boot/extlinux/extlinux.conf, and then restart the OPi OS Arch system to open UART 3, UART 4, and UART 6 simultaneously. If only one needs to be opened, fill in one.

[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf

LABEL Orange Pi

LINUX /Image

FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb

FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-uart3-m1.dtbo
/dtbs/rockchip/overlay/rk3588-uart4-m2.dtbo
/dtbs/rockchip/overlay/rk3588-uart6-m1.dtbo

The red font configuration above needs to be written on one line, and different configurations need to be separated by spaces.

3) After entering the Linux system, first confirm whether there is a device node corresponding to UART in/dev

```
[orangepi@orangepi ~]$ ls /dev/ttyS*
/dev/ttyS3   /dev/ttyS4   /dev/ttyS6   /dev/ttyS7
```

4) Then start testing the UART interface by short circuiting the RX and TX of the UART interface to be tested using DuPont wires

5) Use the **gpio serial** command to test the loopback function of the serial port as shown below. If you can see the print below, it indicates that the serial communication is normal (ttySX needs to be replaced with the corresponding UART node name, please do not copy it)

```
[orangepi@orangepi ~]$ sudo gpio serial /dev/ttySX
Out:    0:  ->  0
Out:    1:  ->  1
Out:    2:  ->  2
Out:    3:  ->  3
Out:    4:  ->  4
Out:    5:  ->  5^C
```

4.6.6. PWM Testing Method

1) From the table below, it can be seen that the Orange Pi 5 Ultra has seven PWM channels available, including PWM 0, PWM 1, PWM 3, PWM 6, PWM 12, PWM 13, and PWM 14

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

1) The corresponding pins of PWM in 40 pins are shown in the table below. PWM0.M0 and PWM0-M2, as well as PWM1_M0 and PWM1_M2, can only be used one at a time and cannot be used simultaneously. They are all the same PWM, just connected to

different pins. Please do not assume that they are two different PWM buses.

PWM bus	Corresponding to 40 pins	Dtbo corresponding configuration
PWM0_M0	Pin 5	pwm0-m0
PWM0_M2	Pin 15	pwm0-m2
PWM1_M0	Pin 3	pwm1-m0
PWM1_M2	Pin 16	pwm1-m2
PWM3_M3	Pin 7	pwm3-m3
PWM6_M1	Pin 27	pwm6-m1
PWM12_M0	Pin 31	pwm12-m0
PWM13_M0	Pin 33	pwm13-m0
PWM14_M0	Pin 35	pwm14-m0

In the OPi OS Arch system, the PWM in the 40pin is turned off by default and needs to be manually turned on to use.

Add the configuration in red font below to/boot/extlinux/extlinux.conf, and then restart the OPi OS Arch system to simultaneously open pwm0, pwm1, pwm3, pwm6, pwm12, pwm13, and pwm14. If only one needs to be opened, fill in one.

[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf

```
LABEL Orange Pi
LINUX /Image
FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb
FDTOVERLAYS    /dtbs/rockchip/overlay/rk3588-pwm0-m0.dtbo
/dtbs/rockchip/overlay/rk3588-pwm1-m0.dtbo
/dtbs/rockchip/overlay/rk3588-pwm3-m3.dtbo
/dtbs/rockchip/overlay/rk3588-pwm6-m1.dtbo
/dtbs/rockchip/overlay/rk3588-pwm12-m0.dtbo
/dtbs/rockchip/overlay/rk3588-pwm13-m0.dtbo
/dtbs/rockchip/overlay/rk3588-pwm14-m0.dtbo
```

The red font configuration above needs to be written on one line, and different configurations need to be separated by spaces.

2) When a PWM is turned on, an additional pwmchipX (where X is a specific number) will appear in**/sys/class/pwm/**. For example, when opening pwm3, checking the pwmchipX under**/sys/class/pwm/**will change from two to three

```
[orangepi@orangepi ~]$ ls /sys/class/pwm/
pwmchip0  pwmchip1  pwmchip2
```

3) Which pwmchip corresponds to pwm3 above? Let's first check the output of the **ls /sys/class/pwm/ -l** command, as shown below:

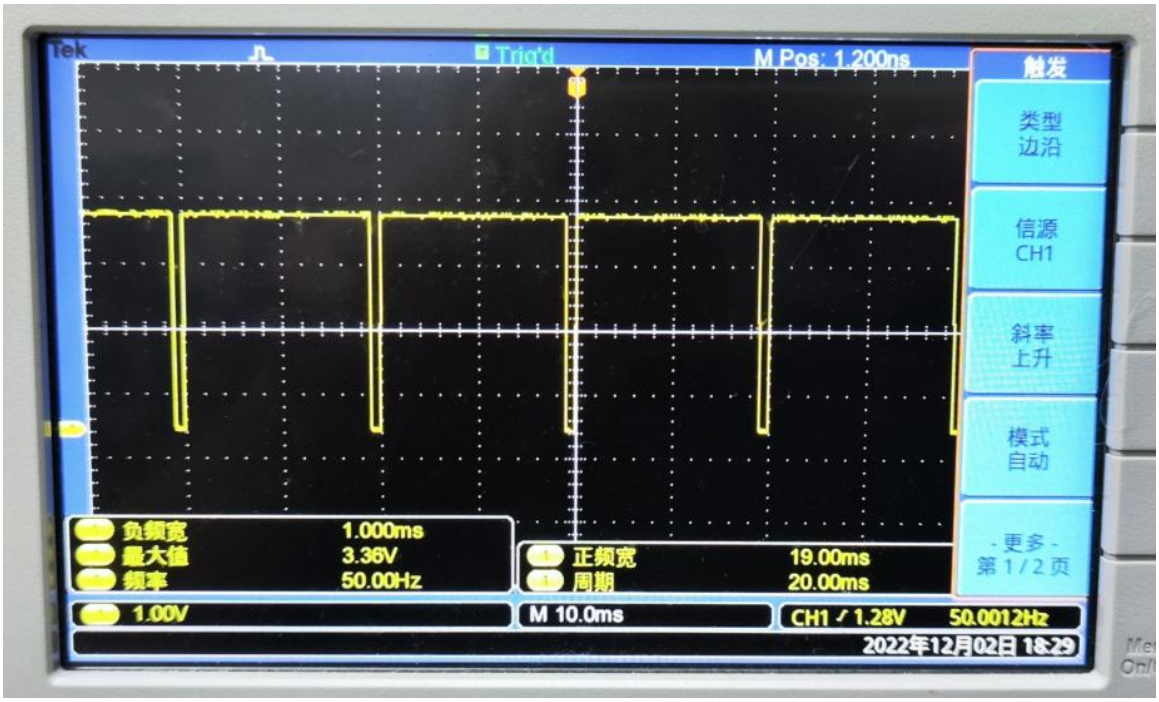
```
[root@orange-os ~]#  ls /sys/class/pwm/ -l
total 0
lrwxrwxrwx 1 root root 0 Jul  5 09:34 pwmchip0 -> ../../devices/platform/fd8b0030.pwm/pwm/pwmchip0
lrwxrwxrwx 1 root root 0 Jul  5 09:34 pwmchip1 -> ../../devices/platform/febe0010.pwm/pwm/pwmchip1
lrwxrwxrwx 1 root root 0 Jul  5 09:34 pwmchip2 -> ../../devices/platform/febf0030.pwm/pwm/pwmchip2
```

4) Then, as shown in the table below, the base address of the pwm3 register is fd8b0030. Looking at the output of the **ls /sys/class/pwm/ -l** command, it can be seen that pwmchip0 is linked to fd8b0030. pwm, so pwm3 corresponds to pwmchip0

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号
			3.3V		1
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7
			GND		9
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15
			3.3V		17
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19
		SPI0_MISO_M2	GPIO1_B1	41	21
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23
			GND		25
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27
		SPI1_CLK_M1	GPIO3_C1	113	29
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35
I2C5_SDA_M2			GPIO4_A7	135	37
			GND		39

5) Then use the following command to make pwm3 output a 50Hz square wave (please switch to the root user first, and then execute the following command)

```
[root@orangepi orangepi]# echo 0 > /sys/class/pwm/pwmchip0/export
[root@orangepi orangepi]# echo 20000000 > /sys/class/pwm/pwmchip0/pwm0/period
[root@orangepi orangepi]# echo 1000000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
[root@orangepi orangepi]# echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
```



6) The testing method for PWM 3 demonstrated above is similar to other PWM testing methods.

4. 6. 7. Testing Methods for CAN

1) According to the table below, the Orange Pi 5 Ultra can use two sets of CAN buses, CAN0 and CAN1

复用功能	复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能	复用功能
			3.3V		1	2		5V			
PWM1_M0 (fd8b0010)	CAN0_RX_M0	I2C2_SDA_M0	GPIO0_C0	16	3	4		5V			
PWM0_M0 (fd8b0000)	CAN0_TX_M0	I2C2_SCL_M0	GPIO0_B7	15	5	6		GND			
		PWM3_IR_M3 (fd8b0030)	GPIO1_A7	39	7	8	13	GPIO0_B5	UART2_TX_M0		
			GND		9	10	14	GPIO0_B6	UART2_RX_M0		
SPI4_MISO_M2	I2C2_SDA_M4	UART6_RX_M1	GPIO1_A0	32	11	12	134	GPIO4_A6			I2C5_SCL_M2
SPI4_MOSI_M2	I2C2_SCL_M4	UART6_TX_M1	GPIO1_A1	33	13	14		GND			
SPI4_CLK_M2	PWM0_M2 (fd8b0000)	I2C4_SDA_M3	GPIO1_A2	34	15	16	35	GPIO1_A3	I2C4_SCL_M3	PWM1_M2 (fd8b0010)	SPI4_CS0_M2
			3.3V		17	18	36	GPIO1_A4			
	UART4_RX_M2	SPI0_MOSI_M2	GPIO1_B2	42	19	20		GND			
		SPI0_MISO_M2	GPIO1_B1	41	21	22	40	GPIO1_B0			
	UART4_TX_M2	SPI0_CLK_M2	GPIO1_B3	43	23	24	44	GPIO1_B4	SPI0_CS0_M2		
			GND		25	26	45	GPIO1_B5	SPI0_CS1_M2		
	PWM6_M1(pwm@febd0020)	I2C8_SDA_M3	GPIO4_C1	145	27	28	144	GPIO4_C0	I2C8_SCL_M3		
		SPI1_CLK_M1	GPIO3_C1	113	29	30		GND			
PWM12_M0 (febf0000)	UART3_TX_M1	CAN1_RX_M0	GPIO3_B5	109	31	32	139	GPIO4_B3			
PWM13_M0 (febf0010)	UART3_RX_M1	CAN1_TX_M0	GPIO3_B6	110	33	34		GND			
PWM14_M0 (febf0020)		SPI1_CS0_M1	GPIO3_C2	114	35	36	143	GPIO4_B7			
I2C5_SDA_M2			GPIO4_A7	135	37	38	112	GPIO3_C0	SPI1_MISO_M1		
			GND		39	40	111	GPIO3_B7	SPI1_MOSI_M1		

In the OPi OS Arch system, the CAN in the 40pin is turned off by default and needs to be manually turned on to use.

Add the configuration in red font below to/boot/extlinux/extlinux.conf, and then restart the OPi OS Arch system to enable CAN0 and CAN1.

[orangepi@orangepi ~]$ sudo vim /boot/extlinux/extlinux.conf
LABEL Orange Pi
LINUX /Image
FDT /dtbs/rockchip/rk3588-orangepi-5-ultra.dtb
FDTOVERLAYS /dtbs/rockchip/overlay/rk3588-can0-m0.dtbo /dtbs/rockchip/overlay/rk3588-can1-m0.dtbo

The red font configuration above needs to be written on one line, and different configurations need to be separated by spaces.

2) After entering the OPi OS Arch system, use the **sudo ifconfig -a** command. If you can see the device nodes of CAN, it means that CAN has been opened correctly

```
[orangepi@orangepi ~]$ sudo pacman -Syy net-tools
[orangepi@orangepi ~]$ sudo ifconfig -a
can0: flags=128<NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 91

can1: flags=128<NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 92
```

3) The pins corresponding to CAN0 and CAN1 are

	CAN0	CAN1
TX pin	Corresponding to pin 5	Corresponding to pin 33

RX pin	Corresponding to pin 3	Corresponding to pin 31

4) Please refer to the section on testing CAN sending and receiving messages using CANalyst II analyzer for the method.

5. Linux SDK - Orangepi build usage instructions

5.1. Compile System Requirements

We can cross compile the Linux image of the development board on an x64 computer, or compile the Linux image of the development board on the Ubuntu 22.04 system. Please choose one according to your preferences.

If using orangepi build to compile Linux images on the Ubuntu 22.04 system of the development board, please ensure proper cooling (especially during SSD startup). If the heat dissipation is not done properly, it is easy to cause file system runaway errors.

5.1.1. Compile using the Ubuntu 22.04 system on the development board

1) The Linux SDK, also known as **orangepi-build**, supports running on **Ubuntu 22.04** on the development board (which has not been tested on other systems), so before downloading orangepi build, please ensure that the Ubuntu version installed on the development board is Ubuntu 22.04. The command to check the installed Ubuntu version on the development board is as follows. If the Release field does not display **22.04**, it means that the current Ubuntu version used does not meet the requirements. Please replace the system before performing the following operations.

```
orangepi@orangepi:~$ lsb_release -a
No LSB modules are available.
Distributor ID:	Ubuntu
Description:	Ubuntu 22.04.1 LTS
Release:	22.04
Codename:	jammy
```

2) Since the kernel and U-boot source code are stored on GitHub, it is important to ensure that the development board can download the code from GitHub properly when compiling the image.

5.1.2. Compile on Ubuntu 22.04 computer using x64

1) The Linux SDK, also known as **orangepi-build**, supports running on computers with **Ubuntu 22.04** installed. Therefore, before downloading orangepi build, please make

sure that the Ubuntu version installed on your computer is Ubuntu 22.04. The command to check the installed Ubuntu version on the computer is as follows. If the Release field does not display **22.04**, it means that the current Ubuntu version used does not meet the requirements. Please replace the system before performing the following operations.

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID:	Ubuntu
Description:	Ubuntu 22.04 LTS
Release:	22.04
Codename:	jammy
```

2) If the computer is installed with a Windows system and does not have Ubuntu 22.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu 22.04 virtual machine on the Windows system. However, please note that do not compile orangepi build on a WSI virtual machine, as orangepi build has not been tested on a WSI virtual machine, so it cannot be guaranteed that orangepi build can be used properly in WSI.

3) The installation image download address for Ubuntu 22.04 **amd64** version is:

https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/22.04/ubuntu-22.04.3-desktop-amd64.iso

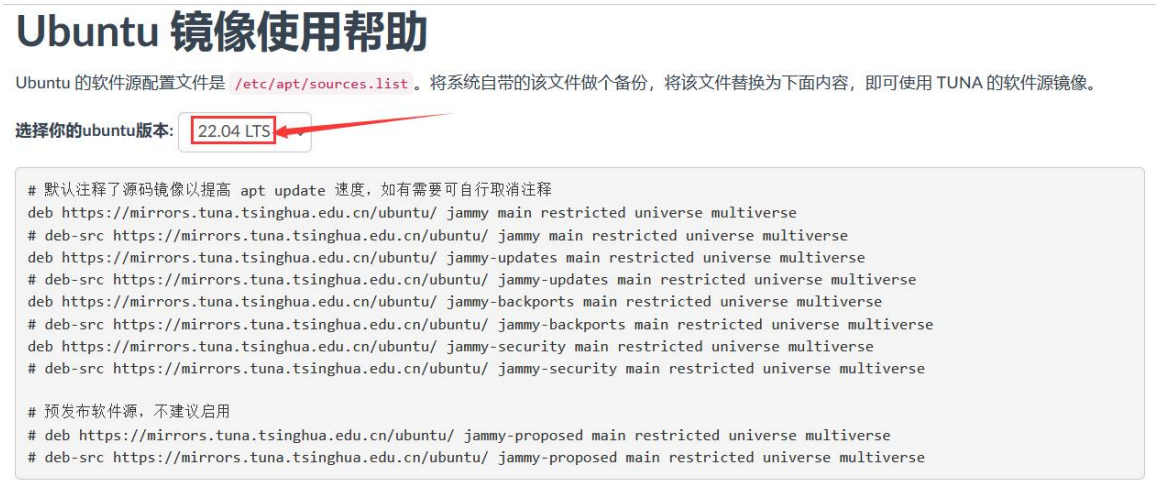
https://repo.huaweicloud.com/ubuntu-releases/22.04/ubuntu-22.04.3-desktop-amd64.iso

4) After installing Ubuntu 22.04 on a computer or virtual machine, please first set the software source of Ubuntu 22.04 to Qinghua Source, otherwise errors may occur during software installation due to network issues

a. The method of replacing Tsinghua Source can refer to the instructions on this webpage

https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/

b. Note that Ubuntu version needs to be switched to 22.04



c. The contents of the **/etc/apt/sources.list** file that needs to be replaced are

```
test@test:~$ sudo mv /etc/apt/sources.list /etc/apt/sources.list.bak
test@test:~$ sudo vim /etc/apt/sources.list
# By default, the source code image has been annotated to improve the speed of apt updates. If necessary, you can remove the annotation yourself
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# Pre release software source, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

d. After replacement, it is necessary to update the package information and ensure that there are no errors

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

e. In addition, since the kernel and U-boot source code are stored on GitHub, it is important to ensure that the computer can download the code from GitHub properly when compiling the image.

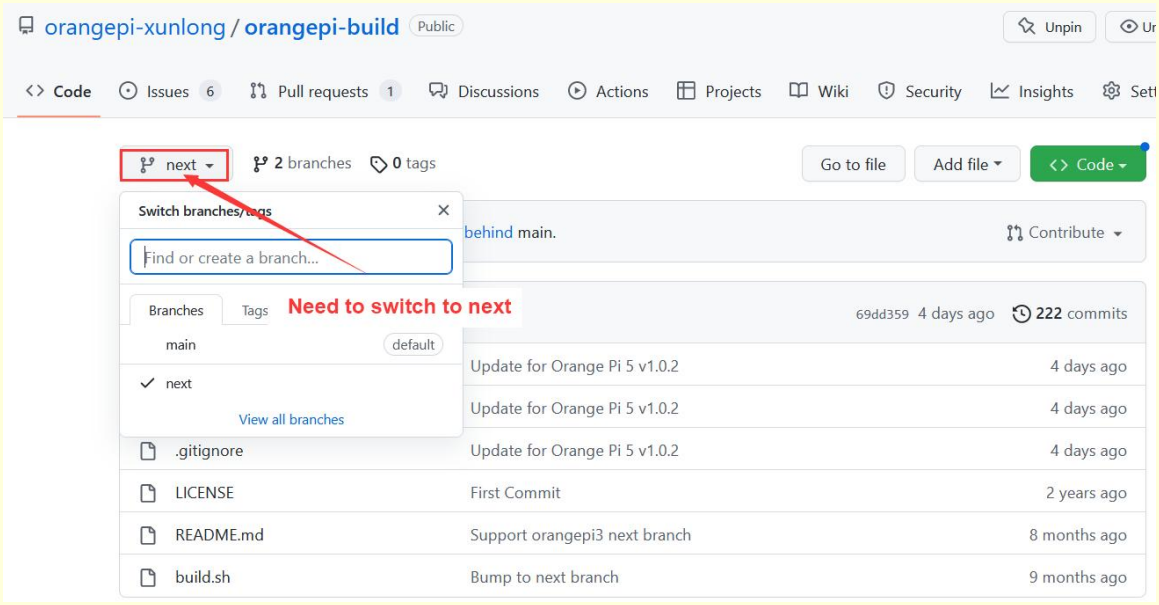
5. 2. Obtain the source code of Linux SDK

5. 2. 1. Download Orangepi build from GitHub

1) The Linux SDK actually refers to the Orangepi build code, which is modified based on the armbian build compilation system. Using Orangepi build, multiple versions of Linux images can be compiled. First, download the code for orangepi build. The command is as follows:

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install -y git
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git -b next
```

Note that the Orange Pi 5 Ultra development board requires downloading the next branch source code of the orangepi build. The git clone command above needs to specify the branch of the orangepi build source code as next.



Downloading the code for orangepi build through the git clone command does not require entering the username and password of the GitHub account (the same applies to downloading other code in this manual). If Ubuntu PC prompts for the username and password of the GitHub account after entering the git clone

command, it is usually due to an incorrect input of the address of the orangepi build repository after git clone. Please carefully check the spelling of the command for errors, rather than 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 on the development board are as follows

branch	u-boot version	Linux kernel version
legacy	u-boot 2017.09	linux5.10
current	u-boot 2017.09	linux6.1

The branch mentioned here and the branch of orangepi build source code are not the same thing, please don't confuse them. This branch is mainly used to distinguish between different versions of kernel source code.

At present, we define the linux5.10 bsp kernel provided by RK as the legacy branch, and the linux6.1 bsp kernel as the current branch.

3) After downloading orangepi build, it will include the following files and folders

a. **build.sh**: Compile startup script
b. **external**: Contains configuration files required for compiling images, specific scripts, and source code for some programs, etc
c. **LICENSE**: GPL 2 License File
d. README.md: Orangepi build documentation
e. **scripts**: General script for compiling Linux images

```
test@test:~/orangepi-build$ ls
build.sh  external  LICENSE  README.md  scripts
```

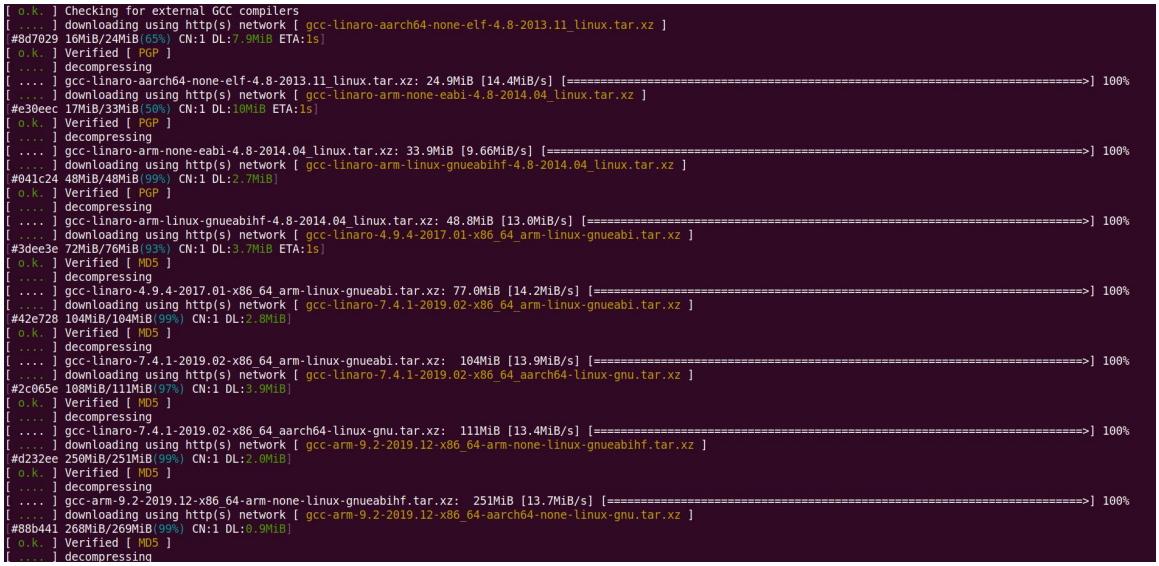
If you download the code for Orangepi build from GitHub, you may find that the Orangepi build does not include the source code for u-boot and Linux kernel, nor does it require a cross compilation toolchain to compile u-boot and Linux kernel. This is normal because these things are stored in other separate GitHub repositories or on certain servers (the addresses will be detailed below). Orangepi build specifies the addresses of u-boot, Linux kernel, and cross compilation toolchain in the script and configuration files. When running Orangepi build, if it finds that these things are not available locally, it will automatically download them from the

corresponding places.

5. 2. 2. Download the cross compilation toolchain

The cross compilation toolchain will only be downloaded when using orangepi build to compile the image on an x64 computer. Compiling the Linux image of the development board in Ubuntu 22.04 will not download cross compilation toolchains, and orangepi build/toolchains will be an empty folder.

1) When Orangepi build runs for the first time, it automatically downloads the cross compilation toolchain and places it in the **toolchains** folder. After running the build.sh script of Orangepi build, it checks whether all the cross compilation toolchains in **toolchains** exist. If they do not exist, it will restart the download. If they do exist, it will be used directly without repeated downloads.



2) The mirror website of the cross compilation toolchain in China is the open source software mirror site of Tsinghua University

https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/

3) After downloading **toolchains**, multiple versions of cross compilation toolchains will be included, and the development board will only use two of them

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu
gcc-arm-11.2-2022.02-x86_64-arm-none-linux-gnueabihf
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 Linux kernel source code is

a. linux5.10 和 linux6.1

gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu

5) The cross compilation toolchain used to compile the u-boot source code is

a. v2017.09

gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu

5.2.3. Explanation of the complete directory structure of orangepi build

1) After downloading the orangepi build repository, it does not include the Linux kernel, U-boot source code, or cross compilation toolchain. The Linux kernel and U-boot source code are stored in separate Git repositories

a. The Git repository where the Linux 5.10 kernel source code is stored is as follows:

https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10-rk35xx

b. The Git repository where the Linux 6.1 kernel source code is stored is as follows:

https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-6.1-rk35xx

c. The git repository where the u-boot source code is stored is as follows:

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2017.09-rk3588

2) When Orangepi build is first run, it will download the cross compilation toolchain, u-boot, and Linux kernel source code. After successfully compiling the Linux image once, the files and folders that can be seen in Orangepi build are:

a. **build.sh**: Compile startup script

b. **external**: Contains configuration files required for compiling the image, scripts

for specific functions, and source code for some programs. The rootfs compressed file cached during the image compilation process is also stored in the external file

c. **kernel**: Store the source code of the Linux kernel. The folder named **orange-pi-5.10-rk35xx** contains the kernel source code of the legacy branch of the RK3588/RK3588S series development board, while the folder named **orange-pi-6.1-rk35xx** contains the kernel source code of the current branch of the RK3588/RK3588S series development board. Please do not manually modify the name of the kernel source code folder. If modified, the compilation system will re download the kernel source code when running
d. **LICENSE**: GPL 2 License File
e. **README.md**: Orangepi build documentation
f. **output**: Store compiled deb packages such as u-boot and Linux, compilation logs, and compiled images
g. **scripts**: General script for compiling Linux images
h. **toolchains**: Store cross compilation toolchain
i. **u-boot**: Store the source code of u-boot, and the folder named **v2017.09-rk3588** contains the u-boot source code of the legacy branch of RK3588/RK3588S series development boards. Please do not manually modify the name of the u-boot source code folder. If it is modified, the compilation system will re download the u-boot source code when running
j. **userpatches**: Store the configuration files required for compiling scripts

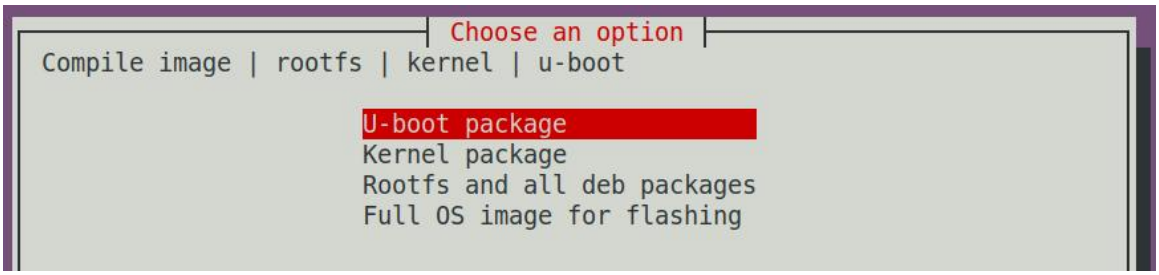
```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts  toolchains  u-boot  userpatches
```

5.3. Compiling u-boot

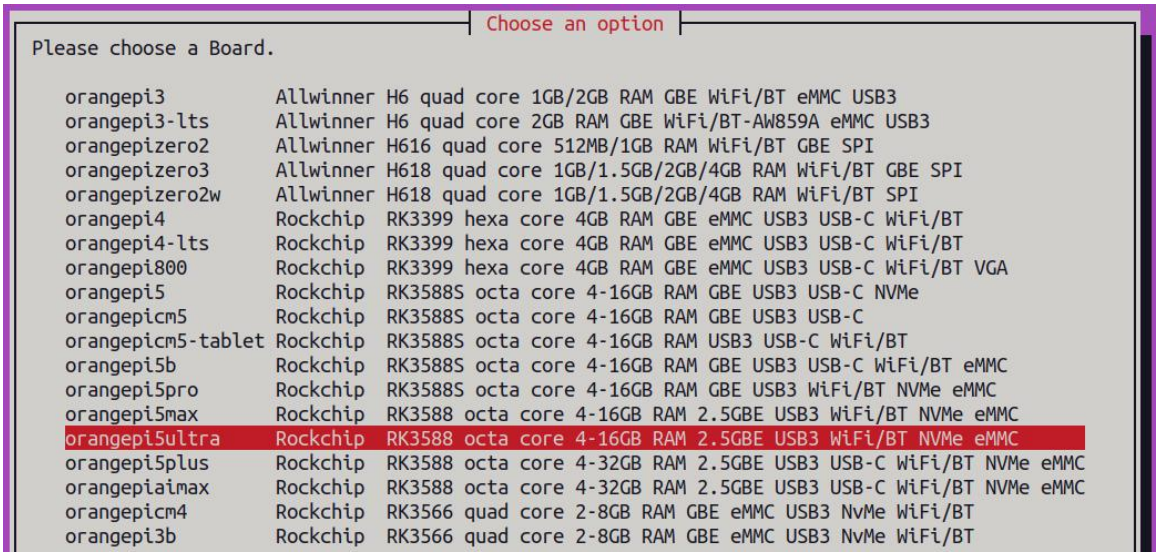
1) Run the build.sh script, remember to grant sudo privileges

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **U-boot package** and press Enter



3) Next, select the model of the development board



4) Then it will start compiling u-boot, and some of the information prompted during compilation is explained as follows

a. Version of u-boot source code

[o.k.] Compiling u-boot [**v2017.09**]

b. Version of cross compilation toolchain

[o.k.] Compiler version [**aarch64-linux-gnu-gcc 7.4.1**]

c. The path of the compiled u-boot deb package

[o.k.] Target directory [**orangepi-build/output/debs/u-boot**]

d. The package name of the compiled u-boot deb package

[o.k.] File name [**linux-u-boot-legacy-orangepi5ultra_1.0.2_arm64.deb**]

e. Compilation time used

[o.k.] Runtime [**1 min**]

f. Repeat the command to compile u-boot, and 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=orangepi5ultra
BRANCH=legacy BUILD_OPT=u-boot KERNEL_CONFIGURE=no ]
```

5) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-legacy-orangepi5ultra_1.0.2_arm64.deb
```

6) The generated deb package of u-boot contains the following files

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-orangepi5ultra_1.0.2_arm64.deb .      (Please note that there is a
'.' at the end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-legacy-orangepi5ultra_1.0.2_arm64.deb    usr
```

b. The decompressed file is shown below

```
test@test:~/orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-legacy-orangepi5ultra_1.0.2_arm64
    │   ├── idbloader.img
    │   ├── rkspi_loader.img
    │   └── u-boot.itb
    └── u-boot
        ├── LICENSE
        ├── orangepi_5_ultra_defconfig
        └── platform_install.sh

3 directories, 6 files
```

7) When the orangepi build compilation system compiles the u-boot source code, it first synchronizes the u-boot source code with the u-boot source code on the GitHub server. Therefore, 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 the u-boot completely before turning off this function, otherwise it will prompt that the u-boot**

source code cannot be found. If it is a compressed source code downloaded from Baidu Cloud Drive, there is no problem because the u-boot source code is already cached). Otherwise, the modifications made will be restored. The method is as follows:

Set the IGNOREUPDATES variable to "yes" in userpatches/config-default.conf

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

8) When debugging u-boot code, you can use the following method to update u-boot in the Linux image for testing

a. Upload the compiled deb package of u-boot 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-orangepi5ultra_1.0.2_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the deb package of the installed u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepi5ultra-legacy
```

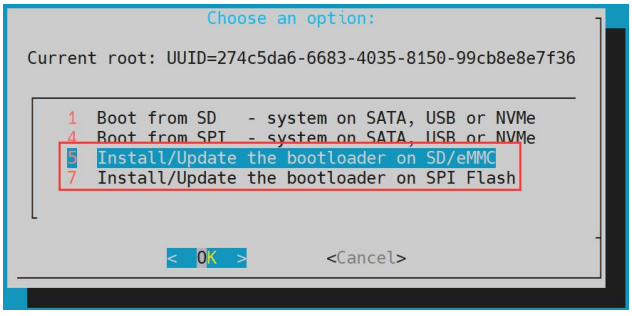
c. Reinstall the newly uploaded deb package for u-boot

```
root@orangepi:~# dpkg -i linux-u-boot-legacy-orangepi5ultra_1.0.2_arm64.deb
```

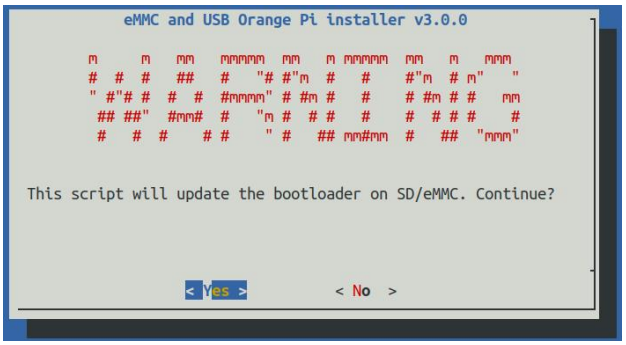
d. Then run the nand sata install script

```
root@orangepi:~# nand-sata-install
```

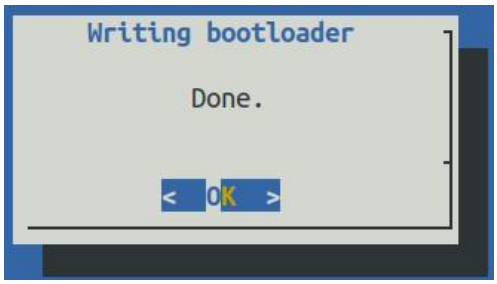
e. Then select **5 Install/Update the bootloader on SD/eMM** to update u-boot in TF card or **7 Install/Update the bootloader on SPI Flash** to update u-boot in SPI Flash



f. After pressing the enter key, a warning will first pop up



g. Pressing the enter key again will start updating u-boot, and after the update is complete, the following information will be displayed



h. Then you can restart the development board to test whether the u-boot modifications have taken effect

9) Other useful information

a. In the u-boot 2017.09 source code, the defconfig configuration file used by the development board is

orangepi-build/u-boot/v2017.09-rk3588/configs/orangepi_5_ultra_defconfig

b. In the U-boot 2017.09 source code, the dts file used for the development board is

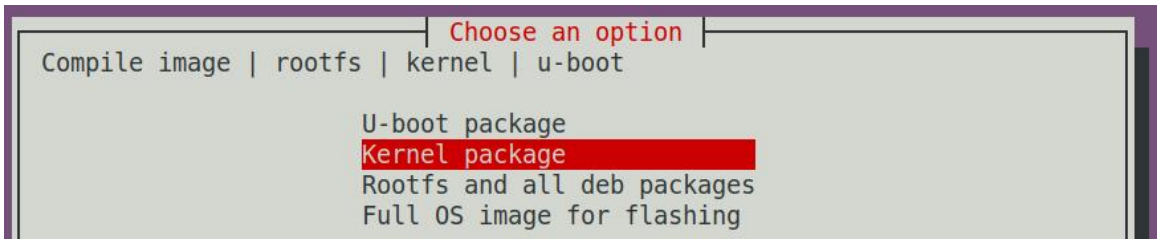
orangepi-build/u-boot/v2017.09-rk3588/arch/arm/dts/rk3588-orangepi-5-ultra.dts

5. 4. Compiling Linux Kernel

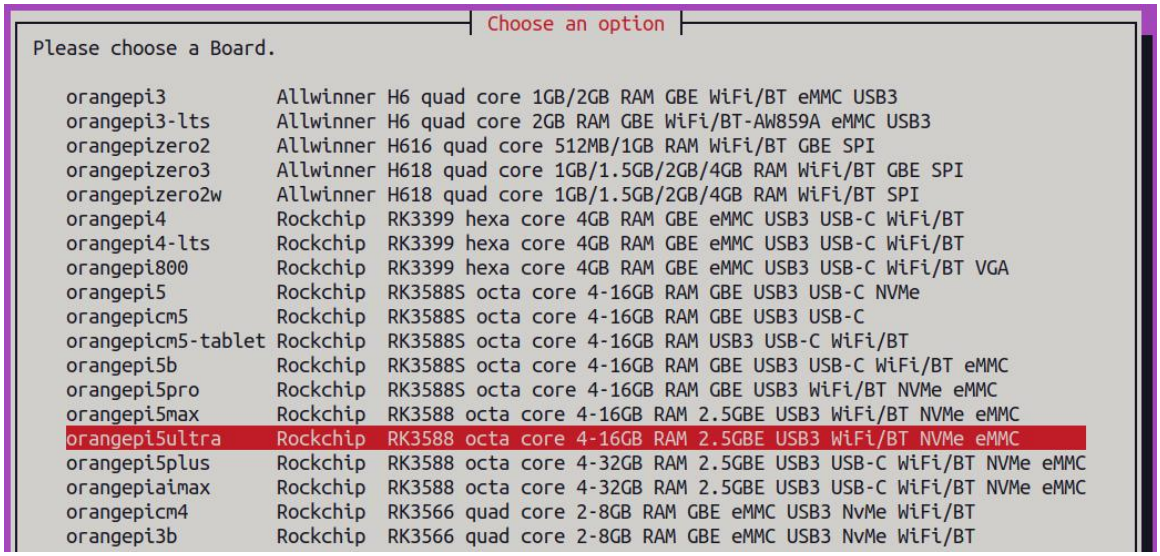
1) Run the build.sh script, remember to grant sudo privileges

```
test@test:~/orangepi-build$ sudo ./build.sh
```

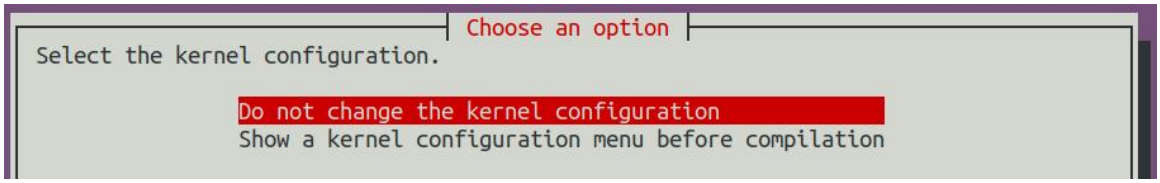
2) Select **Kernel package** and press Enter



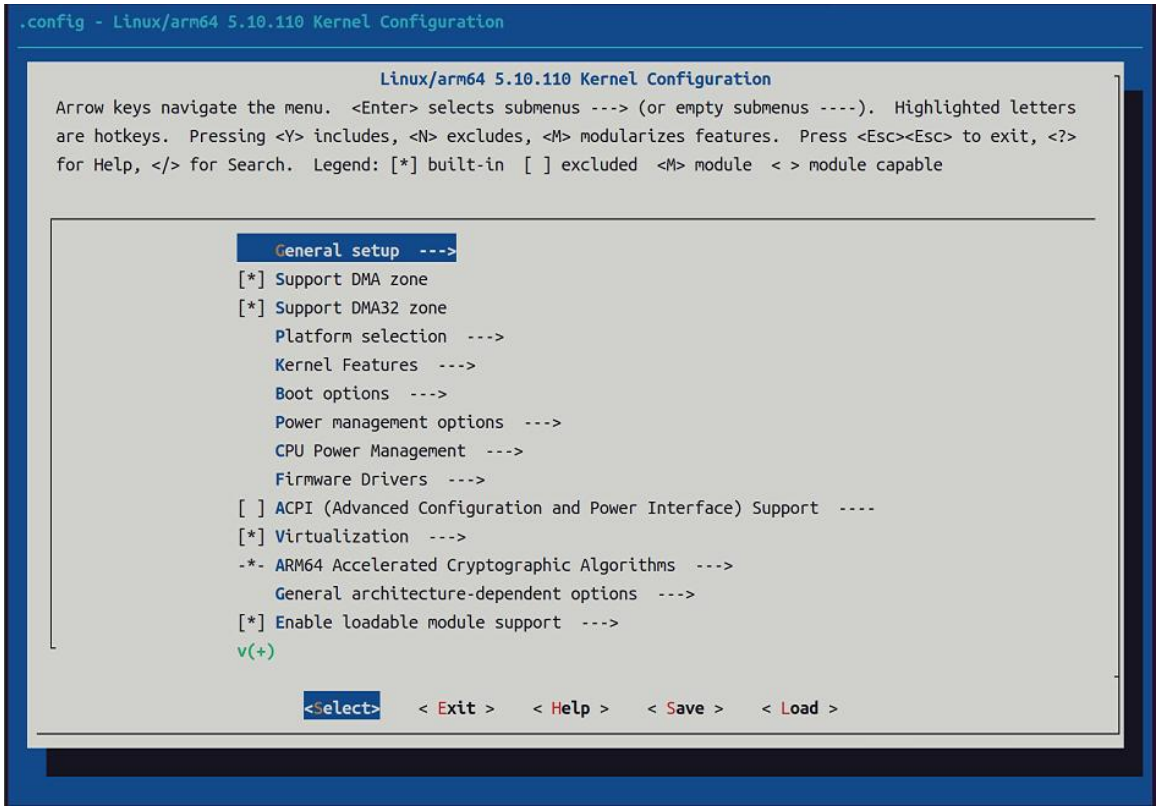
3) Next, select the model of the development board



4) Then it will prompt whether the kernel configuration interface needs to be displayed. If the kernel configuration does not need to be modified, select the first one. If the kernel configuration needs to be modified, select the second one



5) If step 4) selects the option to display the kernel configuration menu (second option), a kernel configuration interface opened through **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration, save and exit after modification, and then start compiling the kernel source code



a. If there is no need to modify the configuration options of the kernel, passing **KERNEL_CONFIGURE=no** when running the build.sh script can temporarily block the pop-up kernel configuration interface

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

b. You can also set **KERNEL_CONFIGURE=no** in the **orangepi-build/userpatches/config-default.conf** default.exe configuration file to permanently disable this feature

c. If the following error appears when compiling the kernel, it is due to the small terminal interface of Ubuntu PC, which causes the **make menuconfig** interface to not display. Please set the terminal of Ubuntu PC to its maximum size and run the build.sh script again

```
  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) The following is a partial explanation of the information prompted when compiling kernel source code

a. Version of Linux kernel source code

[o.k.] Compiling current kernel [**5.10.160**]

b. The version of the cross compilation toolchain used

[o.k.] Compiler version [**aarch64-none-linux-gnu-gcc 11.2.1**]

c. The default configuration file used by the kernel and the path where it is stored

[o.k.] Using kernel config file [**config/kernel/linux-rockchip-rk3588-legacy.config**]

d. The path of the compiled kernel related deb package

[o.k.] Target directory [**orangepi-build/output/debs/**]

e. The package name of the compiled kernel image deb package

[o.k.] File name [**linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb**]

f. Compilation time used

[o.k.] Runtime [**5 min**]

g. Finally, the compilation command for the kernel selected last time will be displayed. The following command can be used to start compiling the kernel source code without selecting it through the graphical interface

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepi5ultra BRANCH=legacy BUILD_OPT=kernel KERNEL_CONFIGURE=no**]

7) View the compiled kernel related deb packages

a. **linux-dtb-legacy-rockchip-rk3588_1.0.2_arm64.deb** Contains dtb files used by the kernel

b. **linux-headers-legacy-rockchip-rk3588_1.0.2_arm64.deb** Contains kernel header files

c. **linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb** Contains kernel images and kernel

modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-legacy-rockchip-rk3588_1.0.2_arm64.deb
output/debs/linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb
output/debs/linux-headers-legacy-rockchip-rk3588_1.0.2_arm64.deb
```

8) The deb package of the generated Linux image contains the following files

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-rockchip-rk3588_1.0.2_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb  usr
```

b. The decompressed file is shown below

```
test@test:~/orangepi-build/output/debs/test$ tree -L 2
.
├── boot
│   ├── config-5.10.160-rockchip-rk3588
│   ├── System.map-5.10.160-rockchip-rk3588
│   └── vmlinuz-5.10.160-rockchip-rk3588
├── etc
│   └── kernel
├── lib
│   └── modules
├── linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb
└── usr
    ├── lib
    └── share
```

9) When the Orangepi build compilation system compiles the Linux kernel source code, it first synchronizes the Linux kernel source code with the GitHub server's Linux kernel

source code. Therefore, if you want to modify the Linux kernel source code, you first need to turn off the source code update function **(you need to compile the Linux kernel source code completely before turning off this function, otherwise it will prompt that the Linux kernel source code cannot be found. If it is a source code compressed package downloaded from Baidu Cloud Drive, there is no problem because the Linux source code is already cached)**. Otherwise, the modifications made will be restored. The method is as follows:

Set the IGNOREUPDATES variable to "yes" in **userpatches/config-default.conf**

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

10) If modifications have been made to the kernel, the following method can be used to update the kernel and kernel modules of the Linux system on the development board

a. Upload the compiled deb package of the Linux kernel to the Linux system on the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-legacy-rockchip-rk3588_1.0.2_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-legacy-rockchip-rk3588
```

c. Reinstall the deb package of the new Linux kernel that was just uploaded

```
root@orangepi:~# dpkg -i linux-image-legacy-rockchip-rk3588_1.0.2_arm64.deb
```

d. Then restart the development board and check if the kernel related modifications have taken effect

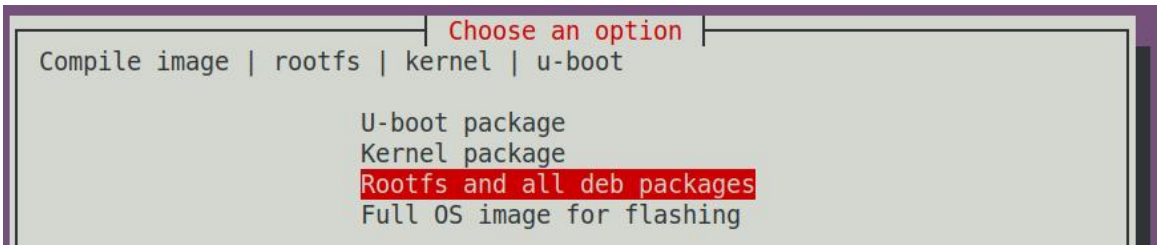
```
root@orangepi:~# reboot
```

5. 5. Compile rootfs

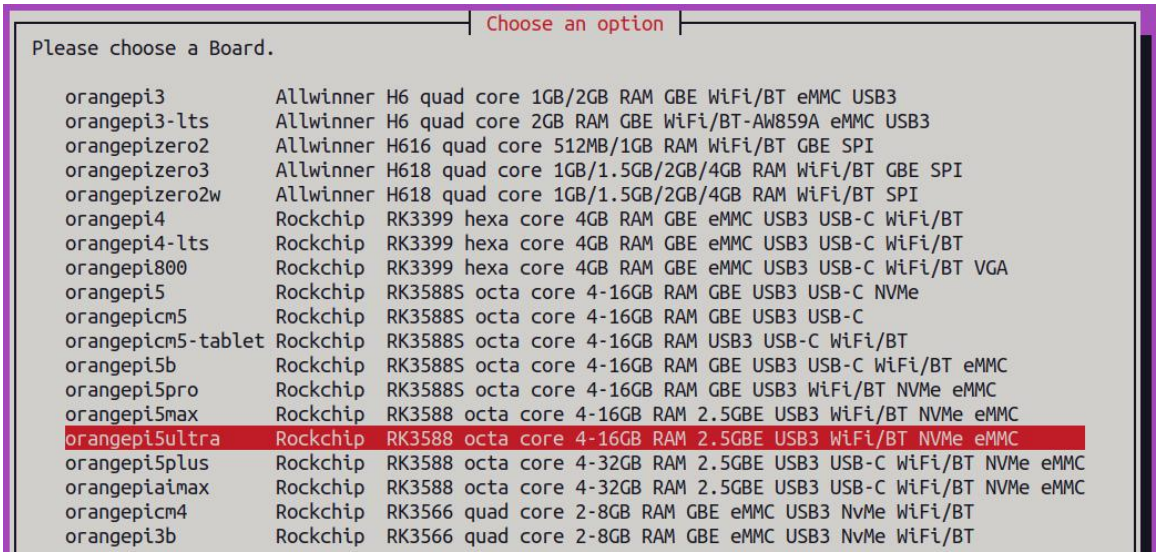
1) Run the build.sh script, remember to grant sudo privileges

```
test@test:~/orangepi-build$ sudo ./build.sh
```

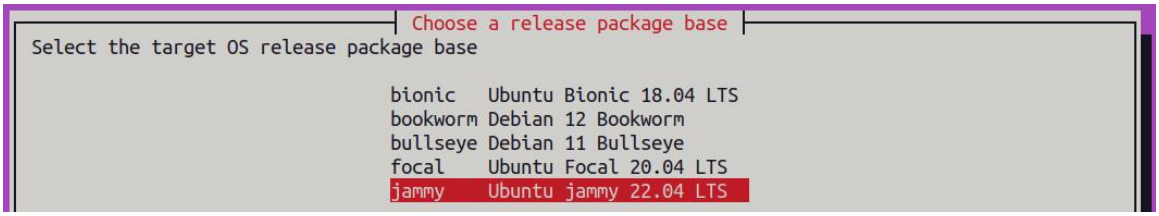
2) Select **Rootfs and all deb packages**, then press enter



3) Next, select the model of the development board



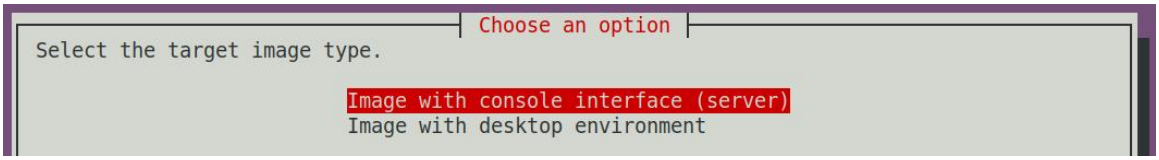
4) Then select the type of rootfs



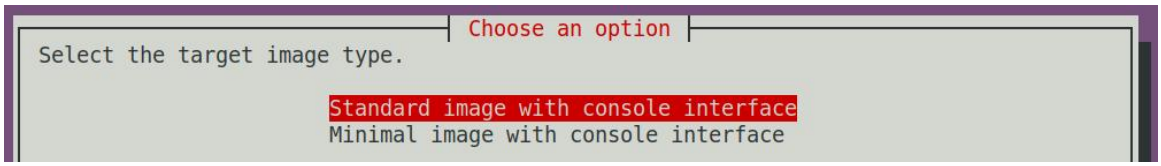
5) Then select the type of image

a. **Image with console interface (server)** Represents a server version image with a relatively small size

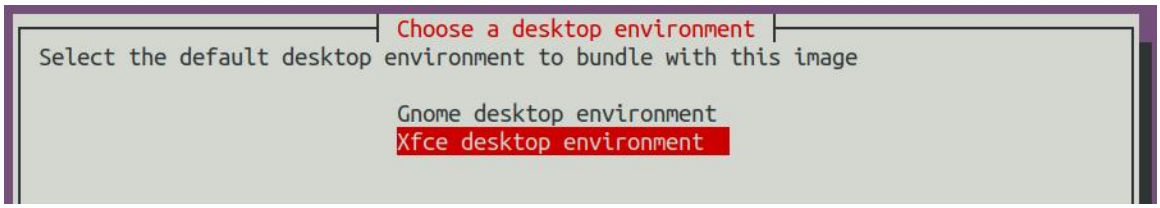
b. **Image with desktop environment** Represents a desktop image with a relatively large volume

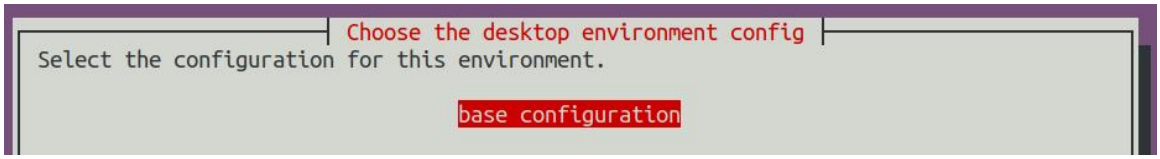


6) If you are compiling the server version image, you can also choose to compile the Standard version or the Minimal version. The Minimal version comes with much less pre installed software than the Standard version **(please do not choose the Minimal version unless you have special requirements, as many things are not pre installed by default and some features may not be available)**

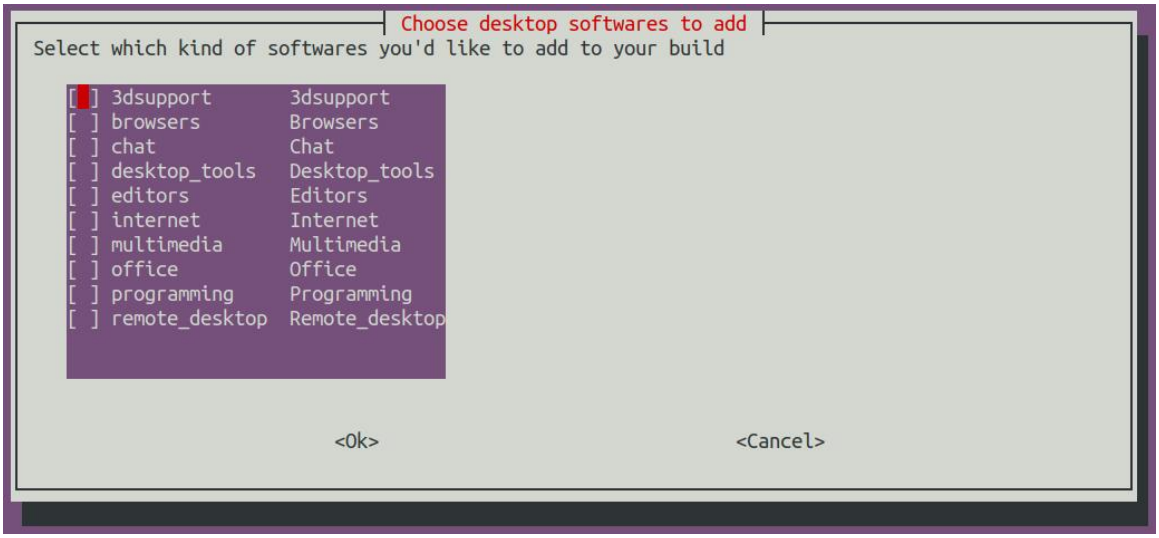


7) If compiling a desktop version of the image, you also need to choose the type of desktop environment. Currently, Ubuntu Jammy mainly maintains XFCE and Gnome desktops, Ubuntu Focal only maintains XFCE desktops, Debian Bullseye mainly maintains XFCE and KDE desktops, and Debian Bookwork mainly maintains XFCE desktops





Then you can choose additional software packages that need to be installed. Please press the enter key here to skip directly.



8) Then it will start compiling rootfs, and some of the information prompted during compilation is as follows

a. Types of rootfs

[o.k.] local not found [Creating new rootfs cache for **jammy**]

b. The storage path of the rootfs compressed file generated by compilation

[o.k.] Target directory [**external/cache/rootfs**]

c. The name of the rootfs compressed file generated by compilation

[o.k.] File name [**jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4**]

d. Compilation time used

[o.k.] Runtime [**13 min**]

9) View the compiled rootfs compressed file

a. **jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4** is a compressed file of rootfs, and the meaning of each field in the name is

a) **jammy** represents the type of Linux distribution of rootfs

b) **xfce** represents rootfs as the desktop version type, and if it is **cli**, it represents the server version type

c) **arm64** represents the architecture type of rootfs

d) **f930ff6ebbac1a72108a2e100762b18f** is an MD5 hash value generated from the package names of all software packages installed by rootfs. As long as the list of software packages installed by rootfs is not modified, this value will not change. The compilation script will use this MD5 hash value to determine

whether to recompile rootfs

b. **jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4.list** lists the package names of all the software packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4
jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4.current
jammy-xfce-arm64.f930ff6ebbac1a72108a2e100762b18f.tar.lz4.list
```

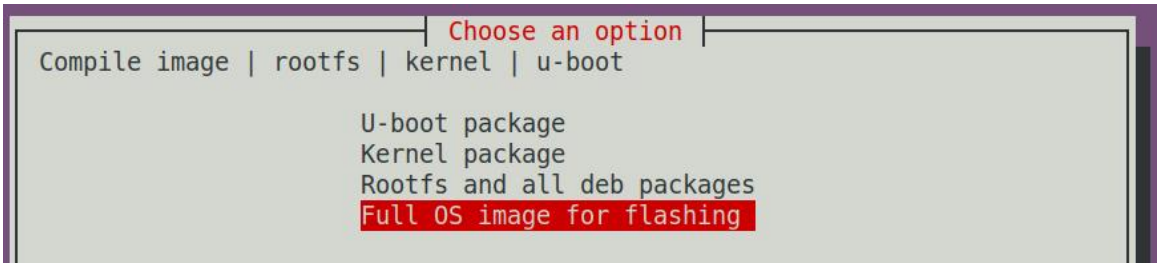
10) If the required rootfs already exist in **external/cache/rootfs**, compiling rootfs again will skip the compilation process and will not restart. When compiling the image, it will also search for available rootfs in **external/cache/rootfs**, and if so, use them directly, which can save a lot of download and compilation time.

5. 6. Compiling Linux Images

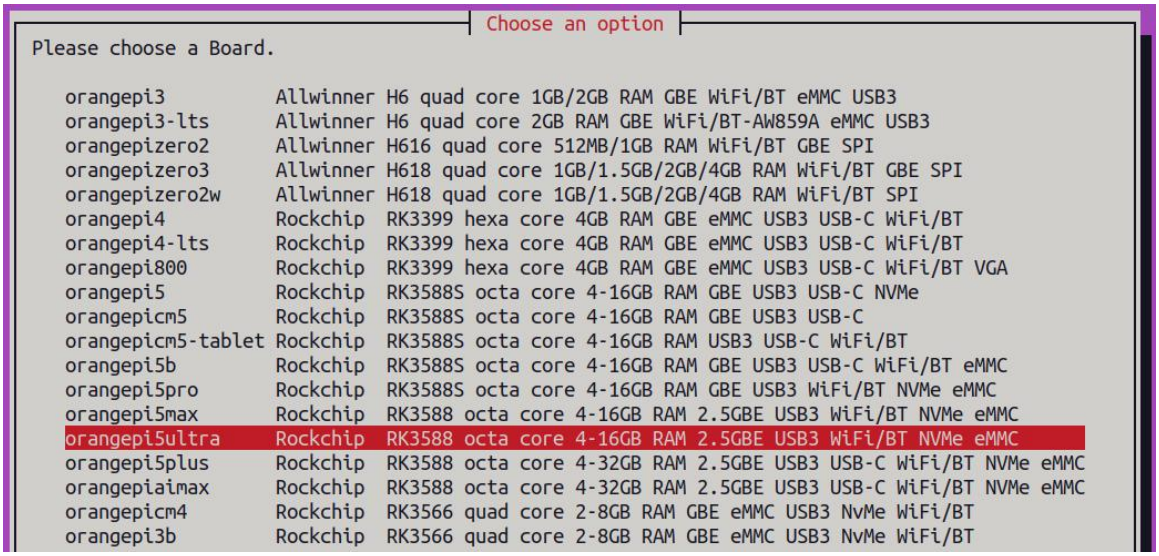
1) Run the build.sh script, remember to grant sudo privileges

```
test@test:~/orangepi-build$ sudo ./build.sh
```

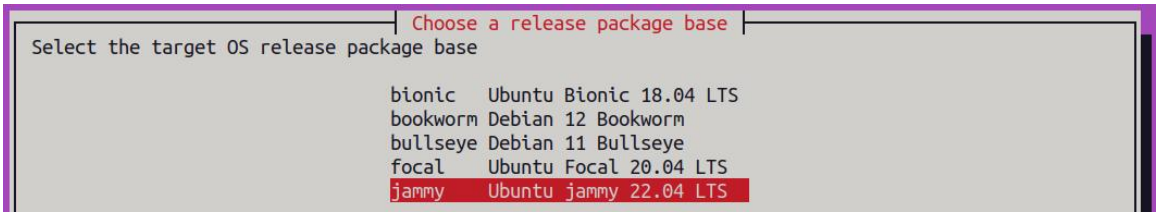
2) Select **Full OS image for flashing** and press Enter



3) Then select the model of the development board



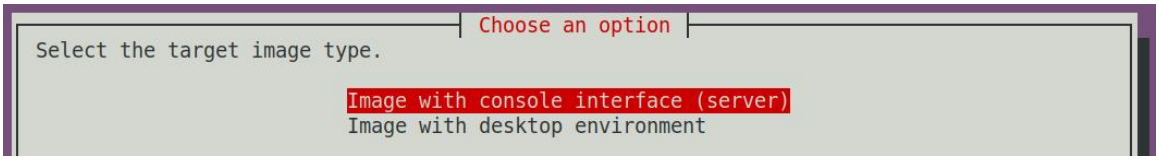
4) Then select the type of rootfs



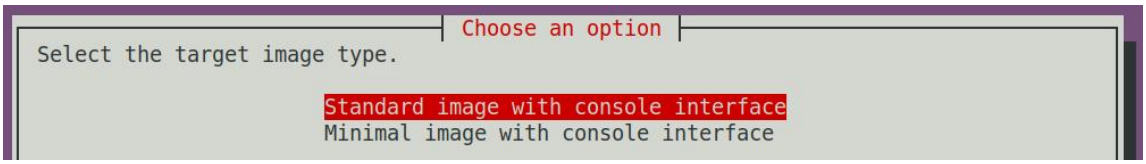
5) Then select the type of image

a. **Image with console interface (server)** Represents a server version image with a relatively small size

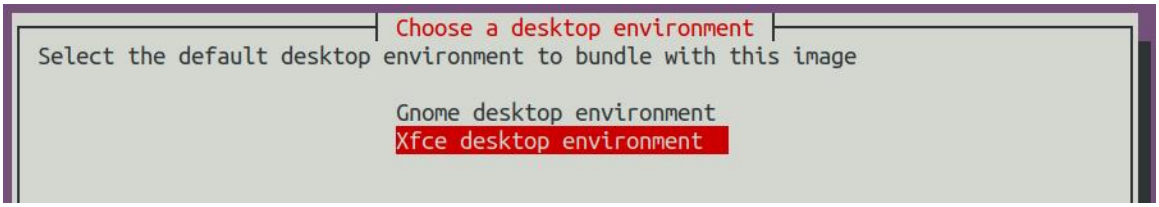
b. **Image with desktop environment** Represents a desktop image with a relatively large volume

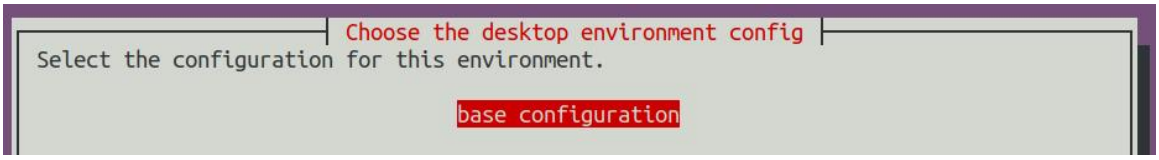


6) If you are compiling the server version image, you can also choose to compile the Standard version or the Minimal version. The Minimal version comes with much less pre installed software than the Standard version **(please do not choose the Minimal version unless you have special requirements, as many things are not pre installed by default and some features may not be available)**

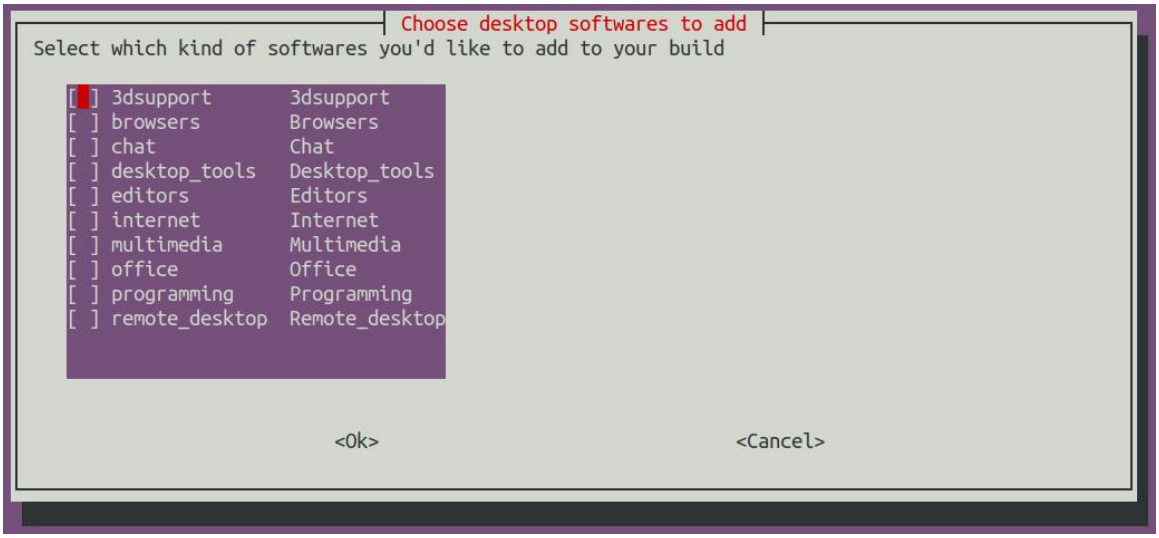


7) If compiling a desktop version of the image, you also need to choose the type of desktop environment. Currently, Ubuntu Jammy mainly maintains XFCE and Gnome desktops, Ubuntu Focal only maintains XFCE desktops, Debian Bullseye mainly maintains XFCE and KDE desktops, and Debian Bookwork mainly maintains XFCE desktops





Then you can choose additional software packages that need to be installed. Please press the enter key here to skip directly.



8) Then it will start compiling the Linux image, and the general process of compilation is

as follows

a. Initialize the compilation environment of Ubuntu PC and install the necessary software packages for the compilation process

b. Download the source code for u-boot and Linux kernel (if cached, only update the code)

c. Compile the u-boot source code and generate the deb package for u-boot

d. Compile Linux source code and generate deb packages related to Linux

e. Creating a deb package for Linux firmware

f. Create a deb package for the orangepi config tool

g. Create deb packages that support board level support

h. If compiling the desktop version image, desktop related deb packages will also be created

i. Check if rootfs have been cached. If not, create a new rootfs. If cached, decompress and use it directly

j. Install the deb package generated earlier into rootfs

k. Make specific settings for different development boards and types of images, such as pre installing additional software packages, modifying system configurations, etc

l. Then create an image file and format the partition, with the default type being ext4

m. Copy the configured rootfs to the partition of the image again

n. Then update initramfs

o. Finally, write the bin file of u-boot to the image using the dd command

9) After compiling the image, the following message will be prompted

a. The storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/orangepi5ultra_1.0.2_debian_bullseye_desktop_xfce_linux5.10.160/
orangepi5ultra_1.0.2_debian_bullseye_desktop_xfce_linux5.10.160.img ]
```

b. Compilation time used

```
[ o.k. ] Runtime [ 19 min ]
```

c. The command to repeatedly compile the image can be used to start compiling the image without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepi5ultra
BRANCH=legacy BUILD_OPT=image RELEASE=bullseye BUILD_MINIMAL=no
BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

6. Instructions for using Android 13 system

6.1. Supported Android versions

Android version	Kernel version
Android 13	Linux5.10

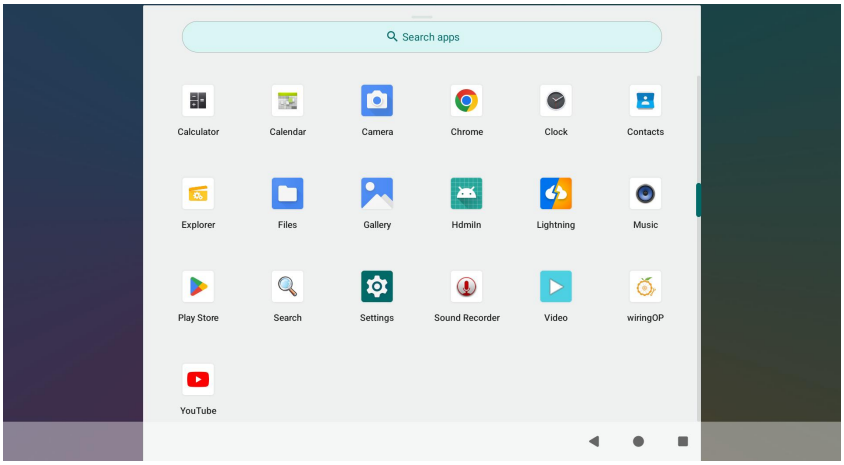
6.2. Adaptation of Android Features

function	Android 13
HDMI TX Video	OK
HDMI TX Audio	OK
HDMI RX Video	OK
HDMI RX Audio	OK
USB2.0x2	OK
USB3.0x2	OK
2.5G Ethernet port	OK
Network port status light	OK
WIFI	OK
Bluetooth	OK
Debug UART	OK
RTC chip	OK
FAN fan interface	OK
EMMC extension interface	OK
GPIO（40pin）	OK
UART（40pin）	OK
SPI（40pin）	OK
I2C（40pin）	OK
PWM（40pin）	OK
TF card startup	OK

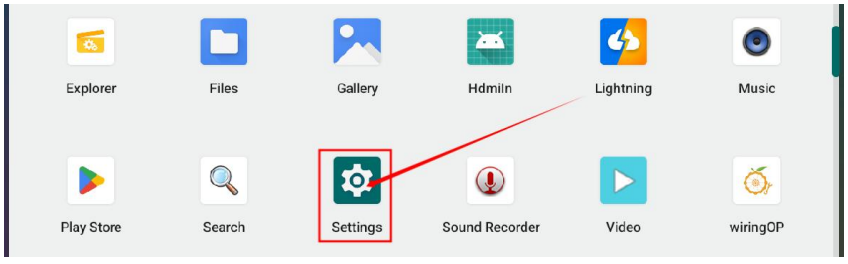
OV13850 camera	**OK**
OV13855 camera	**OK**
SPI+NVME startup	**OK**
LCD	**OK**
MIC	**OK**
Headphone playback	**OK**
Earphone recording	**OK**
Tri color LED light	**OK**
GPU	**OK**
NPU	**OK**
VPU	**OK**
Power on/off button	**OK**

6. 3. WIFI connection testing method

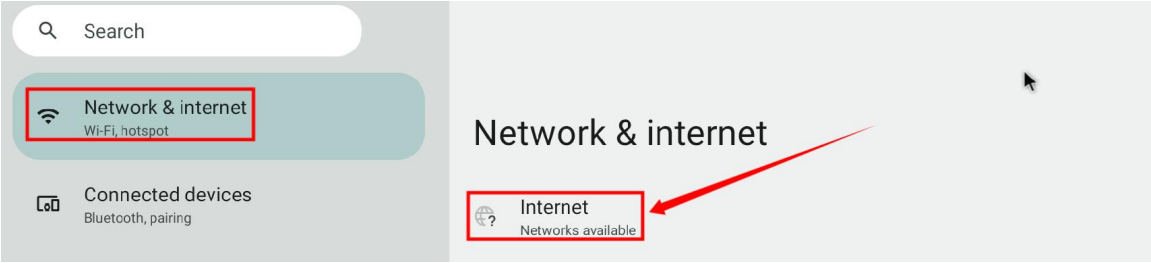
1) First, hold down the left mouse button on the desktop and drag up to open the application list.



2) Then open **Setting**。



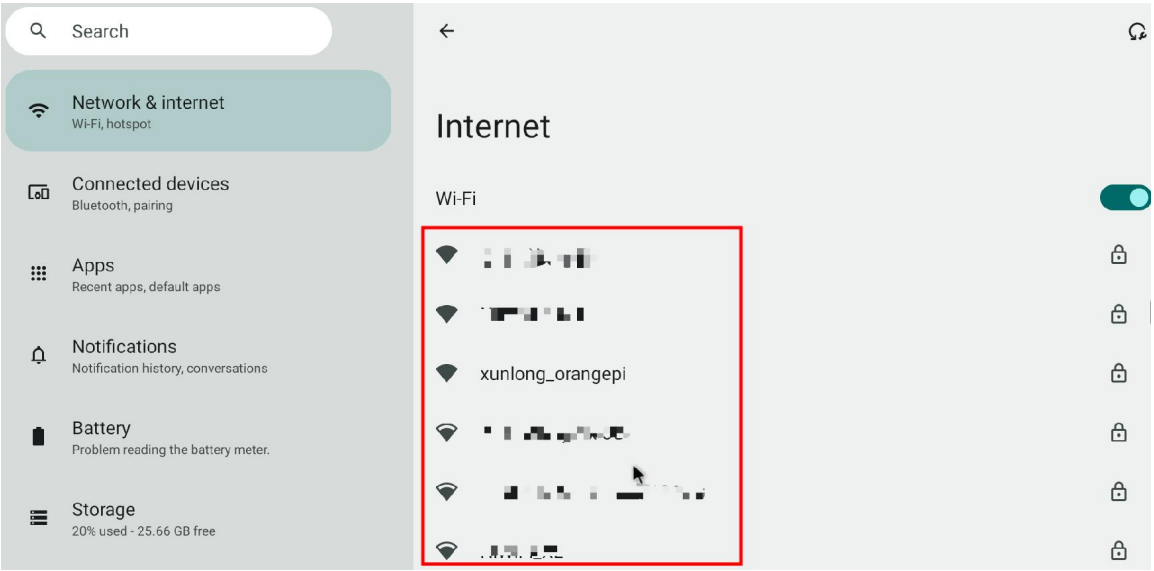
3) Then select the **Internet** option in **Network & internet**.



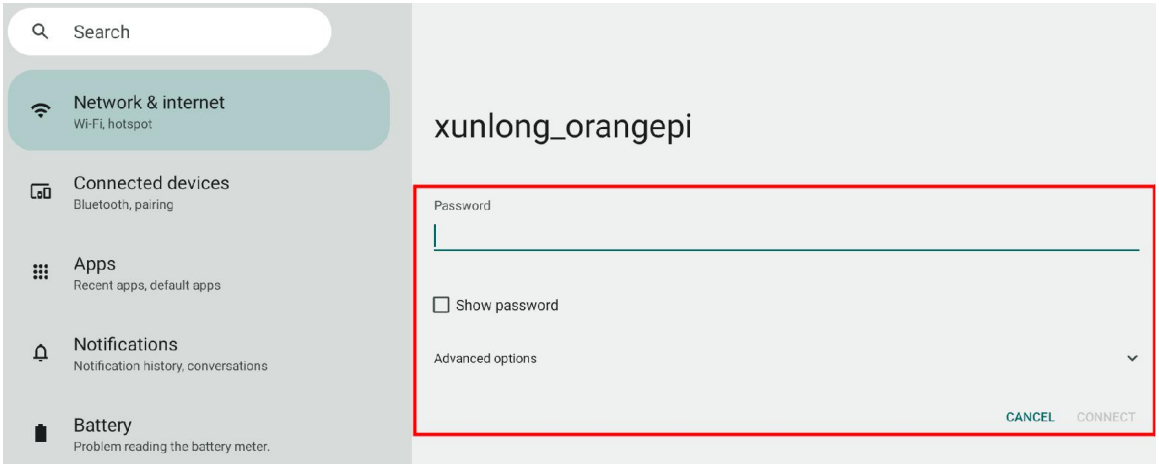
4) Then turn on the **Wi-Fi** switch.



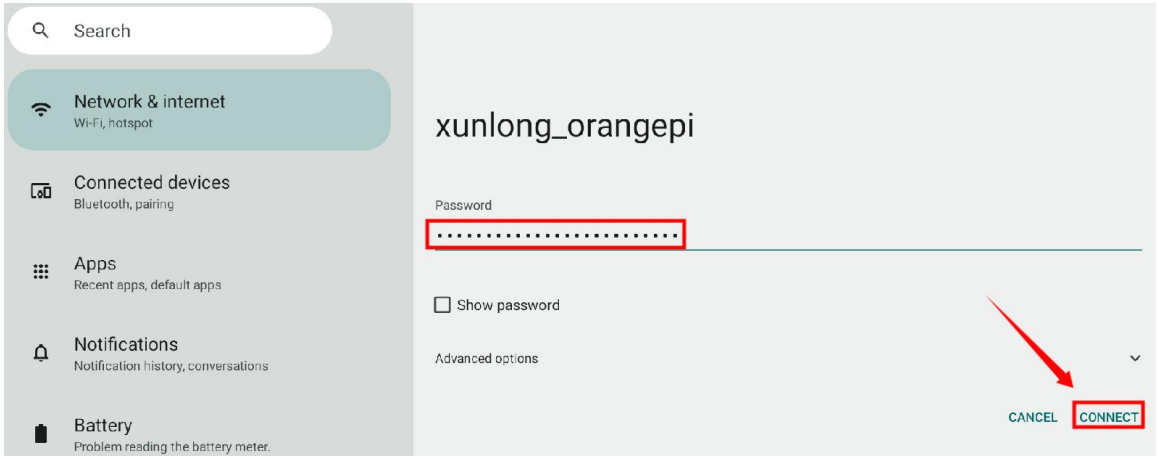
5) If everything is normal after turning on **Wi-Fi**, you can scan nearby Wi-Fi hotspots.



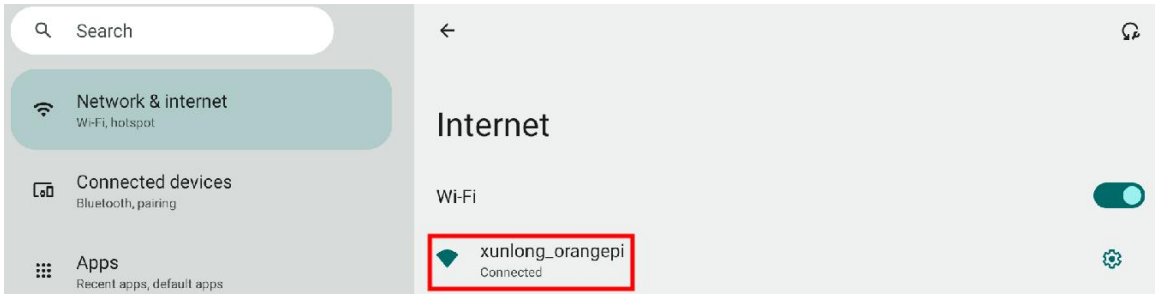
6) Then select the Wi Fi you want to connect to, and the password input interface shown in the following figure will pop up.



7) Then use the keyboard to enter the password corresponding to Wi-Fi, and click **CONNECT** with the mouse to start connecting to Wi-Fi.

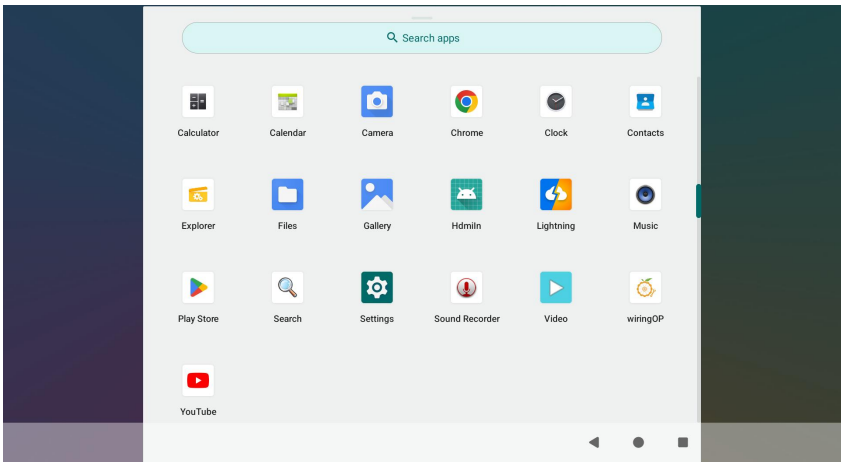


8) The display after successful Wi-Fi connection is shown in the following figure:

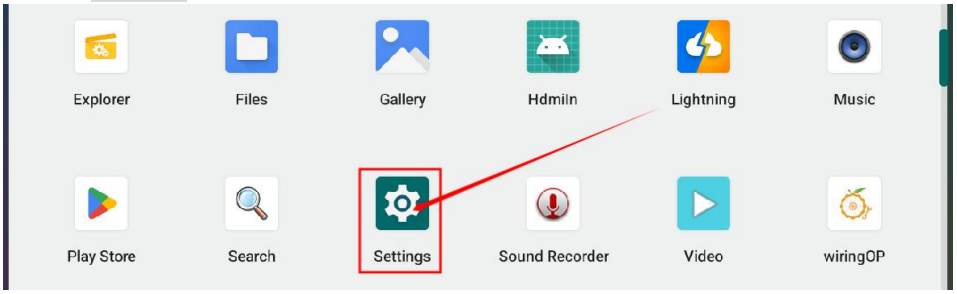


6. 4. Bluetooth testing method

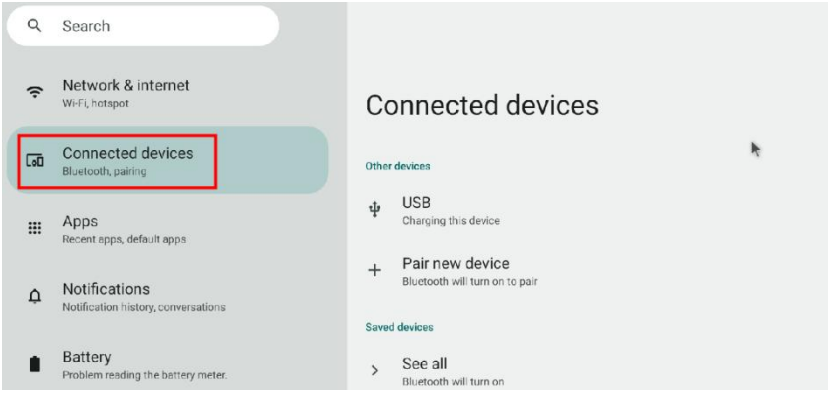
1) First, hold down the left mouse button on the desktop and drag up to open the application list.



2) Then enter **Setting**。

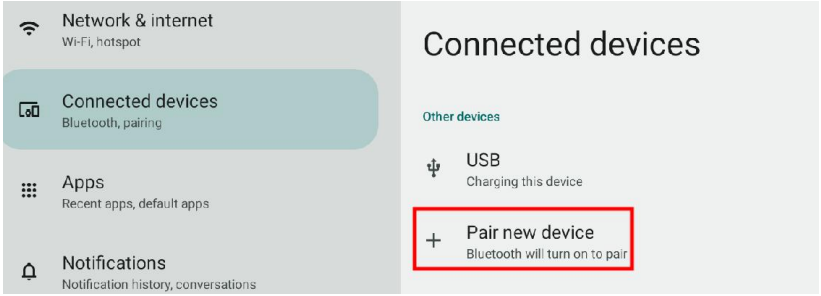


3) Then select **Connected devices**。

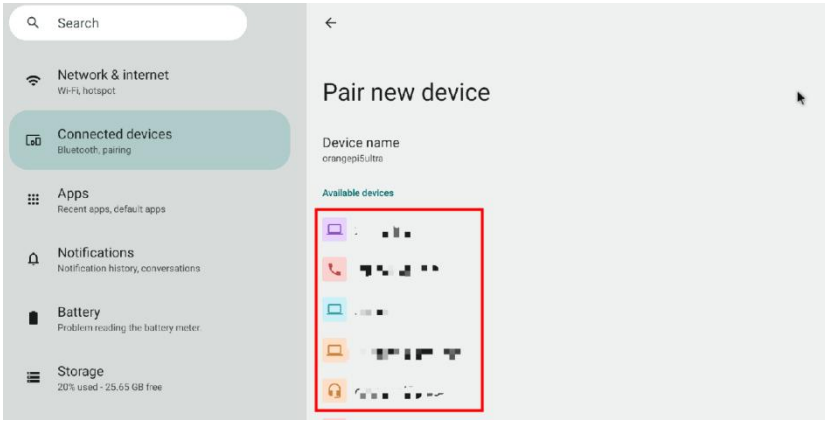


4) Then click **Pair new device** to turn on Bluetooth and start scanning for surrounding

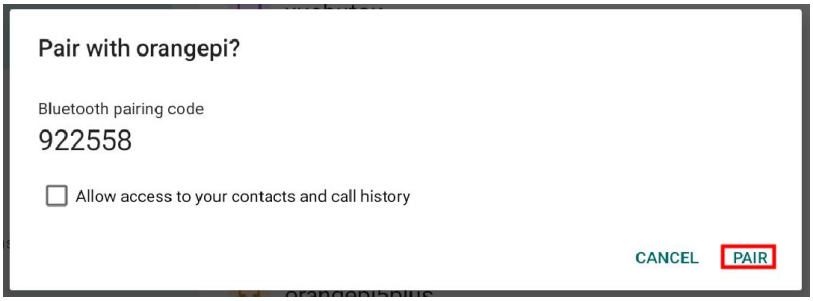
Bluetooth devices.



5) The Bluetooth devices found will be displayed under **Available devices**.



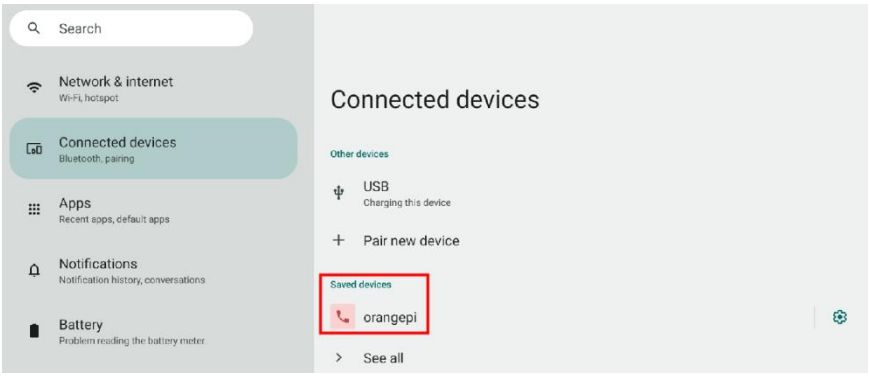
6) Then click on the Bluetooth device you want to connect to start pairing. When the interface below pops up, please use the mouse to select the **Pair** option.



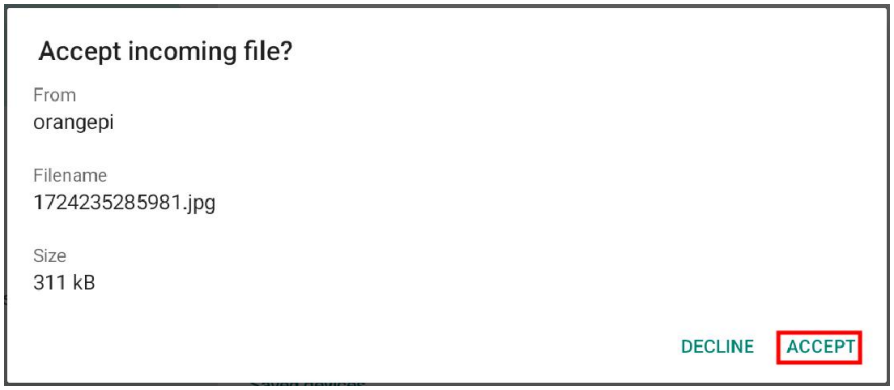
7) The test here is the configuration process of the development board and Android phone Bluetooth. At this time, the confirmation interface below will pop up on the phone, and clicking the pairing button on the phone will start the pairing process.



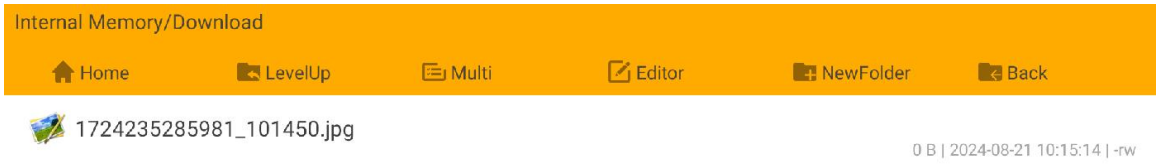
8) After pairing, you can see the paired Bluetooth device as shown in the following figure.



9) At this point, you can use your phone's Bluetooth to send an image to the development board. After sending, you can see the confirmation interface below in the Android system of the development board, and then click **Accept** to start receiving the image sent by your phone.



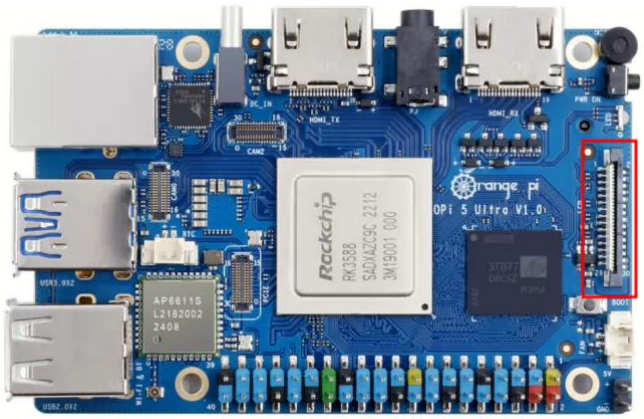
10) The images received by the Android system Bluetooth on the development board can be viewed by opening the **Download** directory in the file manager.



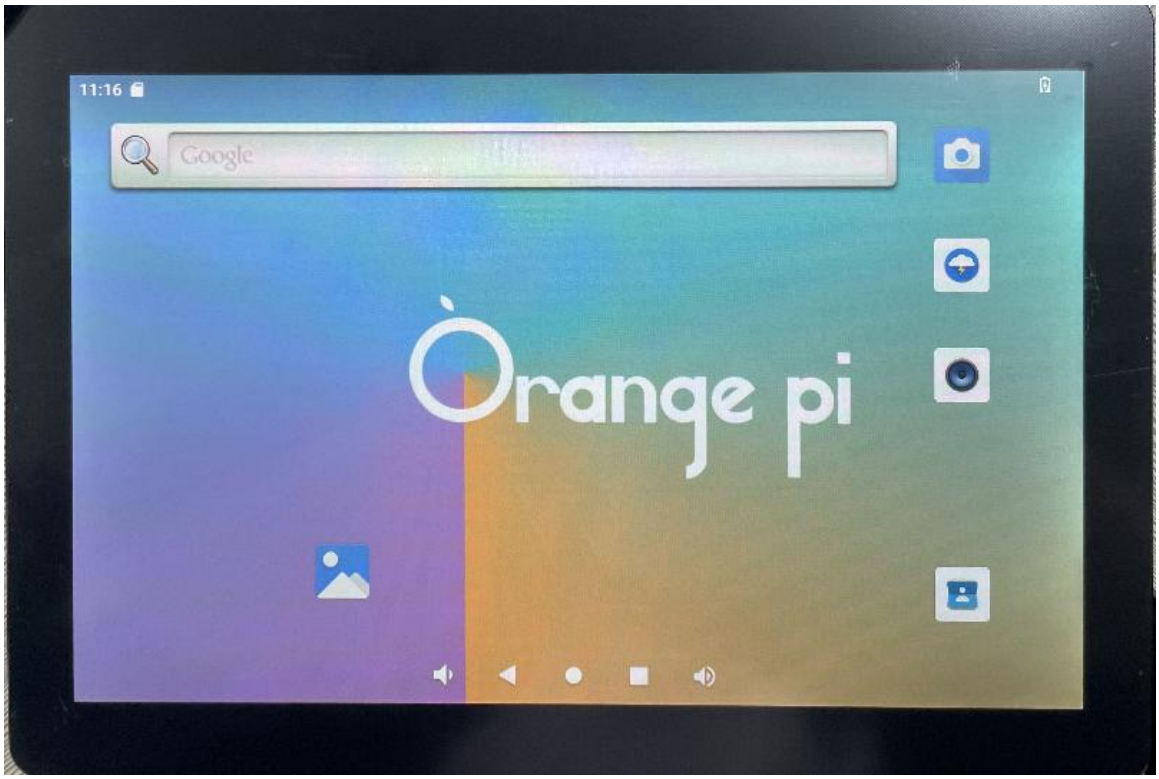
6. 5. 10.1 inch MIPI screen usage method

1) Firstly, the screen needs to be assembled. Please refer to the assembly **method for a 10.1-inch MIPI screen**.

2) The position of the mipi LCD screen interface on the development board is shown in the following figure:



3) Connect the assembled screen to the LCD interface of the development board, connect the Type-C power supply to the board, and power it on. After the system starts up, you can see the screen display as shown in the following figure:



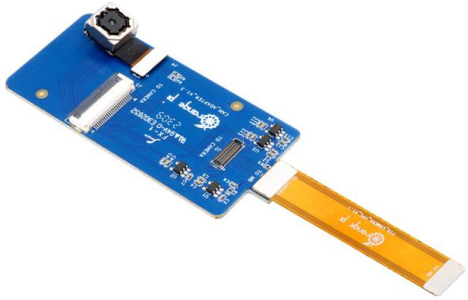
6.6. Testing Method for OV13850 and OV13855 MIPI Cameras

At present, the development board supports two MIPI cameras, OV13850 and OV13855. The specific images are shown below:

a. OV13850 camera with 13 million MIPI interface



b. OV13855 camera with 13 million MIPI interface

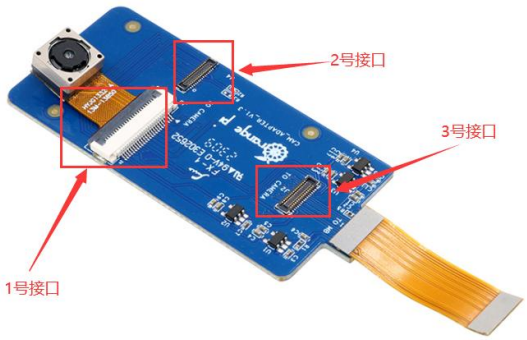


The adapter board and FPC cable used for OV13850 and OV13855 cameras are the same, except that the two cameras are connected to the adapter board at different positions. The FPC cable is shown in the following figure. Please note that the FPC cable has a direction. The end marked as **TO MB** needs to be plugged into the camera interface of the development board, and the end marked as **TO CAMERA** needs to be plugged into the camera adapter board.



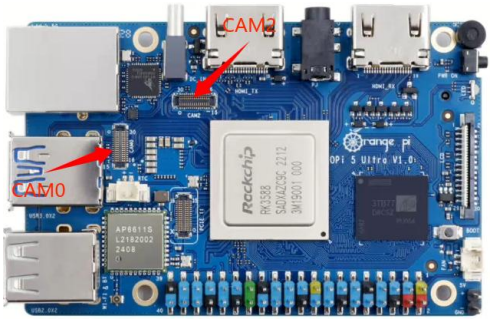
There are a total of 3 camera interfaces on the camera adapter board, and only one can be connected for use at a time, as shown in the following figure. Among them:

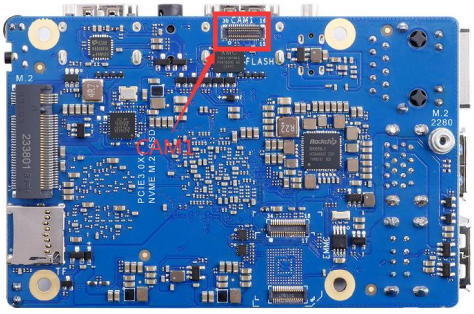
a. **Connect interface 1 to OV13850 camera**
b. **Connect interface 2 to OV13855 camera**
c. Interface 3 is not in use, just ignore it



There are a total of 3 camera interfaces on the Orange Pi 5 Ultra development board. In the Android system, only Cam0 and Cam1 are enabled by default. We define the

positions of Cam0, Cam1, and Cam2 as shown in the following figure:

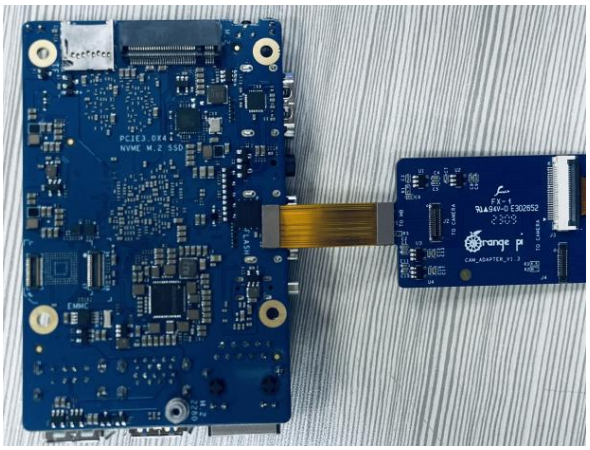




The method of inserting the camera into the Cam0 interface of the development board is as follows:

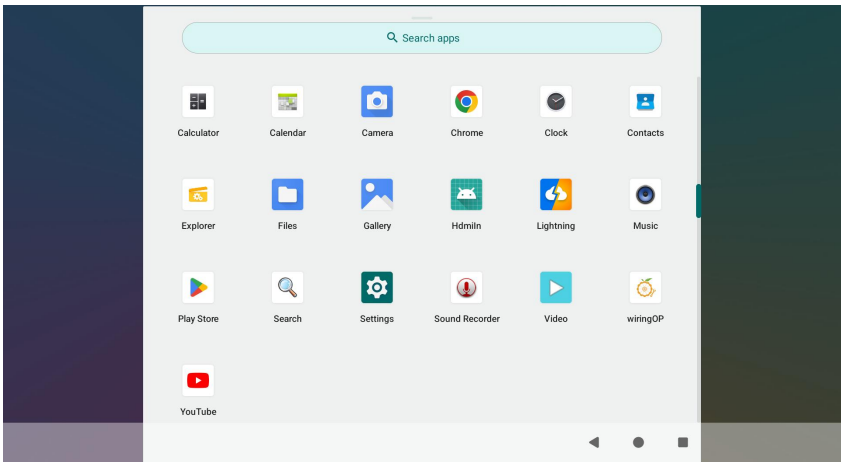


The method of inserting the camera into the Cam1 interface of the development board is as follows:

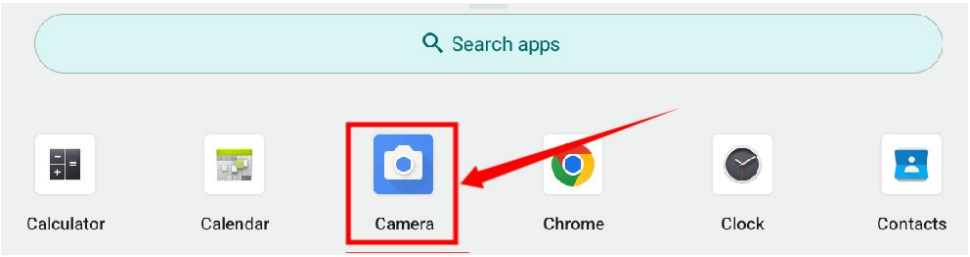


After connecting the camera to the development board, we can use the following method to test the camera:

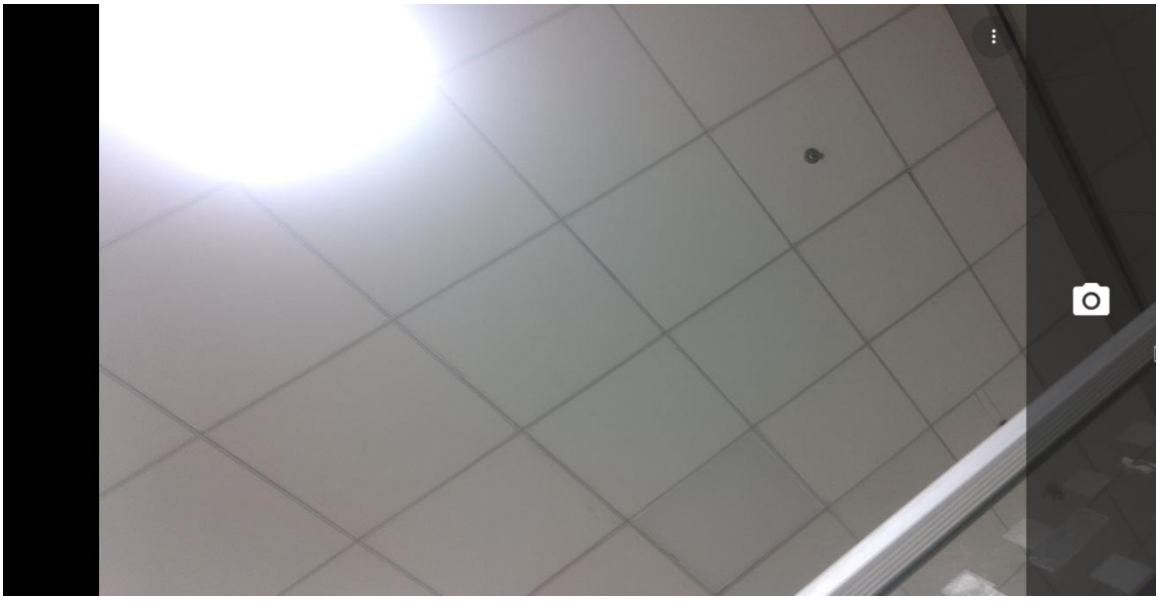
a. First, hold down the left mouse button on the desktop and drag up to open the application list.



b. Then open the camera app.



c. Then you can see the preview screen of the camera.



In addition to a single camera, we can also connect two cameras simultaneously. After connecting the dual cameras, as in the previous steps, open the camera app to see the image of one of the cameras.

The method to switch to another camera is:

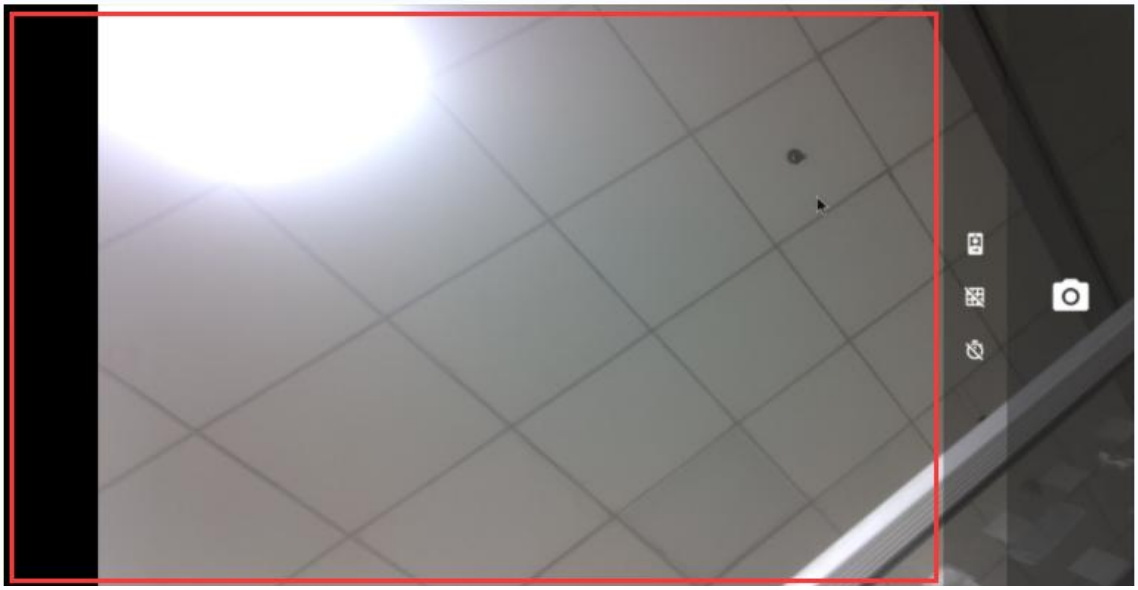
a. First, click on these three points in the upper right corner.



b. Then click on the position shown in the picture to switch cameras.



Press and hold the mouse in the area shown in the red box in the camera app, and then drag it to the right to bring up the switch interface between taking photos and filming.



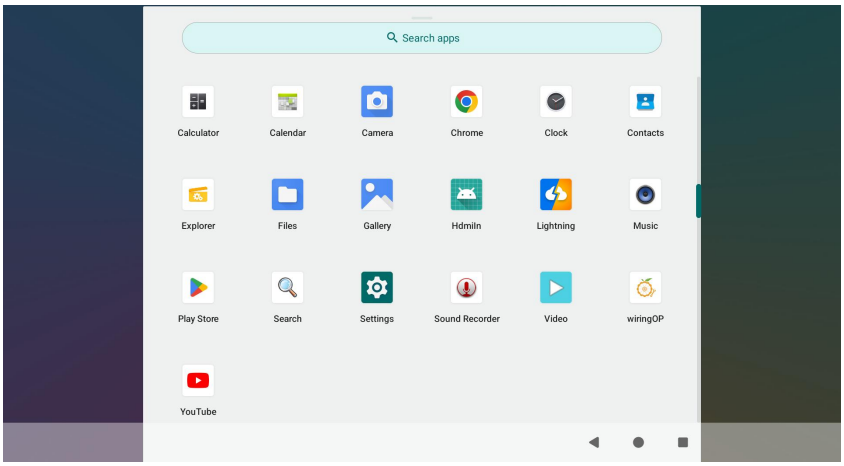
The interface for switching between photography and videography is shown below. Click on **Video** to switch to recording mode.



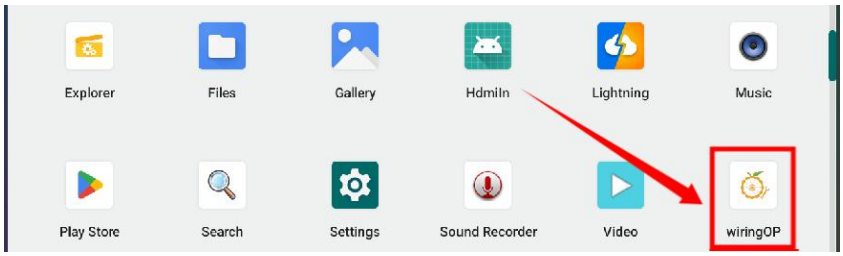
6. 7. 40pin interface GPIO, UART, SPI, and PWM testing

6. 7. 1. 40pin GPIO port test

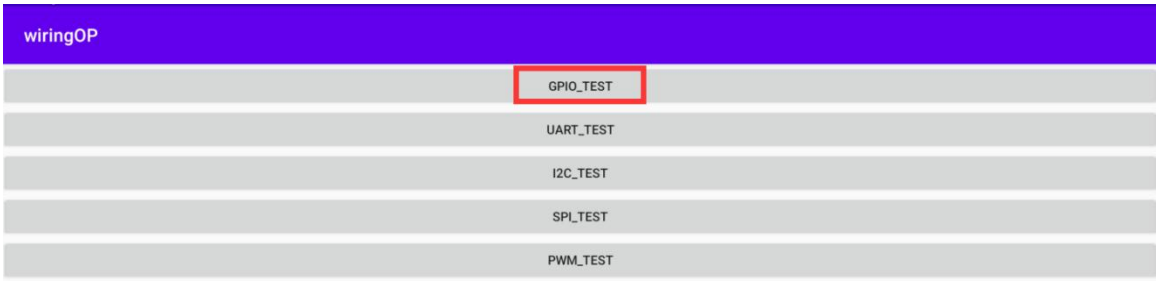
1) First, hold down the left mouse button on the desktop and drag up to open the application list.



2) Then open the WiringoP app.



3) The main interface of WiringOP is shown in the following figure, and then click the **GPIO_TEST** button to open the GPIO testing interface.

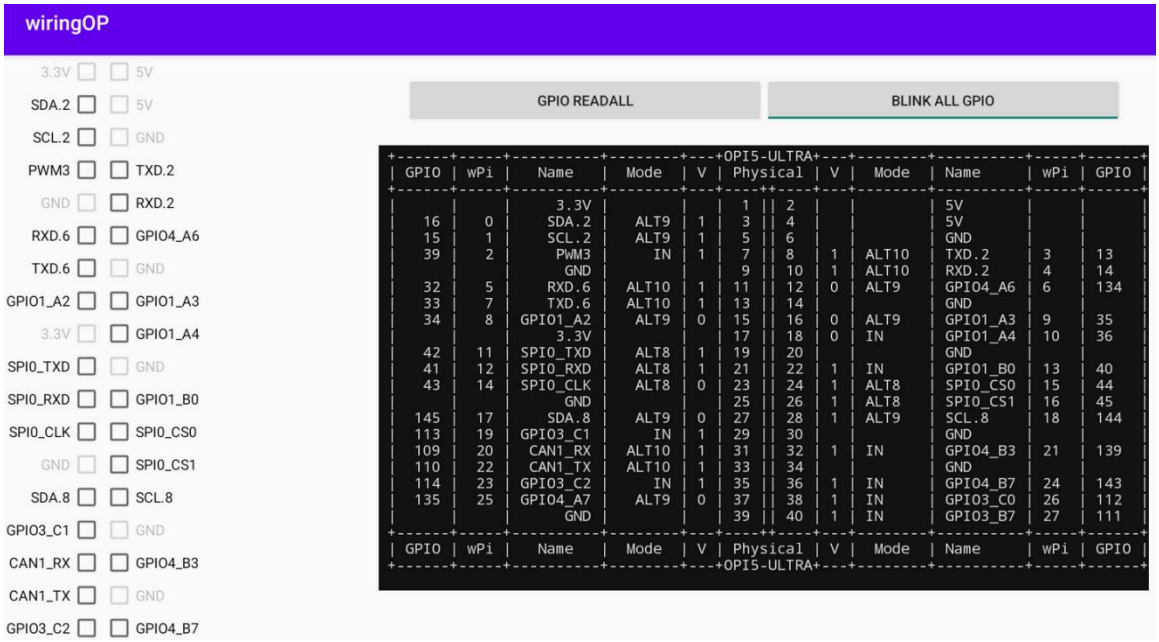


4) The GPIO testing interface is shown in the following figure, where the two rows of **CheckBox** buttons on the left correspond one-to-one with the 40 pin pins. When the **CheckBox** button is selected, the corresponding GPIO pin will be set to **OUT** mode and the pin level will be set to high level; When unchecked, the GPIO pin level is set to low level; When clicking the **GPIO READALL** button on the right, information such as wPi number, GPIO mode, and pin level can be obtained;

When the **BLINK ALL GPIO** button is clicked, the program will control 28 GPIO ports to continuously switch between high and low levels.



5) Click the **GPIO READALL** button to access detailed information for all pins, as shown in the following figure:

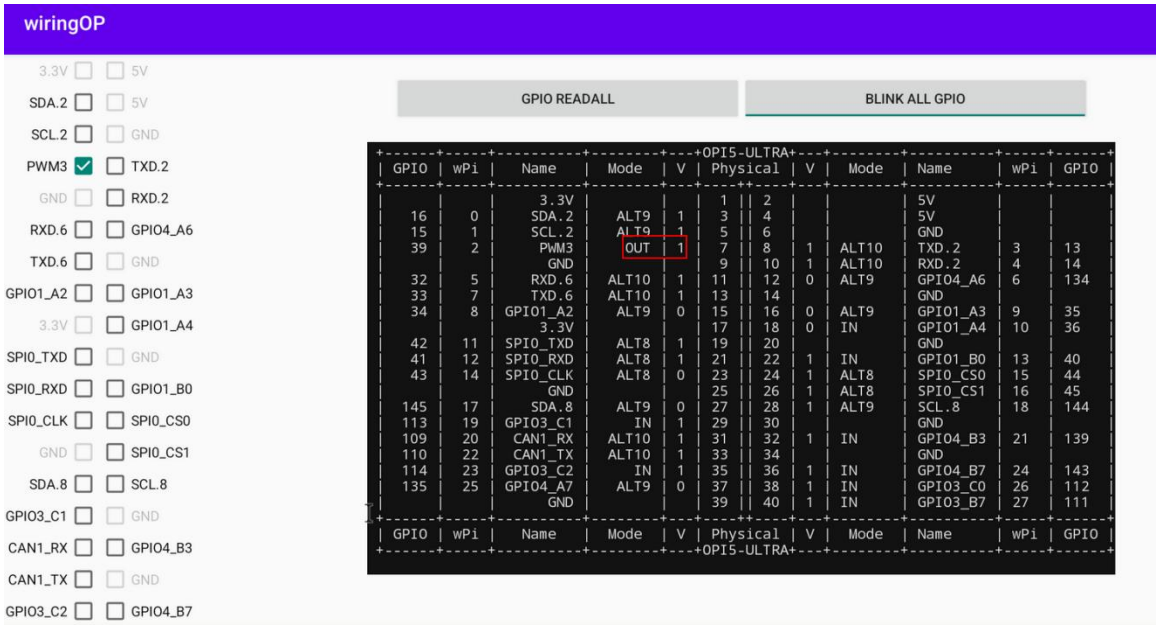


6) There are a total of 28 GPIO ports available in the 40 pin development board. Taking pin 7- corresponding to GPIO as GPO1_A7- corresponding to wPi serial number 2- as an example, we will demonstrate how to set the high and low levels of GPIO ports. Firstly, click on the **CheckBox** button corresponding to pin 7. When the button is selected, pin 7 will be set to 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 has been successfully set.



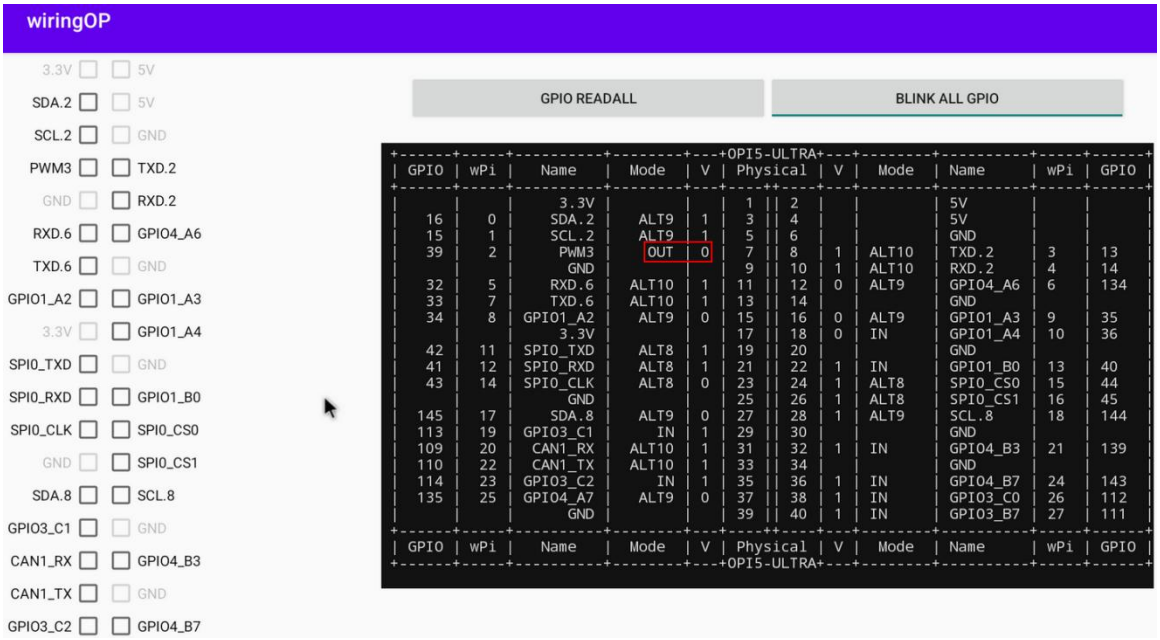
7) Then click the **GPIO READALL** button to see that the current mode of pin 7 is **OUT** and the pin level is high.



8) Click the **CheckBox** button in the following image again to uncheck the status. Pin 7 will be set to 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 has been successfully set



9) Then click the **GPIO READALL** button to see that the current mode of pin 7 is OUT and the pin level is low.

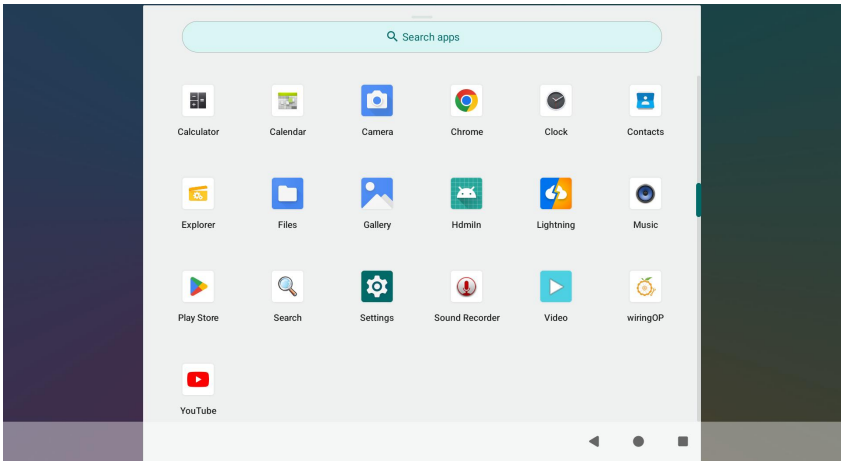


6. 7. 2. 40pin UART Test

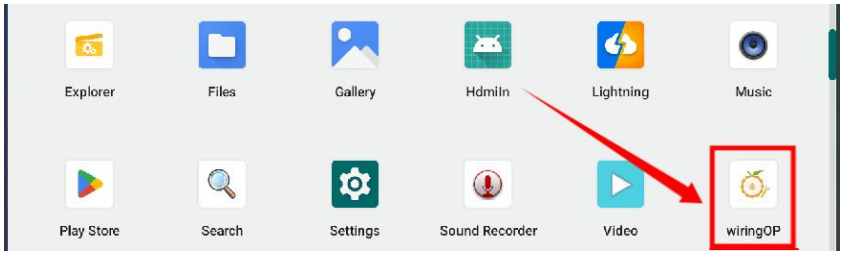
1) In the Android system, two serial ports, UART3 and UART6, are enabled by default. The corresponding pins in the 40 pin configuration are shown in the table below, and the corresponding device nodes are**/dev/ttyS3** and**/dev/ttyS6**.

UART bus	RX corresponds to 40 pins	TX corresponds to 40 pins
UART3	Pin 33	Pin 31
UART6	Pin 11	Pin 13

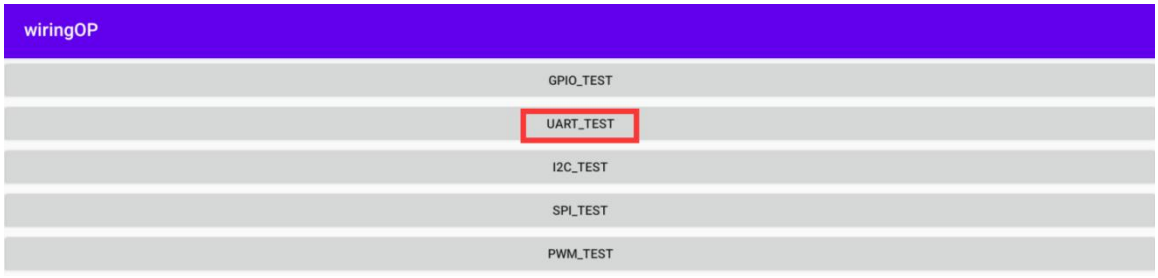
2) First, hold down the left mouse button on the desktop and drag up to open the application list.



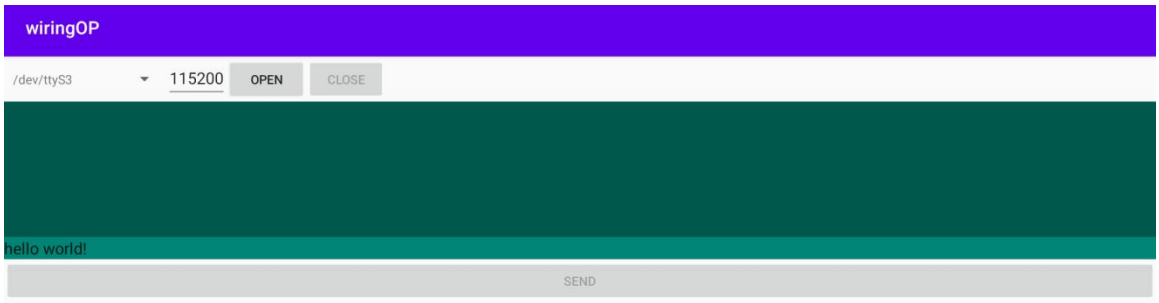
3) Then open the WiringoP app.



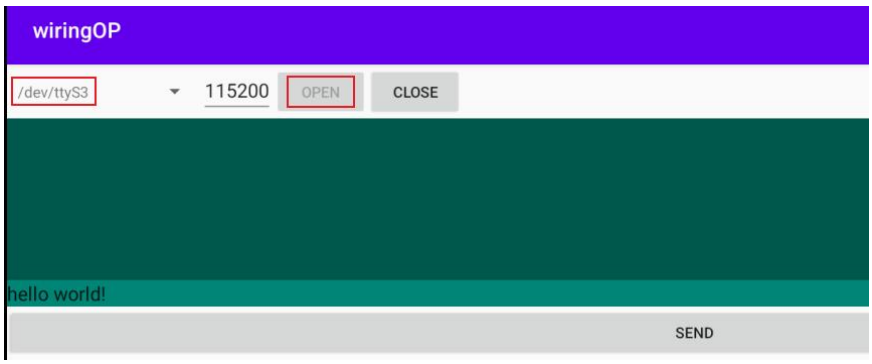
4) The main interface of WiringoP APP is shown in the following figure, and then click the **UART_TEST** button to open the UART test interface.



5) The serial port testing interface of WiringOP APP is shown in the following figure:

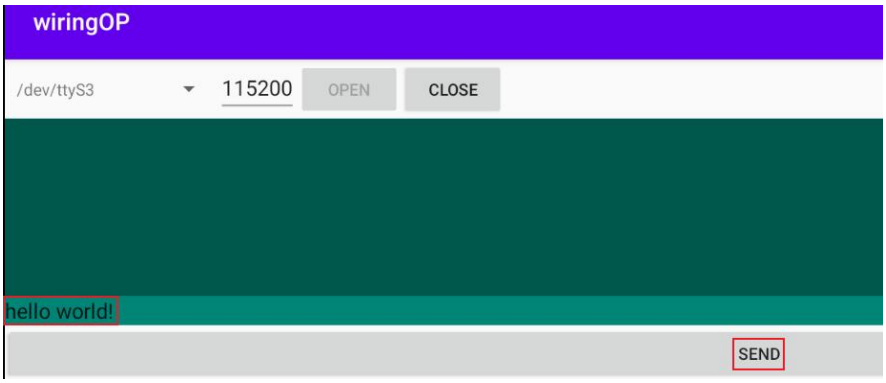


6) Next, enter the desired baud rate in the editing box, and then click the **OPEN** button to open the**/dev/ttyS3** node. After successful opening, the **OPEN** button will become unselectable, while the **CLOSE** and **SEND** buttons will become selectable.

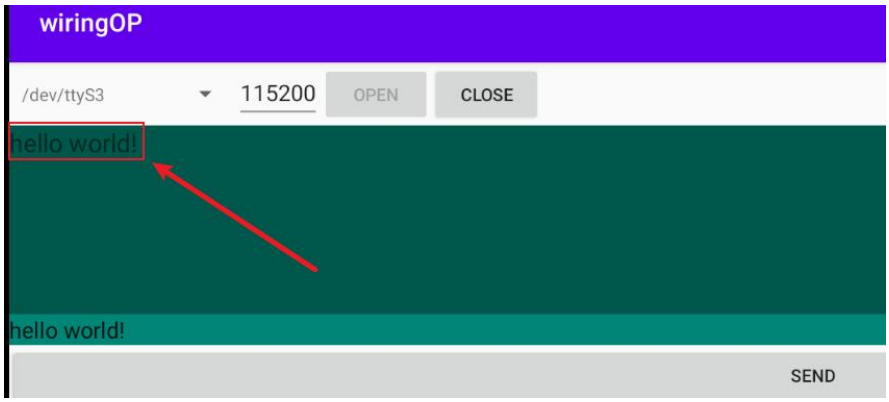


7) Then use DuPont wire to short-circuit the RXD and TXD pins of UART 3.

8) Then you can enter a character in the send edit box below and click the **SEND** button to start sending.



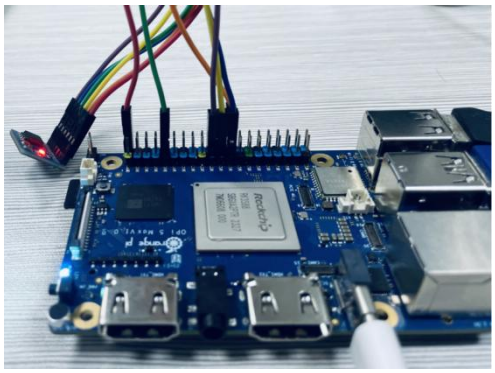
9) If everything is normal, the received string will be displayed in the receiving box.



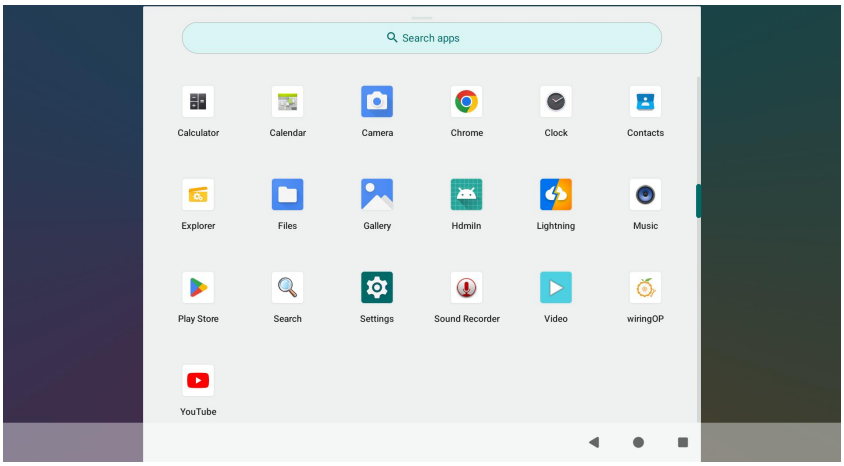
6. 7. 3. 40pin SPI test

1) The **SPI0** group of SPI buses is enabled by default in Android.

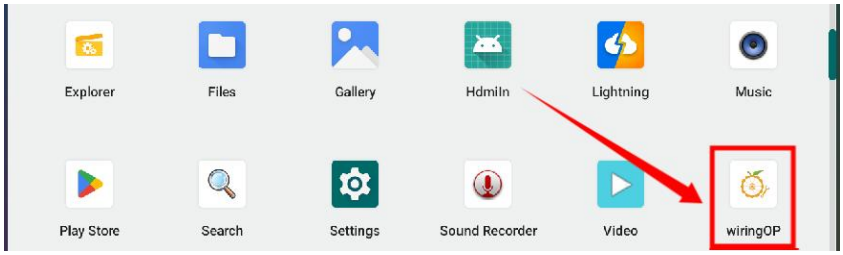
2) Here, the w25q64 module is used to test the SPI interface. First, connect the w25q64 device to the SPI0 interface.



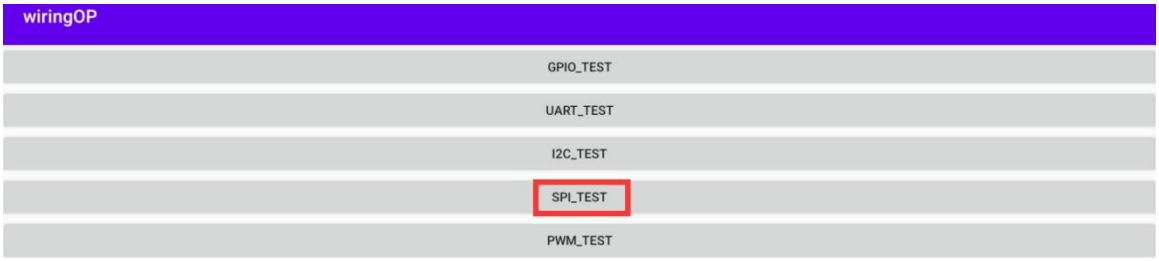
3) First, hold down the left mouse button on the desktop and drag up to open the application list.



4) Then open the WiringoP app.



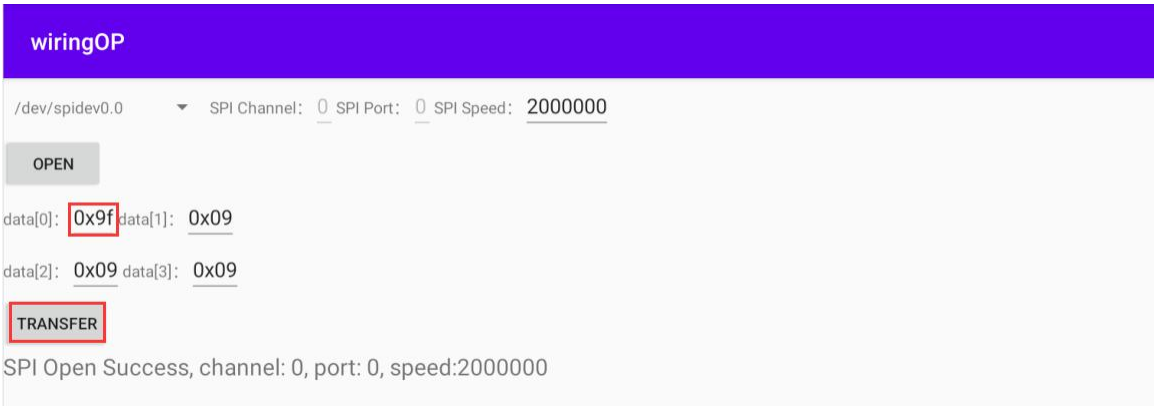
5) The main interface of WiringOP APP is displayed as shown in the following figure. Click the SPI_TEST button to open the SPI testing interface.



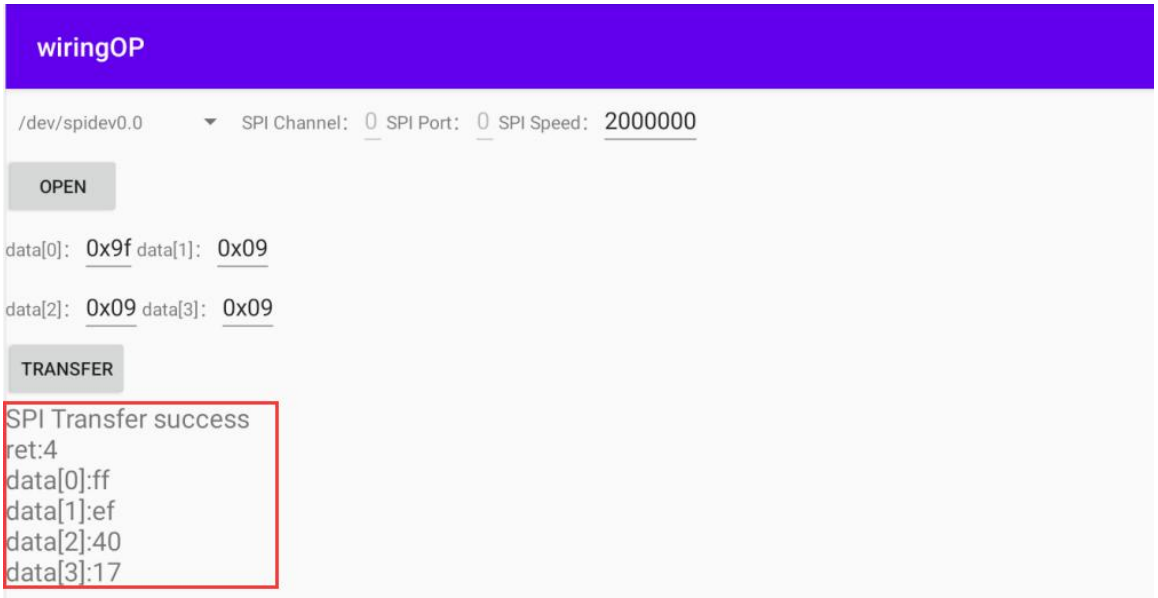
6) Then click the **OPEN** button to initialize SPI.



7) Then fill in the bytes that need to be sent, such as reading the ID information of w25q64, filling in the address 0x9f in data [0], and then clicking the **TRANSFER** button.



8) Finally, the app will display the read ID information.



9) The MANUFACTURER ID of the w25q64 module is EFh, and the Device ID is 4017h, which corresponds to the values read above (h represents hexadecimal).

MANUFACTURER ID	**(MF7 - MF0)**	
Winbond Serial Flash	EFh	
Device ID	**(ID7 - ID0)**	**(ID15 - ID0)**
Instruction	**ABh, 90h, 92h, 94h**	**9Fh**
W25Q64FV (SPI)	16h	4017h
W25Q64FV (QPI)	16h	6017h

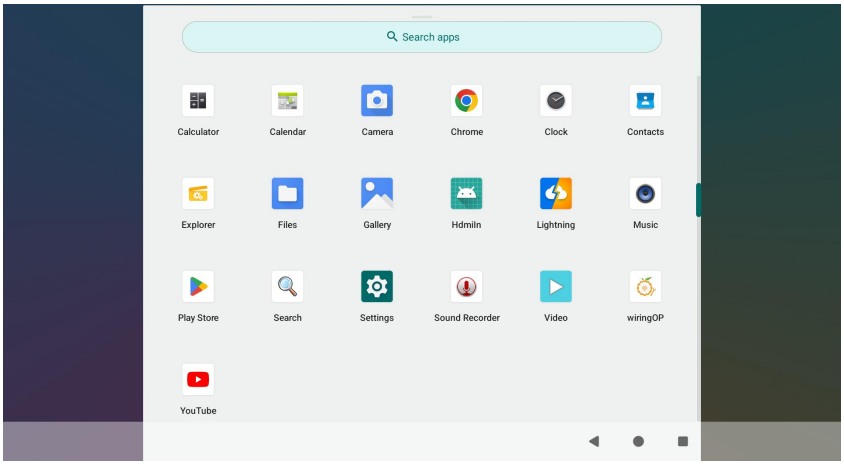
6. 7. 4. 40 pin PWM test

1) Android has enabled **PWM3** and **PWM14** by default, and the corresponding pins are

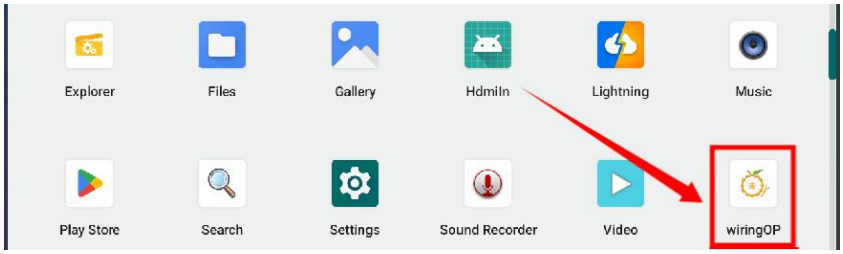
located at the 40pin position as shown in the following figure:

PWM bus	Corresponding to 40 pins
PWM3	Pin 7
PWM14	35

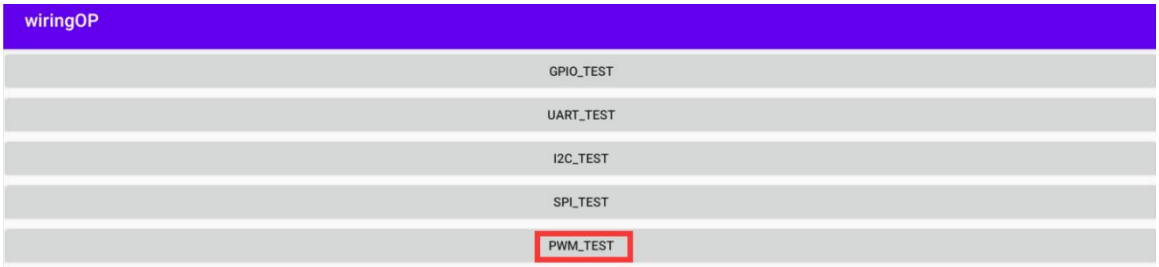
2) First, hold down the left mouse button on the desktop and drag up to open the application list.



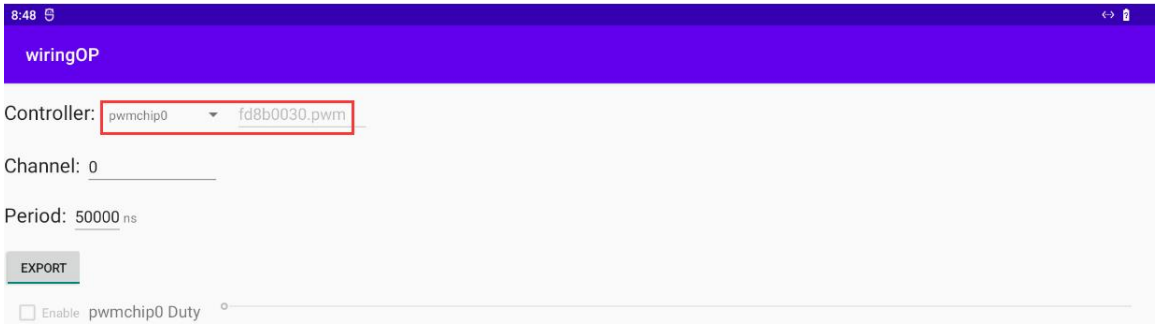
3) Then open the WiringoP app.



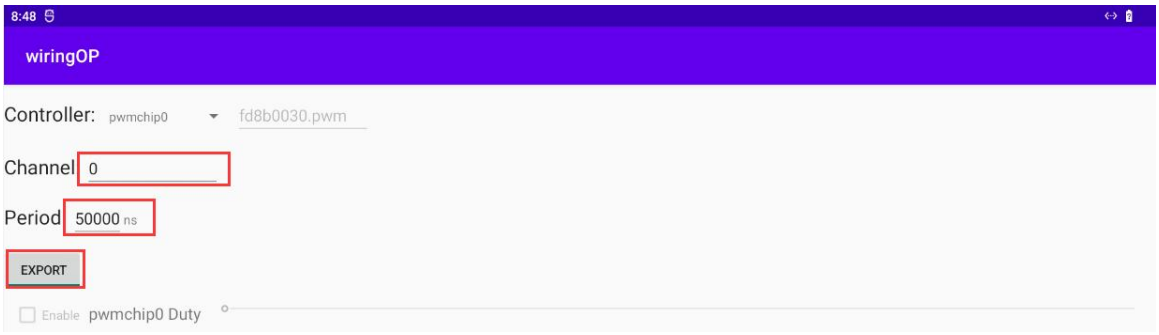
4) Then click the **PWM_TEST** button on the main interface of WirgOP to enter the PWM testing interface.



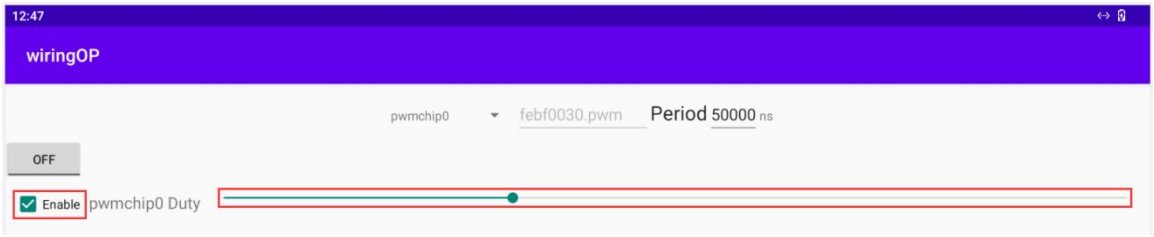
5) The base address of PWM3 is **fe8b0030**, and the base address of PWM14 is **febf0020**. Here, **fe8b0030.pwm** is displayed on the right side of pwmchip0, indicating that **PWM3** has been selected.



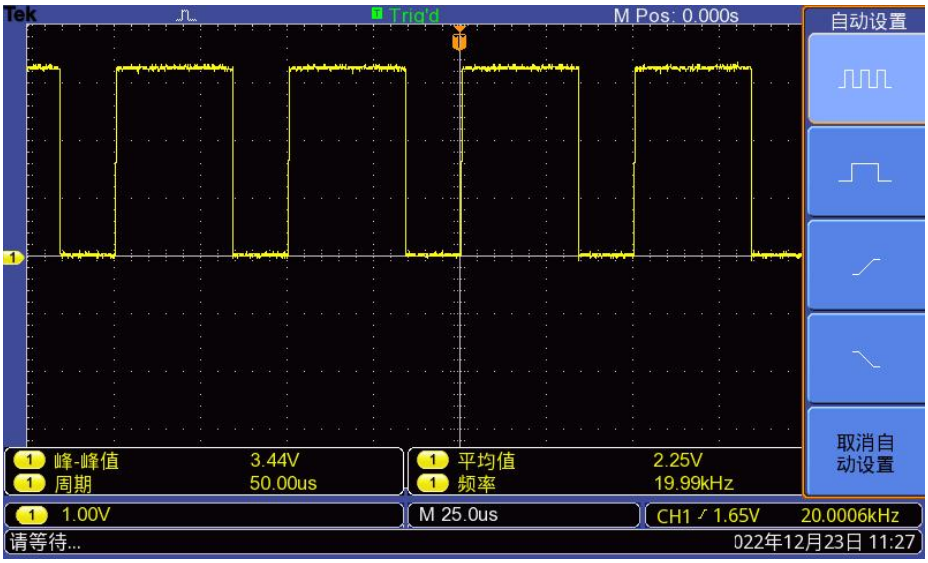
6) Then confirm the PWM channel, which defaults to channel 0, and confirm the PWM cycle. The default configuration is **50000ns**, and the converted PWM frequency is **20KHz**, which can be modified by clicking the **EXPORT** button to export **PWM3**.



7) Then drag the drag bar below to change the PWM duty cycle, and select Enable to output the PWM waveform.



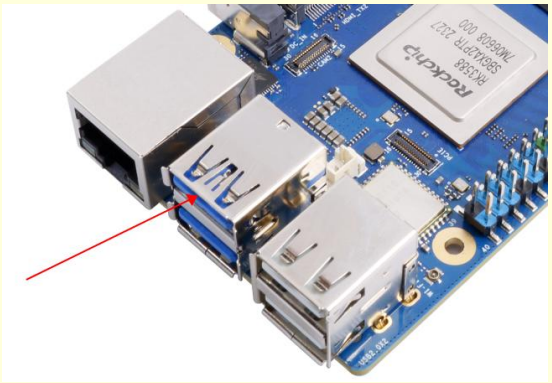
8) Then use an oscilloscope to measure pin 7 of the 40 pins on the development board to see the waveform below.



6. 8. Usage of ADB

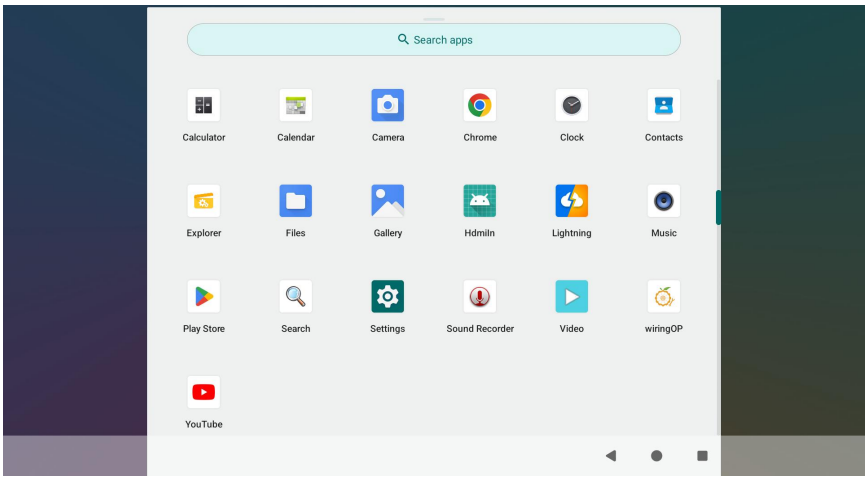
6. 8. 1. Method for Switching USB OTG Mode

The development board has 4 USB interfaces, among which the USB interface marked in red in the figure can support both Host mode and Device mode, while the other 3 USB interfaces only support Host mode.

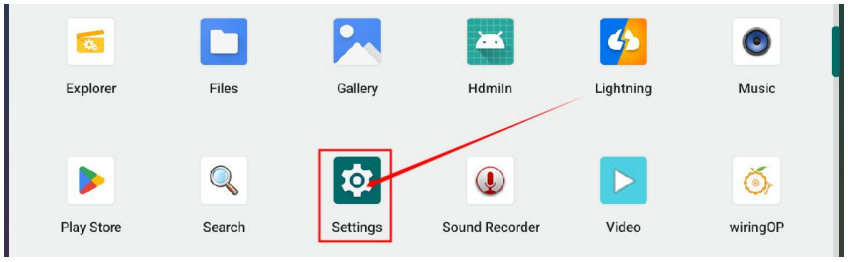


The USB OTG interface defaults to Host mode and can be used to connect USB devices such as mice and keyboards. If you want to use ADB, you need to manually switch to Device mode.

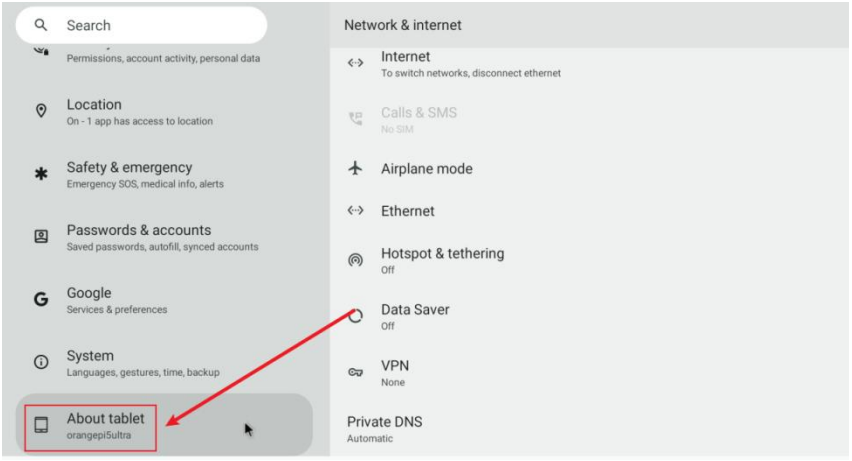
1) First, hold down the left mouse button on the desktop and drag up to open the application list.



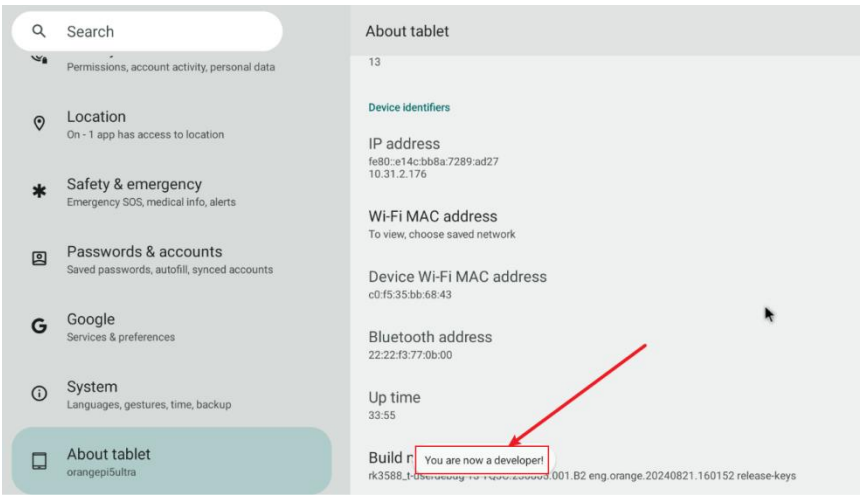
2) Then open **Settings**。



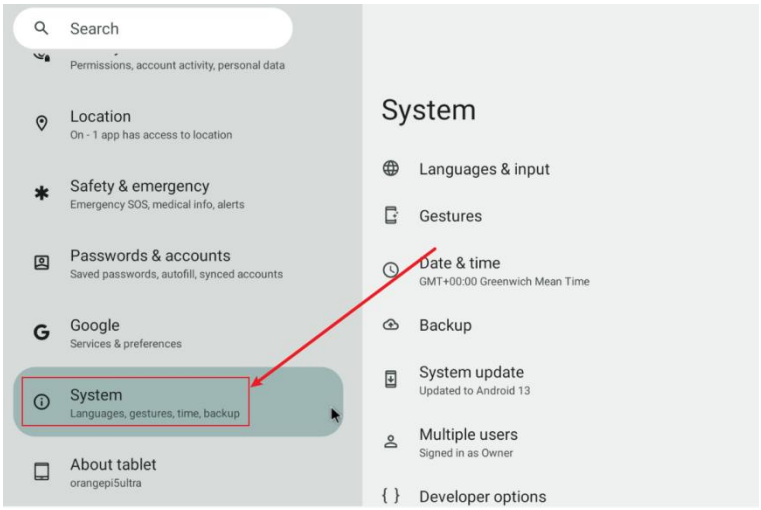
3) Then select **About tablet**。



4) Then click the **Build number** menu bar multiple times with the mouse until **You are now a developer!** The prompt.



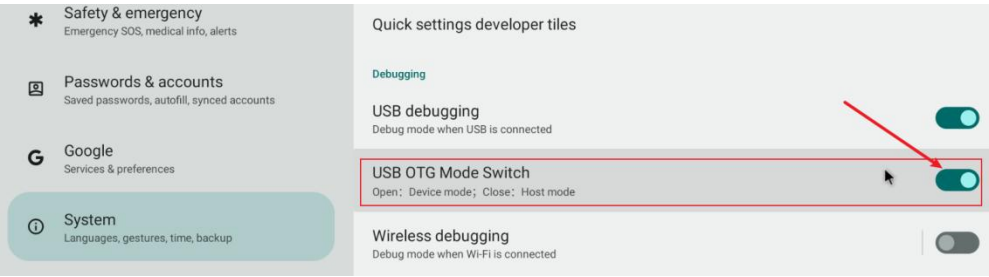
5) Then click to return to the previous menu and select **System**.



6) Then select **Developer options**。



7) Finally, locate the **USB OTG Mode Switch** switch, **turn it on to switch to Device mode, and turn it off to switch to Host mode**.



6.8.2. Using a data cable to connect adb for debugging

1) Firstly, prepare a high-quality USB 2.0 male to male data cable.



1) Then refer to the **method of switching USB OTG mode** to switch USB OTG to device mode.

2) Then use a USB 2.0 male to female data cable to connect the development board to the USB interface of the computer (please also use a TypeC power supply to power the development board).

3) Install adb tools on Ubuntu PC.

```
test@test:~$ sudo apt update
test@test:~$ sudo apt -y install adb
```

4) The recognized ADB devices can be viewed through the following command.

```
test@test:~$ adb devices
List of devices attached
S63QCF54CJ      device
test@test:~$ lsusb
Bus 003 Device 006: ID 2207:0006
```

5) Then you can log in to the Android system through adb shell on Ubuntu PC.

```
test@test:~$ adb shell
```

```
console:/ $
```

6） Execute the command to remount the Android system.

```
test@test:~$ adb root
test@test:~$ adb remount
```

7） Then you can transfer the file to the Android system.

```
test@test:~$ adb push example.txt /system/
```

6. 8. 3. Debugging with network connection adb

Using network adb does not require a USB Typc C interface data cable to connect the computer and development board, but communicates through the network. Therefore, first make sure that the wired or wireless network of the development board is connected, and then obtain the IP address of the development board, which will be used later.

1) Ensure that the **service.adb.tcp.port** of the Android system is set to port number 5555.

```
console:/ # getprop | grep "adb.tcp"
[service.adb.tcp.port]: [5555]
```

2) If **service.adb.tcp.port** is not set, you can use the following command to set the port number of the network adb.

```
console:/ # setprop service.adb.tcp.port 5555
console:/ # stop adbd
console:/ # start adbd
```

3) Install adb tools on Ubuntu PC.

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install -y adb
```

4) Then connect the network adb on Ubuntu PC.

```
test@test:~$ adb connect 192.168.1.xxx      (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
```

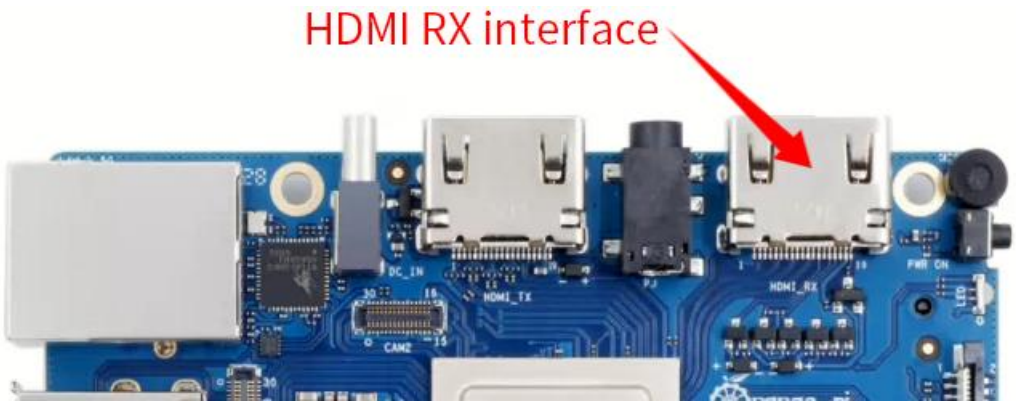
```
connected to 192.168.1.xxx:5555

test@test:~$ adb devices
List of devices attached
192.168.1.xxx:5555          device
```

5) Then you can log in to the Android system through the adb shell on an Ubuntu PC.

```
test@test:~$ adb shell
console:/ #
```

6. 9. Testing Method for HDMI RX

1) The location of the HDMI RX interface on the development board is as follows:

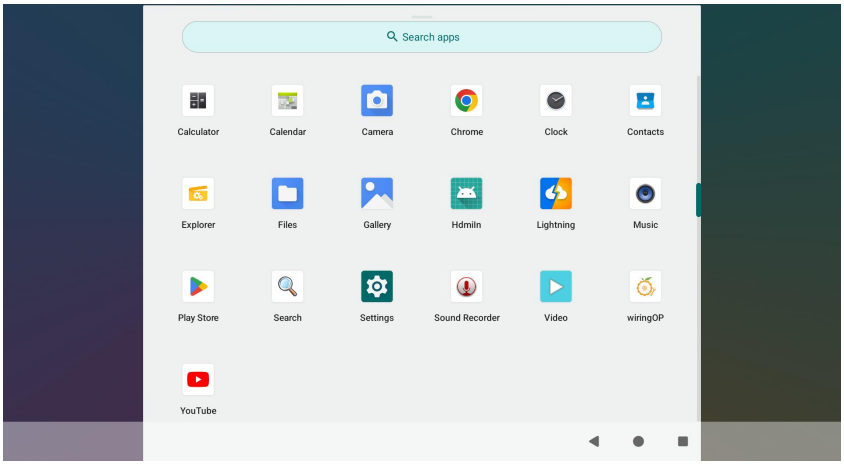


2) Then use the HDMI to HDMI cable shown in the figure below to connect the HDMI output of other devices to the HDMI RX interface of the development board.

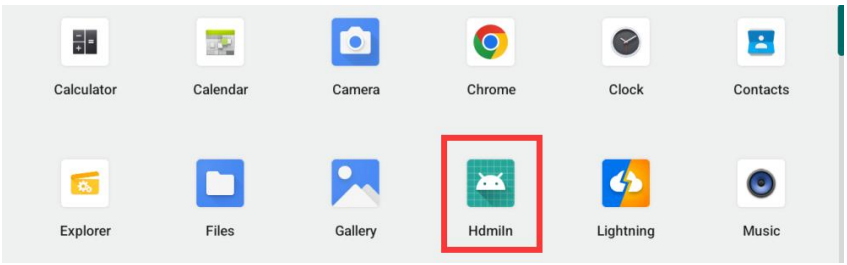


3) Ensure that the HDMI output of the device connected to the HDMI RX interface is normal, then hold down the left mouse button on the desktop and drag up to open the

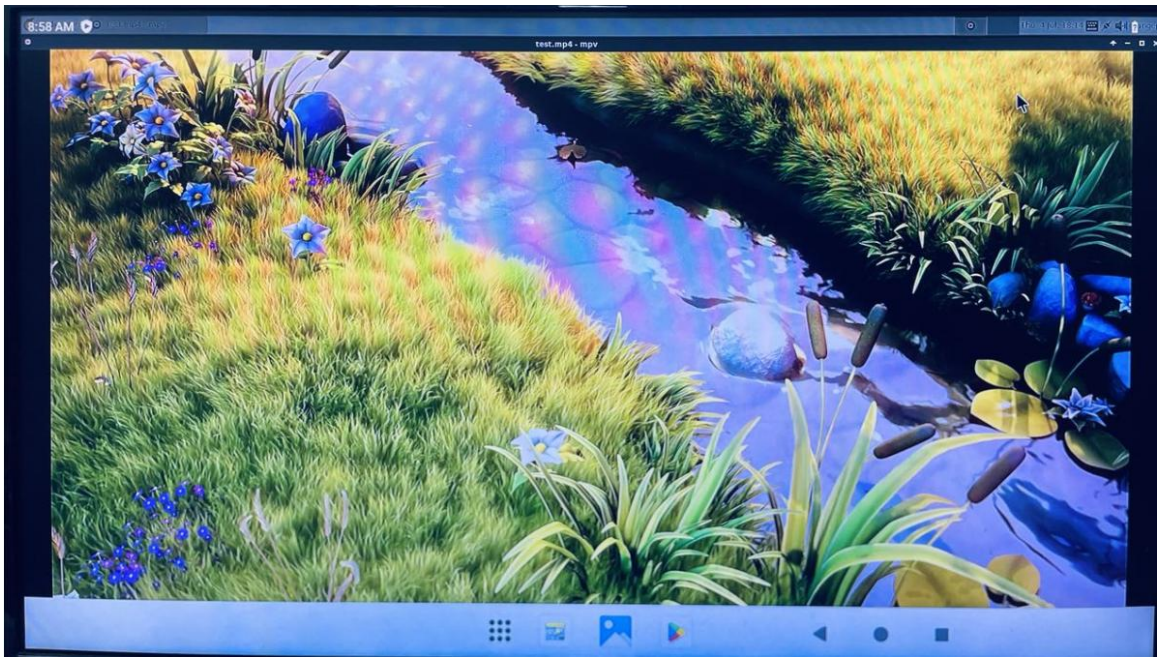
application list.



4) Then open the HDMI In test app.



5) Then you can see the video input of the HDMI RX (in the picture below, the HDMI RX displays the image of the HDMI output of the Opi5 development board, and a video is currently being played). At the same time, the audio input of the HDMI RX will also be output from the HDMI TX interface or headphone interface of the development board.



7. Compilation method of Android 13 source code

7. 1. Download the source code of Android 13

1) Firstly, download the compressed Android 13 source code from Baidu Cloud Drive or Google Cloud Drive.

2) After downloading the compressed Android 13 source code, please check if the MD5 checksum is correct. If it is not correct, please download the source code again.

```
test@test:~$ md5sum -c md5sum
Android_13.tar.gz00: OK
Android_13.tar.gz01: OK
Android_13.tar.gz02: OK
Android_13.tar.gz03: OK
Android_13.tar.gz04: OK
Android_13.tar.gz05: OK
Android_13.tar.gz06: OK
Android_13.tar.gz07: OK
Android_13.tar.gz08: OK
```

3) Then multiple compressed files need to be merged into one and decompressed.

```
test@test:~$ cat Android_13.tar.gz0* | tar -xvzf -
```

7. 2. Compile the source code for Android 13

1) First, install and compile the software packages required for Android 13 source code.

```
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
test@test:~$ sudo apt-get install -y u-boot-tools
```

2) There is a make.sh compilation script in the source code, with the following compilation parameters:

a. **-B**：Compile uboot
b. **-K**：Compile kernel
c. **-a**： Compile android
d. **-F**：Compile uboot、kernel and android
e. **-M**：Generate partition images in the rockdev directory
f. **-u**：Package to generate a complete image that can ultimately be launched
g. **-b**：Specify the development board model

3) Compile uboot, kernel, and Android and package them into the final bootable complete image.

```
test@test:~$ cd Android_13
test@test:~/ Android_13$ ./make.sh -FMu -b orangepi5ultra --nvme --gapps
```

4) After compilation, the following information will be printed.

```
********rkImageMaker ver 2.1********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
```

```
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
Make update image ok!
```

5) The final generated image file will be placed in the **rockdev/Image-rk3588_t** directory. Among them, **update.img** is the TF card boot image, and **update_spi_nvme.img** is the NVME SSD boot image.

```
test@test:~/Android_13$ cd rockdev/Image-rk3588_t
test@test:~/Android_13/rockdev/Image-rk3588_t$ ls update*
update.img update_spi_nvme.img
```

8. OpenWRT System User Manual

8.1. OpenWRT version

OpenWRT version	Kernel version
v22.03.4	Linux5.10.110

8.2. OpenWRT adaptation situation

function	OpenWRT
USB2.0x2	OK
USB3.0x2	OK
3 pin debugging serial port	OK
TF card startup	OK
2.5G PCIe Ethernet port	OK

Network port status light	OK
LED light	OK
RTL8821CU USB network card	OK
RTL8723BU USB network card	OK
FAN fan interface	OK
EMMC extension interface	OK

8. 3. The first boot to expand rootfs

1) When starting the OpenWRT system for the first time, the **resize-rootfs.sh** script will be executed to expand rootfs, and it will automatically restart after the expansion is completed.

2) After logging into the system, you can use the **df -h** command to check the size of rootfs. If it matches the actual capacity of the storage device (TF card, eMMC, or NVME SSD), it indicates that the automatic expansion is running correctly.

```
root@OpenWrt:~# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                14.8G     14.7G     91.6M  99% /
tmpfs                   495.5M      6.1M    489.4M   1% /tmp
tmpfs                   512.0K         0    512.0K   0% /dev
/dev/root                14.8G     14.7G     91.6M  99% /opt/docker
```

8. 4. Method of logging into the system

8. 4. 1. Login via serial port

1) Firstly, you can refer to the chapter on debugging serial port usage for instructions on how to use the serial port.

2) The OpenWrt system will automatically log in as the **root** user by default, and the display interface is shown below:

```
BusyBox v1.35.0 (2023-04-09 12:27:46 UTC) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 22.03.4, r20123-38ccc47687
 -----------------------------------------------------
=== WARNING! =====================================
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
--------------------------------------------------
```

8. 4. 2. Login to the system via SSH

Please note that in the OpenWrt system of Orange Pi 5 Ultra, the network port is configured as a LAN port by default.

1) Firstly, connect the LAN port of the board to the network port of the computer using an Ethernet cable, so that the network port of the computer can obtain the IP address through DHCP.

2) The default LAN port IP of the board is set to **192.168.2.1**, so the computer can obtain IP addresses starting with **192.168.2** at this time.

3) If the computer is installed with Ubuntu system, you can execute the following command to log in to the system through SSH. By default, you can log in directly without a password.

```
test@ubuntu:~$ ssh root@192.168.2.1
```

4) The display after successfully logging into the system is shown in the following figure:



5) If the computer is installed with Windows system, you can refer to the method introduced in the section of SSH remote login development board under Windows to log in.

8. 4. 3. Login to LuCI Management Interface

Please note that in the OpenWrt system of Orange Pi 5 Ultra, the network port is configured as a LAN port by default.

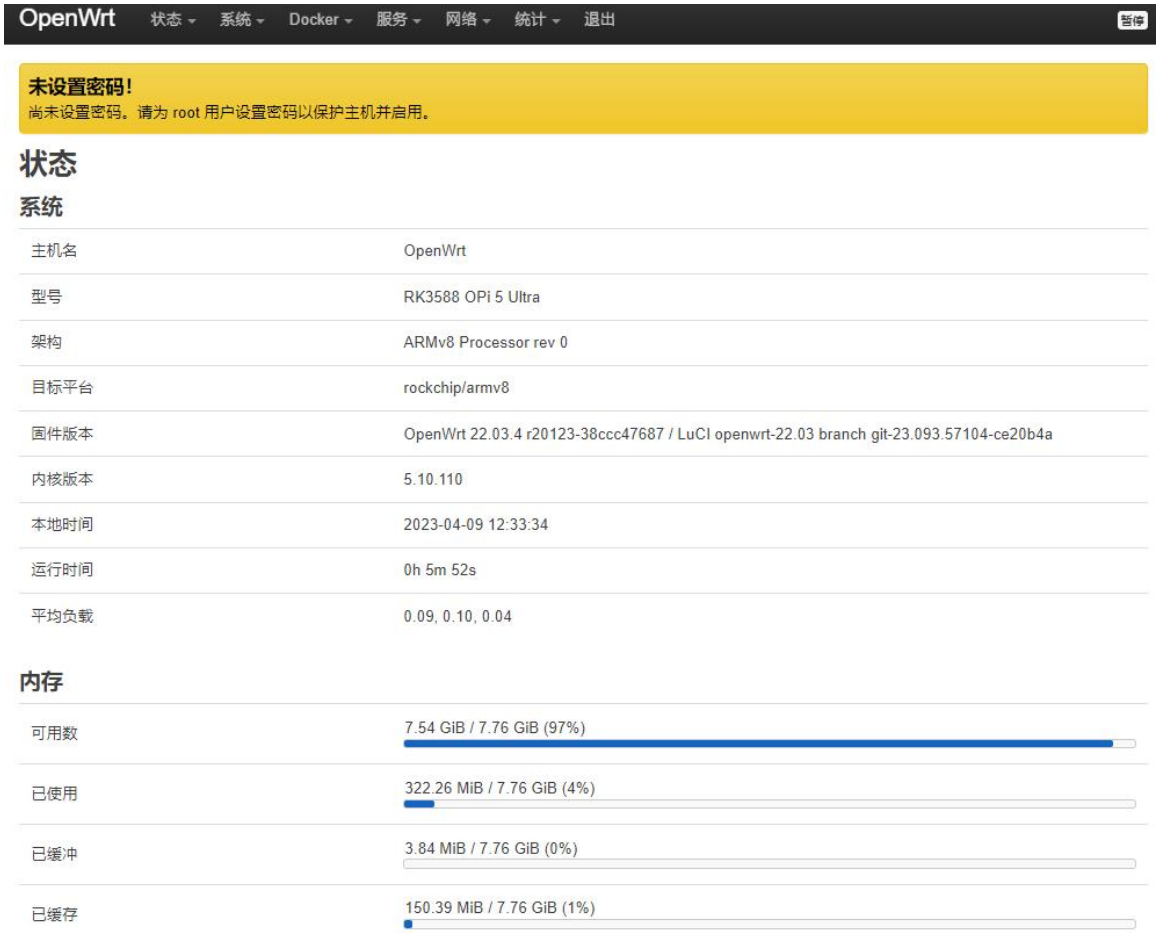
1) Firstly, connect the LAN port of the board to the network port of the computer using an Ethernet cable, so that the network port of the computer can obtain the IP address through DHCP.

2) The default LAN port IP of the board is set to **192.168.2.1**, so the computer can obtain IP addresses starting with **192.168.2** at this time.

3) You can log in to the LuCI interface by entering the IP address **192.168.2.1** in the browser on your computer.



4) **The OpenWrt system does not have a password set by default**, so simply click the **login** button. After successful login, the interface will display as shown in the following figure:



8. 4. 4. Log in to the terminal through the LuCI management interface

Please note that in the OpenWrt system of Orange Pi 5 Ultra, the network port is configured as a LAN port by default.

1) Firstly, connect the LAN port of the board to the network port of the computer using

an Ethernet cable, so that the network port of the computer can obtain the IP address through DHCP.

2) The default LAN port IP of the board is set to **192.168.2.1**, so the computer can obtain IP addresses starting with **192.168.2** at this time.

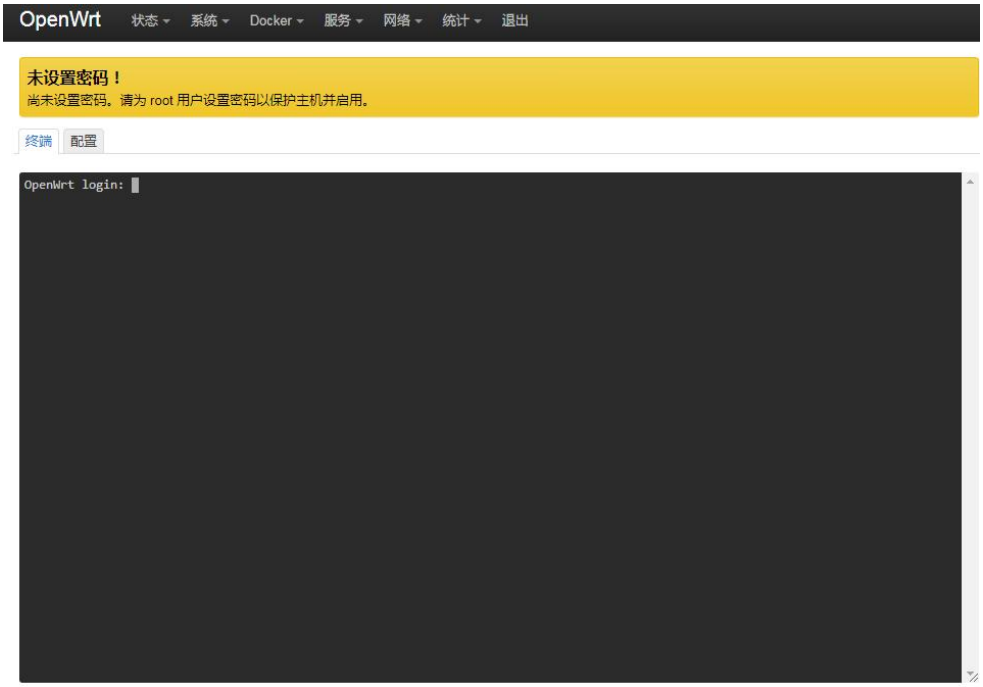
3) You can log in to the LuCI interface by entering the IP address **192.168.2.1** in the browser on your computer.



4) Select **"Terminal"** in the **"Services"** column of the navigation bar and click to enter.



5) At this point, the terminal interface is shown in the following figure:



6) Enter the username root to log in



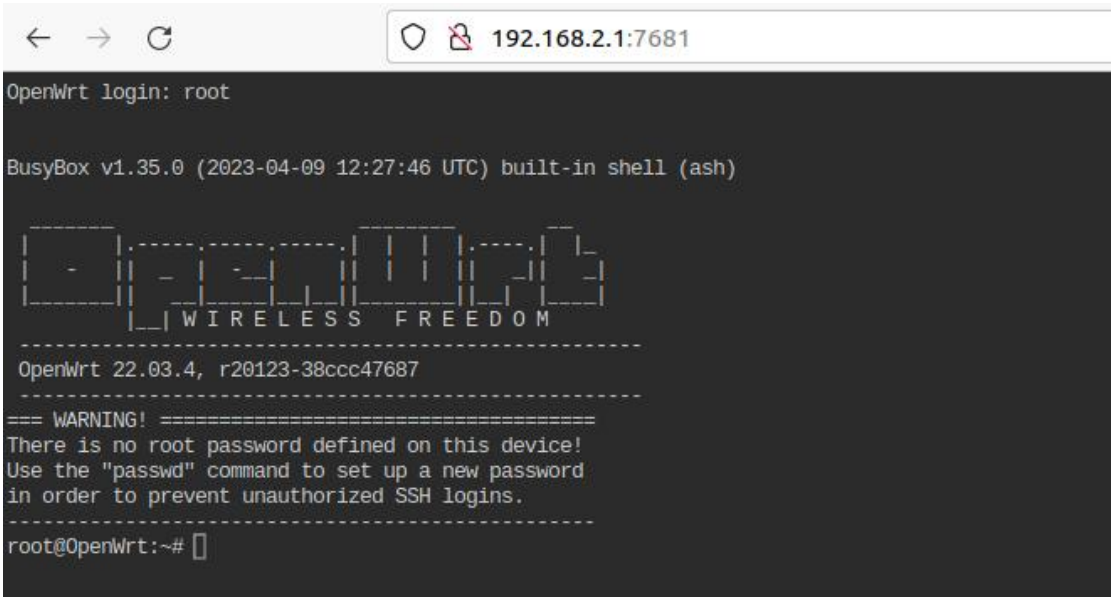
8. 4. 5. Login to the terminal using IP address and port number

Please note that in the OpenWrt system of Orange Pi 5 Ultra, the network port is configured as a LAN port by default.

1) Firstly, connect the LAN port of the board to the network port of the computer using an Ethernet cable, so that the network port of the computer can obtain the IP address through DHCP.

2) The default LAN port IP of the board is set to **192.168.2.1**, so the computer can obtain IP addresses starting with **192.168.2** at this time.

3) Then enter **192.168.2.1:7681** in the browser to log in to the OpenWRT terminal.



8.5. Method of modifying LAN port IP address through command line

1) In the OpenWrt system, a command-line tool uci is provided, which can easily modify, add, delete, and read the contents of configuration files. For detailed instructions, please refer to the official documentation.

2) First, use the following command to obtain the network configuration. The corresponding configuration file is**/etc/config/network**, and you can see that the value of the **network.lan.ipaddr** item is **192.168.2.1**.

```
root@OpenWrt:~# uci show network
...
network.lan=interface
```

```
network.lan.device='br-lan'
network.lan.proto='static'
network.lan.ipaddr='192.168.2.1'
network.lan.netmask='255.255.255.0'
network.lan.ip6assign='60'
....
```

3) Then enter the following command to modify the **network.lan.ipaddr** option.

```
root@OpenWrt:~# uci set network.lan.ipaddr='192.168.100.1'
```

4) Then enter the following command to complete the submission, which is written to the configuration file.

```
root@OpenWrt:~# uci commit
```

If the IP address in red font matches the one to be set, it indicates that the modification was successful.

```
root@OpenWrt:~# cat /etc/config/network
...
config interface 'lan'
        option device 'br-lan'
        option proto 'static'
        option netmask '255.255.255.0'
        option ip6assign '60'
        option ipaddr '192.168.100.1'
...
```

5) Restart the network through Ubuntu. Please refer to the official documentation for instructions on how to use Ubuntu.

```
root@OpenWrt:~# ubus call network restart
```

6) At this point, entering the command shows that the IP address of the LAN port is already **192.168.100.1**.

```
root@OpenWrt:~# ifconfig br-lan
br-lan    Link encap:Ethernet  HWaddr FE:55:13:A3:EF:E7
          inet addr:192.168.100.1  Bcast:192.168.100.255  Mask:255.255.255.0
```

```
inet6 addr: fd60:c4cd:1033::1/60 Scope:Global
UP BROADCAST MULTICAST  MTU:1500  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B)  TX bytes:370 (370.0 B)
```

8. 6. Method for changing root password

8. 6. 1. Modify via Command Line

1) First, enter passwd root in the system command line, and the following prompt message will appear. At this time, you can enter the password you want to set and press Enter to confirm.

```
root@OpenWrt:/# passwd root
Enter new UNIX password:
```

2) Next, you will be prompted to re-enter the password. At this point, enter the password again to confirm and press enter.

```
Retype password:
```

3) The successfully modified display is as follows:

```
passwd: password for root changed by root
```

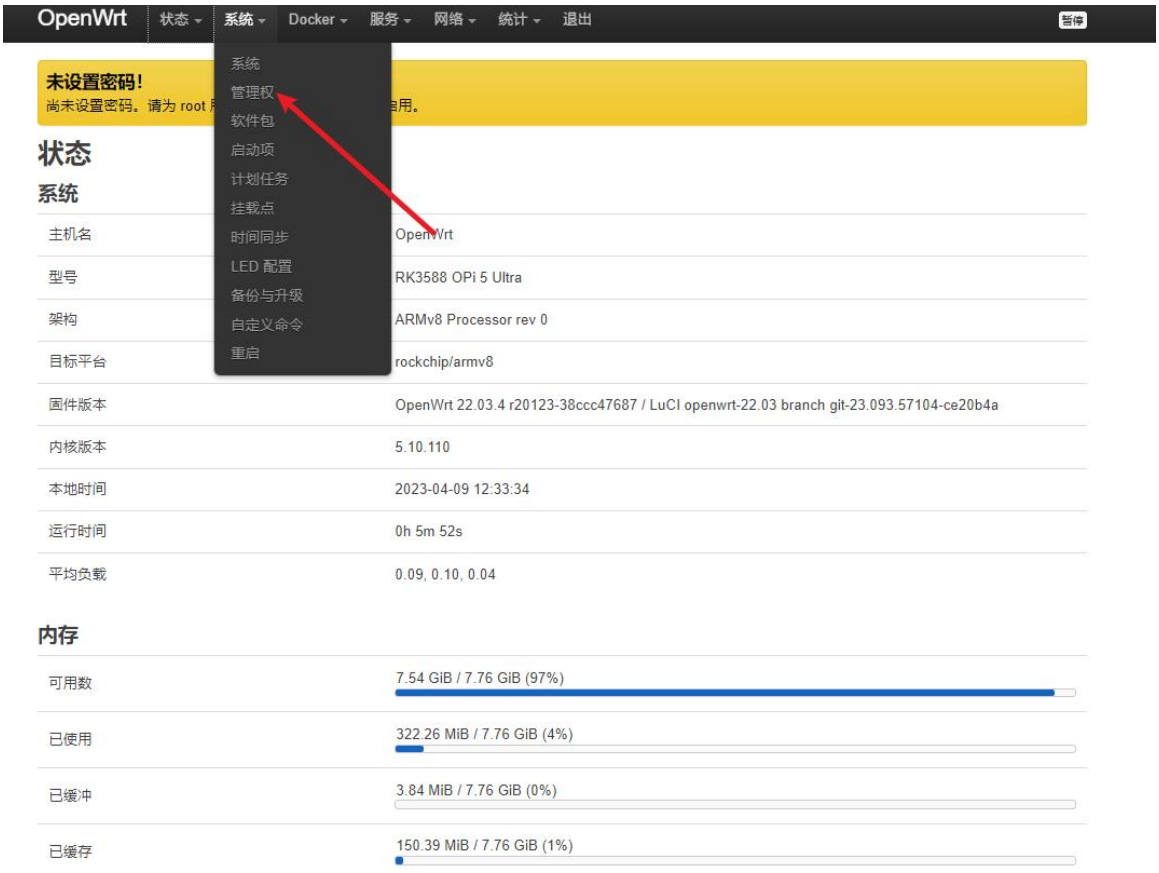
8. 6. 2. Modify through LuCI management interface

1) Firstly, refer to the **login LuCI management interface** to enter the OpenWRT management interface.

2) Then follow the steps below to change the password.

a. Find the **"System"** option in the navigation bar and click on it.

b. In the vertical bar options below the system, select **"Management Rights"** and click.



c. Select the **'Router Password'** option on the tab page.



3) Modify and save the router password.

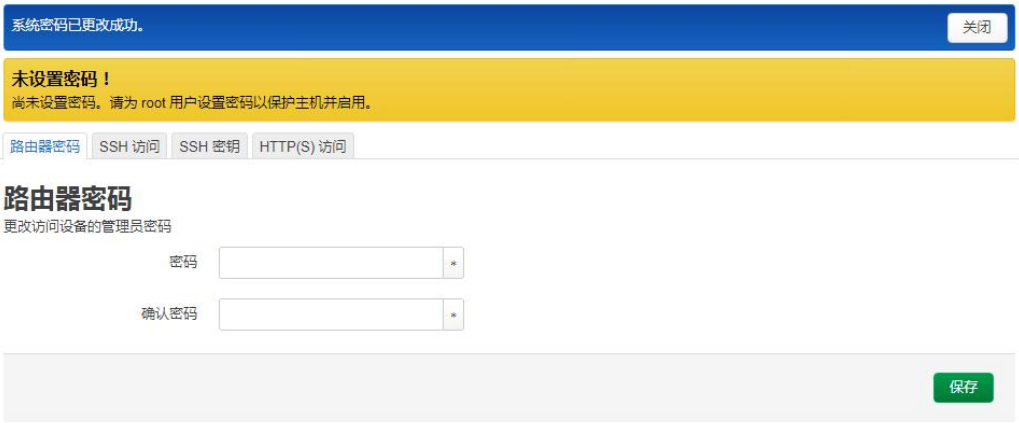
a. Enter the password you have set in the **"Password"** and **"Confirm Password"** dialog boxes (if unsure if the password is entered correctly, click the "*" icon behind the dialog box to display the input characters).

b. Click **'Save'** to save the newly modified password.



Note: In the "Password" and "Confirm Password" dialog boxes, the password entered twice needs to be consistent.

4) After the password is successfully changed, a pop-up box will appear saying **"System password has been changed successfully"**. At this time, logging in to OpenWRT requires a password.



8. 7. USB interface testing

8. 7. 1. Mounting USB storage devices at the command line

1) First, insert the USB drive into the USB interface of the Orange Pi development board.

2) If you can see the output of sdX by executing the following command, it means that the USB drive recognition is successful.

```
root@OpenWrt:~# cat /proc/partitions | grep "sd*"
major minor  #blocks  name
   8        0   15126528 sda
```

3) The mount command can be used to mount the USB drive to/mnt, and then you can view the files on the USB drive.

```
root@OpenWrt:~# mount /dev/sda /mnt/
root@OpenWrt:~# ls /mnt/
test.txt
```

4) After mounting, you can use the df -h command to view the capacity usage and mounting point of the USB flash drive.

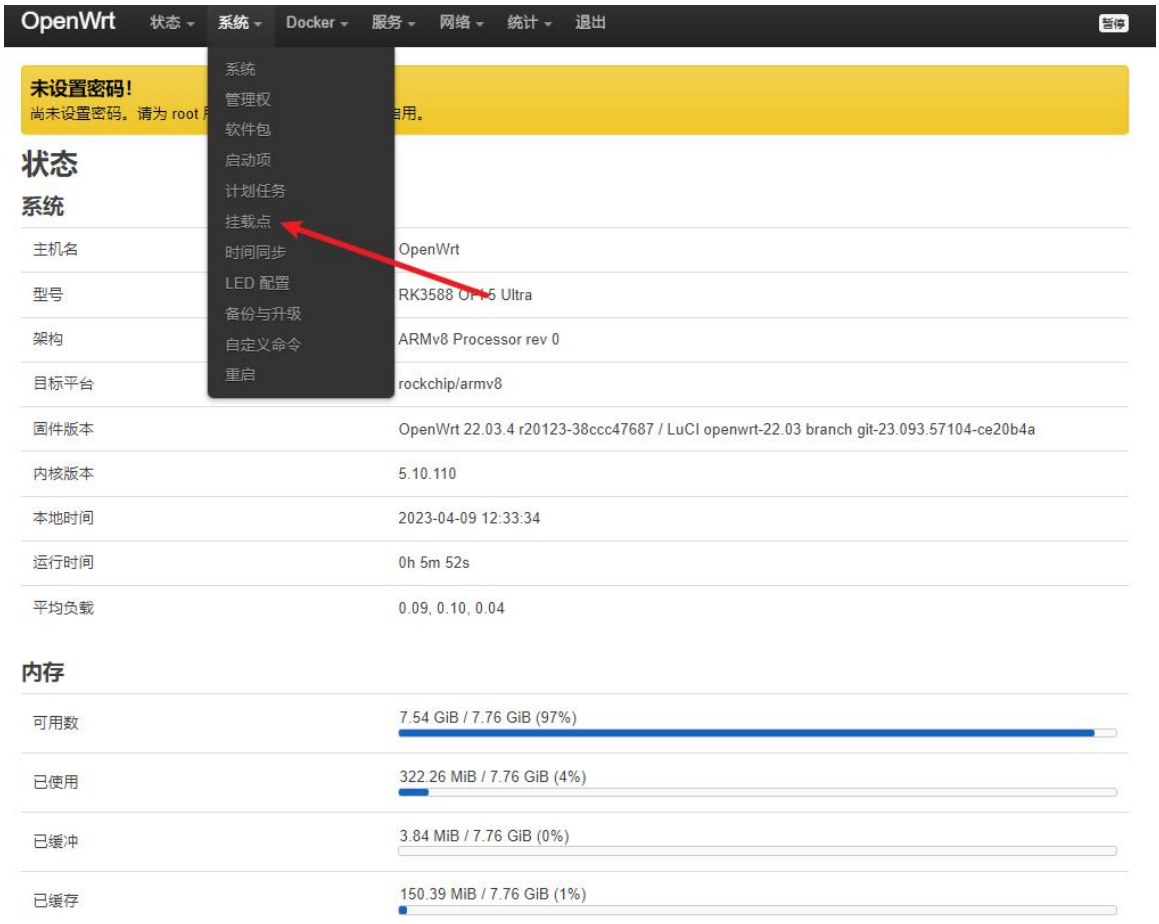
```
root@OpenWrt:~# df -h | grep "sd"
/dev/sda                14.4G       187.2M        14.2G   1% /mnt
```

8.7.2. Mounting USB storage devices on the LuCI management interface

1) Firstly, connect the USB flash drive (or other storage device) to the development board via USB 2.0.

2) Then follow the **login LuCI management interface** to enter the LuCI management interface.

3) Then, in the LuCI management interface, click on "System ->Mount Point" to enter the mount point configuration interface.



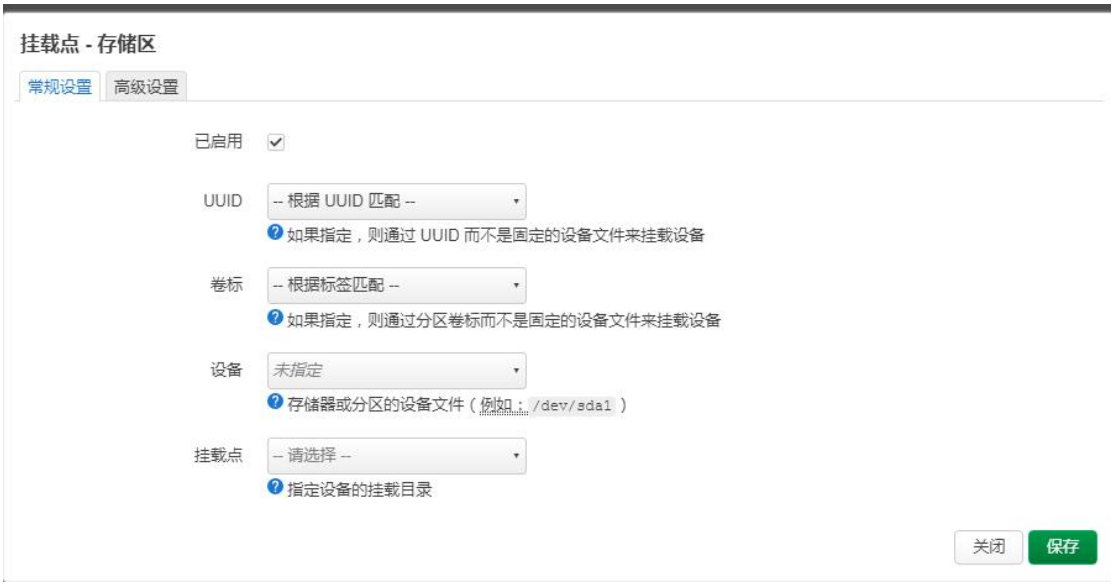
4) Then follow the steps below to add a mount point.

a. Find 'Mount Point' at the bottom of the global settings interface for mount points.

b. Below the mounting point, select the "Add" button and click to enter.



c. Next, a pop-up window will appear below.



d. Then you can start mounting the storage device.

a) Check **'Enabled'**.

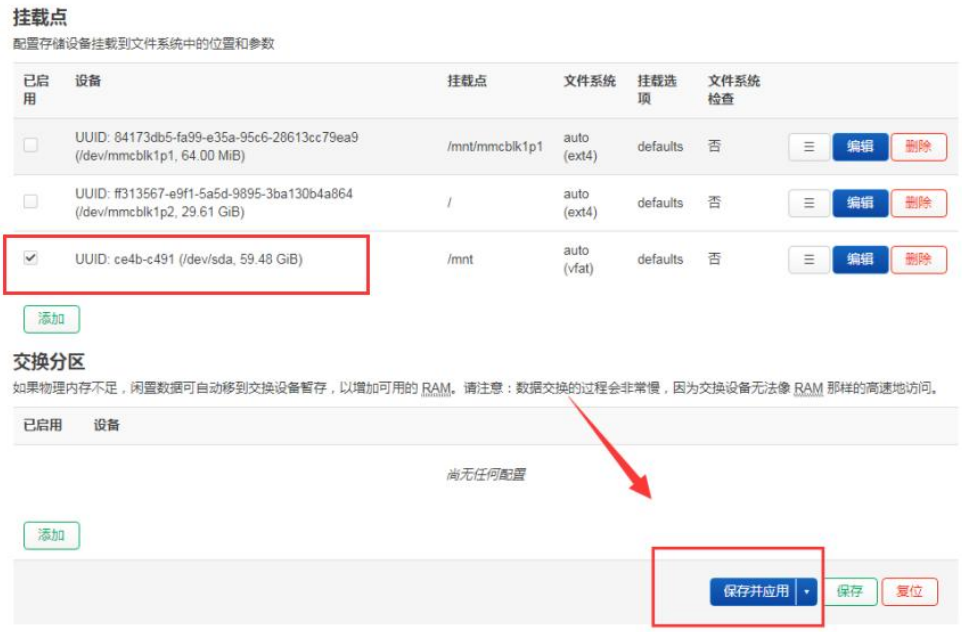
b) Select the actual connected device/dev/sda in the UUID column of the general settings (choose according to your own device).

c) Select "**Custom**" in the mount point column and fill in the target directory to be mounted to. Taking the**/mnt** directory as an example, enter to confirm.

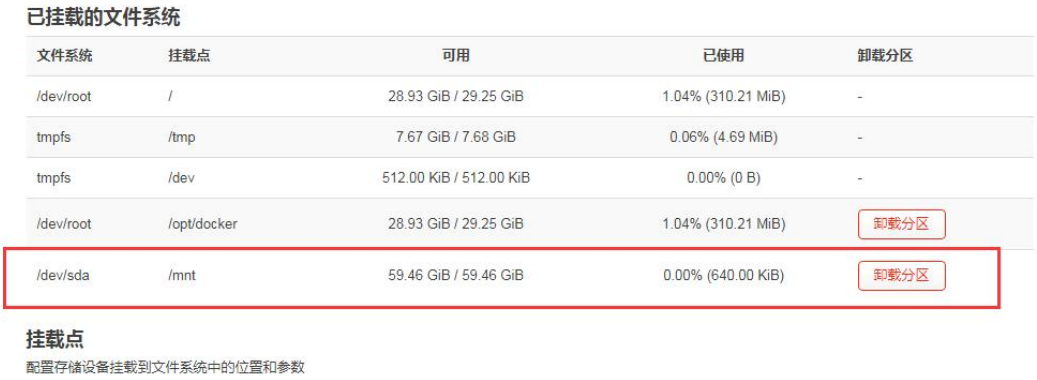
d) Then click the "**Save**" button in the bottom right corner.



5) Then you will return to the mount point global settings page and click "**Save and Apply**" in the bottom left corner of the page to make the mount point effective.



6) After saving, you can see in the **'Mounted File System'** that the storage device has been successfully mounted.



8. 8. USB Wireless Network Card Test

The currently tested USB wireless network cards that can be used are shown below. For other models of USB wireless network cards, please test them yourself. If they cannot be used, you need to port the corresponding USB wireless network card driver.

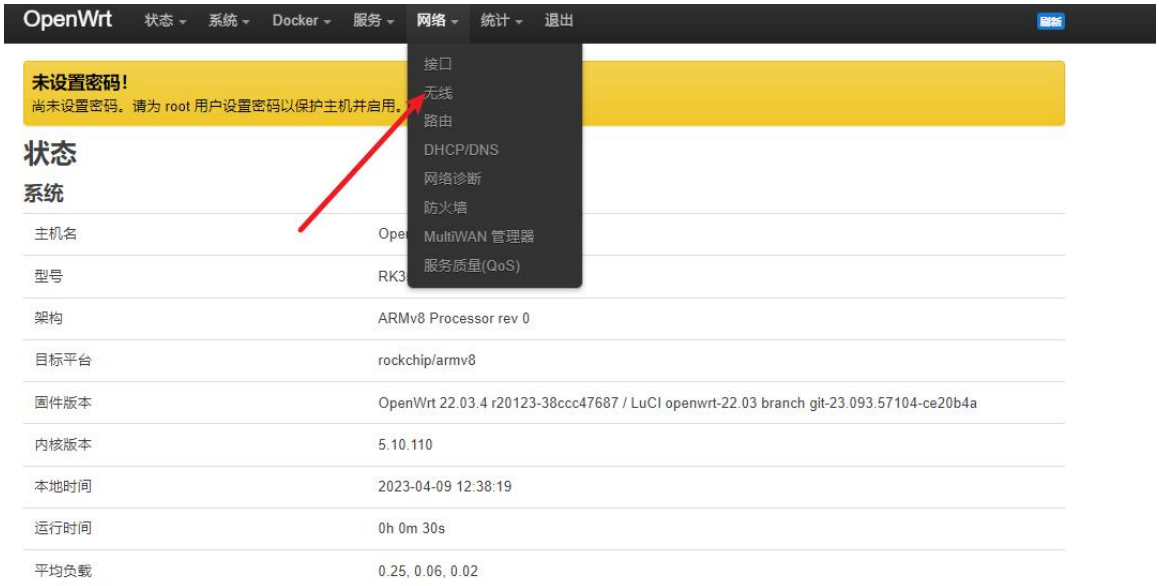
Serial number	model	

1	RTL8723BU support 2.4G WIFI+BT4.0	
2	RTL8821CU support 2.4G +5G WIFI support BT 4.2	

8. 8. 1. Method of using a USB wireless network card to connect to a WIFI hotspot

1）Insert the USB wireless network card into the USB port of the development board, and then connect the power supply to power on the development board.

2）After the system startup is complete, click on **Network ->Wireless** to enter the wireless WiFi configuration interface.



3）The default wireless configuration of OpenWRT system is **Master** mode. For the convenience of the next operation, we will remove the default wireless connection.



4）Then click on the bottom right corner of the page to **save** and make the configuration effective.



5）Then click the **scan** button to scan the surrounding WiFi hotspots.



6）Then a window will pop up displaying available WiFi hotspots. Click the **"Join Network"** button to the right of the desired WiFi hotspot to connect.



7）Then a interface will pop up to connect to the WiFi hotspot. We will enter the hotspot password at the location shown in the figure below, and then click the **submit** button.



8）Then the following interface will pop up, click the **save** button in the bottom right corner.



9）Finally, you will return to the main interface of wireless configuration, click **save and apply**, and wait for the configuration to be applied.



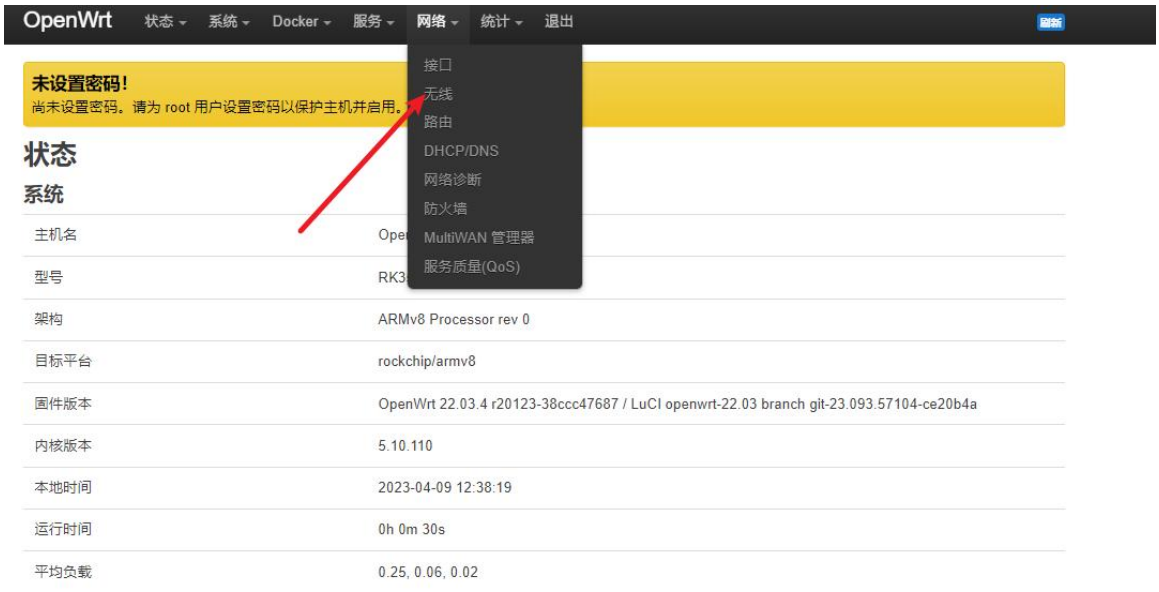
10）After successfully connecting to the WiFi hotspot, the interface displays as shown in the following figure.



8.8.2. Method for creating a WIFI hotspot using a USB wireless network card

1）Insert the USB wireless network card into the USB port of the development board, and then connect the power supply to power on the development board.

2）The system startup is complete, click on **Network ->Wireless** to enter the wireless WiFi configuration interface.



3）The default wireless configuration of OpenWRT system is **Master** mode. For the convenience of the next operation, we will remove the default wireless connection.



4）Then click on the bottom right corner of the page to **save** and make the configuration effective.



5）Then click the **add** button on the right.



6） In the pop-up tab **device configuration**, we set the parameters as shown in the following figure.



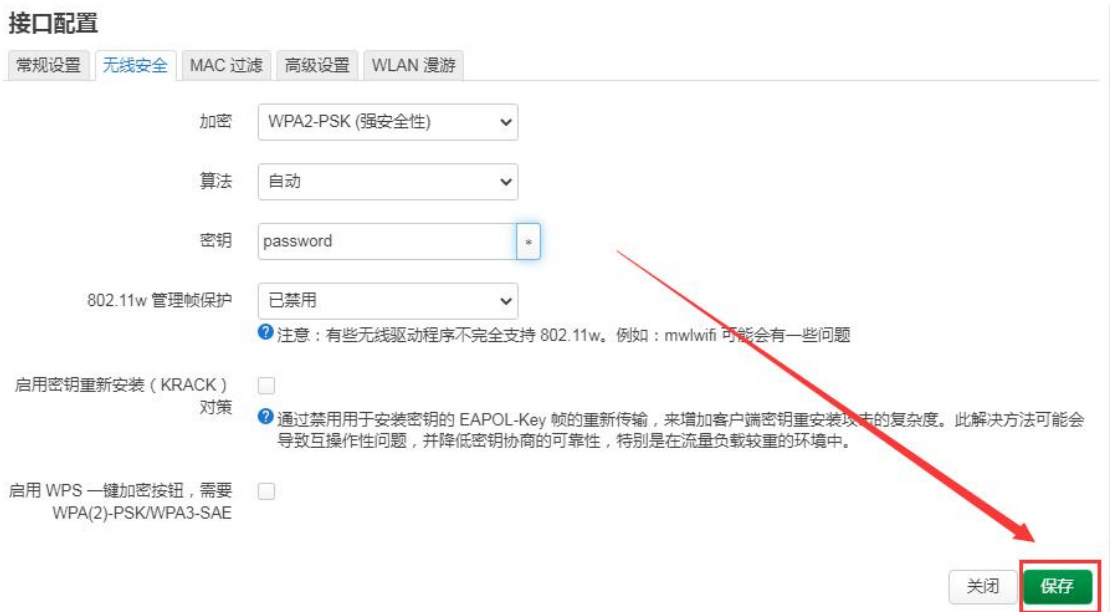
7）Then in **Interface Configuration ->General Setting**s, set the mode to **Access Point AP**, **ESSID** (Wireless Network Name) to **OpenWrt**, and network to **LAN**.



8）Then in Interface **Configuration ->Wireless** Security, select **WPA2-PSK** as the encryption algorithm; Set the key (wireless password) to **password**.



9）After completing the above settings, click on the bottom right corner of the page to **save**, and then the tab will exit.



10）Then click on the bottom right corner of the page to **save and apply**, and wait for the configuration to be applied.



11）The display interface for successfully creating a hotspot is shown in the following figure:



12）Then use your phone or computer to search for the corresponding WiFi SSID for connection. After successful connection, as shown in the following figure:



8. 9. Installing software packages through the command line

8. 9. 1. Installing through OPkg on the terminal

1) Update the list of available software packages

```
root@OpenWrt:/# opkg update
```

2) Get software list

```
root@OpenWrt:/# opkg list
```

3) Install the specified software package

```
root@OpenWrt:/# opkg install <Package Name>
```

4) View installed software

```
root@OpenWrt:/# opkg list-installed
```

5) Uninstall software

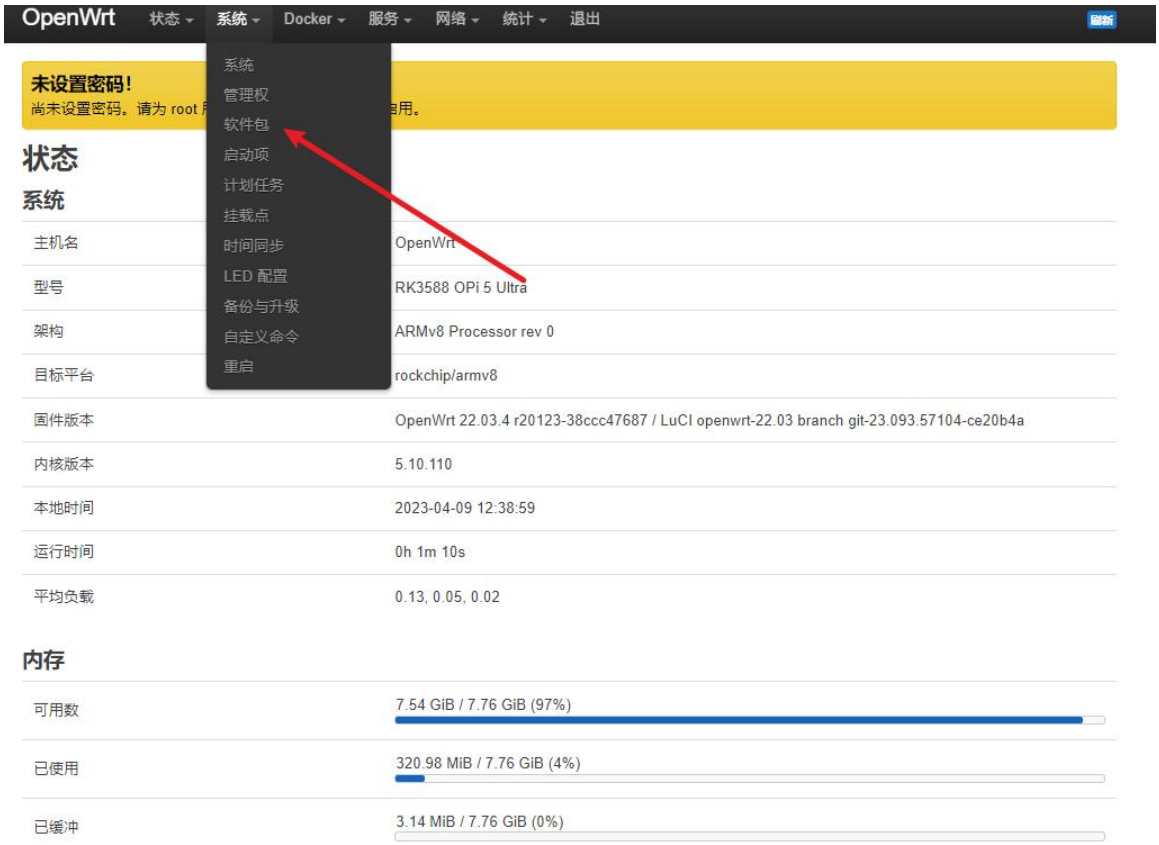
```
root@OpenWrt:/# opkg remove <Package Name>
```

8.10. OpenWRT management interface installation software package

If you need to add software packages, you can install them through the OpenWRT management interface.

8.10.1. View the list of available software packages in the system

1) First, enter the software package management page
   a. Find the **"System"** option in the navigation bar and click to enter
   b. In the vertical bar options below the system, select **"Software Package"** and click to enter



2) Then the main page of the software package will appear, as shown in the following figure, to obtain the list of available software

a. In the **"Operation"** option of the software package, click **"Update List"** to obtain the list of available software packages

b. On the tab page, click **"Available"** to view the currently available software packages

c. View the current number of available software packages

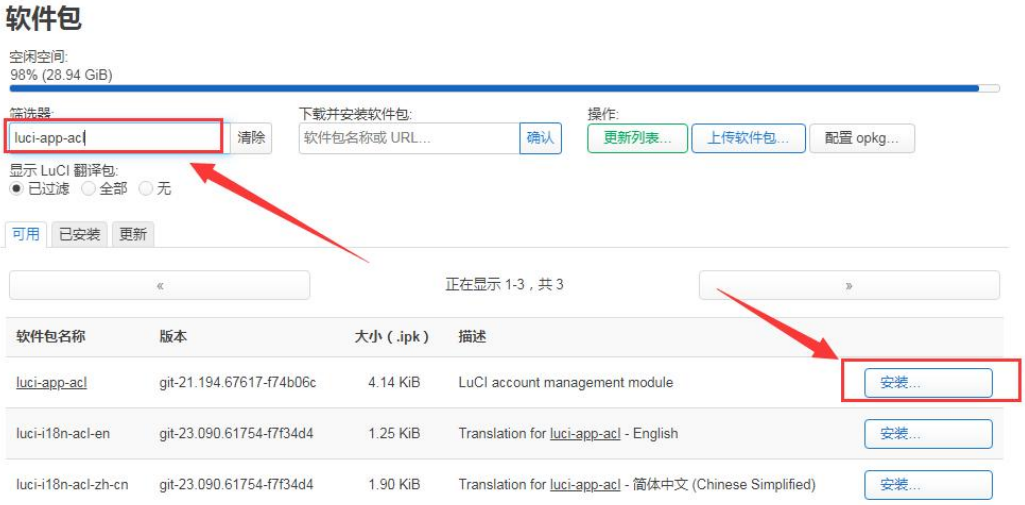


8. 10. 2. Example of Installing Software Packages

1) Taking the installation of the software package "**luci-app-acl**" as an example

a. In the package management interface of OpenWRT, click on the filter dialog box and enter "**luci-app-acl**"

b. In the list of software packages, you can see the version, package size, and description information of the "**luci-app-acl**" package, and then click the **"Install"** button



c. Then the following pop-up window will appear, click **"Install"** to proceed



d. Then wait for the installation to complete



e. The display after installation is as follows



2) Check if the software package has been successfully installed
   a. In the package management interface of OpenWRT, click on the filter dialog box and enter **"luci-app-acl"**
   b. Select and click **'Available'** on the tab page
   c. The **"luci-app-acl"** package will be displayed in the package list and updated to **'installed'** status

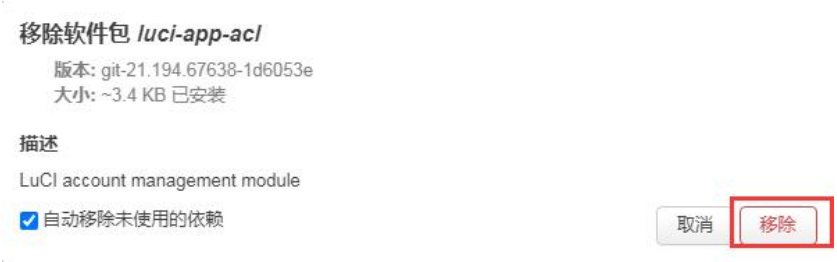


8. 10. 3. Example of Removing Software Packages

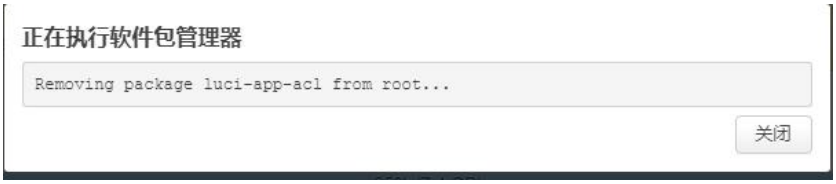
1) Taking the removal of the software package **"luci-app-acl"** as an example
   a. In the package management interface of OpenWRT, click on the filter dialog box and enter **"luci-app-acl"**
   b. Select **'Installed'** on the tab page to display a list of installed software packages
   c. Click **'Remove'** on the right to remove the corresponding software package



d. Then a pop-up window will appear below, click **'Remove'** to proceed



e. After successful removal, the display interface is as follows



2) Check if the software package has been successfully removed

a. In the package management interface of OpenWRT, click on the filter dialog box and enter **"luci-app-acl"**

b. Select and click **'Installed'** on the tab page

c. The **"luci-app-acl"** package will not be displayed in the package list, and the **"luci-app-acl"** package has been successfully removed



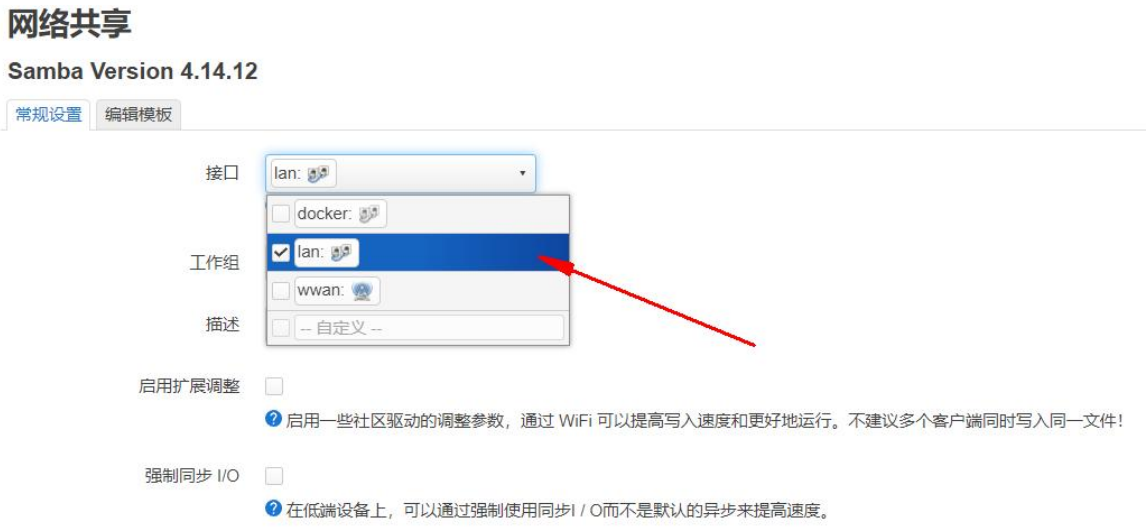
8. 11. Using Samba Network Sharing

There are two main software options for implementing OpenWRT LAN file sharing: Samba and NFS. Samba system has good compatibility, while NFS performs better. For users who need to use Windows devices, it is recommended to choose Samba.

1) Enter the Samba network share management page
   a. Find the **"Services"** option in the navigation bar and click to enter
   b. In the vertical bar options below the service, select **"Network Sharing"** and click to enter



   a. Select the interface that Samba service needs to listen on
   b. Select **"General Settings"** in the navigation bar of network sharing and click to enter
   c. Set the interface to **"Lan"**



2) Set the shared directory for network sharing

a. Click "Add" in the "Shared Directory" section of the "General Settings" for network sharing to share the directory address

b. Enter the name of the shared folder as "**mmt**" under the name

c. Under the path of the shared directory, select the location of the shared directory "/"

d. Check **'browseable'** and **'allow anonymous users to run'**

e. Click **'Save and Apply'** to save the configuration

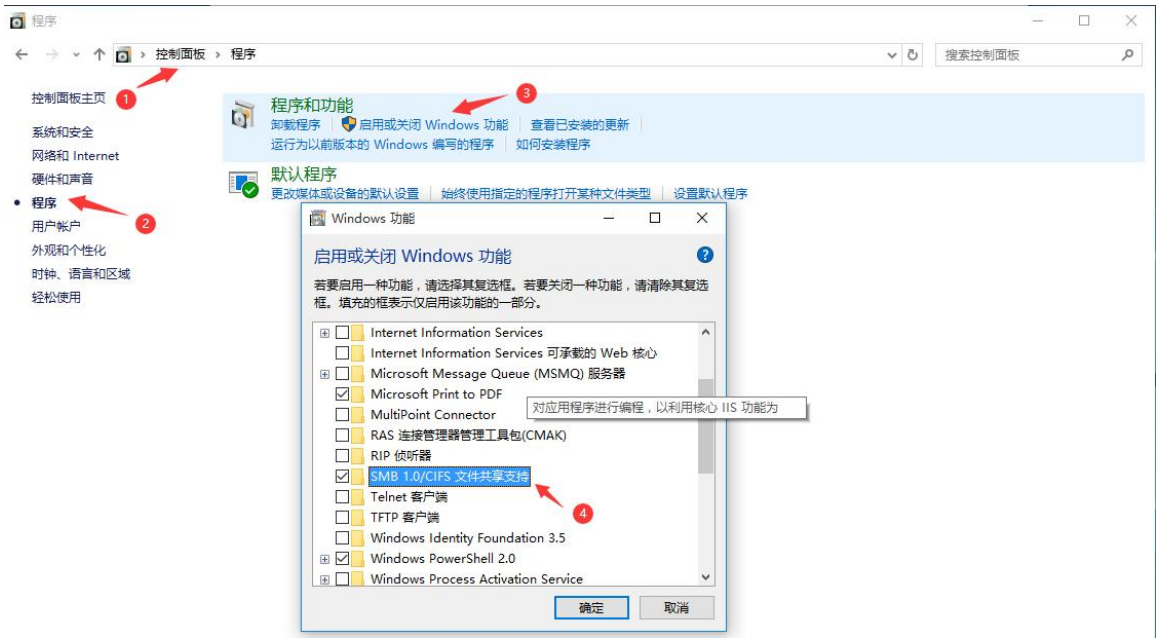


3) Windows 10 starts network discovery and sharing

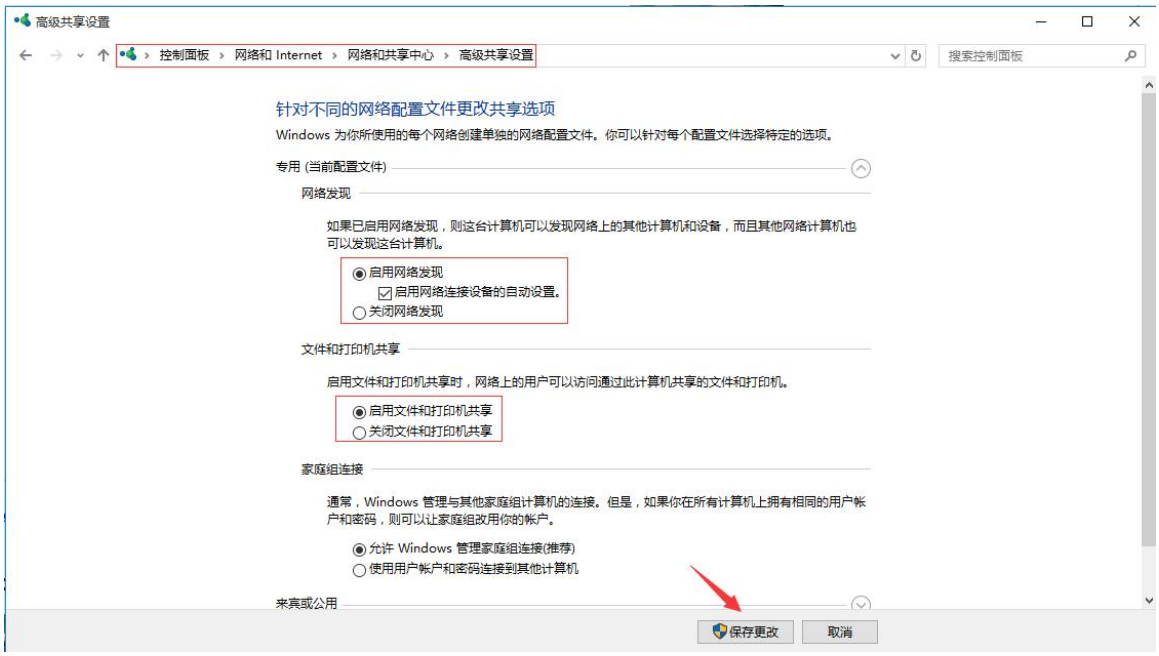
Note: To access Samba on the Windows 10 system, it is necessary to first confirm whether Windows 10 has started network discovery and sharing. If it has not been started, the following settings should be made first.

a. Enable access to Samba v1/v2

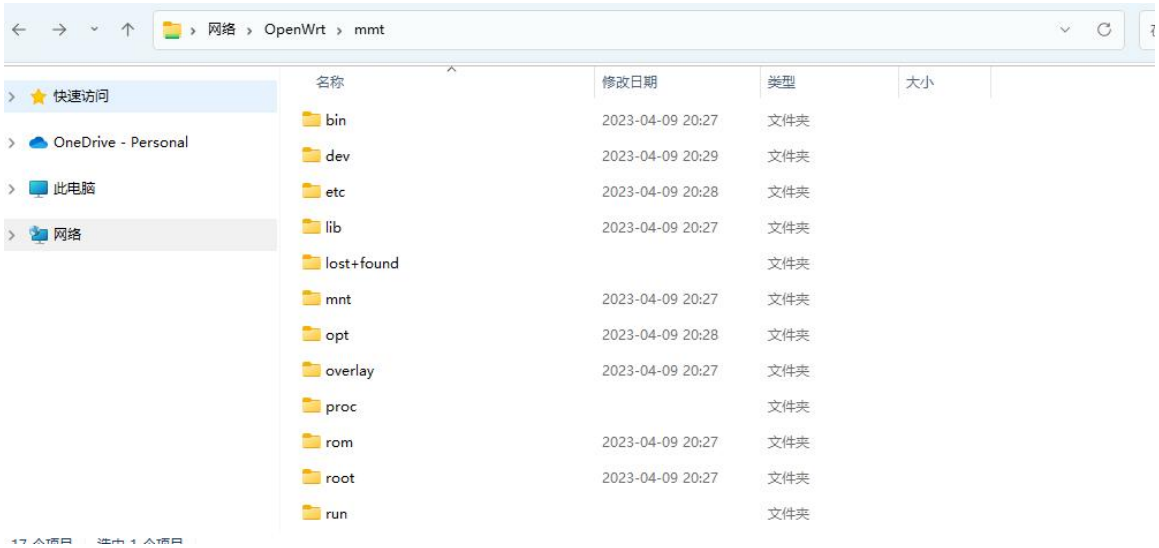
a) Enter the Control Panel of Windows 10
b) Click on "Programs" in the left navigation bar of the control panel
c) Select 'Enable or Disable Windows Features' in Programs and Features
d) Check 'SMB 1.0/CIFS file sharing support' in the pop-up box to enable or disable Windows features
e) Click 'OK' to configure the application



b. Open Windows 10 Network Discovery
a) Enter the Control Panel of Windows 10
b) Select "Network and Internet" in the control panel
c) Then open the "Network and Sharing Center"
d) Click on 'Advanced Sharing Settings'
e) Open **'Enable Network Discovery'** and **'Enable File and Printer Sharing'**
f) Click 'Save Changes' to save the network discovery configuration for Windows 10



4) After setting up, enter \ \ OpenWrt in the address bar of the resource manager to access the shared directory. The username is root and the password is the password set by the development board host

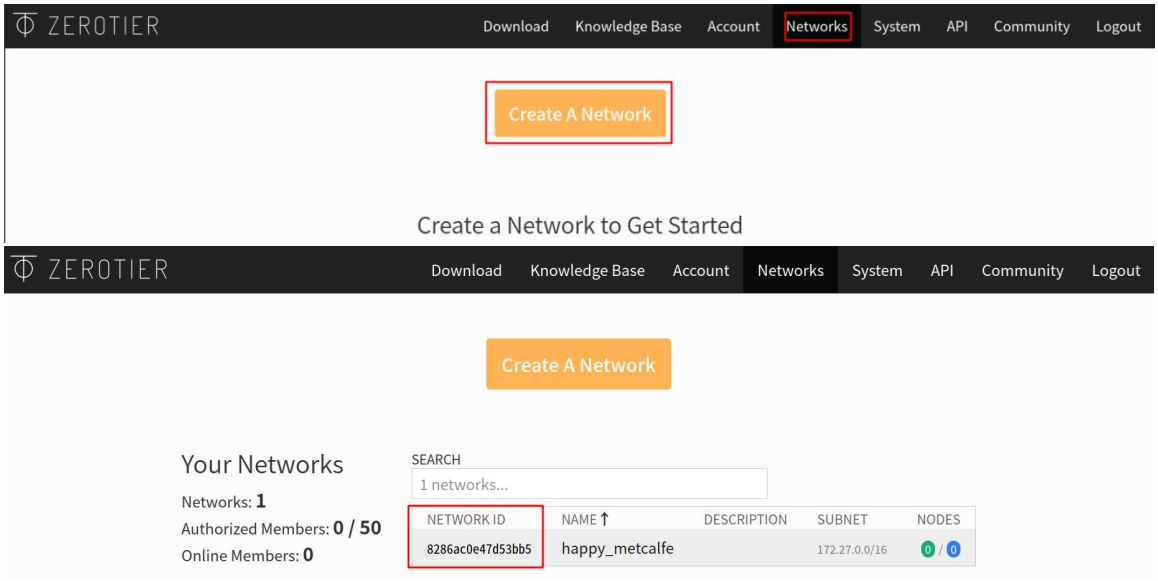


8. 12. Zerotier User Manual

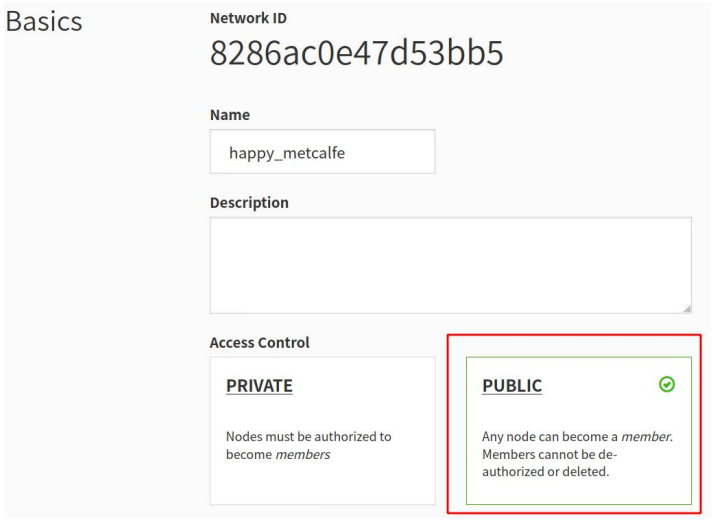
The OpenWRT system has pre installed the zerotier client. After creating a virtual LAN on the zerotier official website, the client can directly join it through the

Network ID. The specific operation is shown below.

1) Log in to the zerotier official website **https://my.zerotier.com/network** After registering and logging in, click Network ->Create A Network to create a virtual LAN



2) Click to enter the network console page, where you can set the privacy option to public, so that network nodes that join do not need to be verified



3) Below, the address will be automatically assigned. Here, you can choose your own network segment, and the selected one is 172.27. **



4) Enter the following command on the OpenWRT terminal to join the virtual LAN created above, **where 8286ac0e47d53bb5 is the Network ID of the virtual LAN created above**

```
root@OpenWrt:/# zerotier-one -d                    #Start the zerotier client
root@OpenWrt:/# zerotier-cli join 8286ac0e47d53bb5  #Join the network
```

5) By entering ifconfig on the terminal, it can be seen that there is already a newly added **ztks54inm2** device with an IP address of **172.27.214.213**

```
root@OpenWrt:/# ifconfig
ztks54inm2 Link encap:Ethernet  HWaddr F6:4E:DE:BF:D8:52
          inet addr:172.27.214.213  Bcast:172.27.255.255  Mask:255.255.0.0
          inet6 addr: fe80::e82f:d0ff:fe5a:867e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:2800  Metric:1
          RX packets:18 errors:0 dropped:0 overruns:0 frame:0
          TX packets:48 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1720 (1.6 KiB)  TX byte81 (8.2 KiB)
```

6) Install the zerotier client on another device (using Ubuntu 18.04 as an example), execute the following command to install, and restart the computer after installation is complete

```
test@ubuntu:~$ curl -s https://install.zerotier.com | sudo bash
```

7) After restarting, join the virtual LAN based on the Network ID, and you can also see

that the IP address assigned by zerotier has been obtained. At this time, the Ubuntu PC and the development board are in the same LAN, and they can communicate freely

```
test@ubuntu:~$ sudo zerotier-cli join 8286ac0e47d53bb5
test@ubuntu:~$ ifconfig
ztks54inm2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 2800
        inet 172.27.47.214  netmask 255.255.0.0  broadcast 172.27.255.255
        inet6 fe80::5ce1:85ff:fe2b:6918  prefixlen 64  scopeid 0x20<link>
        ether f6:fd:87:68:12:cf  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 46  bytes 10006 (10.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

8) Test whether two terminals can communicate

```
root@OpenWrt:/# ping 172.27.47.214 -I ztks54inm2
PING 172.27.47.214 (172.27.47.214): 56 data bytes
64 bytes from 172.27.47.214: seq=0 ttl=64 time=1.209 ms
64 bytes from 172.27.47.214: seq=1 ttl=64 time=1.136 ms
64 bytes from 172.27.47.214: seq=2 ttl=64 time=1.203 ms
64 bytes from 172.27.47.214: seq=3 ttl=64 time=1.235 ms
^C
--- 172.27.47.214 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.136/1.195/1.235 ms
```

9) Zerotier other commonly used commands

```
root@OpenWrt:/# zerotier-one -d                    #Start the zerotier client
root@OpenWrt:/# zerotier-cli status                #Obtain address and service status
root@OpenWrt:/# zerotier-cli join # Network ID     #Join the network
root@OpenWrt:/# zerotier-cli leave # Network ID    #Leave the internet
root@OpenWrt:/# zerotier-cli listnetworks          #List networks
OPENWRT_DEVICE_REVISION="v0"
OPENWRT_RELEASE="OpenWrt 22.03.4 r20123-38ccc47687"
```

9. Compilation method of OpenWRT source code

9.1. Download OpenWRT source code

1) First, execute the following command to download the openwrt-22.03 branch code

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install -y git
test@test:~$ git clone https://github.com/orangepi-xunlong/openwrt.git -b openwrt-22.03
```

2) After downloading the OpenWRT code, the following files and folders will be included

```
test@test:~/openwrt$ ls
BSDmakefile  Config.in  include  Makefile  README.md  scripts  toolchain
Config  feeds.conf.default  LICENSE  package  rules.mk  target  tools
```

9.2. Compile OpenWRT source code

1) Firstly, install the following dependency packages (currently only tested for compilation on Ubuntu 22.04. If compiling on other versions of Ubuntu systems, please install the corresponding dependency packages according to the error message)

```
test@test:~/openwrt$ sudo apt update
test@test:~/openwrt$ sudo apt install -y ack antlr3 asciidoc autoconf    \
automake autopoint binutils bison build-essential    \
bzip2 ccache cmake cpio curl device-tree-compiler fastjar \
flex gawk gettext gcc-multilib g++-multilib git gperf haveged \
help2man intltool libc6-dev-i386 libelf-dev libglib2.0-dev \
libgmp3-dev libltdl-dev libmpc-dev libmpfr-dev \
libncurses5-dev \libncursesw5-dev libreadline-dev libssl-dev \
libtool lrzsz mkisofs msmtp nano ninja-build p7zip p7zip-full \
```

```
patch pkgconf python2.7 python3 python3-pyelftools \
libpython3-dev qemu-utils rsync scons squashfs-tools \
subversion swig texinfo uglifyjs upx-ucl unzip \
vim wget xmlto xxd zlib1g-dev
```

2) Then execute **./scripts/feeds update -a** and **./scripts/feeds install -a** download dependency package

```
test@test:~/openwrt$ ./scripts/feeds update -a
test@test:~/openwrt$ ./scripts/feeds install -a
```

3) Then choose to use the configuration file for OrangePi 5 Ultra

```
test@test:~/openwrt$ cp configs/orangepi-5-ultra-rk3588_defconfig .config
```

4) Then execute the following command to make the configuration effective

```
test@test:~/openwrt$ make defconfig
```

5) Execute the following command to start compiling the openwrt source code

```
test@test:~/openwrt$ make V=s
```

6) After compilation, the path where the image is generated is:

```
bin/targets/rockchip/armv8/openwrt-rockchip-armv8-xunlong_orangepi-5-ultra-ext4-sysupgrade.img.gz
```

10. Appendix

10. 1. User Manual Update History

Version	Date	Update Explanation
v1.0	2024-10-10	Initial version

10. 2. Image update history

Date	Update Explanation
2024-10-10	Orangepi5ultra_1.0.0_ubuntu_focal_server_linux5.10.160.7z Orangepi5ultra_1.0.0_ubuntu_jammy_server_linux5.10.160.7z Orangepi5ultra_1.0.0_debian_bullseye_server_linux5.10.160.7z Orangepi5ultra_1.0.0_debian_bookworm_server_linux5.10.160.7z Orangepi5ultra_1.0.0_ubuntu_focal_desktop_xfce_linux5.10.160.7z Orangepi5ultra_1.0.0_ubuntu_jammy_desktop_xfce_linux5.10.160.7z Orangepi5ultra_1.0.0_debian_bullseye_desktop_xfce_linux5.10.160.7z Orangepi5ultra_1.0.0_debian_bookworm_desktop_xfce_linux5.10.160.7z Orangepi5ultra_1.0.0_ubuntu_jammy_desktop_gnome_linux5.10.160.7z Orangepi5ultra_1.0.0_ubuntu_jammy_server_linux6.1.43.7z Orangepi5ultra_1.0.0_debian_bookworm_server_linux6.1.43.7z Orangepi5ultra_1.0.0_ubuntu_jammy_desktop_xfce_linux6.1.43.7z Orangepi5ultra_1.0.0_debian_bookworm_desktop_xfce_linux6.1.43.7sz OrangePi5Ultra_RK3588_Android13_v1.0.0.tar.gz OrangePi5Ultra_RK3588_Android13_spi-nvme_v1.0.0.tar.gz Opios-droid-aarch64-opi5ultra-24.09-linux5.10.160.tar.gz Opios-droid-aarch64-opi5ultra-24.09-linux5.10.160-spi-nvme.tar.gz Opios-arch-aarch64-gnome-opi5ultra-24.08-linux5.10.160.img.xz

	openwrt-rockchip-armv8-xunlong_orangepi-5ultra-ext4-sysupgrade_v1.0.img.gz * Initial version