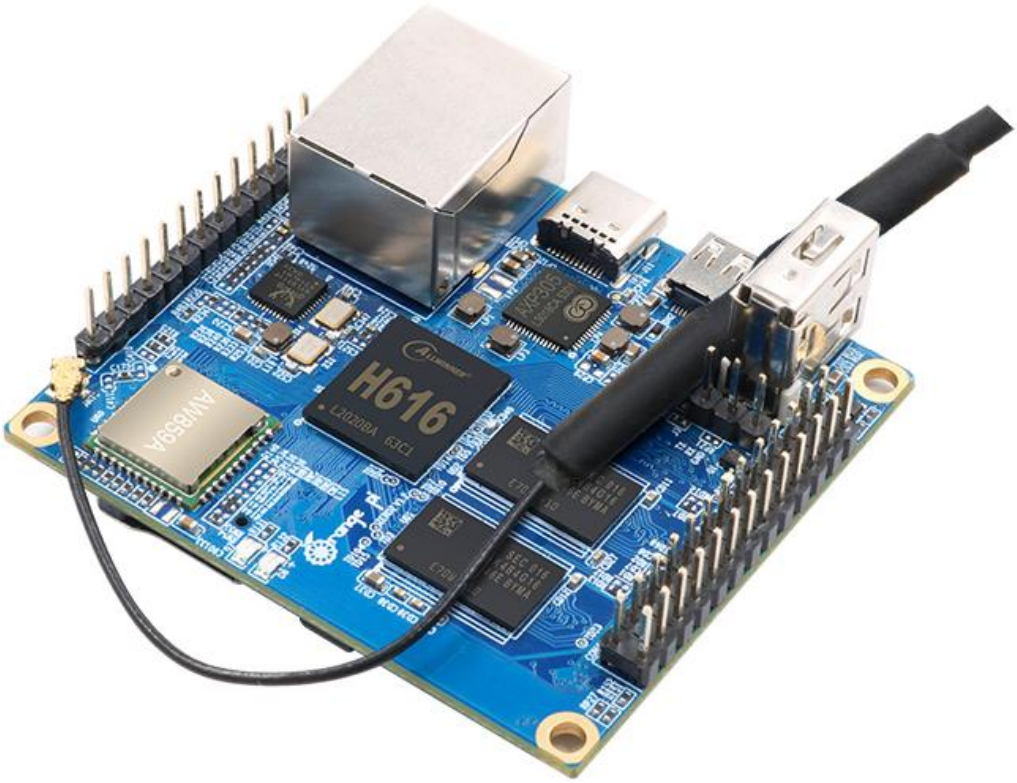
Orange Pi Zero 2

User Manual



Contents

Version Update History

Version	Date	Update Instructions
v3.6	2022-04-20	1. Linux system USB microphone test method 2. How to Test Audio in Desktop Linux System 3. Linux5.16 I2C, SPI, UART and PWM test methods 4. Linux: Tencent ncnn neural network forward computing framework test method 5. Installation method of Ubuntu20.04 ROS 2 Galactic 6. Debian12 Linux5.16 system GPU test instructions 7. Add a new version of orangepi-build instructions 8. Add some cautionary notes
v3.7	2022-05-20	1. Support Ubuntu22.04 server and desktop image 2. Ubuntu22.04 Linux5.16 GPU test instructions 3. Ubuntu22.04 method of installing browser 4. Setting method for automatic login of root user in Linux desktop system 5. How to disable the desktop in the Linux desktop version system 6. Linux: The method of manually expanding the capacity of the rootfs partition in the TF card 7. Linux: Method to reduce the capacity of rootfs partition in TF card 8. How to hide the mouse cursor on the touch screen of the Linux desktop version system 9. Linux: installation and testing method of face_recognition face recognition library 10. Debian: How to install OpenMediaVault 5.x and 6.x 11. Installation method of Ubuntu 20.04 ROS 1 Noetic 12. Linux: How to install Pi-hole 13. Debian10: Tensorflow installation method 14. Linux: Introduction to the use of GotoHTTP

1. Basic Features of Orange Pi Zero 2

1.1. What is Orange Pi Zero 2

Orange Pi is an open source single-board -computer, a new generation of arm64 development boards, which can run Android TV 10, Ubuntu and Debian . The Orange Pi Zero 2 uses the Allwinner H616 Rockchip and has 1GB DDR3 .

1.2. Purpose of Orange Pi Zero 2

We can use it to build:

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

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

1.3. Who is Orange Pi Zero 2 designed for?

Orange Pi Zero2 is for anyone who wants to start creating with technology – not just consuming it. It's a simple, fun, useful tool that you can use to start taking control of the world around you.

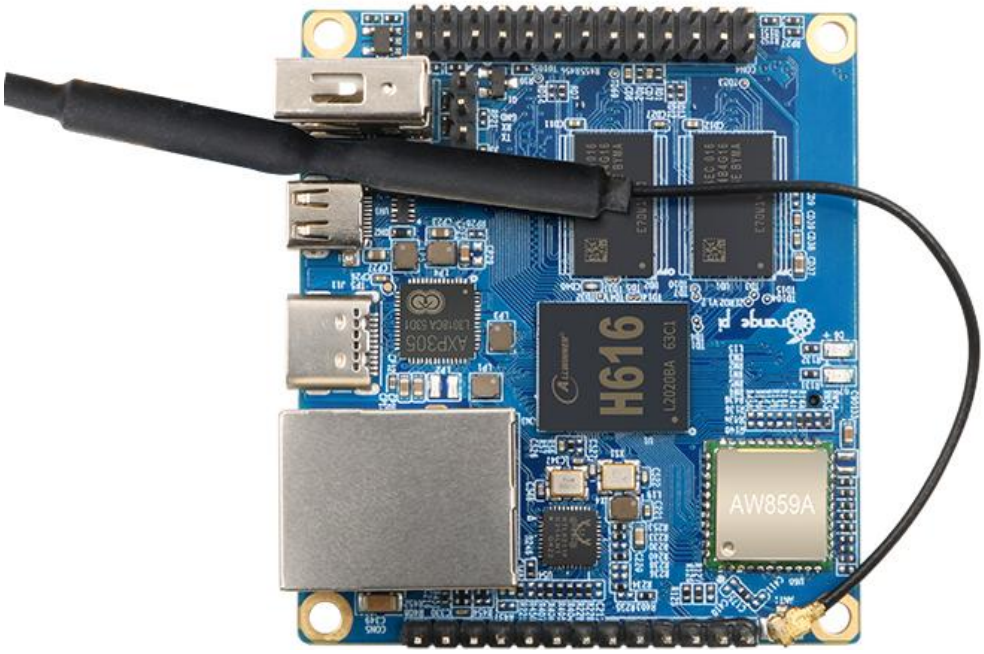
1.4. Hardware Features of Orange Pi Zero 2

Hardware Introduction	
CPU	Allwinner H616 64-bit high-performance Quad-core Cortex-A53 processor
GPU	Mali G31 MP2

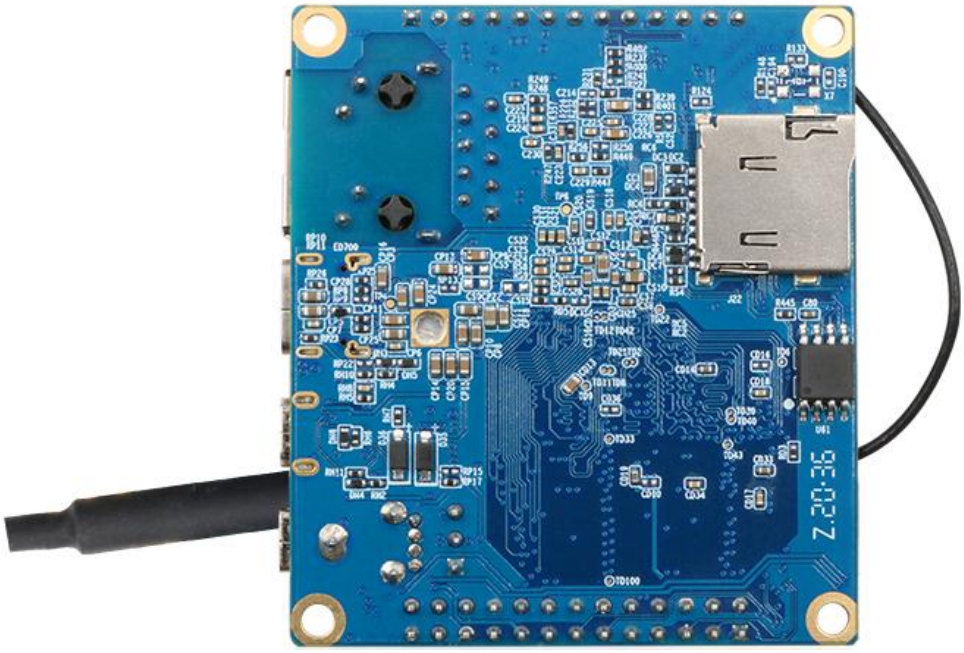
	Supports OpenGL ES 1.0/2.0/3.2、OpenCL 2.0
Memory(SDRAM)	1GB DDR3 (Shared with GPU）
Onboard Storage	TF card slot、2MB SPI Flash
Onboard Network	Support 1000M/100M/10M Ethernet
WIFI+BT	• AW859A Chip、Support IEEE 802.11 a/b/g/n/ac、BT5.0
Video Outputs	• Micro HDMI 2.0a • TV CVBS output, Support PAL/NTSC（Via 13pin interface board）
Audio output	• Micro HDMI • 3.5mm audio port (Via 13pin interface board)
Power Source	USB Type C interface input
USB 2.0 Ports	3*USB 2.0 HOST （Two of them are via 13pin interface board）
26pin header	With I2Cx1、SPIx1、UARTx1 and multiple GPIO ports
13pin header	With USB 2.0 HOSTx2、TV-OUT、LINE OUT、IR-RX、and 3*GPIO ports
Debug serial port	UART-TX、UART-RX and GND
LED	Power led & Status led
IR receiver	Support IR remote control (via 13pin interface board)
Supported OS	Android10 TV、Ubuntu、Debian and so on
Appearance specification	
Dimension	85mm×56mm
Weight	30g
range Pi™ is a trademark of the Shenzhen Xunlong Software Co.,Limited	

1.5. The Top view and Bottom view of Orange Pi Zero 2

Top view：

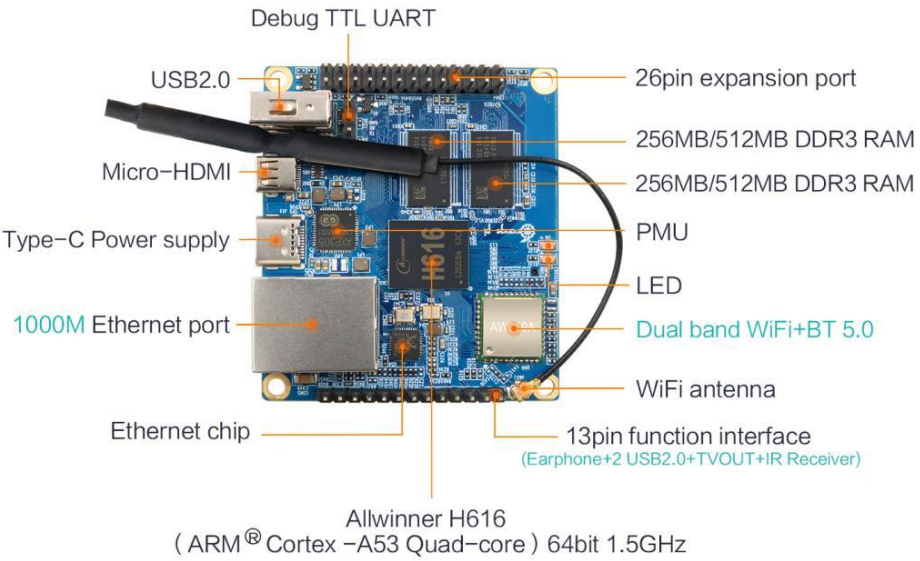


Bottom view：

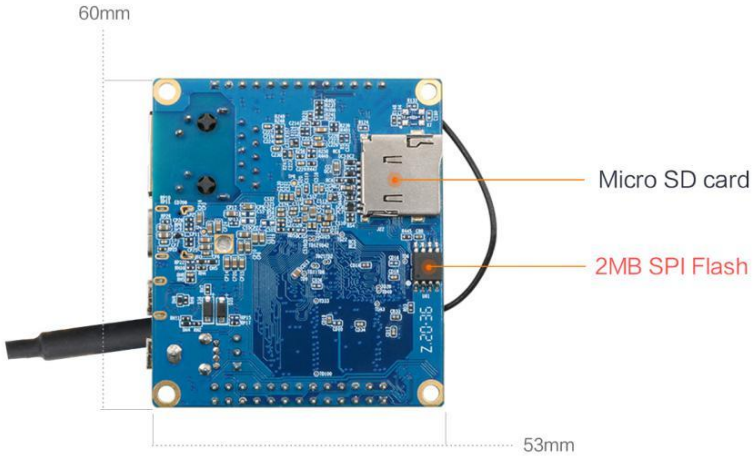


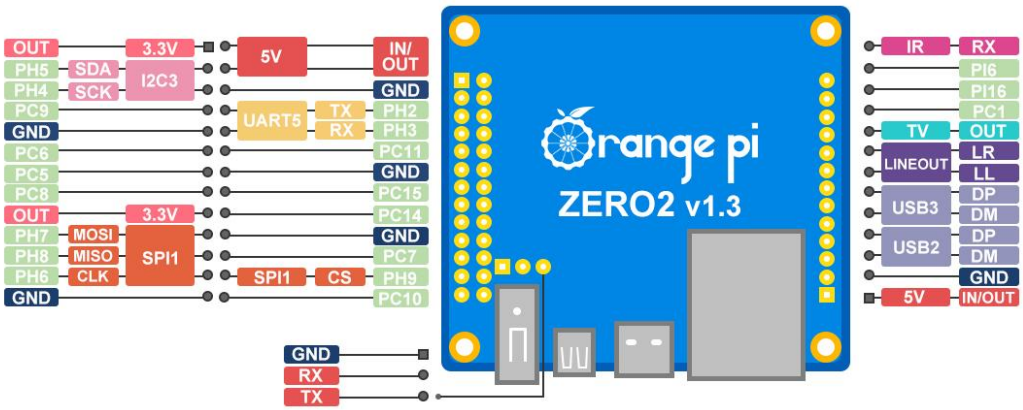
1.6. The Ports Detail of Orange Pi Zero 2

Top view



Bottm view





The diameter of the four positioning holes is 3.0mm

2. Introduction to the use of the development board

2.1. Prepare the necessary accessories

1) TF card, class10 or above high-speed SanDisk card with a minimum capacity of 8GB

Use other brands of TF cards (not SanDisk TF cards), as shown in the picture below (including but not only to these cards), some friends have reported that there will be problems during the system startup process, such as the system being stuck halfway during startup, Or the reboot command cannot be used normally, and it was finally solved after change to using the SanDisk TF card. So if you are using a non-SanDisk TF card and find that problems during system startup or use, please replace the TF card to the SanDisk TF card and then test

some TF cards have system startup problems on Orange Pi Zero 2

In addition, the TF card that can be used normally on other types of development boards does not guarantee that the Orange Pi Zero 2 will also start normally by using this TF card. Please pay special attention to this point.

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

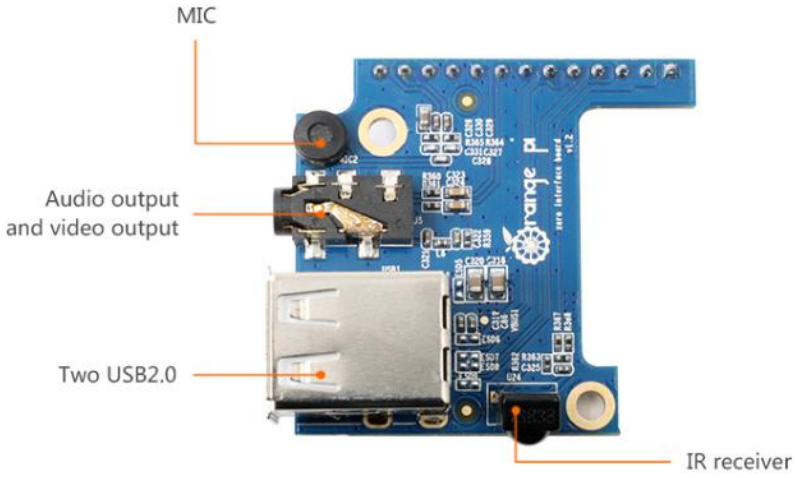
3) Micro HDMI to HDMI cable, used to connect the development board to an HDMI monitor or TV for display

Note, please do not use the wider Micro HDMI adapter shown in the figure below. Due to the small distance between the USB interface, Micro HDMI interface and Type-C power interface of the development board, the three may not be able to be used at the same time. Plug into the development board.

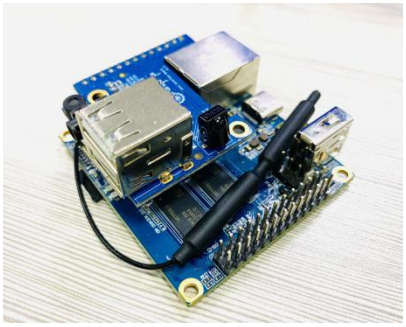
4) Power supply, if there is a 5V/2A or 5V/3A power supply head, you only need to prepare a data cable with the USB to Type C interface like as shown in the picture on the left below, and you can also use a cable similar to the picture shown on the right below. 5V/2A or 5V/3A high-quality USB Typc C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

5) 13pin expansion board

a. The actual expansion board is as follows



b. The way to insert the expansion board into the development board is as follows, note: not to inserted in the wrong direction

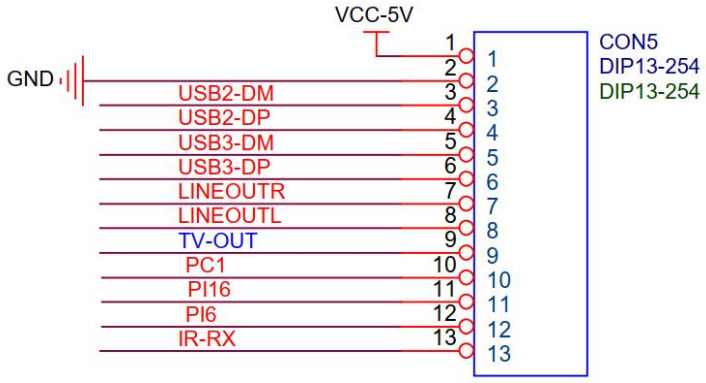


c. The 13pin pin header on the Orange Pi Zero 2 development board can be connected to the expansion board to expand the functions that are not available on the development board. The functions that the expansion board can use include:

1	Mic	**No support, no support, no support! ! !** The 13pin expansion board is a general-purpose expansion board, suitable for various development boards of Orange Pi, but the 13pin interface of the Orange Pi Zero2 does not have the Mic function, so although there is a Mic on the 13pin expansion board, it is on the Orange Pi Zero 2. Can not be used, the 13pin expansion board is mainly used to expand other functions except Mic on the Orange Pi Zero 2.

		If you need to use the MIC function, it can be extended through the USB interface,Chapter 3 of the manual describes how to use the USB MIC.
2	Analog audio and video output interface	Support, it can be used to connect headphones to play music, or connect to TV through AV cable to output analog audio and video signals (Android only).
3	USB 2.0 Host x 2	Supported, used to connect USB keyboard, mouse and USB storage device.
4	IR function	Support, can control Android system through infrared remote control

d. The schematic diagram about the 13pin pin header of the Orange Pi Zero 2 development board is as follows



6) USB interface mouse and keyboard, as long as it is a standard USB interface mouse and keyboard, the mouse and keyboard can be used to control the Orange Pi development board

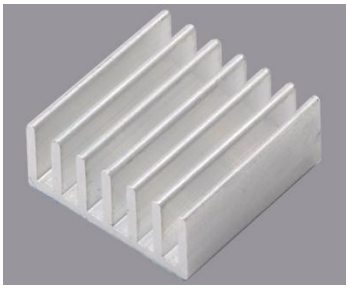
7) Infrared remote control, mainly used to control the Android system

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

8) Fast or Gigabit Ethernet cable to connect the development board to the Internet

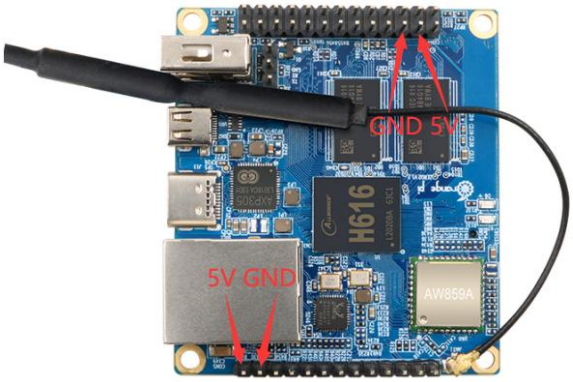
9) AV video cable, if you want to display video through the AV interface instead of the HDMI interface, then you need to connect the development board to the TV through the AV video cable

10) Heat sink, if you are worried that the temperature of the development board is too high, you can add a heat sink, and the heat sink can be attached to the H616 chip



11) 5V cooling fan, as shown in the figure below, there are 5V and GND pins on the 26pin and 13pin interfaces of the development board that can be connected to the cooling fan. The distance between the 26pin and 13pin pin headers is 2.54mm, and the power interface of the cooling fan refers to this Specifications to buy

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



12) Matching shell, transparent shell and black shell are available

Note that the matching case of the Orange Pi Zero 2 cannot hold the 13pin expansion board.

13) USB to TTL module and DuPont cable, when using the serial port debugging function, USB to TTL module and DuPont cable are required to connect the development board and computer

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

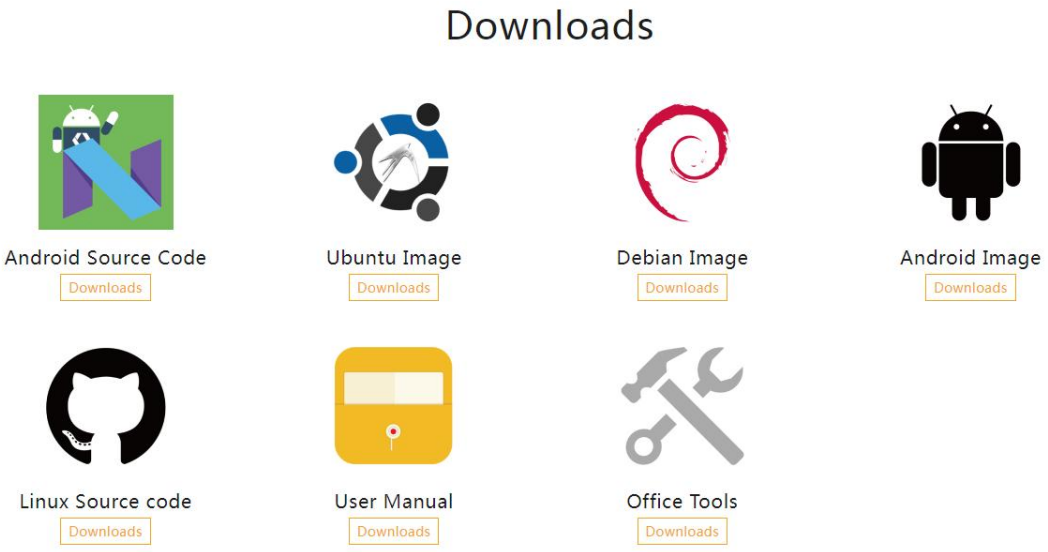
14) A PC with Ubuntu and Windows operating systems installed

1	Ubuntu14.04.6 PC	Optional, used to compile Android source code
2	Ubuntu18.04 PC	Optional, used to compile Linux source code
3	Windows PC	For burning Android and Linux images

2.2. Download the image of the development board and related materials

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

http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-Zero-2.html



2) The data mainly includes

a. **Android source code：** Save on Google network disk

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

c. **User Manual and Schematics**：Chip related data sheets will also be placed here
d. **Official tool:** Mainly includes the software that needs to be used in the use of the development board
e. **Android Image**：Save on Google network disk
f. **Ubuntu Image**：Save on Google network disk
g. **Debian Image**：Save on Google network disk

2.3. Method of Burning Linux Image to TF Card Based on Windows PC

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

2.3.1. The way to use balenaEtcher to burn a Linux image

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

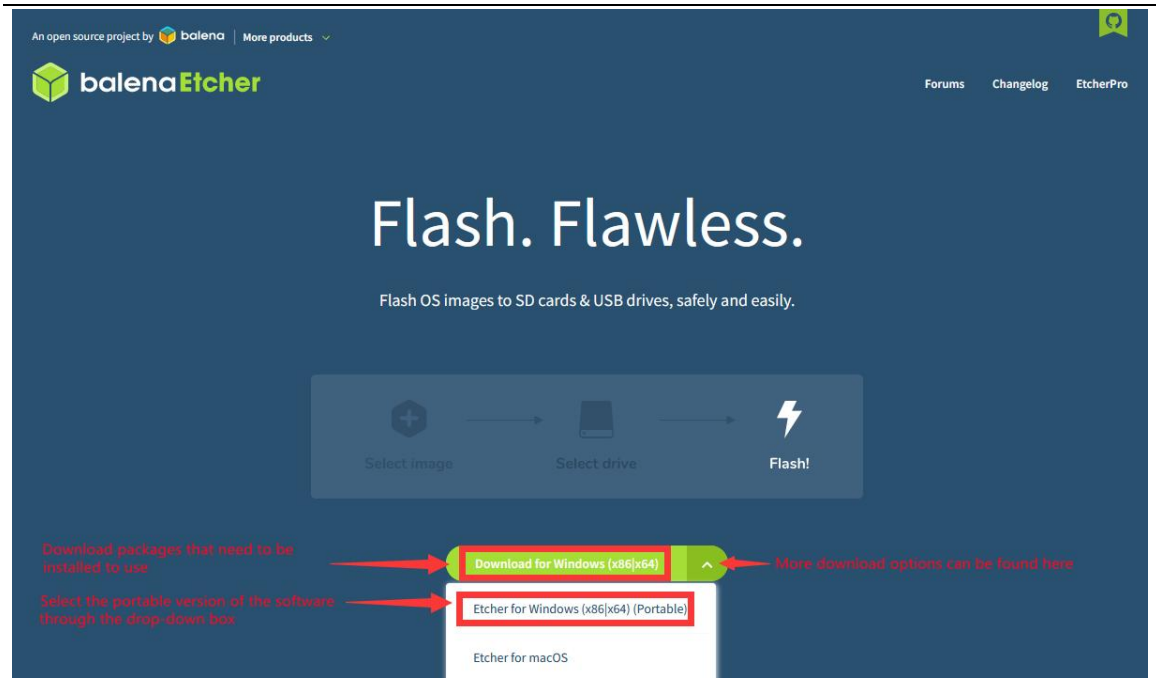
2) Then insert the TF card into the card reader and insert it into the computer

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

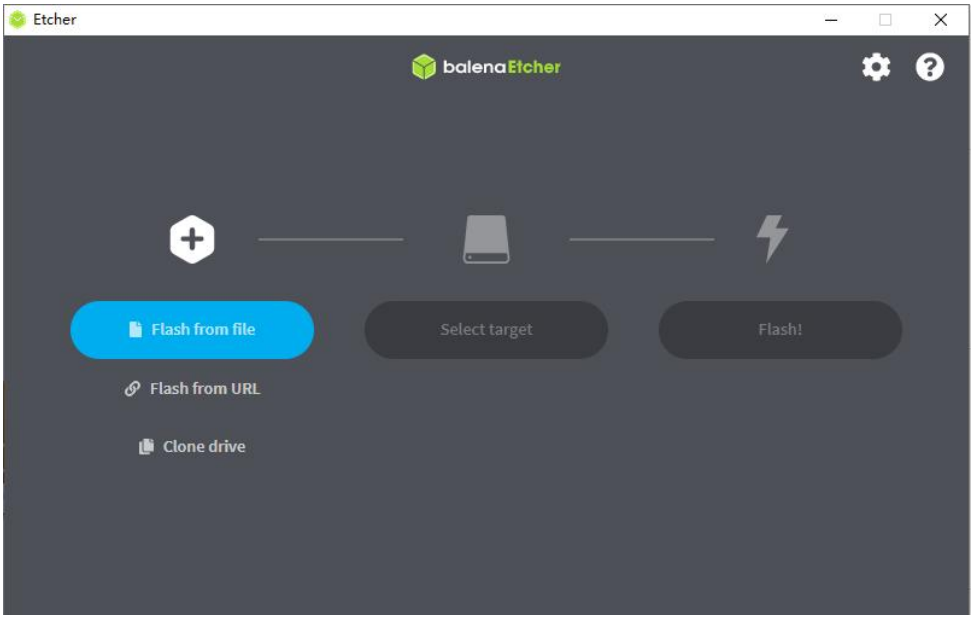
4) Then download the burning software for the Linux image——**balenaEtcher**，the download address is

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

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



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



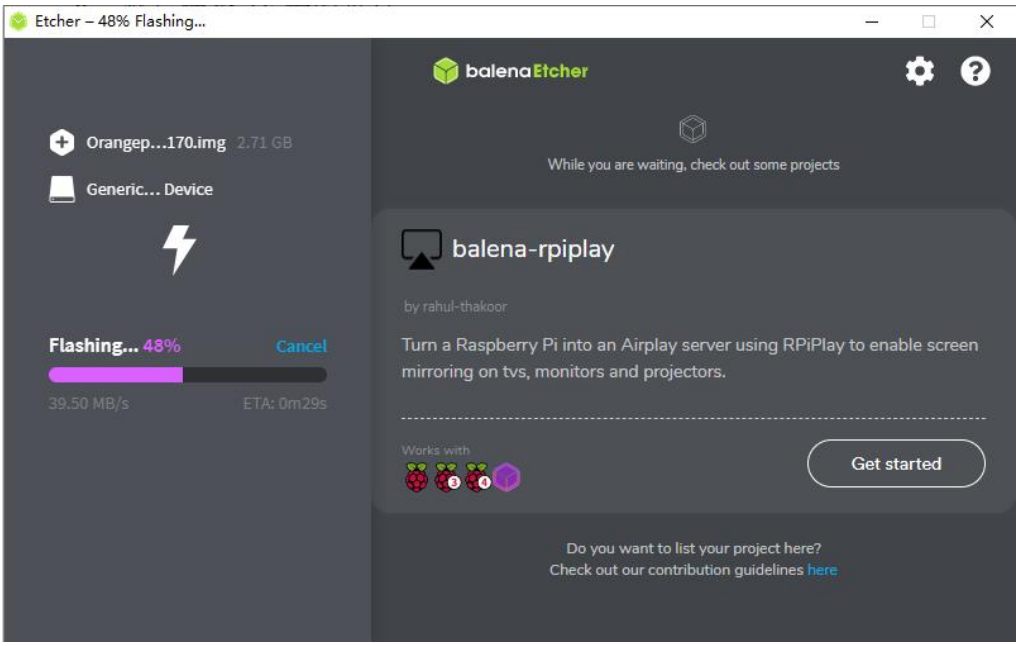
7) The specific steps to use balenaEtcher to burn a Linux image are as follows

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

b. Then select the symbol of the TF card

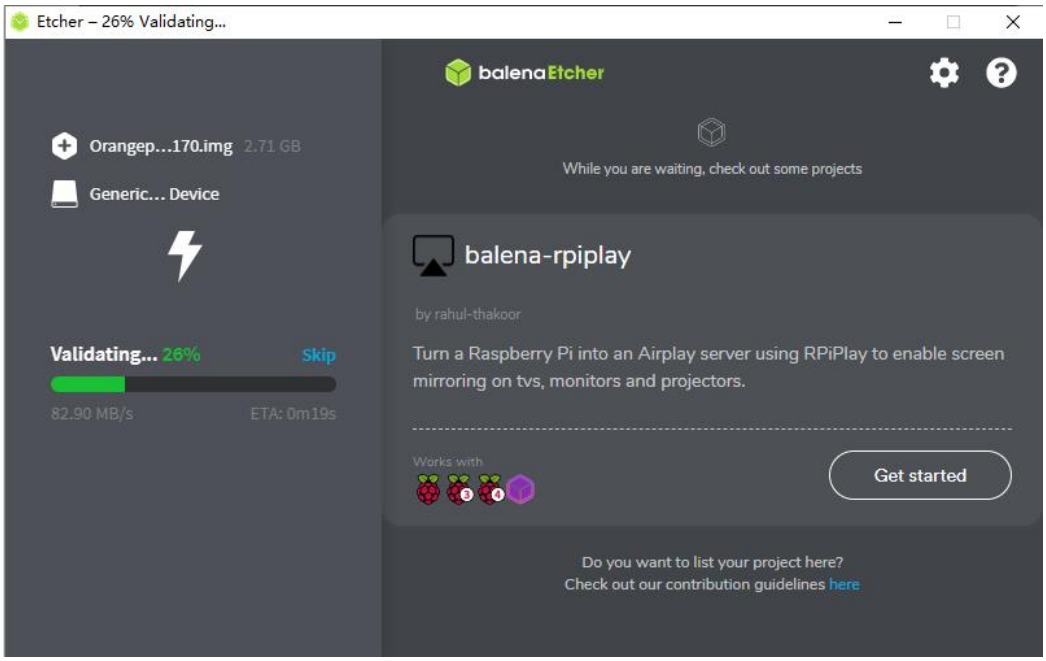
c. Finally, click Flash to start burning the Linux image to the TF card

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

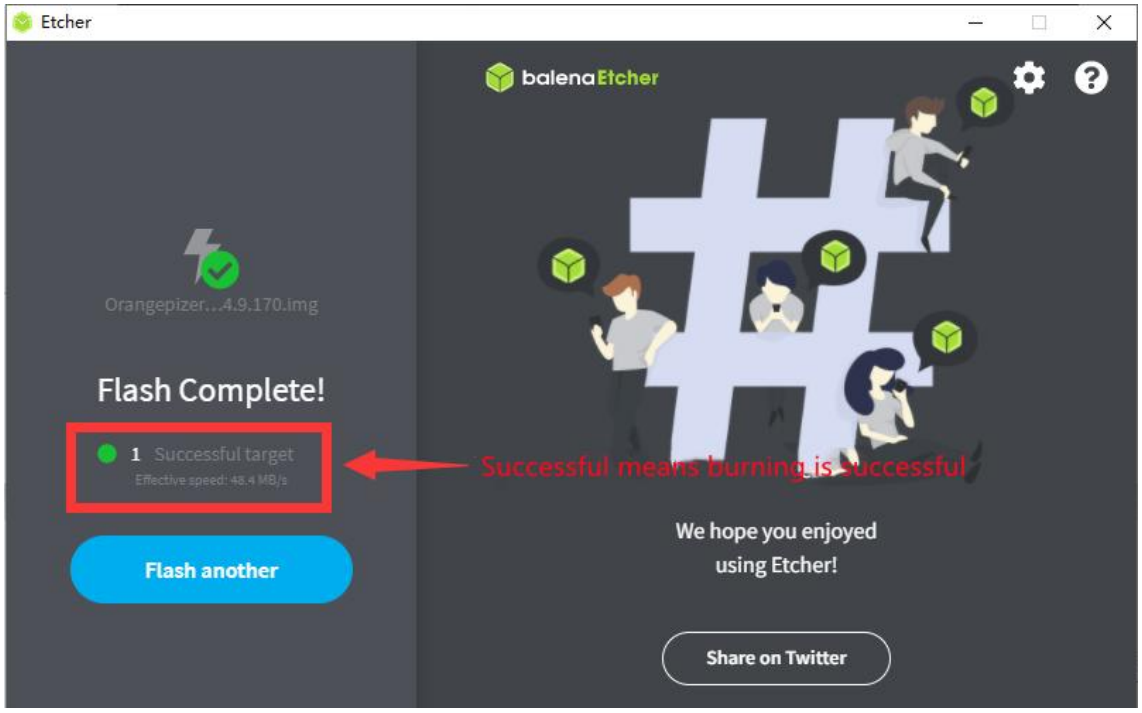


9) After the Linux image is burned, balenaEtcher will also verify the image burned to the TF card by default to ensure that there is no problem in the burning process. As shown in the figure below, a green progress bar indicates that the image has been burned, and

balenaEtcher is verifying the burned image.



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



2.3.2. The Way to use Win32Diskimager to burn Linux image

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

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

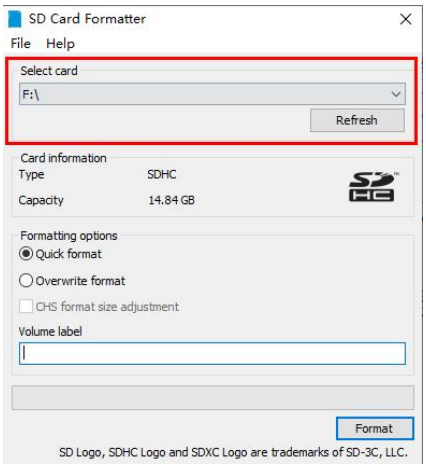
3) Then format the TF card

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

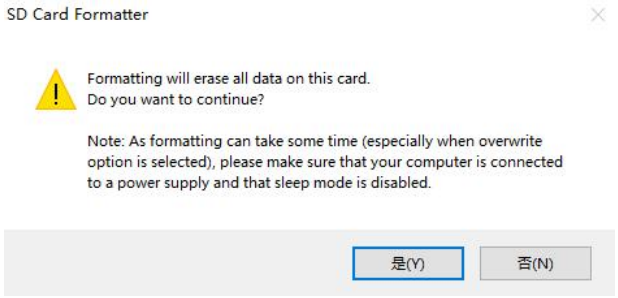
https://www.sdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip

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

c. If only the TF card is inserted into the computer, the symbol of the TF card will be displayed in the "**Select card**" column. If multiple USB storage devices are inserted into the computer, you can select the symbol corresponding to the TF card through the drop-down button.



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



e. After formatting the TF card, the message shown below will pop up, click OK.

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

5) Use **Win32Diskimager** to burn Linux image to TF card

a. The download page of Win32Diskimager is

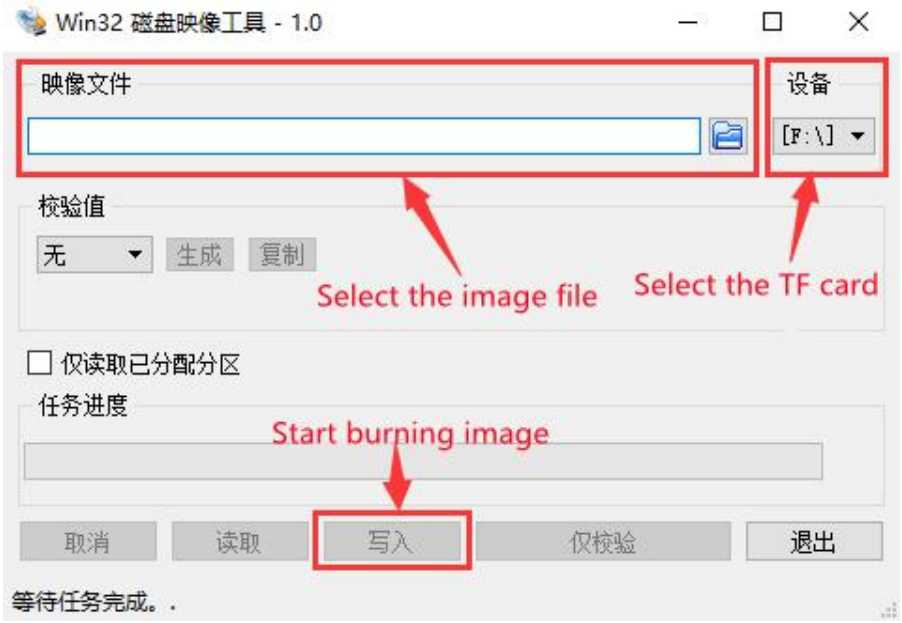
http://sourceforge.net/projects/win32diskimager/files/Archive/

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

a) First select the path of the image file

b) Then confirm that the symbol of the TF card is consistent with the one displayed in the "**Device**" column

c) Finally click "**Write**" to start burning



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

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

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

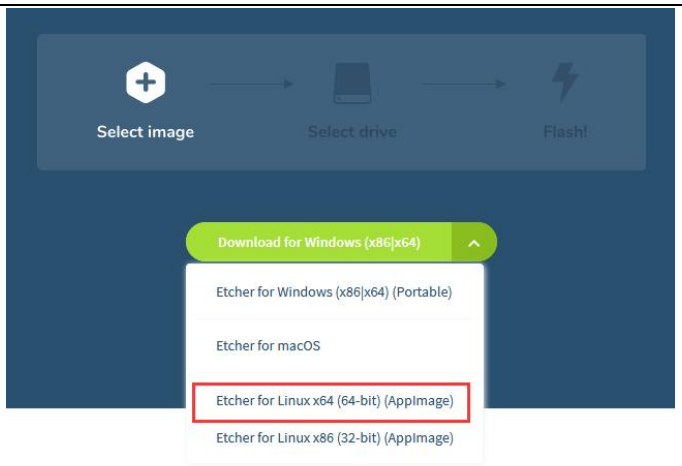
1) First prepare a TF card with 8GB or more. The transmission speed of the TF card must be **class10** or above. It is recommended to use a TF card from a brand such as SanDisk

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

3) Download balenaEtcher software, the download address is

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

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



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

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

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

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

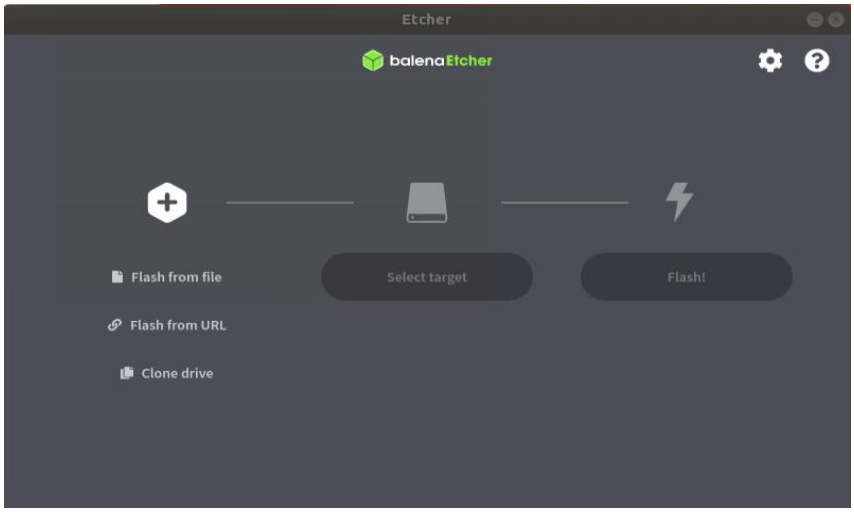
```
test@test:~$ 7z x Orangepizero2_2.2.0_ubuntu_focal_desktop_linux4.9.170.7z
test@test:~$ ls Orangepizero2_2.2.0_ubuntu_focal_desktop_linux4.9.170.*
Orangepizero2_2.2.0_ubuntu_focal_desktop_linux4.9.170.7z
Orangepizero2_2.2.0_ubuntu_focal_desktop_linux4.9.170.sha      #checksum file
Orangepizero2_2.2.0_ubuntu_focal_desktop_linux4.9.170.img      #image file
```

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

download again

```
test@test:~$ sha256sum -c *.sha
orangepizero2_2.2.0_ubuntu_bionic_server_linux4.9.170.img: success
```

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

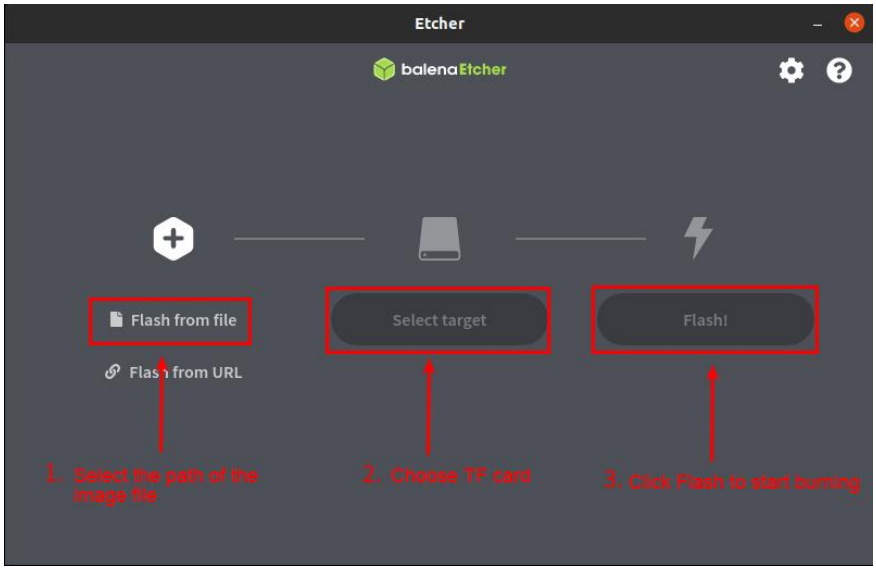


9) The specific steps to use balenaEtcher to burn a Linux image are as follows

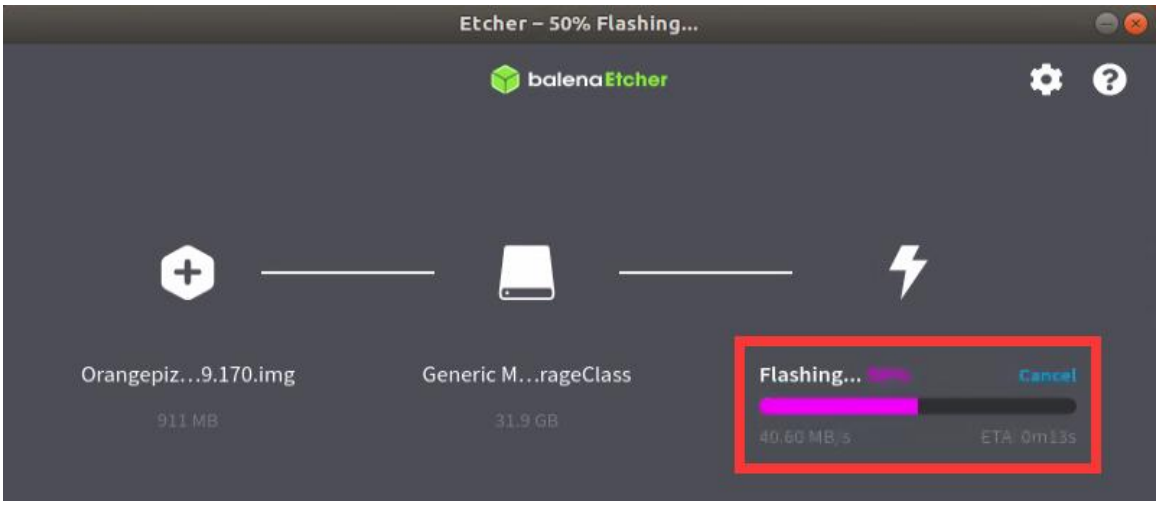
a. First select the path of the Linux image file to be burned

b. Then select the symbol of the TF card

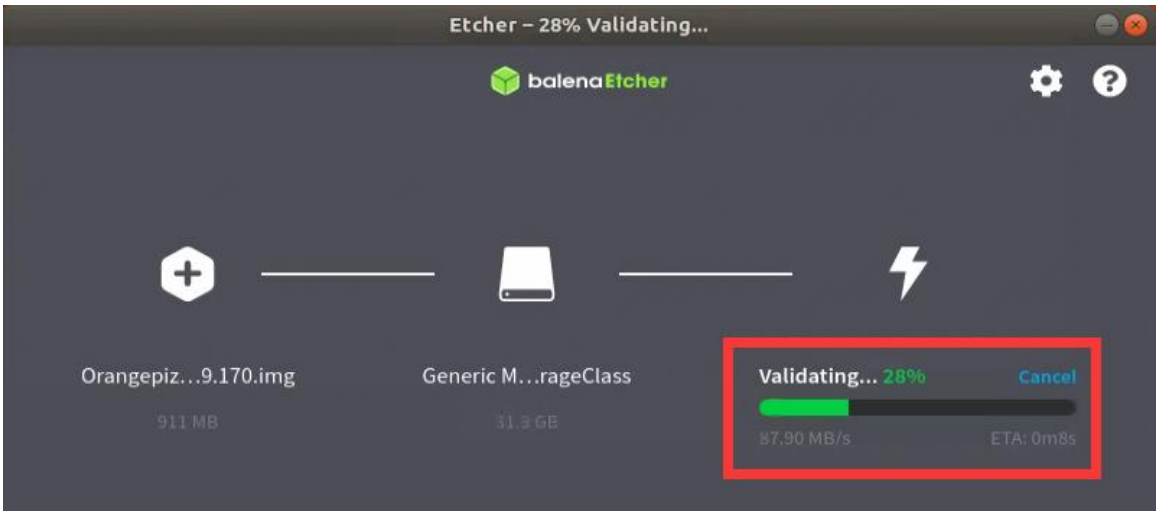
c. Finally, click Flash to start burning the Linux image to the TF card



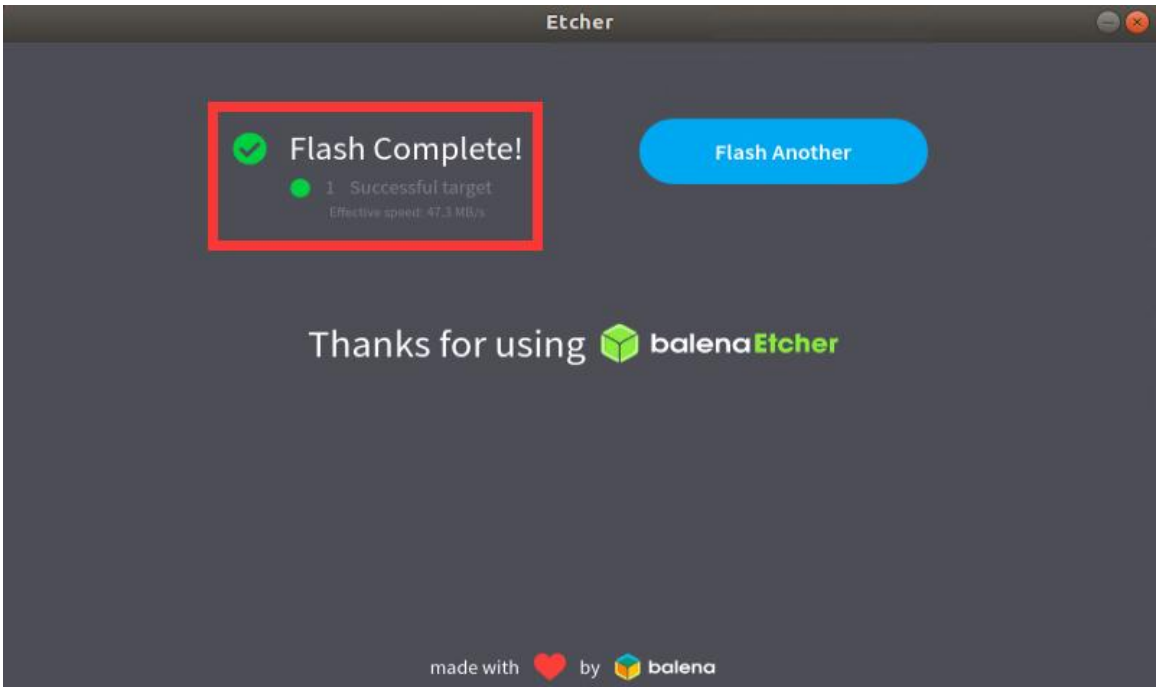
10) The interface displayed in the process of balenaEtcher burning the Linux image is shown in the figure below. In addition, the progress bar shows purple to indicate that the Linux image is being burned to the TF card.



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



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



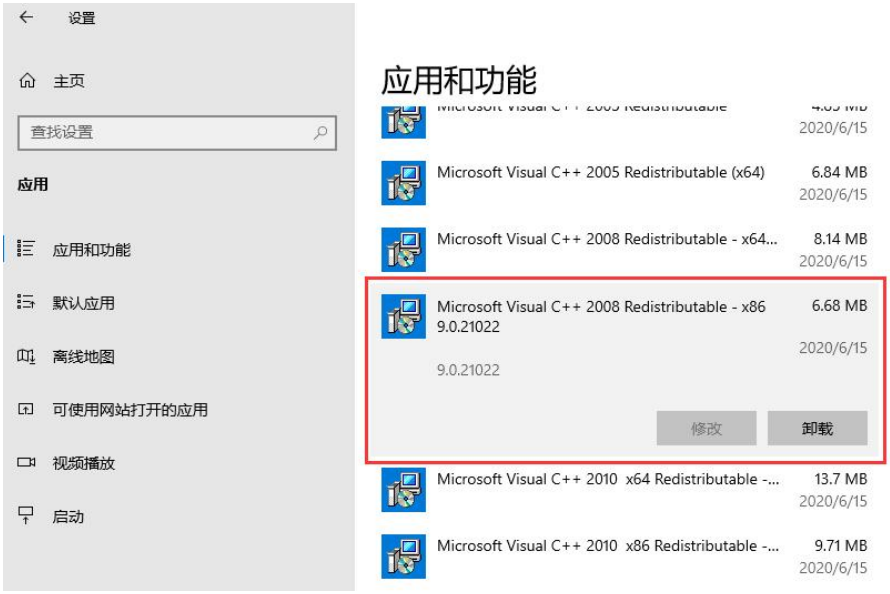
2.5. The Way to burn Android image to TF card

The Android image of the development board can only be burned into the TF card using the PhoenixCard software under the Windows platform, and the version of the PhoenixCard software must be PhonixCard-4.2.8.

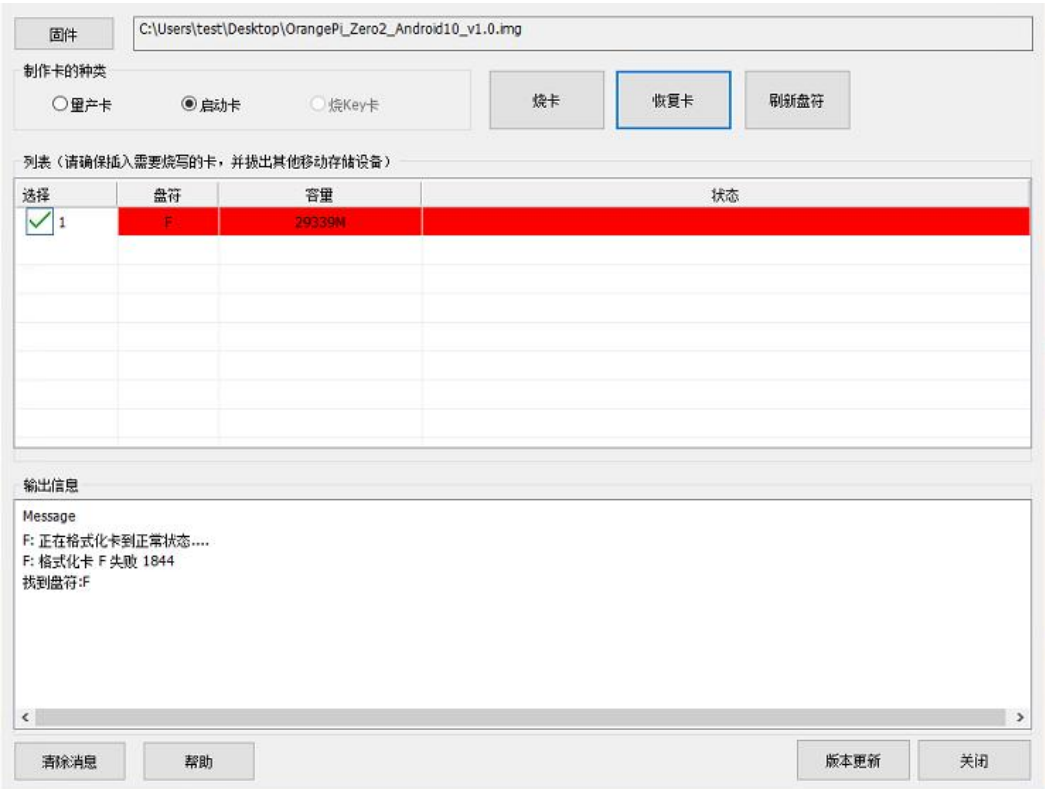
Please do not use software that burns Linux images, such as Win32Diskimager or balenaEtcher, to burn Android images.

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

1) First make sure that the Windows system has installed **Microsoft Visual C++ 2008 Redistrbutable - x86**

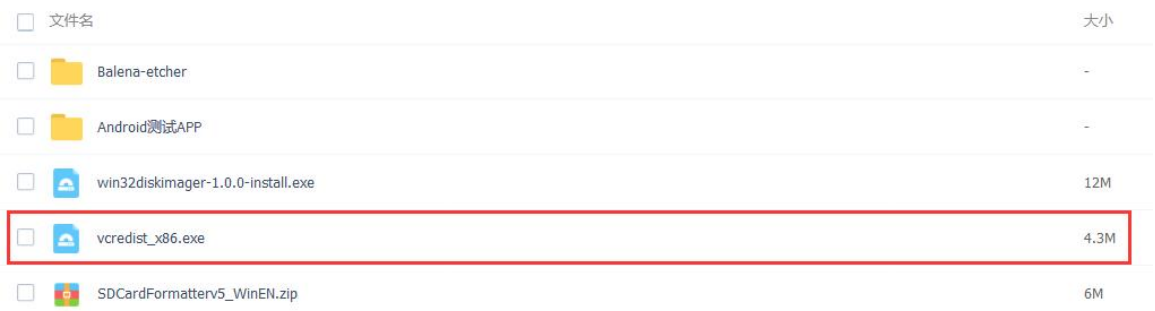


2) If **Microsoft Visual C++ 2008 Redistrbutable - x86** is not installed，Using **PhoenixCard** to format TF card or burn Android image will prompt the error like as the following



3) The installation package of **Microsoft Visual C++ 2008 Redistrbutable - x86** can be

downloaded from the official tool of Orange Pi Zero 2, or you can download it from Microsoft's official website.



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

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

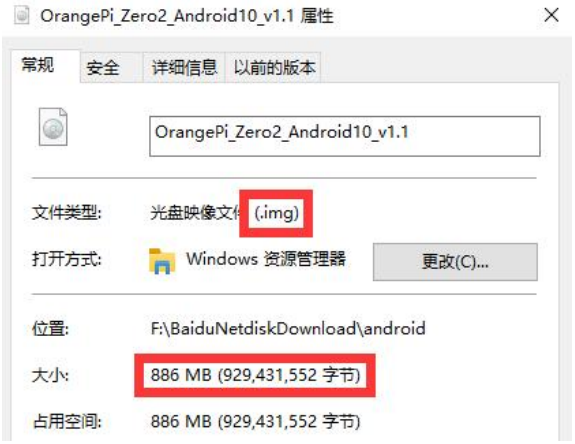
6) Download the Android10 image and PhoenixCard burning tool from the data download page of Orange Pi, please make sure that the version of PhonenixCrad tool is PhonixCard-4.2.8, please do not use PhonixCard software lower than 4.2.8 to burn Orange Pi Zero 2 Android 10 image, the Android 10 image flashed by the PhonixCard tool earlier than this version may have problems

7) Then use the decompression software to decompress the downloaded Android image compression package. In the decompressed file, the file ending with "**.img**" is the Android image file

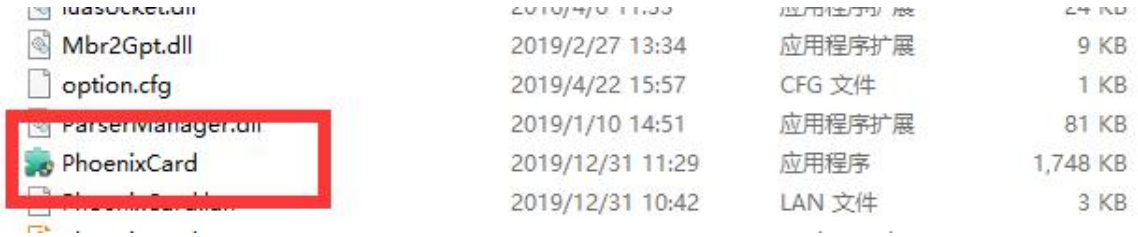
a. **If you don't know how to decompress the compressed package of the Android image, you can install a 360 compression software**

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

c. The first file close to 900M is the Android image file to be burned. Check its properties as shown in the figure below



8) Then use the decompression software to decompress **PhonixCard4.2.8.zip**, this software does not need to be installed, just find the PhoenixCard in the decompressed folder and open it



9) After opening PhoenixCard, if the TF card is recognized normally, the disk letter and memory storage of the TF card will be displayed in the middle list. Please make sure that

the displayed disk letter is consistent with the disk letter of the TF card you want to burn. If there is no display, you can try to unplug the TF card, or click the "**Refresh disk letter**" button in PhoenixCard

10) After confirming the disk letter, first format the TF card and click the "**Restore Card**" button in PhoenixCard (if the "**Restore Card**" button is gray and cannot be pressed, you can click the "Refresh Disk Letter" button first)

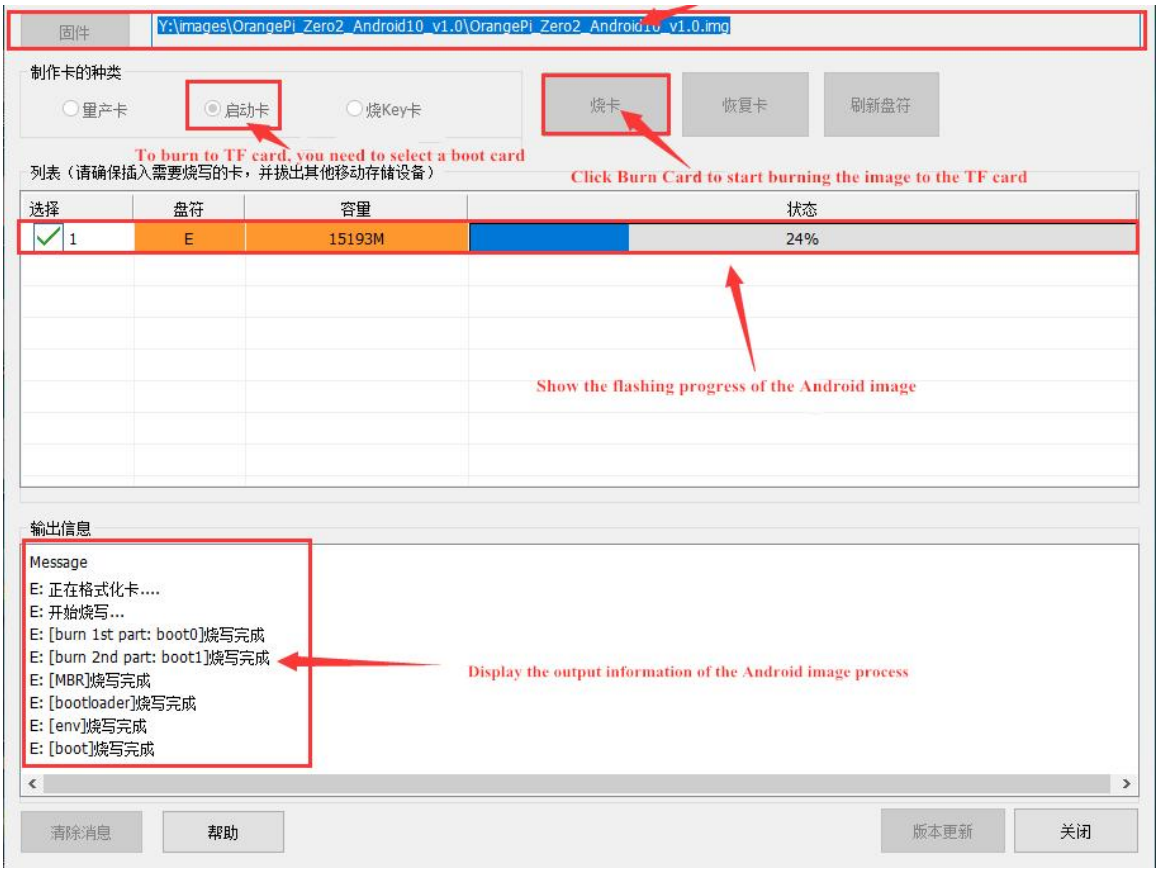
If there is a problem with formatting, please try to unplug and insert the TF card and then test. If there is still a problem after re-plugging the TF card, you can restart the Windows computer or try another computer.

11) Then start writing Android image to TF card

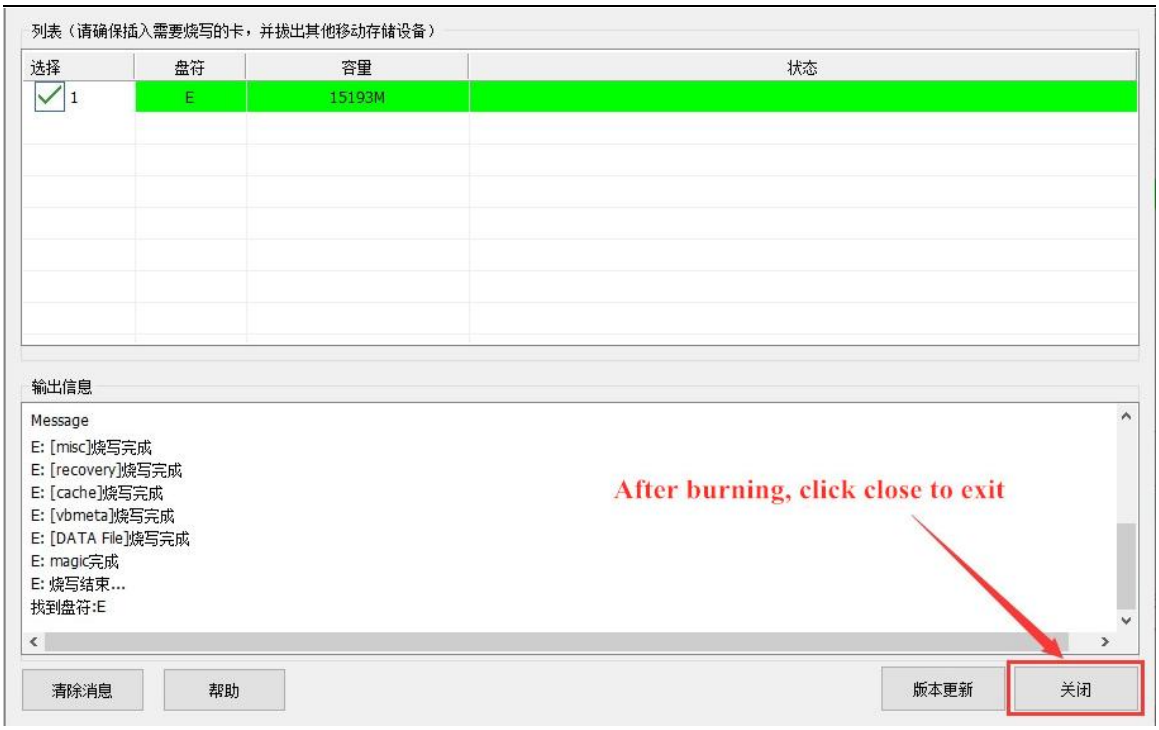
a. First select the path of the Android image in the "**Firmware**" column

b. Select "**Startup Card**" in "**Type of Production Card**"

c. Then click the "**burn card**" button to start burning



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



After burning the Android system, you can only see a 128 MB partition on the TF card in Windows, and the displayed partition is as shown in the figure below (some computers may pop up more than a dozen disk partitions, but only the 128 MB partition can be opened.), please note that this is normal, please don't think that the TF card is burnt out. The reason for this is because the Android system has a total of 17 partitions, and the other 16 partitions cannot be recognized normally in the Windows system. At this point, please feel free to unplug the TF card and insert it into the development board to start.

After the Android system starts, use the following command to see the 17 partitions in the TF card:

```
console:/ # ls /dev/block/mmcblk0*
/dev/block/mmcblk0     /dev/block/mmcblk0p14 /dev/block/mmcblk0p4
/dev/block/mmcblk0p1   /dev/block/mmcblk0p15 /dev/block/mmcblk0p5
/dev/block/mmcblk0p10 /dev/block/mmcblk0p16 /dev/block/mmcblk0p6
/dev/block/mmcblk0p11 /dev/block/mmcblk0p17 /dev/block/mmcblk0p7
/dev/block/mmcblk0p12 /dev/block/mmcblk0p2  /dev/block/mmcblk0p8
/dev/block/mmcblk0p13 /dev/block/mmcblk0p3  /dev/block/mmcblk0p9
console:/ #
```

Using the df -h command, you can see that the 32 GB TF card has about 26 GB of space after burning the Android system (the 17 partitions will not all be mounted to the Android system, focus on these to see partitions).

```
console:/ # df -h
Filesystem              Size  Used Avail Use% Mounted on
/dev/block/dm-0         588M  586M  1.7M 100% /
tmpfs                   485M  616K  484M   1% /dev
tmpfs                   485M     0  485M   0% /apex
/dev/block/dm-1         112M  112M  404K 100% /vendor
/dev/block/dm-2         106M  106M  332K 100% /product
/dev/block/mmcblk0p17    27G  895M   26G   4% /data
/dev/block/mmcblk0p7    614M  512K  613M   1% /cache
/dev/block/mmcblk0p11    11M   72K   11M   1% /metadata
tmpfs                   485M     0  485M   0% /storage
/data/media              27G  895M   26G   4% /storage/emulated
/mnt/media_rw/0000-0000 128M  5.3M  122M   5% /storage/0000-0000
/dev/block/mmcblk0p16    16M     0   16M   0% /Reserve0
/dev/block/mmcblk0p1    128M  5.3M  122M   5% /mnt
console:/ #
```

2.6. Start the Orange Pi development board

1) Insert the burnt image TF card into the TF card slot of the Orange Pi development board

2) The development board has a Micro HDMI interface, you can connect the development board to a TV or HDMI display through a Micro HDMI to HDMI cable

3) If you purchased a 13pin expansion board, you can plug the 13pin expansion board into the 13pin interface of the development board

4) Connect the USB mouse and keyboard to control the Orange PI development board

5) The development board has an Ethernet port, which can be plugged into a network cable for Internet access

6) Connect a high-quality power adapter with a 5V/2A (5V/3A can also be used) USB Type C interface

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

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

7) Then turn on the switch of the power adapter. If everything is normal, the HDMI display can see the startup screen of the system.

8) If you want to view the output information of the system through the debug serial port, please use the serial port cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section on how to use the debugging serial port.

The first thing to note is that if the power is turned on without inserting the TF card that has already burned the system, the development board will not have any lights on, including the two LED lights and network port lights on the development board, So please don't judge the quality of the development board by this method.

The Way to judge whether the system has started normally:

1) If the HDMI display is connected, the judgment method is very simple. As long as the HDMI display normally displays the interface of the system, it means that the system has been started normally;

2) If there is no HDMI display, you can connect the development board to the computer through the serial cable, and check the startup status of the system by debugging the log information output by the debug serial port. If the serial port output stops at the login interface of the terminal, it means that the system has been started normally;

3) If there is no HDMI display and debug serial ports cable, you can judge the startup status of the system through the two LED lights on the development board. If the green LED light is on, the system has generally started normally. If it is powered on and wait for a period of time After that, only the red LED light is on, or the red and green LED lights are not on, indicating that the system has not started normally.

If the system fails to start or cannot enter the login interface normally, please check the following step first

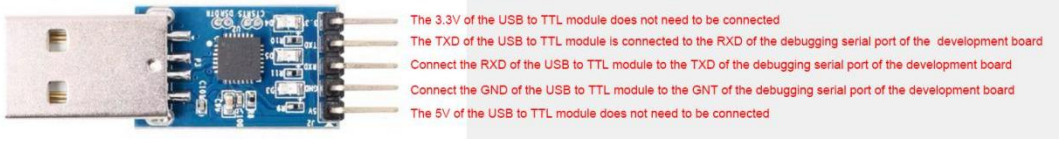
1) Check whether the downloaded image is damaged, which can be judged by

calculating the checksum attached to the image;
2) If there is any problem in the process of burning the image to the TF card, you can re-burn the image and test it again;
3) Make sure there is no problem with the power adapter and the power cord, you can try another one;
4) Make sure that the TF card meets the requirements of the Orange Pi development board. If there is an extra TF card, you can try to change the TF card and then burn the image and test it again;
5) If all of the above are OK, please save the output log of the debugging serial port during the system startup process (preferably in the form of txt text instead of taking pictures), and then report the problem to the customer service.

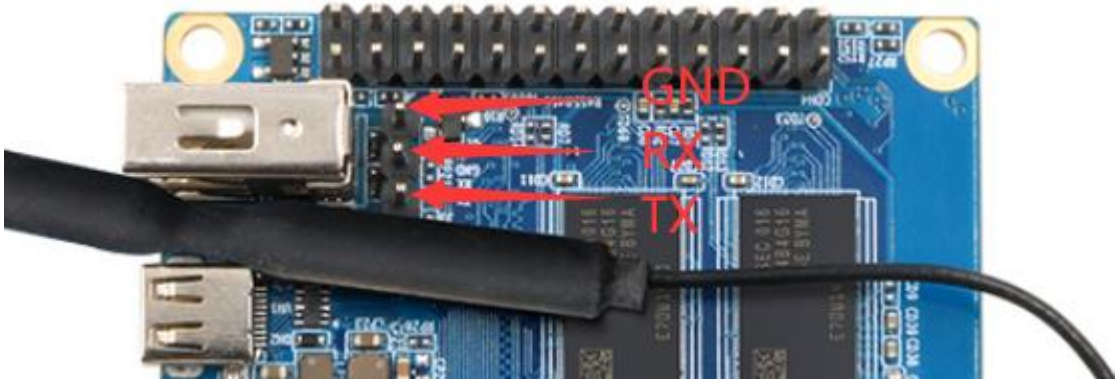
2.7. The method to debug the serial port

2.7.1. Connection instructions for debugging serial port

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



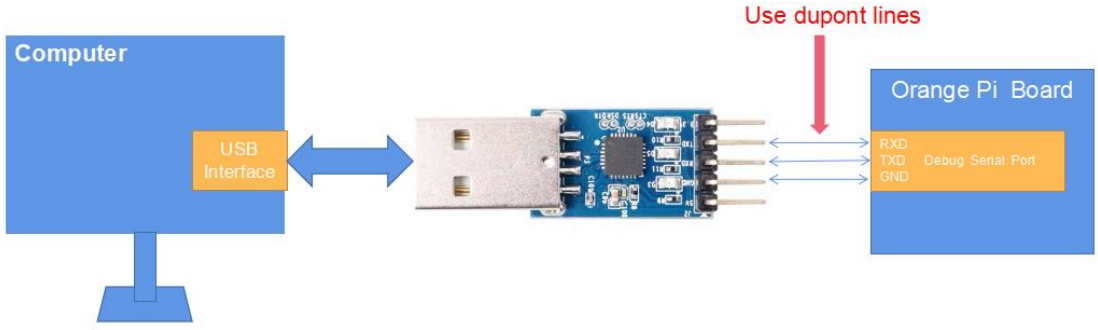
2) The corresponding relationship between the debug serial port GND, TX and RX pins of the development board is shown in the figure below



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

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

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



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

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

2.7.2. The Way 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 interface of the Ubuntu computer. If the connection and recognition of the USB to TTL module is normal, you can see the corresponding device node name under **/dev** of the Ubuntu PC, remember this node name, and set the serial port later software will be used

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

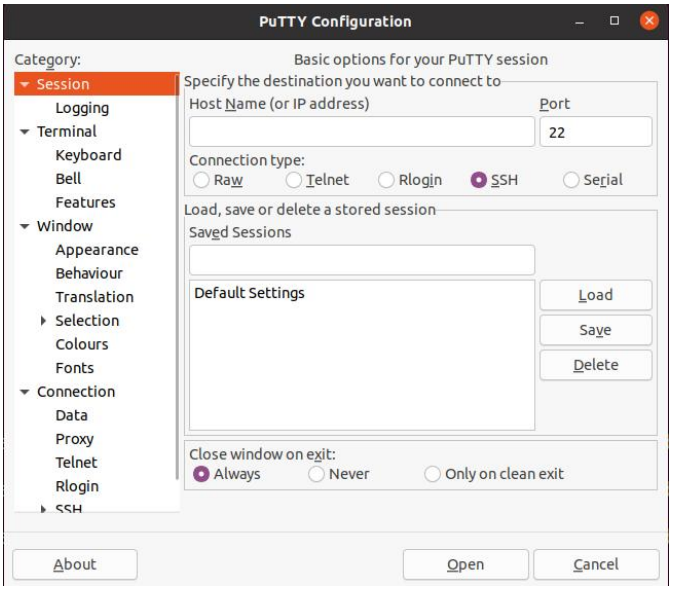
2) putty Then use the following command to install putty on Ubuntu PC

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

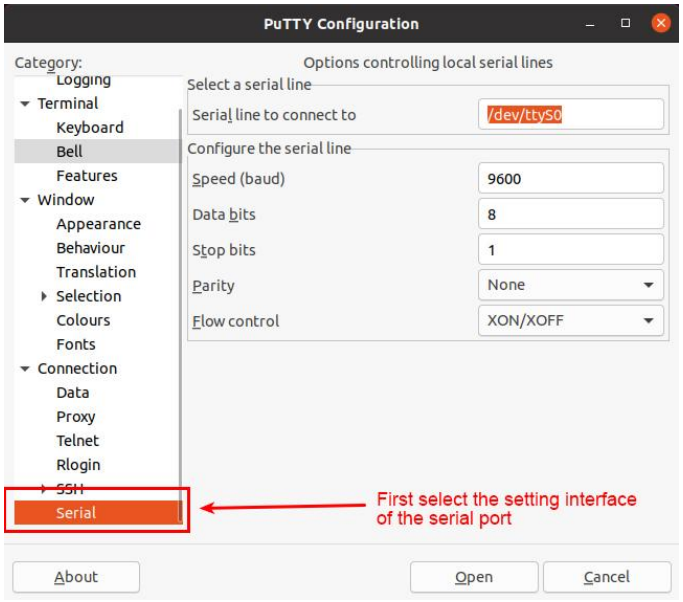
3) Then run putty, remember to add **sudo permissions**

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

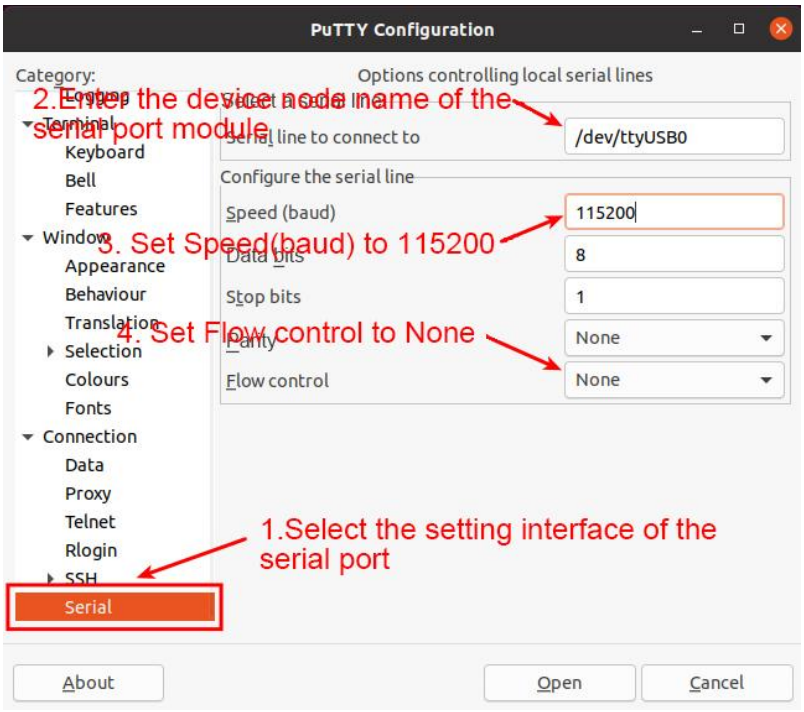
4) After executing the putty command, the following interface will pop up



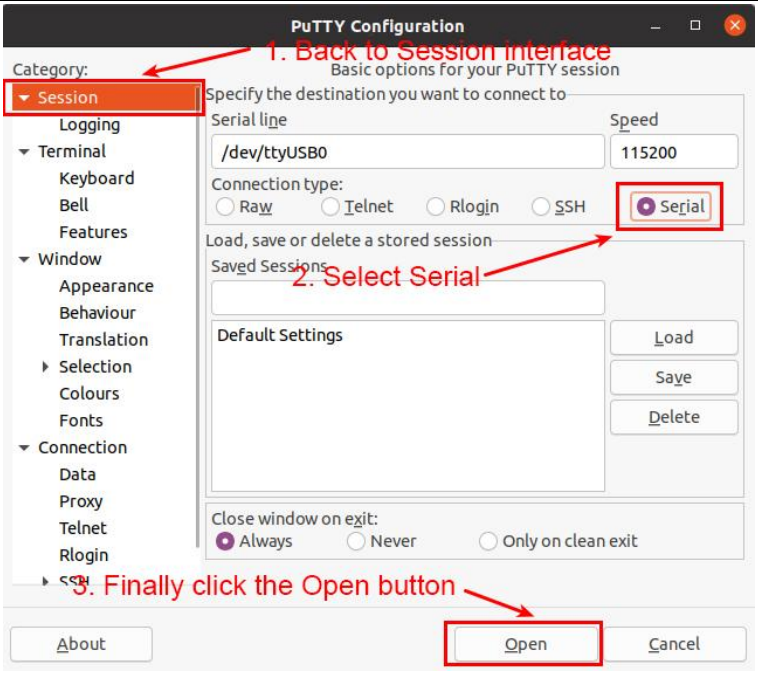
5) First select the setting interface of the serial port



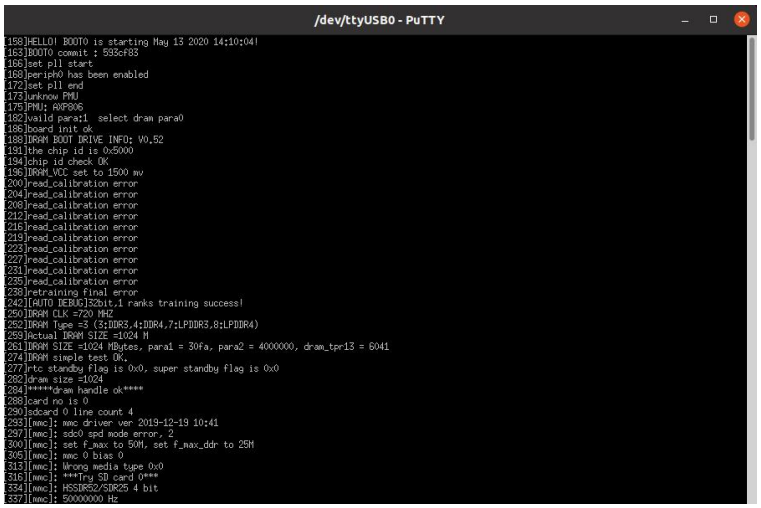
6) Then set the parameters of the serial port
   a. Set **Serial line to connect to** to **/dev/ttyUSB0**（Modify it to the corresponding node name, usually as **/dev/ttyUSB0**）
   b. Set **Speed(baud)** to **115200**（serial port baud rate）
   c. Set **Flow control** to **None**



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



8) Then start the development board, you can see the Log information output by the system from the open serial terminal



2.7.3. The Way to use the debug serial port on Windows platform

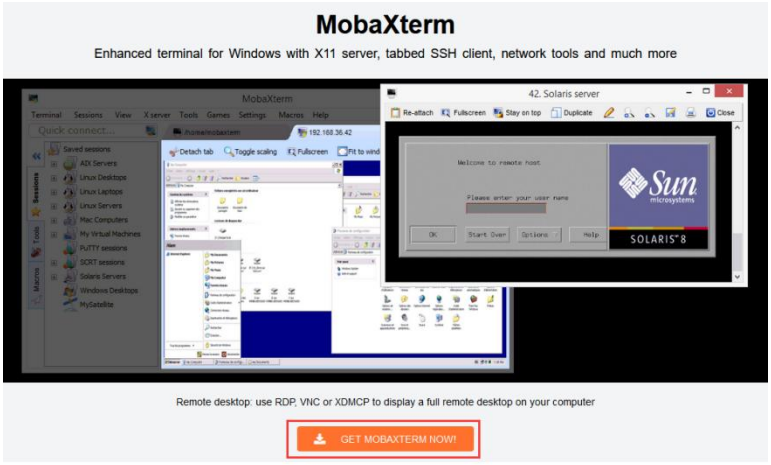
There are many serial port debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following shows how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

1) Download MobaXterm

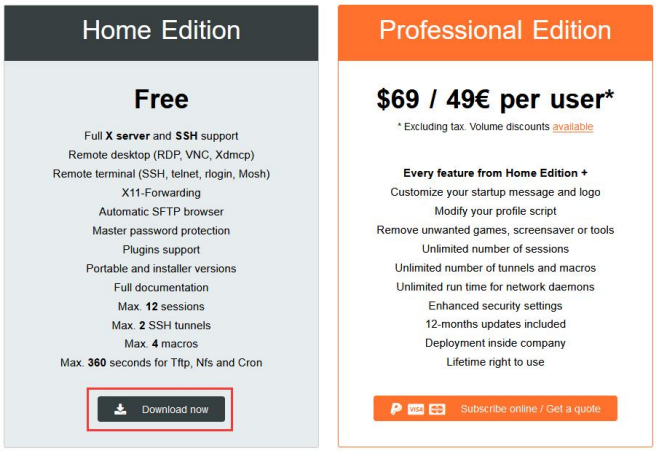
a. Download MobaXterm URL as follows

https://mobaxterm.mobatek.net/

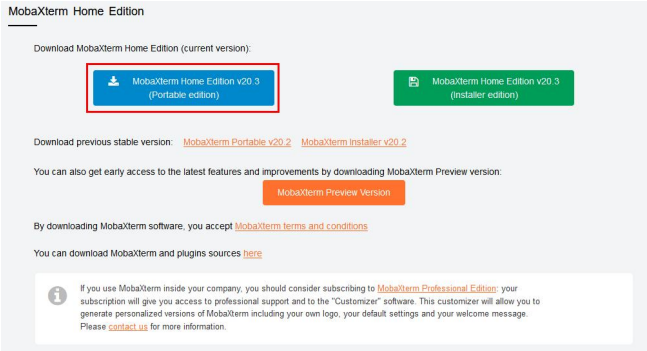
b. After entering the MobaXterm download page, click **GET XOBATERM NOW!**



c. Then choose to download the Home version



d. Then select the Portable portable version. After downloading, you don't need to install it, just open it and you can use it



2) After downloading, use the decompression software to decompress the downloaded

compressed document, you can get the executable software of MobaXterm, and then double-click to open it

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

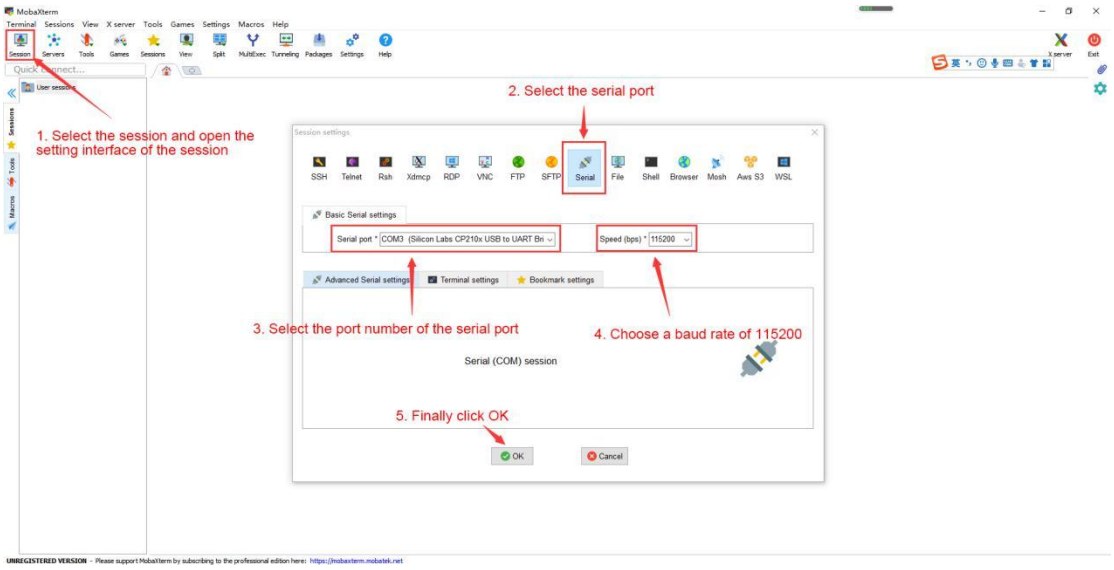
a. Open the session settings interface

b. Select the serial port type

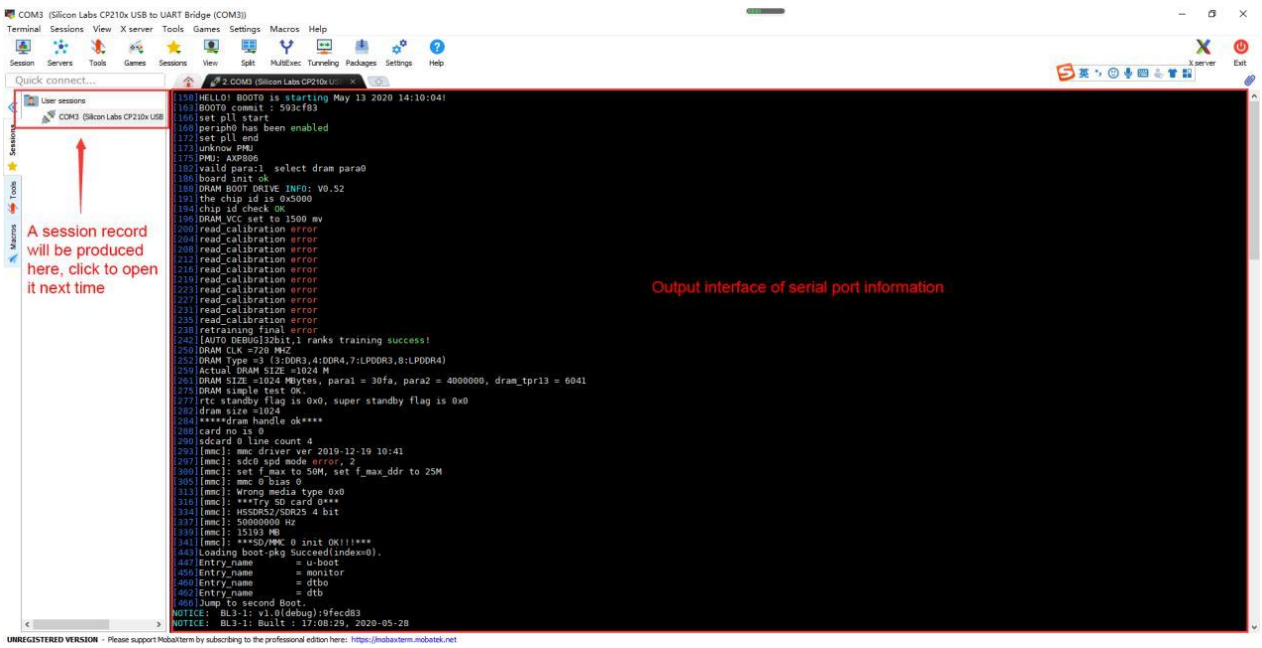
c. Select the port number of the serial port (select the corresponding port number according to the actual situation). If you cannot see the port number, please use the 360 driver master to scan and install the driver for the USB to TTL serial port chip.

d. Select the baud rate of the serial port to be **115200**

e. Finally click the **"OK"** button to complete the setting



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



2.8. Instructions for power supply using the 5v pin in the 26pin or 13pin interface of the development board

The power supply method we recommend for the development board is to use the power cord of the 5V/2A or 5V/3A Type C interface to plug into the Type C power interface of the development board to supply power. If you need to use the 5V pin in the 26pin or 13pin interface to power the development board, please make sure that the power cable used can meet the power supply requirements of the development board. If there is unstable use, please switch back to the Type C power supply

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

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

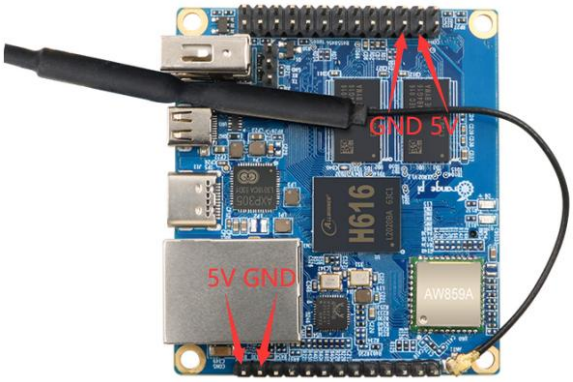
2) Use the 5V pin in the 26pin or 13pin interface to power the development board. The connection method of the power cable is as follows

a. The USB A port of the power cord shown in the figure above needs to be plugged into the 5V/2A or 5V/3A power adapter connector (**it is not recommended to plug it into the USB port of the computer to supply power. If there are too many peripherals connected to the development board, use will be unstable**)

b. The red Dupont wire needs to be plugged into the 5V pin of the 26pin or 13pin interface of the development board

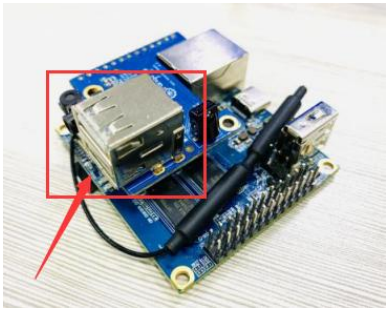
c. The black Dupont wire needs to be plugged into the GND pin of the 26pin or 13pin interface

d. The positions of the 5V pins and GND pins of the 26pin and 13pin interfaces in the development board are shown in the figure below, **Remember not to connect the wrong pin**



2.9. The method of using the 13pin interface of the development board to expand the USB interface

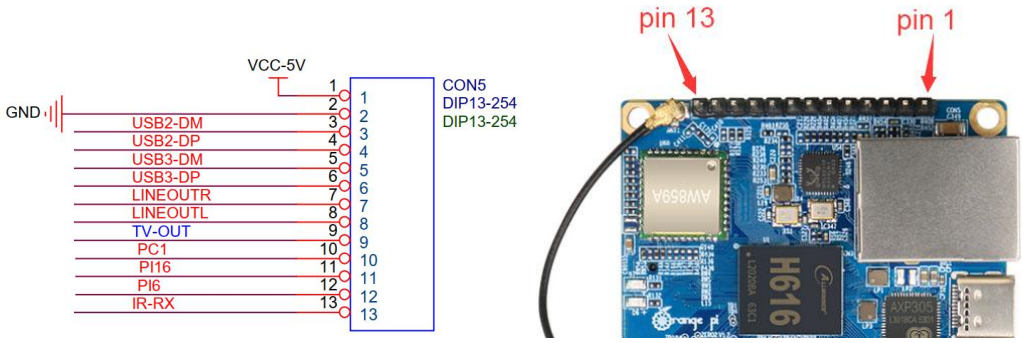
1) If you have purchased the 13pin expansion board of Orange Pi, insert the expansion board into the 13pin interface of the development board, you can expand 2 USB interfaces



2) If there is no 13pin expansion board, you can use a 4pin 2.54mm DuPont to USB2.0 female cable to expand the USB interface. The specific method is as follows:

a. First, you need to prepare a 4pin 2.54mm DuPont to USB2.0 female cable (this cable can be bought on Online store, please search and buy by yourself), as shown in the following figure:

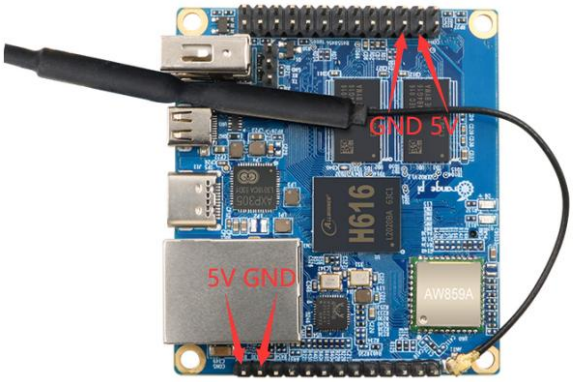
b. The schematic diagram of 13pin interface is shown below



c. The wiring of USB2 is as follows

d. The wiring of USB3 is as follows

e. If you need to connect two USB devices on the 13pin interface at the same time, you will find that the 5V and GND pins on the 13pin interface are not enough. At this time, one of the USB devices can use the 5V and GND pins in the 26pin interface. The location is as shown below shown:



3. Instructions for Debian and Ubuntu systems

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

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

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

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

In addition, people often ask if systems from other development boards can be used on the Orange Pi development board. The answer is no, because different

development boards use chips and circuit connections are generally different. A system developed for a certain development board cannot be used on other development boards.

3.1. Supported linux image types and kernel versions

Linux Image type	core version	server version	Desktop version
Ubuntu18.04 - Bionic	Linux4.9	support	support
Ubuntu 20.04 - Focal	Linux4.9	support	support
Debian 10 - Buster	Linux4.9	support	support
Ubuntu 20.04 - Focal	Linux5.16	support	support
Ubuntu 22.04 - Jammy	Linux5.16	support	support
Debian 11 - Bullseye	Linux5.16	support	support
Debian 12 - Bookworm	Linux5.16	support	support

Note that Debian 12 - Bookworm is an official Debian development system, not the current stable Debian release. Mainly used for developer experience. If it is a product, please do not choose this system considering stability issues.

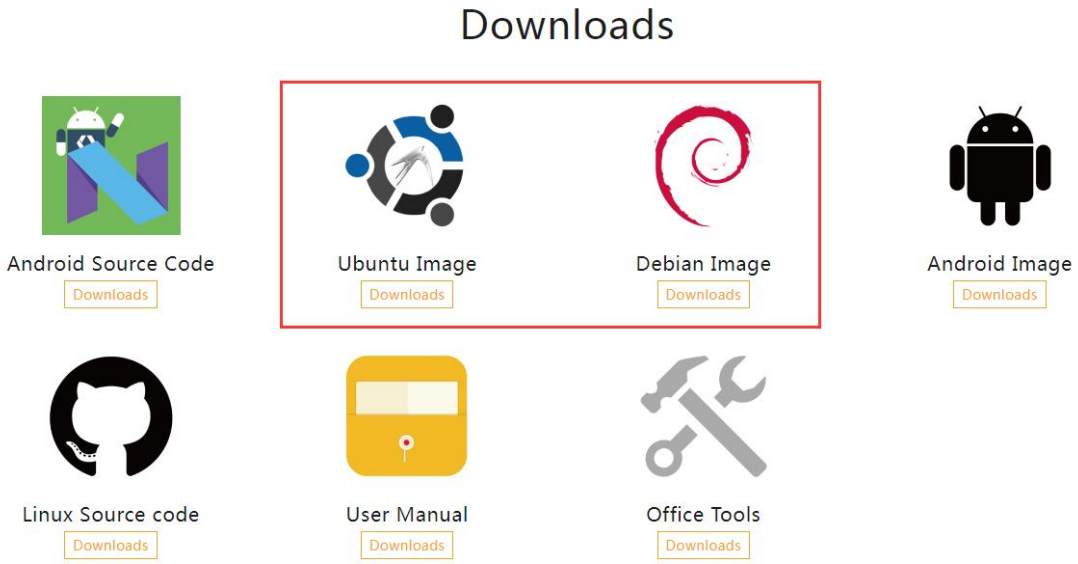
Debian 12 - Bookworm With the official development process of Debian, the packages in the software repositories will be continuously updated (including removing, adding packages and fixing problems encountered). So the packages in Debian 12 - Bookworm are a bit more up-to-date than those in other stable versions of Debian. This means that some new technologies and functions can be experienced in advance. For example, the H616 GPU can be used in the Debian12 Linux5.16 system.

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

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

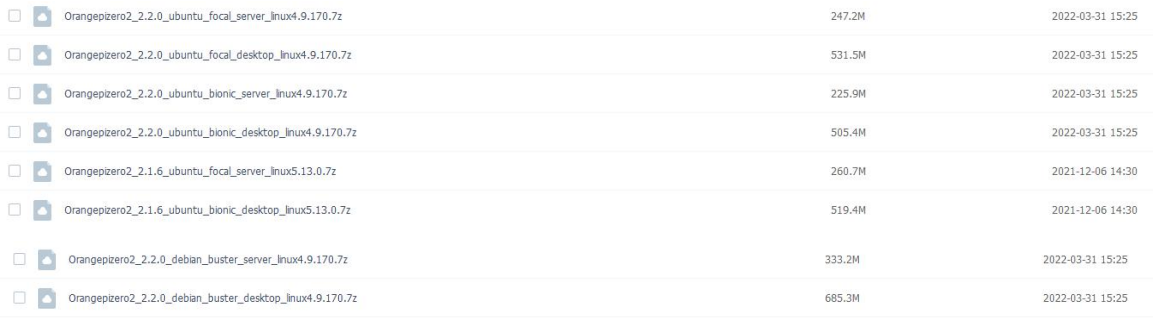
adapted in the code. Please refer to Chapter 5 and Chapter 6 for the compilation method of Linux image.

1) You can see the following download options on the data download page of Orange Pi. **Ubuntu images and Debian images are generally referred to as Linux images.**



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

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



2) Naming rules for Linux images

Development board model_version number_Linux distribution type_distribution code_server or desktop_core version

a. **The model of the development board：Both are Orangepizero2**。The model names of different development boards are generally different. Before burning the image, please make sure that the model name of the selected image matches the development board.

b. **version number**：such as **2.x.x or 3.x.x**，This version number will increase with the update of the image function. In addition, the last number of the version number of the Linux image of the development board is an even number.

c. **Types of Linux distributions**： Currently supports Ubuntu and Debian. Since Ubuntu is derived from Debian, there is not much difference in use between the two systems. However, the default configuration of some software and the use of commands are still slightly different. In addition, Ubuntu and Debian both have software repositories that are maintained and supported, and there are also slight differences in the supported and installable software packages. These require personal experience in order to have a deeper understanding. For more details, you can refer to the official documentation provided by Ubuntu and Debian.

d. **Distribution code**： Used to distinguish different versions of specific Linux distributions such as Ubuntu or Debian. Among them, **bionic** and **focal** are both Ubuntu distributions, bionic means Ubuntu18.04, focal means Ubuntu20.04, and jammy means Ubuntu22.04. The biggest difference between different versions is that many software in the software warehouse maintained by the new version of Ubuntu system are Newer than those in older Ubuntu systems, such as Python and GCC compilation toolchains. buster is the specific version code of Debian, buster means Debian10, **bullseye** means Debian11, and Debian11 is the latest stable version released by Debian. **bookworm** stands for Debian 12, the next official Debian development version.

e. **Server or Desktop**： It is used to indicate whether the system has a desktop environment. If it is **server**, it means that the system does not have a desktop environment installed. The storage space and resources occupied by the image are relatively small, and the command line is mainly used to operate the control system. If it is **desktop**, it means that the XFCE4 desktop environment is installed by default in the system, and the storage space and resources occupied by the image are relatively large. You can connect the monitor, mouse and keyboard to operate the operating system through the interface. Of course, the desktop version of the system can also be operated through the command line

like the server version of the system.

f. **Core version**：Used to indicate the version number of the linux kernel, currently supported **linux4.9.170** and **linux5.16.17**。

3.2. Linux core driver matching

Features	Linux4.9	Linux5.16
HDMI video	OK	OK
HDMI audio	OK	OK
USB2.0 x 3	OK	OK
TF card start	OK	OK
network card	OK	OK
Infrared receiver	OK	OK
WIFI	OK	OK
Bluetooth	OK	OK
Headphone Audio	OK	OK
USB Camera	OK	OK
LED Light	OK	OK
26pin GPIO	OK	OK
I2C3	OK	OK
SPI1	OK	OK
UART5	OK	OK
PWM	OK	OK
Temperature Sensor	OK	OK
Hardware watchdog	OK	OK
Mali GPU	NO	OK
Video codec	NO	NO
TV-OUT	NO	NO

3.3. Description of the linux command format in this manual

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

```
```

As shown below, the content in the yellow box indicates the content that needs

special attention, except for the commands in it

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

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

```
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 an ordinary user. When executing a privileged command, you need to add **sudo**

c. **root@orangepi:~#** The prompt indicates that this command was entered in the **linux system of the development board,**The # at the end of the prompt indicates that the current user of the system is the root user and can execute any command you want to execute.

d. **test@test:~$** The prompt indicates that the command was entered in the Ubuntu PC or Ubuntu virtual machine, not the linux system of the development board. The $ at the end of the prompt indicates that the current user of the system is an ordinary user. When executing a privileged command, you need to add sudo

e. **root@test:~#** The prompt indicates that the command was entered in the Ubuntu PC or Ubuntu virtual machine, not the linux system of the development board.The # at the end of the prompt indicates that the current user of the system is the root user and can execute any command you want to execute.

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

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

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

```
bootlogo=false
console=serial
```

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

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

3.4. Linux System login instructions

3.4.1. Linux System default login account and password

Account	Password
root	orangepi
orangepi	orangepi

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

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

3.4.2. The Way to set the automatic login of the Linux system terminal

1) The Way to automatically log in to the terminal as a root user

a. First enter the following command to create a configuration file for terminal automatic login

```
root@orangepi:~# mkdir -p /etc/systemd/system/getty@.service.d/
root@orangepi:~# mkdir -p /etc/systemd/system/serial-getty@.service.d/
root@orangepi:~# cat <<-EOF >    \
```

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

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

```
   Loading Ramdisk to 4997f000, end 49fff958 ... OK
   reserving fdt memory region: addr=48000000 size=1000000
## Linux machid: 00000000, FDT addr: 7be86d60

Starting kernel ...


orangepizero2 login: root (automatic login)

Last login: Wed Apr  6 01:42:44 UTC 2022 on tty1
  ___  ____  _   _____                 ____
 / _ \|  _ \(_) |__  /___ _ __ ___     |___ \
| | | | |_) | |   / // _ \ '__/ _ \ _____) |
| |_| |  __/| |  / /|  __/ | | (_) |  / __/
 \___/|_|   |_| /____\___|_|  \___/  |_____|

Welcome to Orange Pi Orange Pi 2.2.0 Buster with Linux 4.9.170-sun50iw9

System load:   1.14 0.29 0.10   Up time:       0 min
Memory usage:  12 % of 960MB    IP:           192.168.1.82
CPU temp:      67°C
Usage of /:    4% of 29G

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


root@orangepizero2:~#
```

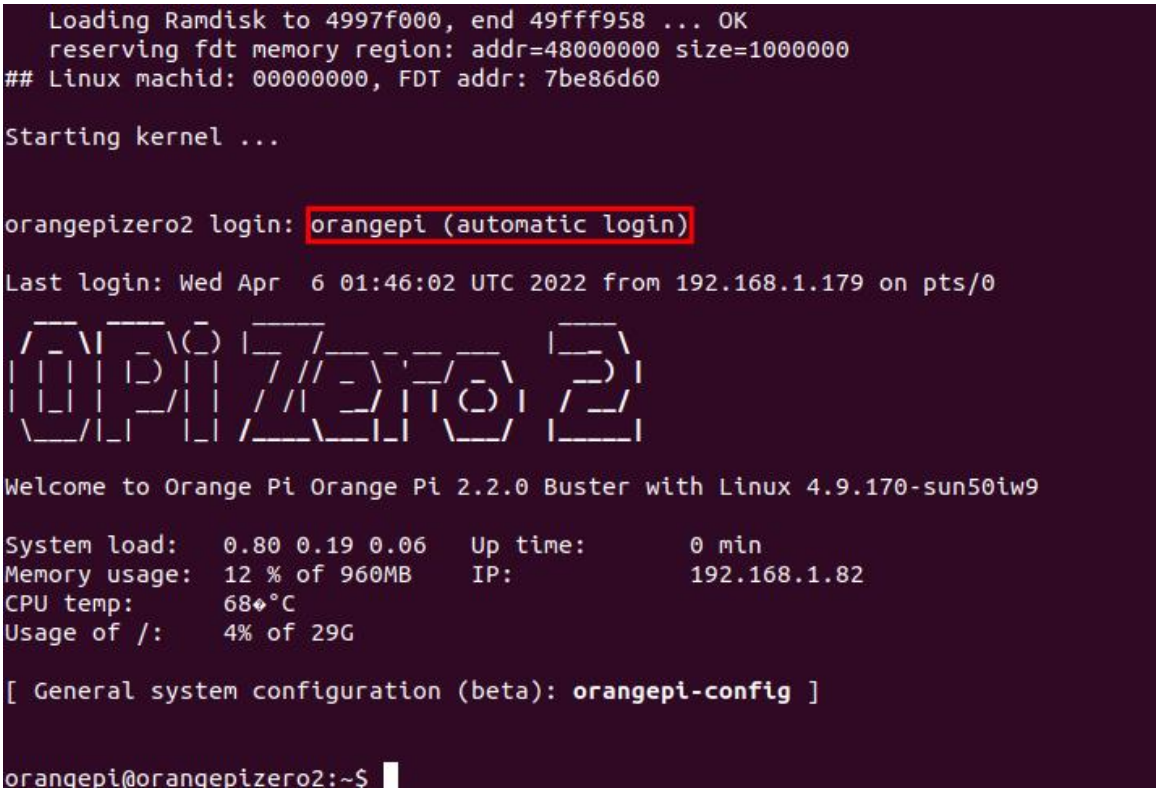
2) Method for orangepi users to automatically log in to the terminal

a. First enter the following command to create a configuration file for terminal automatic login

```
root@orangepi:~# mkdir -p /etc/systemd/system/getty@.service.d/
root@orangepi:~# mkdir -p /etc/systemd/system/serial-getty@.service.d/
root@orangepi:~# cat <<-EOF >   \
```

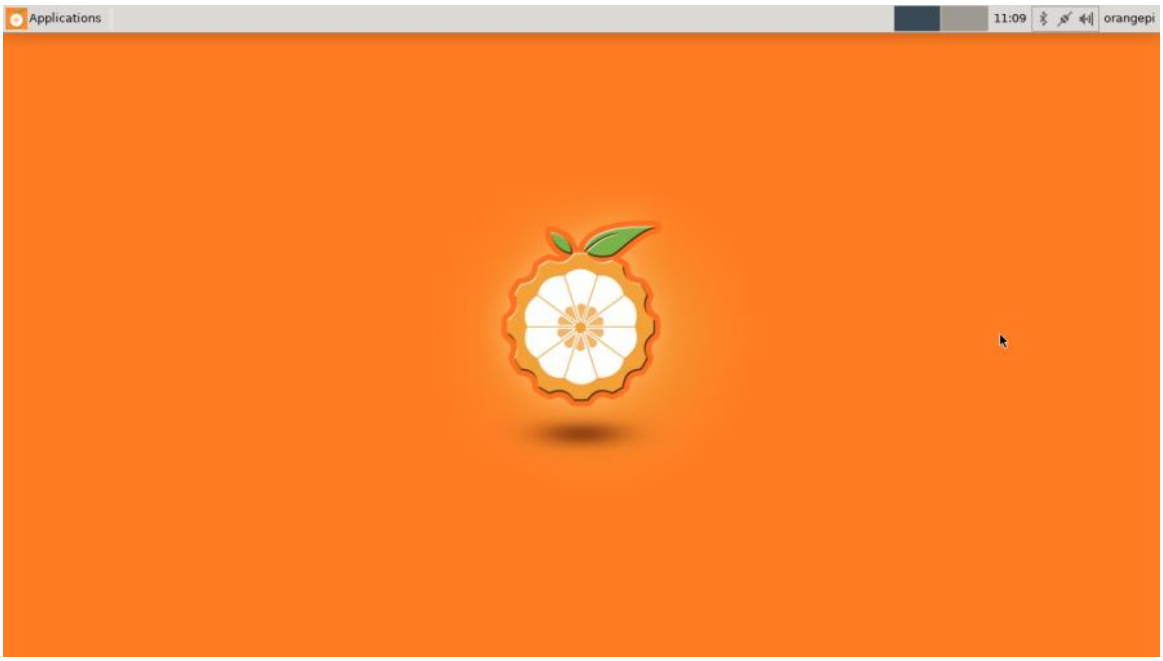
```
/etc/systemd/system/serial-getty@.service.d/override.conf
[Service]
ExecStartPre=/bin/sh -c 'exec /bin/sleep 10'
ExecStart=
ExecStart=-/sbin/agetty --noissue --autologin orangepi %I \$TERM
Type=idle
EOF
root@orangepi:~# cp /etc/systemd/system/serial-getty@.service.d/override.conf   \
/etc/systemd/system/getty@.service.d/override.conf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

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



3.4.3. Linux Desktop version system automatic login instructions

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



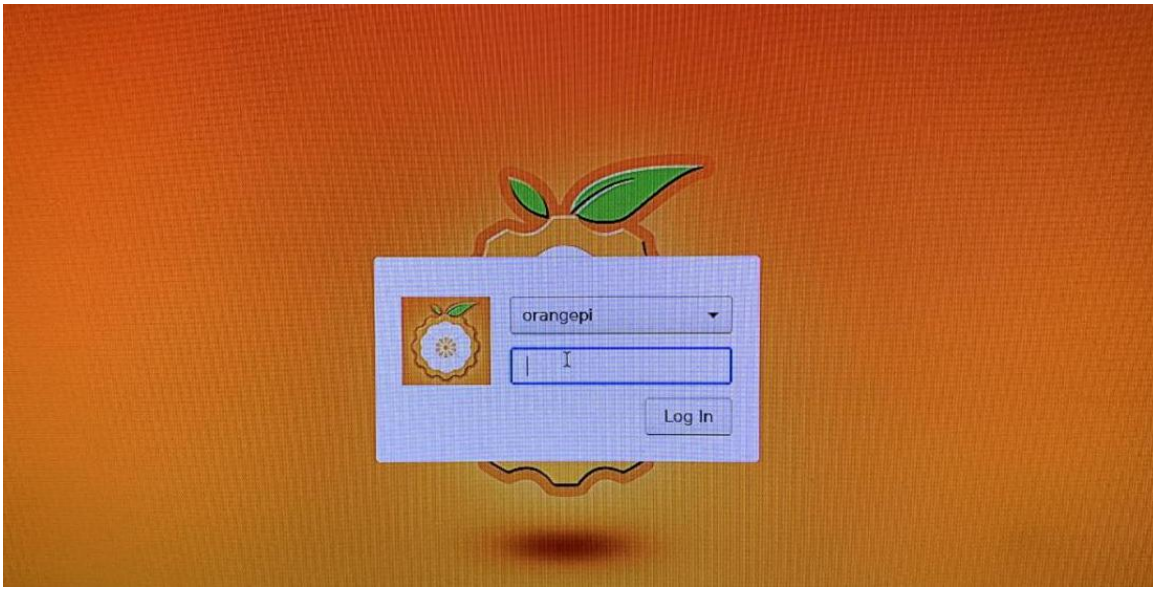
2) Modify the configuration in **/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf** to prevent the desktop version system from automatically logging in to the desktop. The modification command is as follows, or you can open the configuration file and modify it directly

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

3) After modification, the configuration of /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf is as follows

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

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



3.4.4. Setting method for automatic login of root user in Linux desktop system

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

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

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

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

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

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

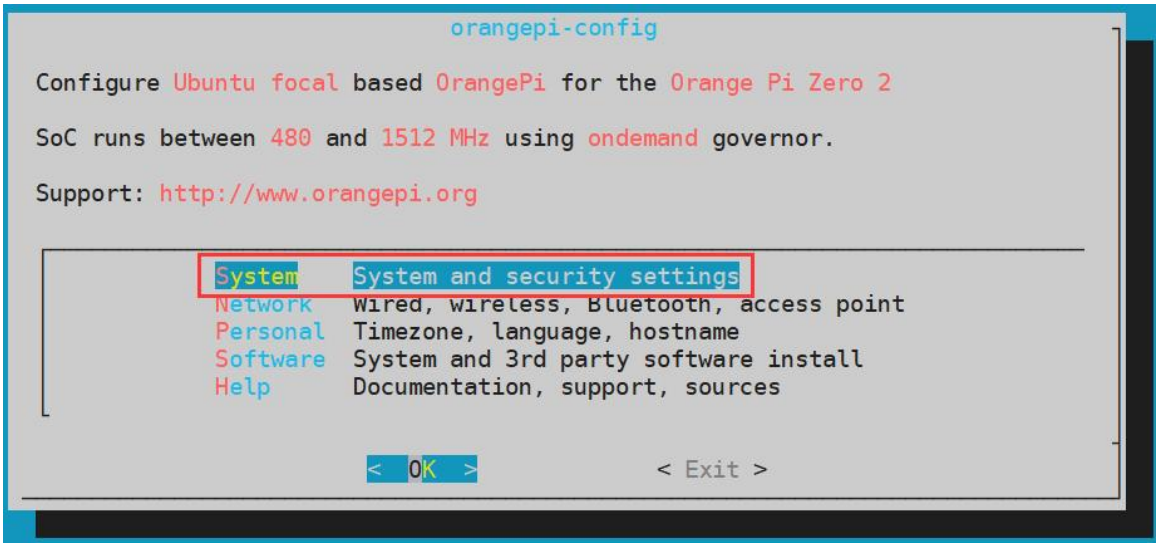
Also note that this is not a bug, since pulseaudio is not allowed to run under root.

3.4.5. The Way to disable the desktop in the Linux desktop system

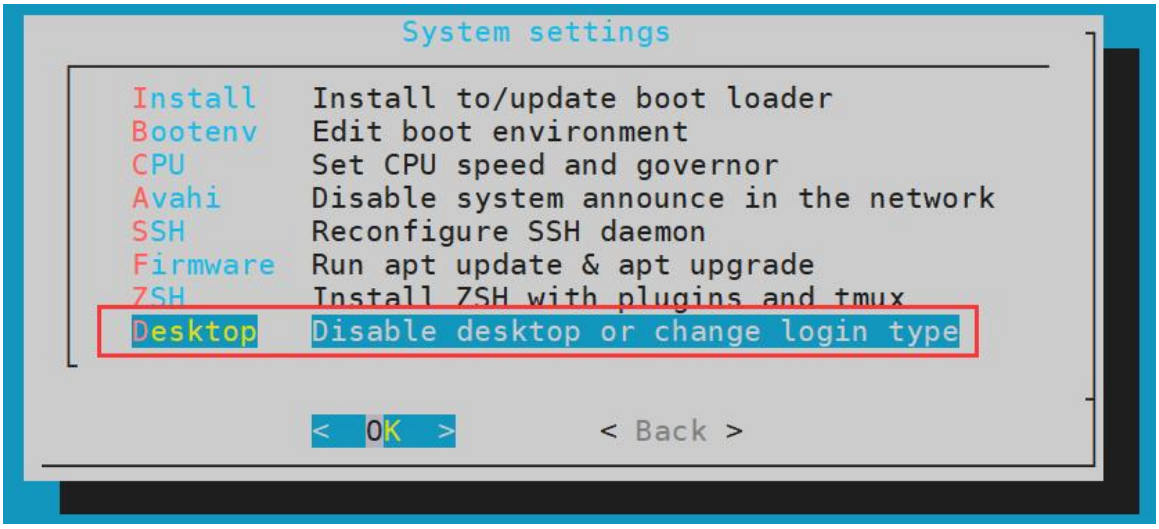
1) First enter the following command on the command line，**Please remember to add sudo permissions**

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

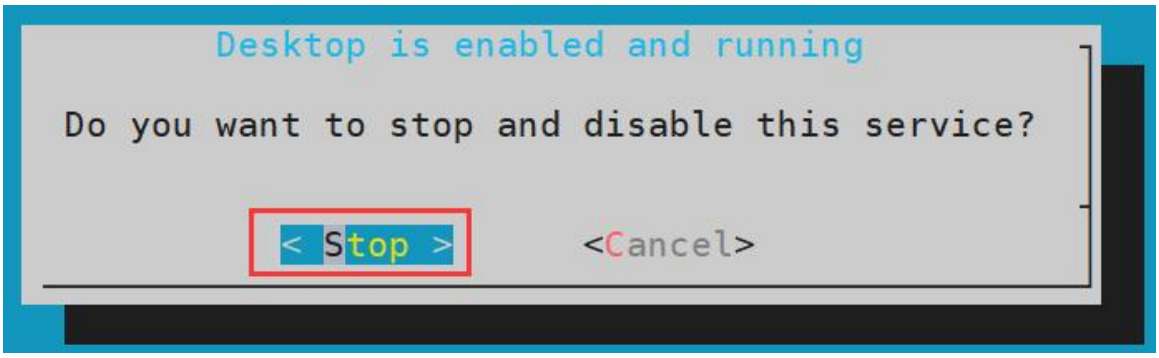
2) Then select**System**



3) Then select **Desktop**



4) Then select **<Stop>**



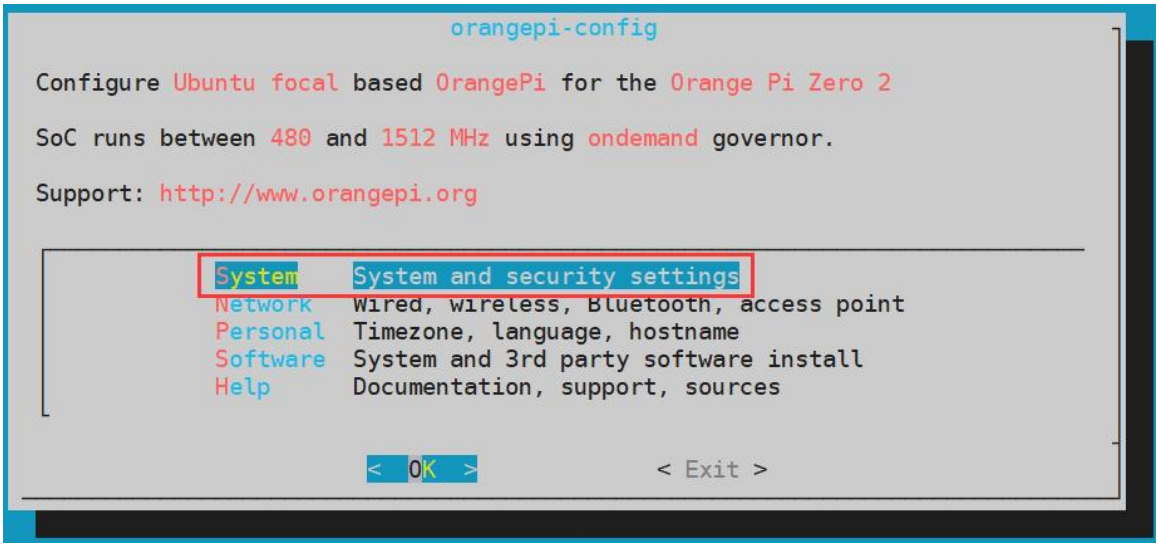
5) **Then restart the Linux system and you will find that the desktop will not be displayed**

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

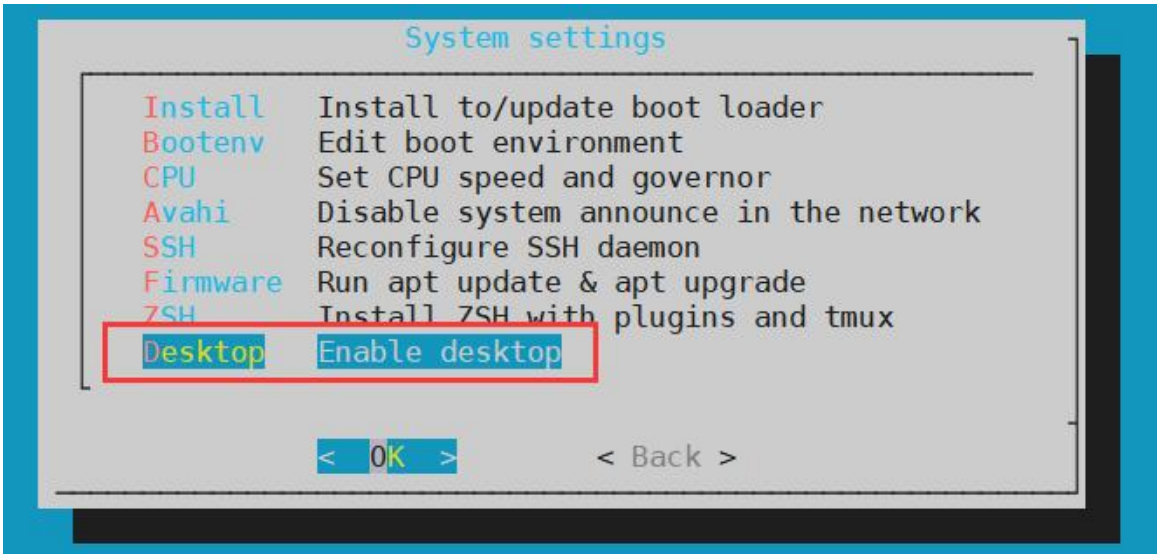
a. First, enter the following command in the command line, please remember to add sudo permissions

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

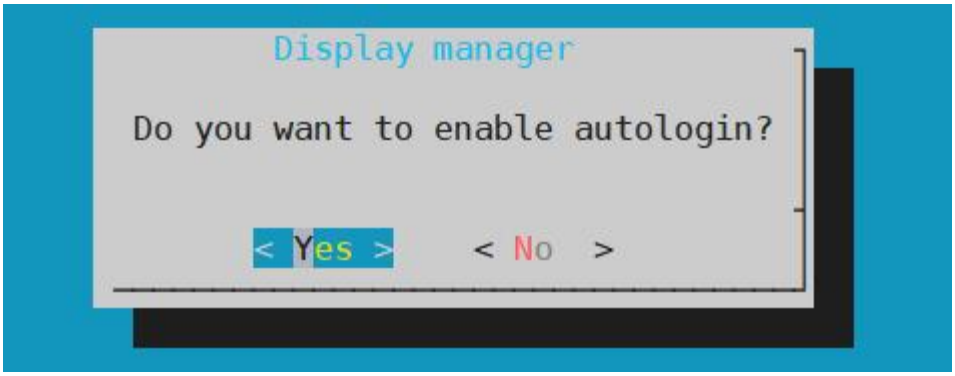
b. Then select **System**



c. Then select **Desktop Enable desktop**



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



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

3.5. On-board LED light test description

1) There are two LED lights on the development board, one green light and one red light. When the system starts, the default display of the LED lights is as follows:

	green light	red light
u-boot startup phase	light off	light up
The Core boots into the system	light up	light off
GPIO □	PC13	PC12

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

a. First enter the setting directory of the green light

```
root@orangepi:~# cd /sys/class/leds/orangepi\:green\:status
```

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

```
root@orangepi:/sys/class/leds/orangepi:green:status# echo 0 > brightness
```

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

```
root@orangepi:/sys/class/leds/orangepi:green:status# echo 1 > brightness
```

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

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

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

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

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

a. First enter the setting directory of the red light

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

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

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

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

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

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

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

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

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

3.6. Operation instructions for the capacity of the rootfs partition of the Linux system in the TF card

3.6.1. The first boot will automatically expand the capacity of the rootfs partition in the TF card

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

Note that if this step is not performed, it will not affect the automatic expansion of the Linux system of the development board. Here I just want to explain how to check the capacity of the TF card after burning the Linux image on the TF card.

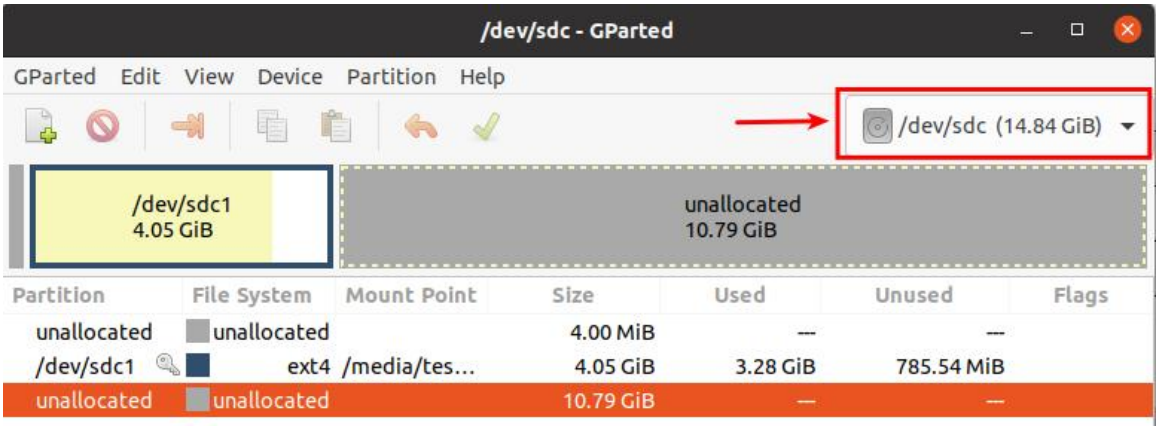
a. First install the software gparted in the Ubuntu computer

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

b. Then open gparted

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

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



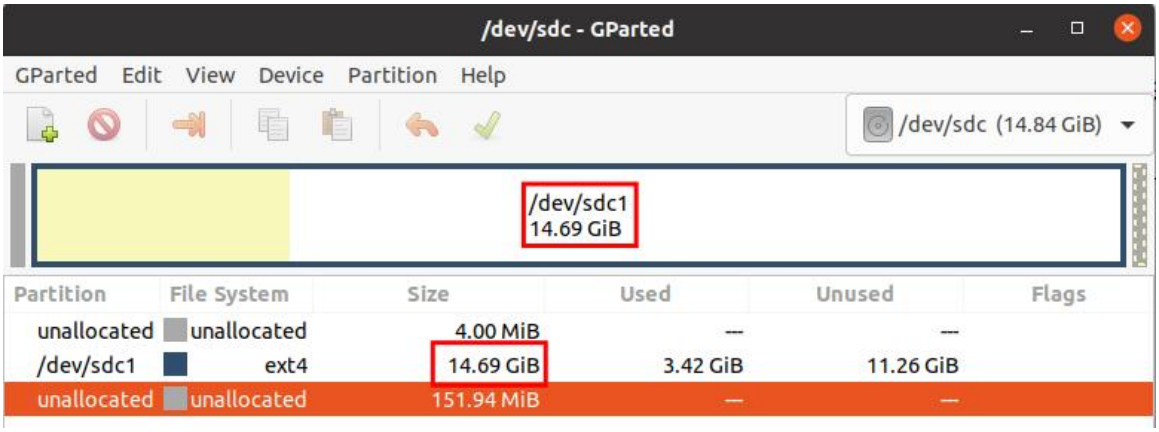
d. The above picture shows the situation of the TF card after burning the Linux desktop version system. It can be seen that although the total capacity of the TF card is 16GB (displayed as 14.84GiB in GParted), the rootfs partition (/dev/ sdc1) actually only allocated 4.05GiB, leaving 10.79GiB unallocated

2) Then you can insert the TF card with the programmed Linux system into the development board to start. When the TF card starts the Linux system for the first time, it will call the orangepi-resize-filesystem script through the systemd service orangepi-resize-filesystem.service to automatically perform The expansion of the rootfs partition, so there is no need to manually expand

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

```
orangepi@orangepi:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            430M     0  430M   0% /dev
tmpfs           100M  5.6M   95M   6% /run
/dev/mmcblk0p1   15G   915M   14G   7% /
tmpfs           500M     0  500M   0% /dev/shm
```

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



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

3.6.2. The method of prohibiting the automatic expansion of the rootfs partition capacity in the TF card

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

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

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

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

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

4) Then enter the root directory of the Linux system in the TF card and create a new file named **.no_rootfs_resize**

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

5) Then you can uninstall the TF card, then pull out the TF card and insert it into the development board to start. When the linux system starts, when the file **.no_rootfs_resize** is detected in the **/root** directory, the rootfs will not be automatically expanded.

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

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

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

Note, please execute the following command under the root user

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

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

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

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

3.6.3. The Way to manually expand the capacity of the rootfs partition in the TF card

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

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

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

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

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

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

4) Then enter the root directory of the Linux system in the TF card and create a new file named .no_rootfs_resize

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

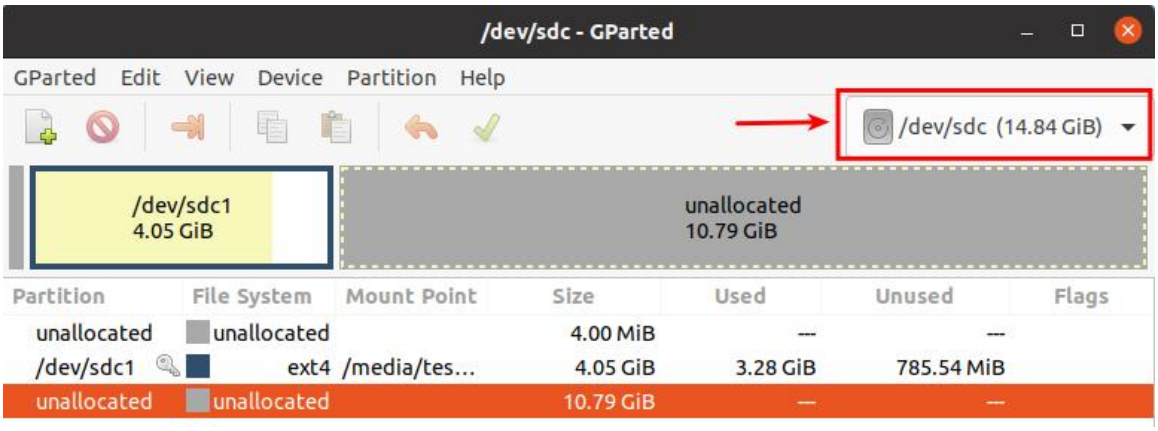
5) Then install the software gparted on the Ubuntu computer

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

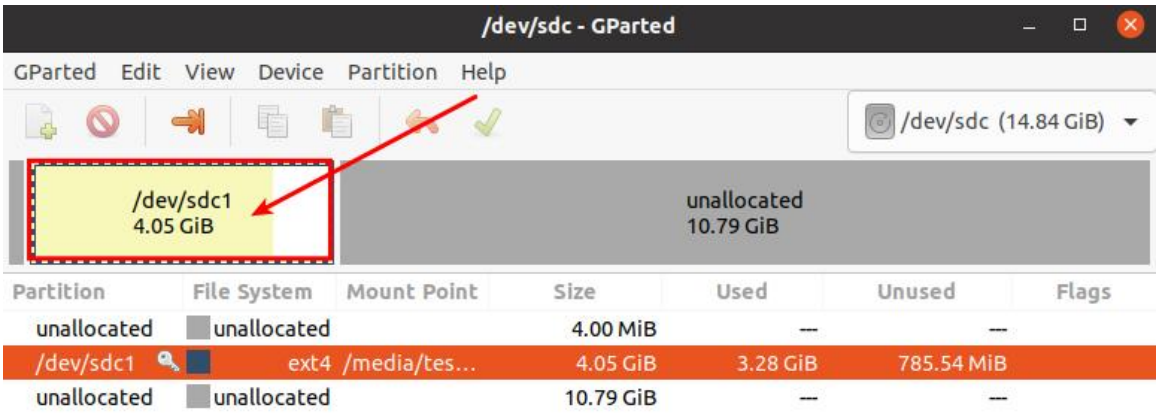
6) Then open gparted

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

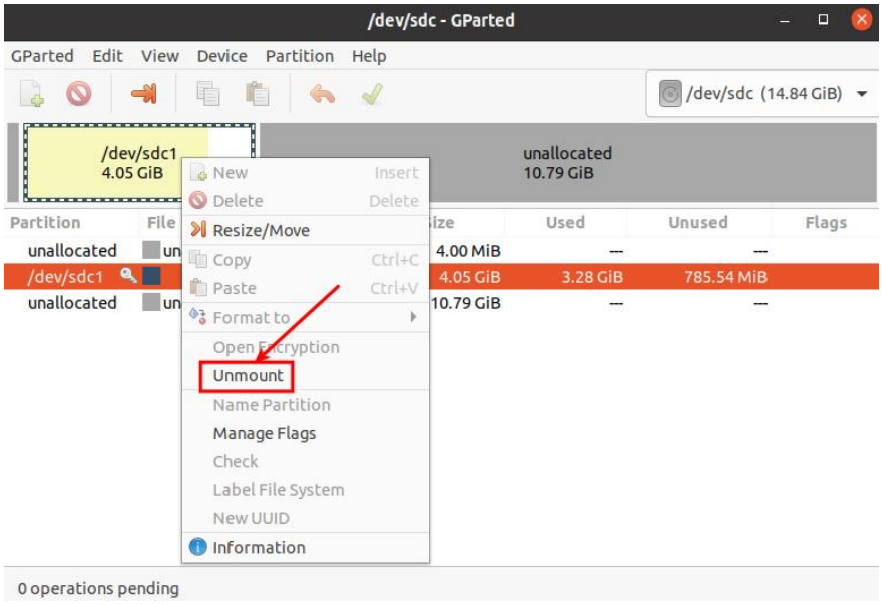
7) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity. The figure below shows the TF card after burning the Linux desktop version system. It can be seen that although the total capacity of the TF card is 16GB (displayed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



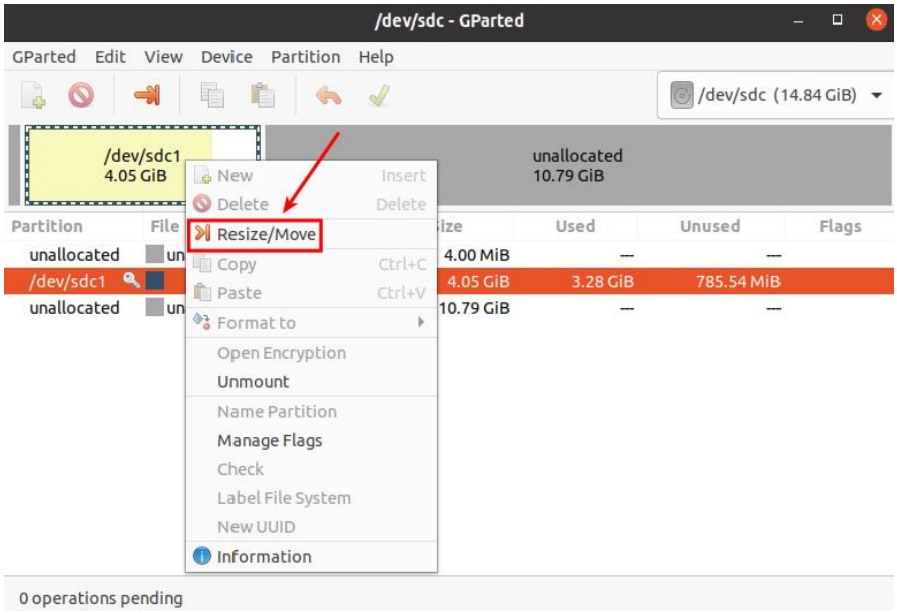
8) Then select the rootfs partition (/dev/sdc1)



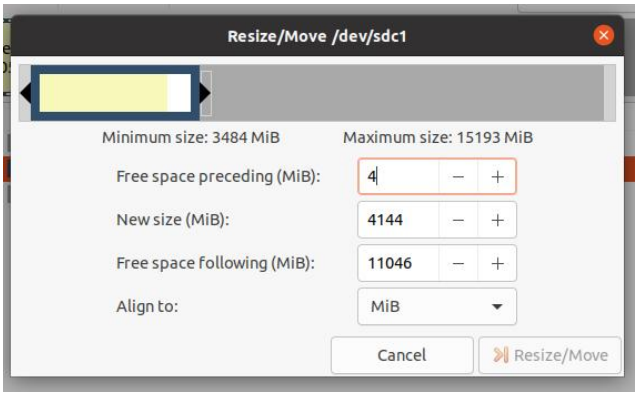
9) Click the right mouse button again to see the operation options shown in the figure below. If the TF card has been mounted, you need to Umount the rootfs partition of the TF card first.



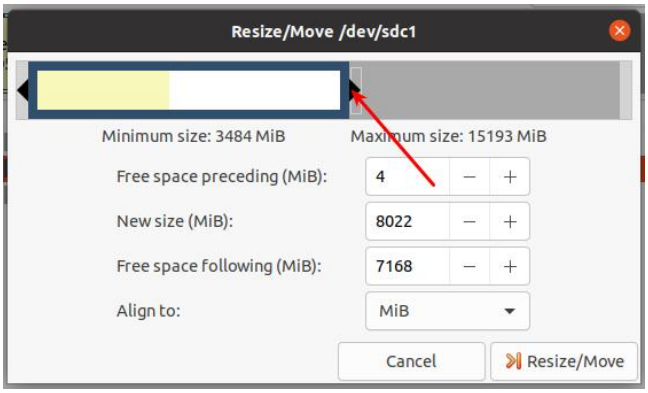
10) Then select the rootfs partition again, right-click, and select **Resize/Move** to start expanding the size of the rootfs partition



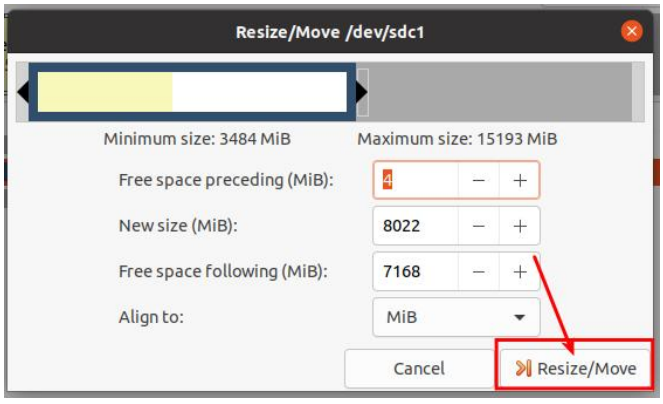
11) **Resize/Move**After the option is opened, the following settings interface will pop up



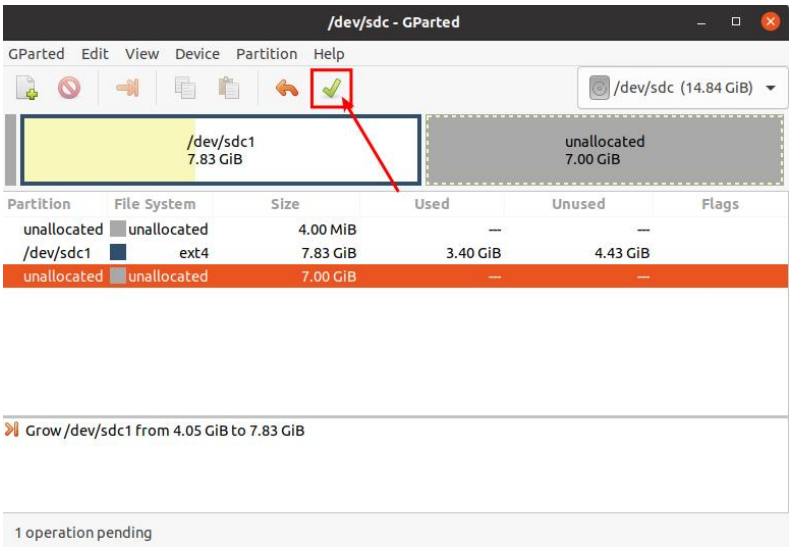
12) Then you can directly drag the position shown in the figure below to set the size of the capacity, or you can set the size of the rootfs partition by setting the number in **New sieze (MiB)**



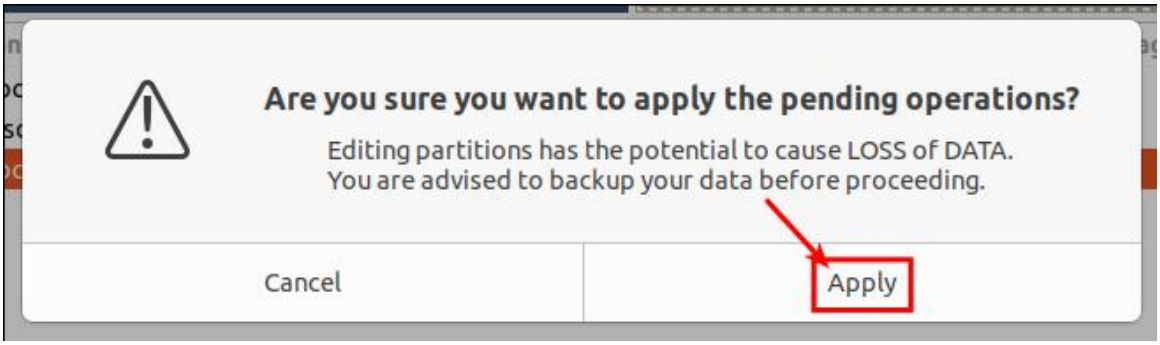
13) After setting the space, click Resize/Move in the lower right corner



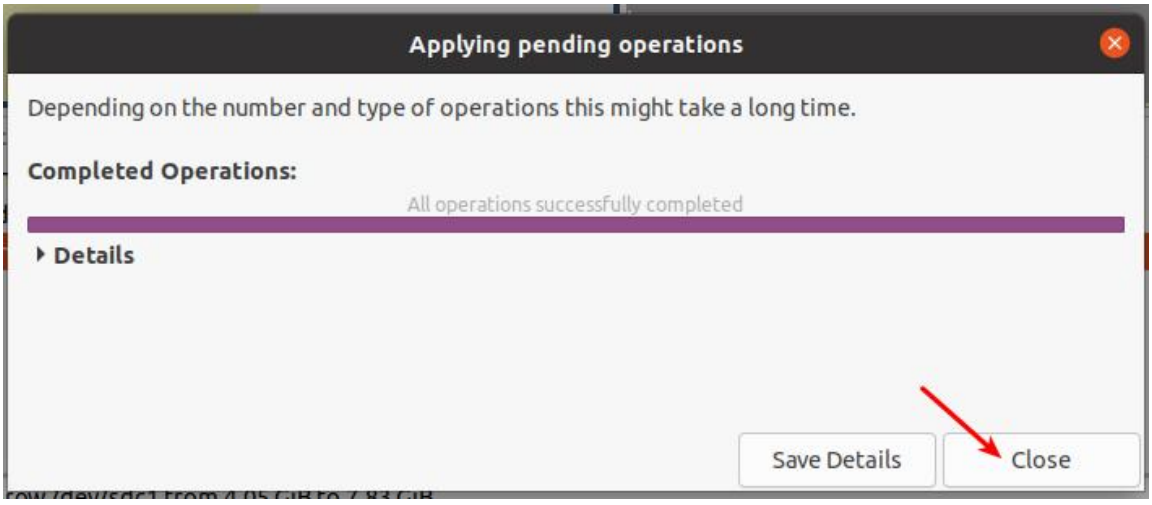
14) Finally, after confirming that it is correct, click on the **green**√ as shown in the figure below.



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



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



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

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

3.6.4. Method to reduce the space of rootfs partition in TF card

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

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

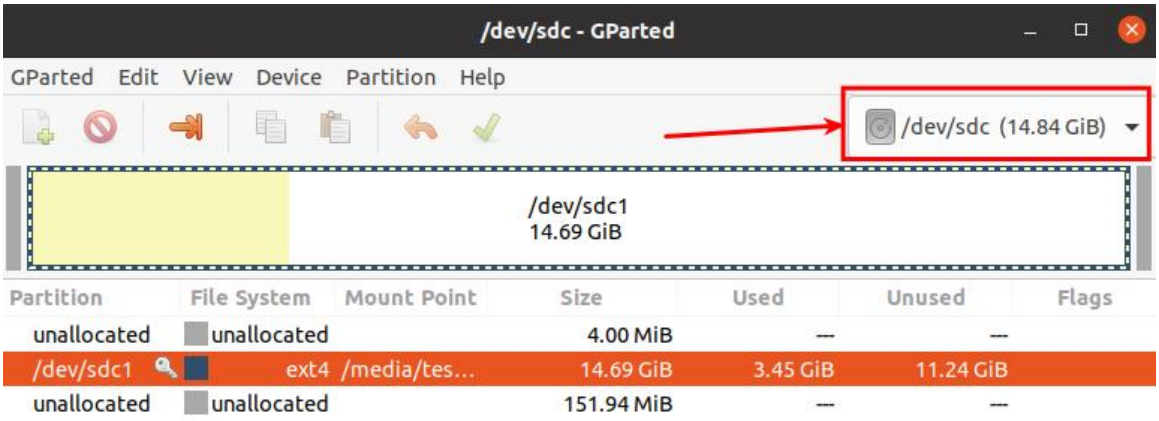
2) Then install the software gparted on the Ubuntu computer

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

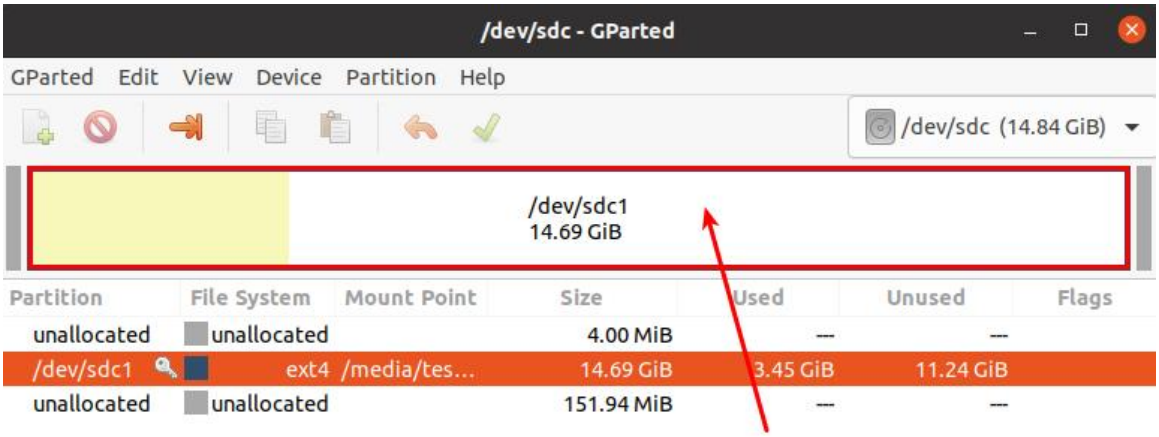
3) Then open gparted

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

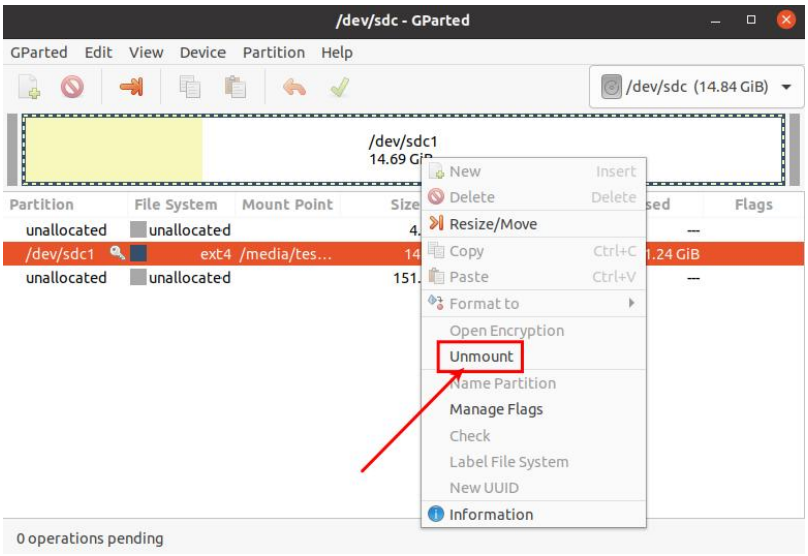
4) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity



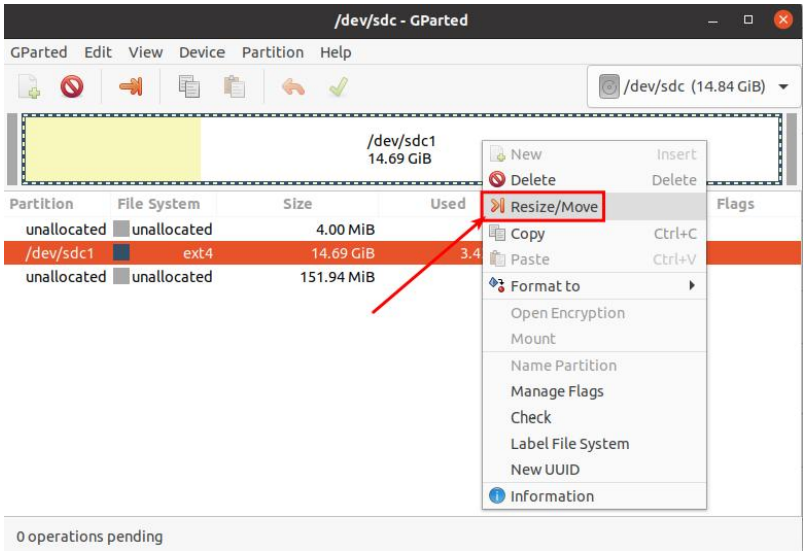
5) Then select the rootfs partition (/dev/sdc1)



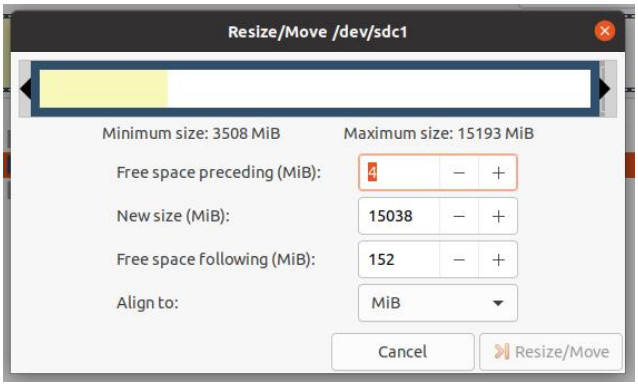
6) Click the right mouse button again to see the operation options shown in the figure below. If the TF card has been mounted, you need to Umount the rootfs partition of the TF card first.



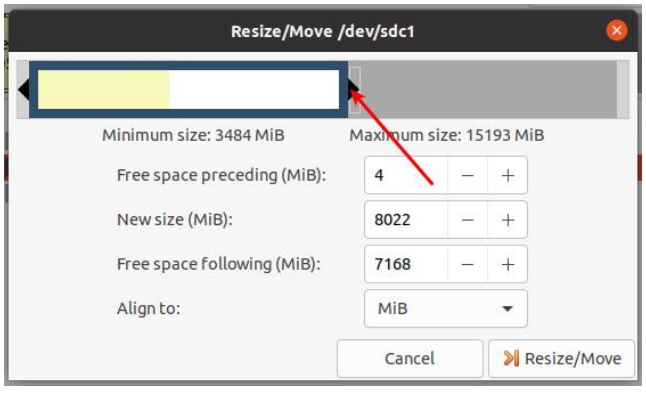
7) Then select the rootfs partition again, right-click, and select **Resize/Move** to start setting the size of the rootfs partition



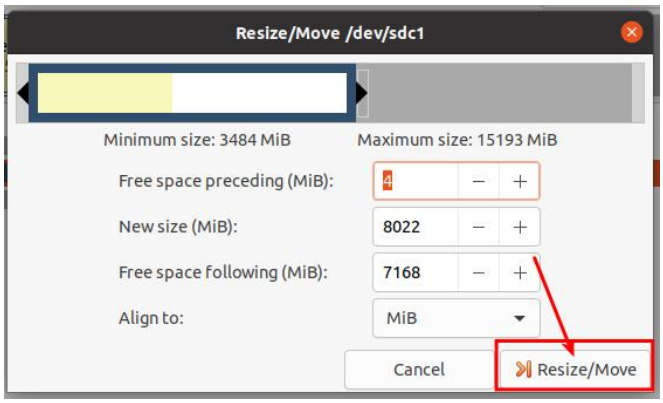
8) **Resize/Move** After the option is opened, the following settings interface will pop up



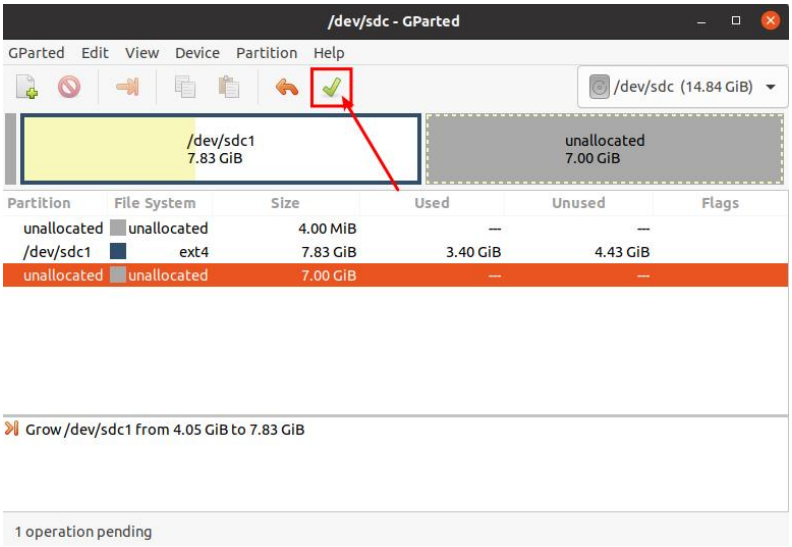
9) Then you can directly drag the position shown in the figure below to set the size of the capacity, or you can set the size of the rootfs partition by setting the number in **New sieze (MiB)**



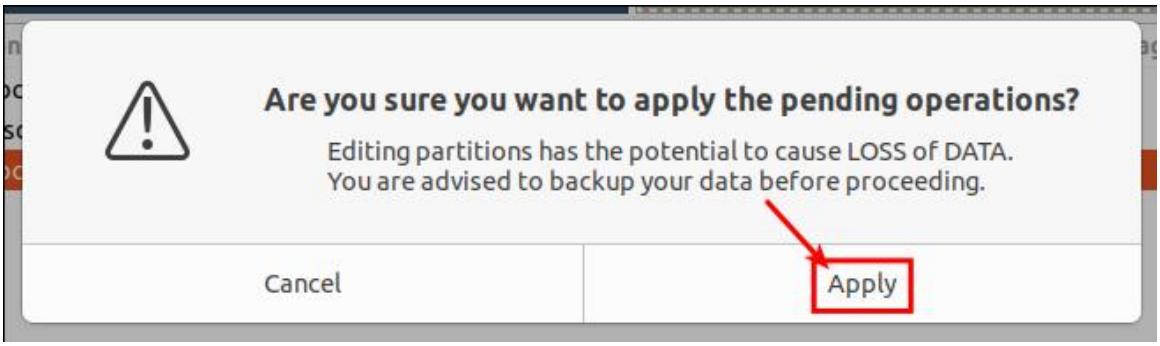
10) After setting the space, click **Resize/Move** in the lower right corner



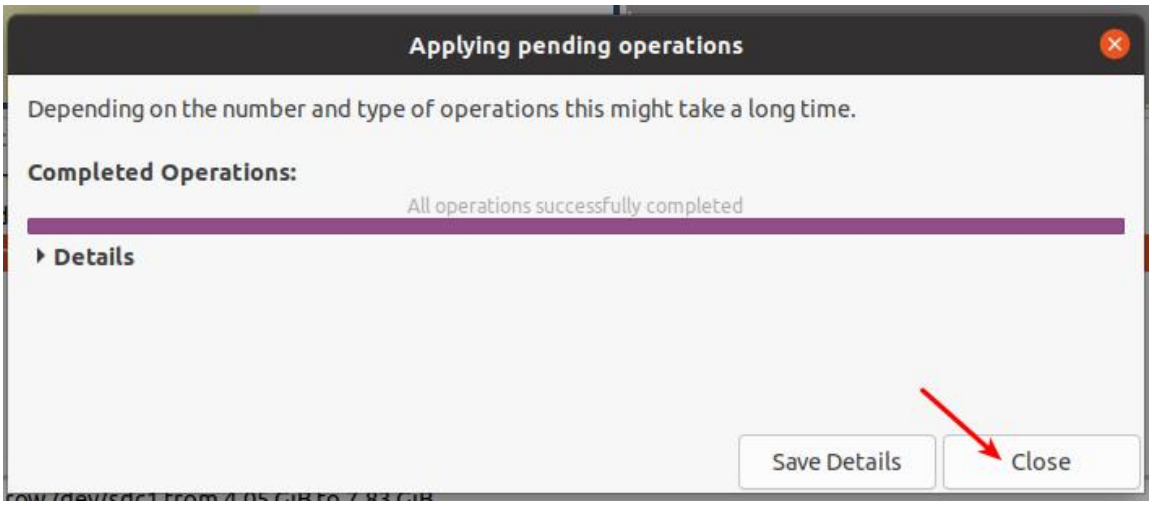
11) Finally, after confirming that it is correct, click on the **green**√as shown in the figure below.



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



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



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

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

3.7. The Way to modify the linux log level (loglevel)

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

```
Starting kernel ...

Uncompressing Linux... done, booting the kernel.

Orange Pi 2.2.0 Focal ttyS0

orangepi login:
```

2) When there is a problem with the startup of the Linux system, you can use the following method to modify the value of loglevel, so as to print more log information to

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

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

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

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

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

3.8. Network connection test

3.8.1. Ethernet port test

1) First, insert one end of the network cable into the Ethernet interface of the development board, and connect the other end of the network cable to the router, and ensure that the network is smooth

2) After the system starts, it will automatically assign an IP address to the Ethernet card through **DHCP，No other configuration is required**

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

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

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

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

1. Connect the HDMI display, then log in to the system and use the ip addr show eth0 command to view the IP address

2. Enter the ip addr show eth0 command in the debug serial terminal to view the IP address

3. If there is no debugging serial port and no HDMI display, you can also view the IP address of the network port of the development board through the management interface of the router. However, this method often fails to see the IP address of the development board. If you can't see it, the debug method looks like this:

A）First, check whether the Linux system has been started normally. If the green light of the development board is on, it is generally started normally. If only the red light is on, or the red and green lights are not on, the system has not started normally;

B）Check whether the network cable is plugged in tightly, or try another network cable;

C）Try another router (the router has encountered many problems, such as the router cannot assign an IP address normally, or the IP address has been assigned normally but cannot be seen in the router);

D）If there is no router to replace, you can only connect the HDMI display or use the debug serial port to view the IP address.

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

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

```
orangepi@orangepi:~$ ping www.baidu.com -I eth0
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
```

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

3.8.2. WIFI connection test

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

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

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

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

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

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

```
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,

a. **wifi_name** Need to be replaced with the name of the WIFI hotspot you want to connected

b. **wifi_passwd** Need to change to the password of the WIFI hotspot you want to connected

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

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

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

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

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

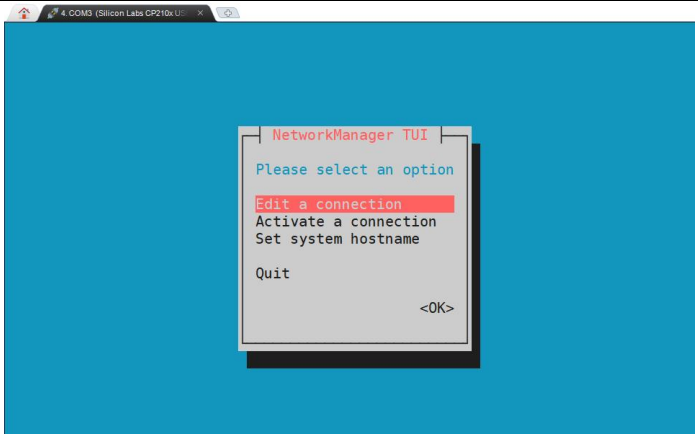
3.8.2.2. The server version image connects to WIFI through a graphical method

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

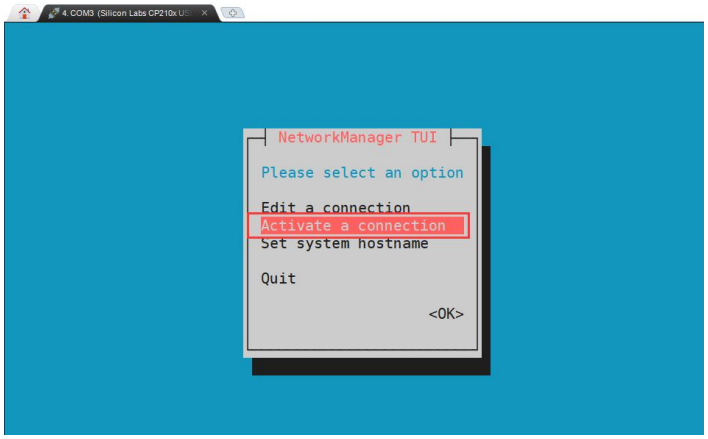
2) Then enter the nmtui command in the command line to open the wifi connection interface

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

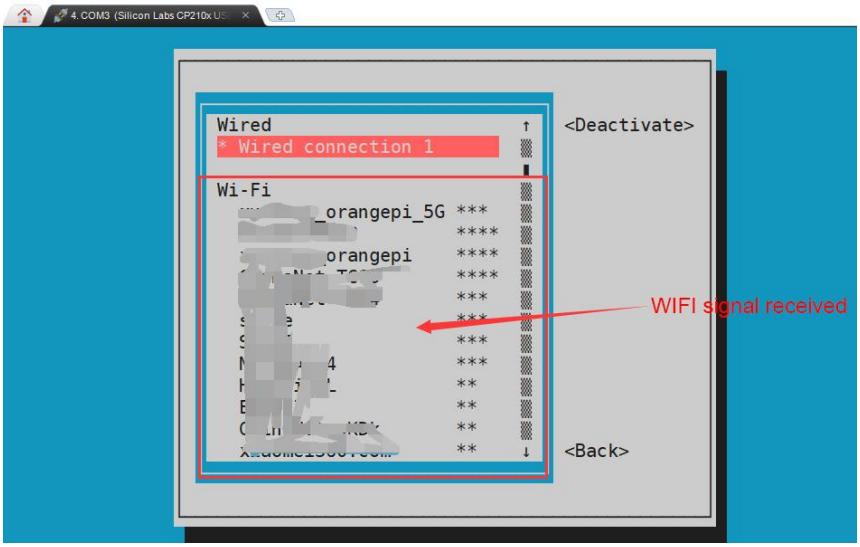
3) Enter the nmtui command to open the interface as shown below



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

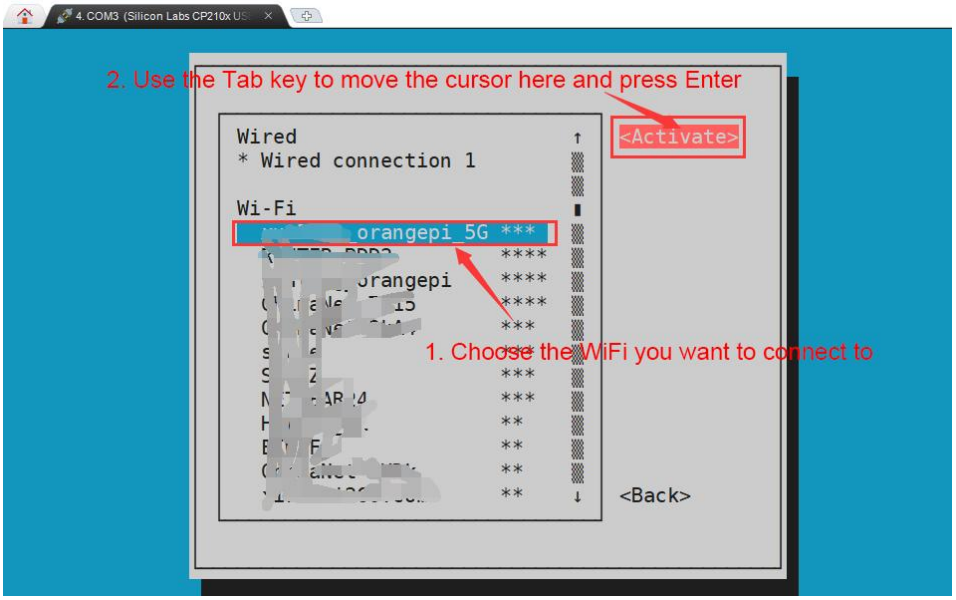


5) Then you can see all the searched WIFI hotspots

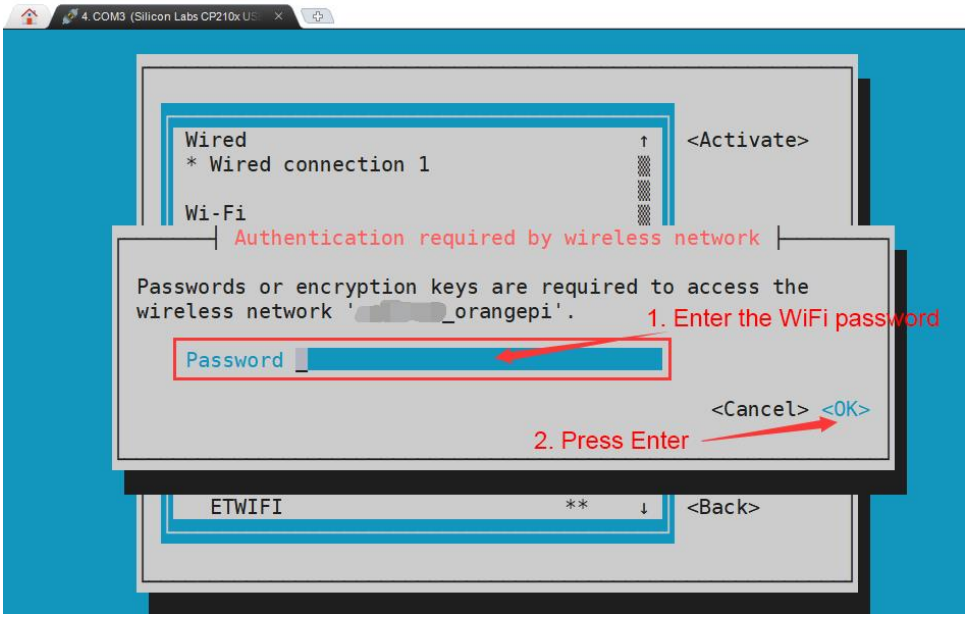


6) Select the WIFI hotspot you want to connect, then use the Tab key to position the

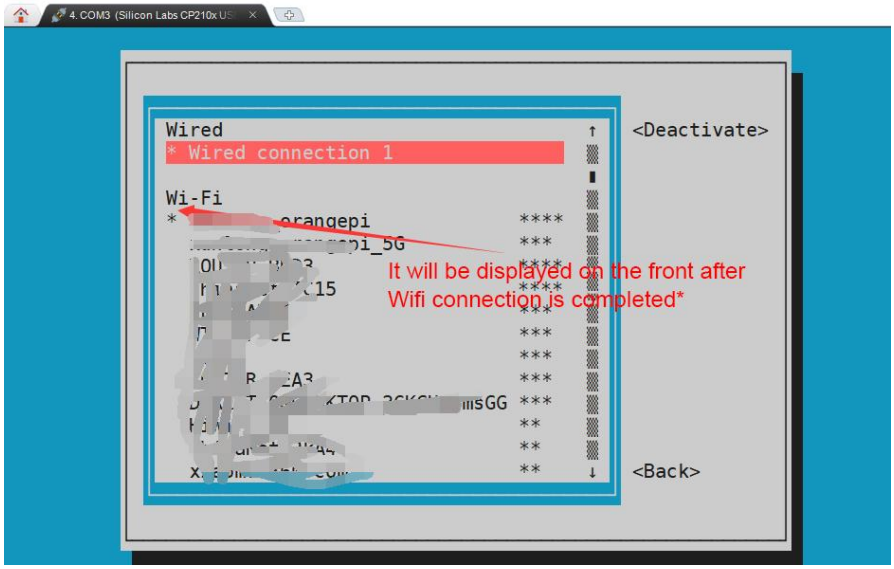
cursor to **Activate** and press Enter



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



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



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

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

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

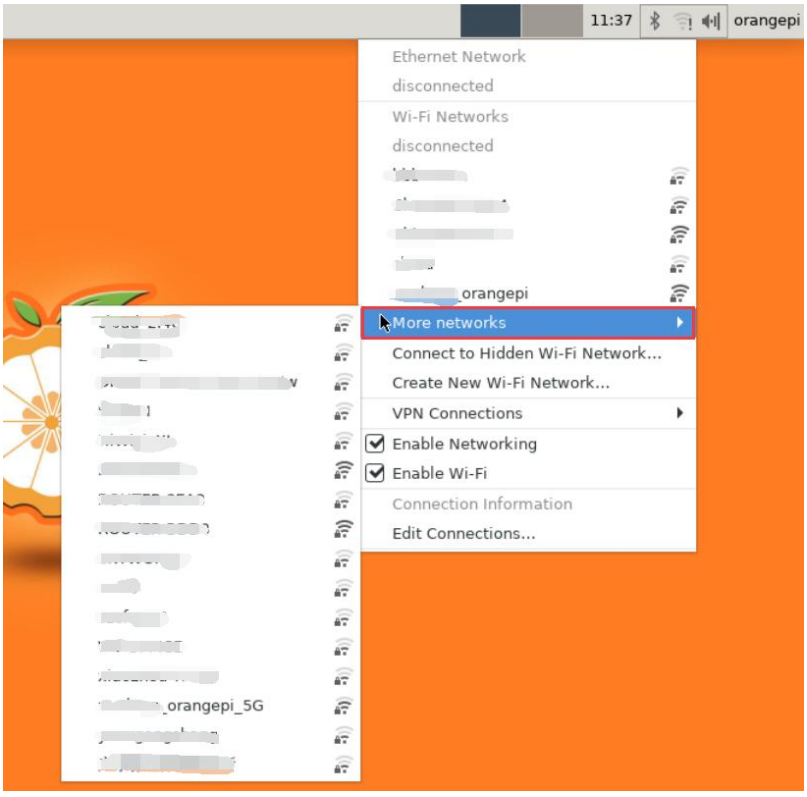
```
orangepi@orangepi:~$ ping www.orangepi.org -I wlan0
PING www.orangepi.org (182.92.236.130) from 192.168.1.49 wlan0: 56(84) bytes of data.
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=1 ttl=52 time=43.5 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=2 ttl=52 time=41.3 ms
64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

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

3.8.2.3. Test method for desktop image

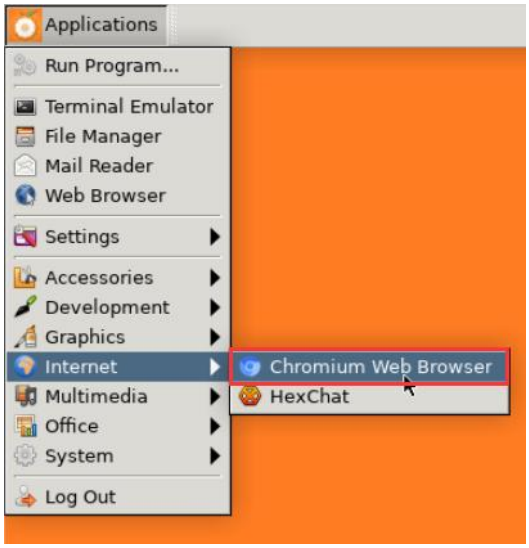
1) Click the network configuration icon in the upper right corner of the desktop (please do not connect the network cable when testing WIFI)

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

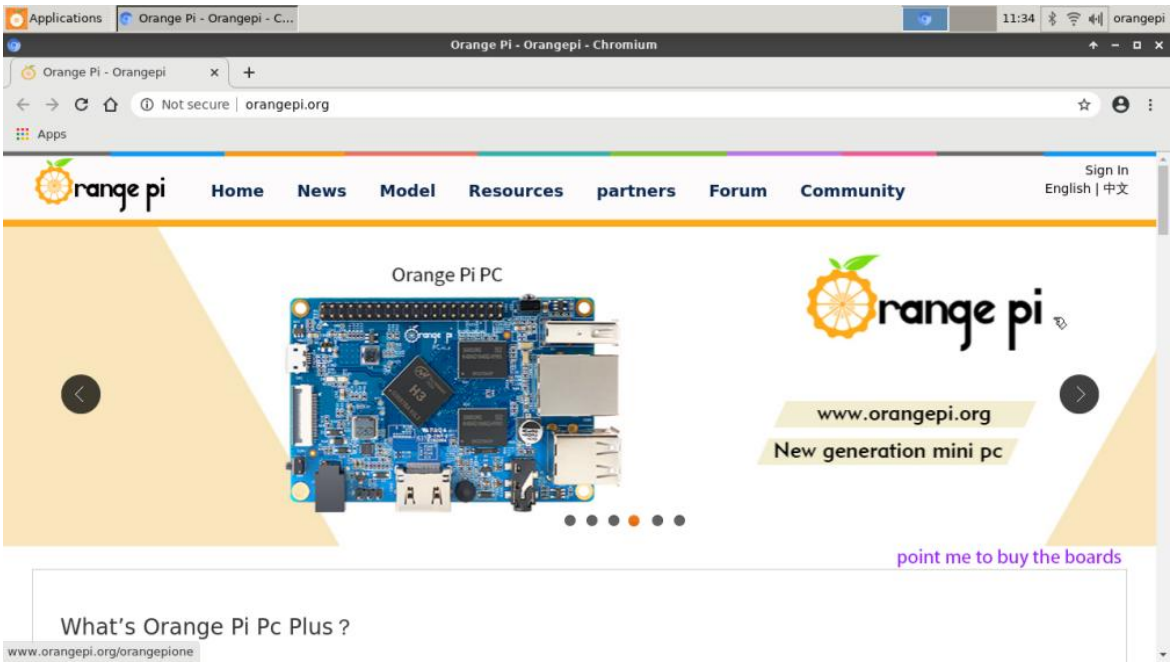


3) Then enter the password of the WIFI hotspot, and then click **Connect** to start connecting to the WIFI

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



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



3.8.3. Way to use Hostapd to establish a WIFI hotspot

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

If you do not need to access the external network, but only need to connect to the WIFI hotspot transmitted by the development board, then it is also possible to not connect the network cable.

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

1) First enter the **orangepi-config** command in the terminal

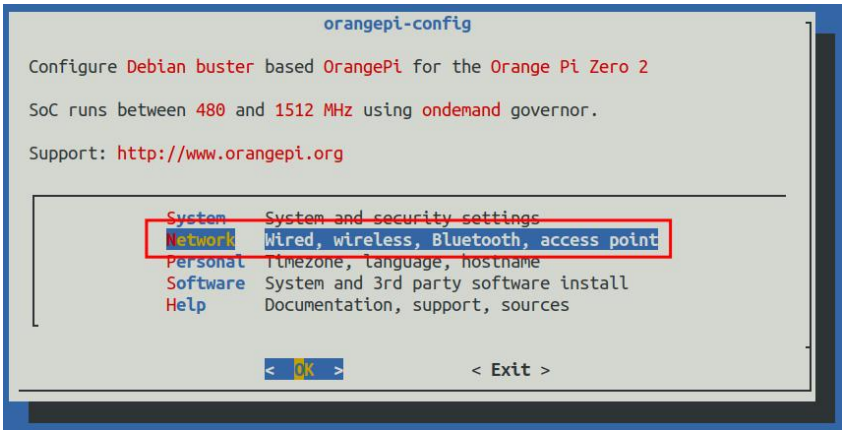
```
root@orangepi:~$ sudo orangepi-config
```

If the network cable is not connected, the following information will be prompted after running orangepi-config, then press Enter to continue.

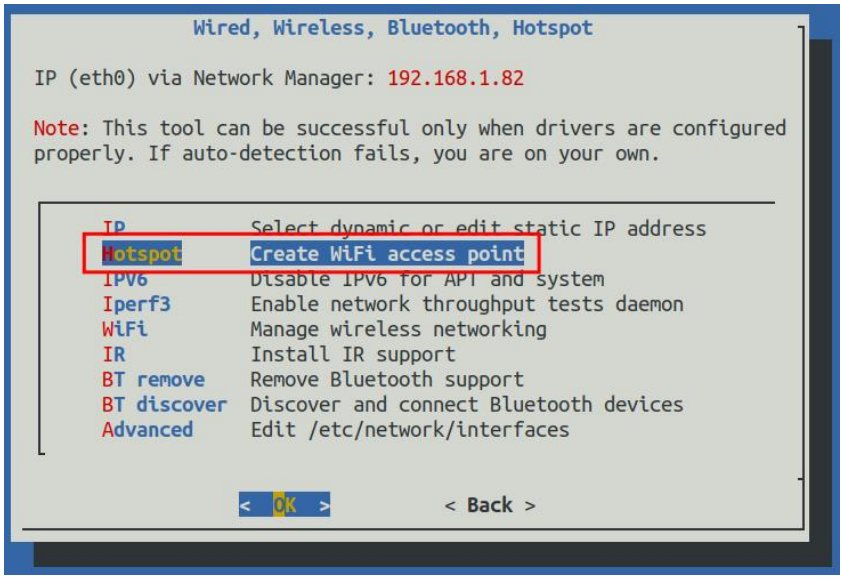
```
orangepi@orangepi:~$ sudo orangepi-config
Warning: Configuration cannot work properly without a working internet connection.
```

Press CTRL C or any key to ignore and continu

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



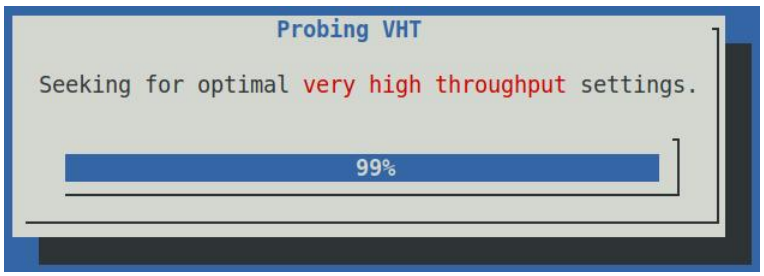
3) Then select **Hotspot Create WiFi access point** option to start setting up Hotspot



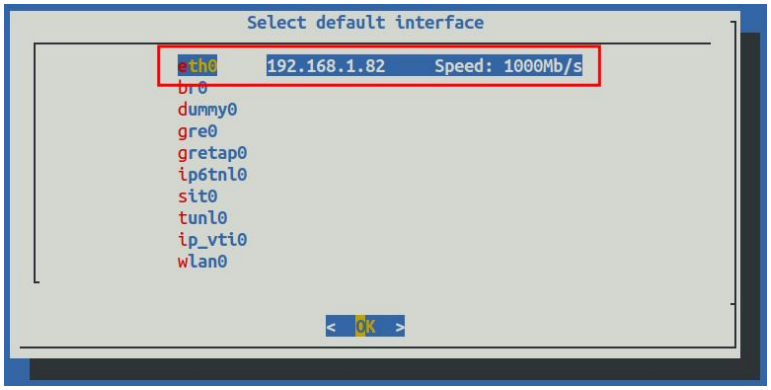
4) When the following selection box pops up, select **wlan0**

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

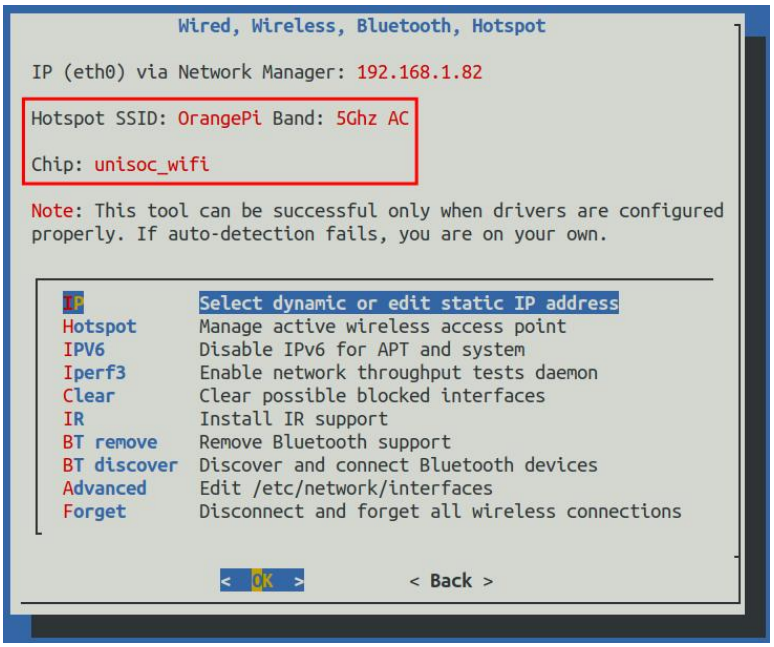
5) Then orangepi-config will start a series of settings, just wait patiently at this time



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



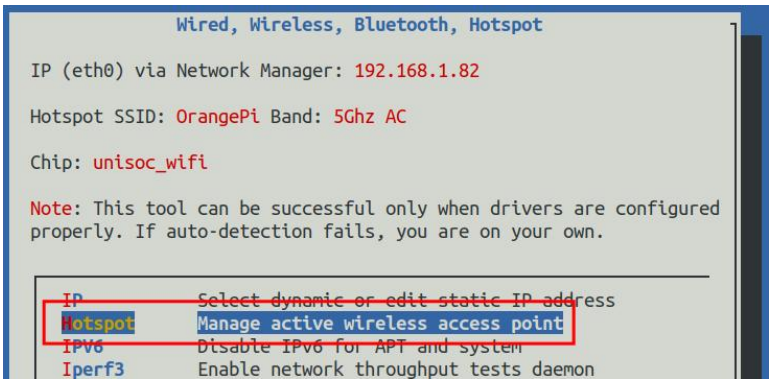
7) After all settings are completed, if orangepi-config displays the following interface, it means that Hostapd is set correctly



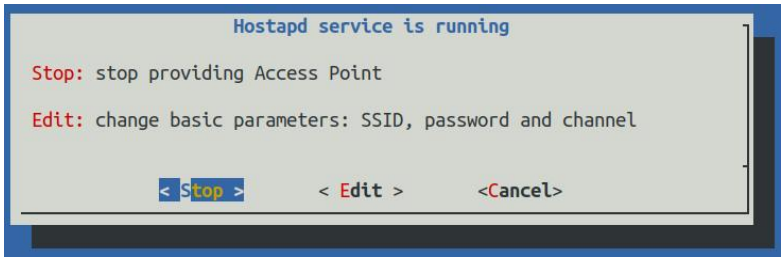
8) The name of the WIFI hotspot set by Hotspot is: **OrangePi** by default, and the password is: **12345678**. If everything is normal, the mobile phone or computer can search for the WIFI hotspot named OrangePi. Hostapd is set up correctly. The following figure is a schematic diagram of the WIFI hotspot emitted by the mobile phone connected to the development board:

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

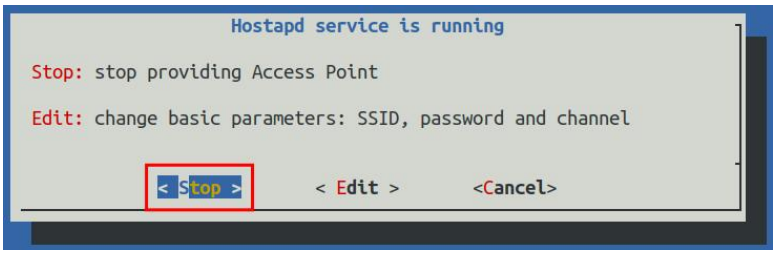
a. First select **Hotspot** in config-config



b. Then you can see the following selection interface

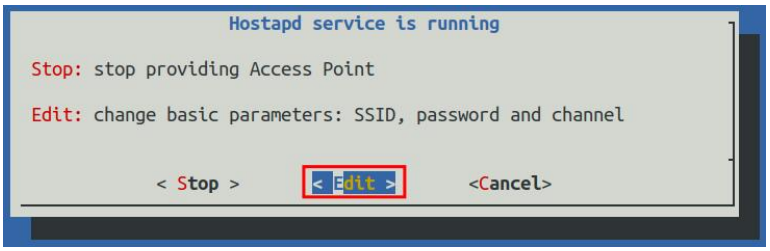


c. Select **Stop** to stop the Hostapd service

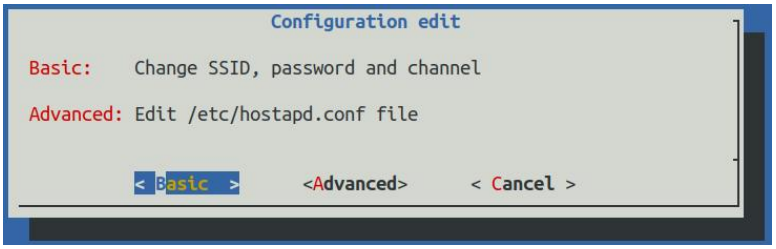


d. Select **Edit** to edit the configuration of Hostapd

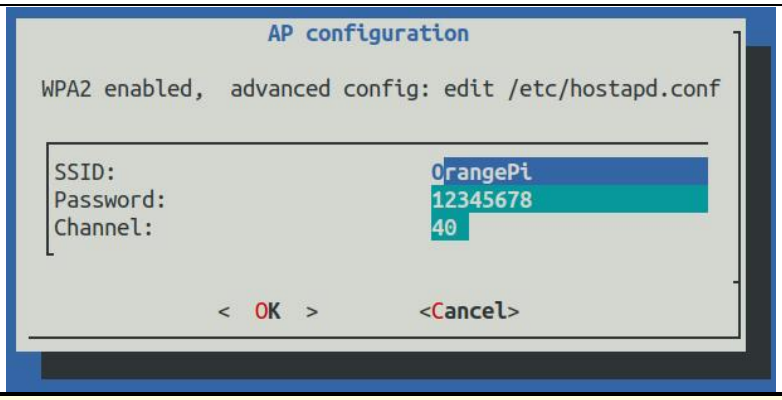
a) The **Edit** option location of Hostapd is shown in the figure below



b) **Edit** after the option is turned on, it will be as shown in the figure below

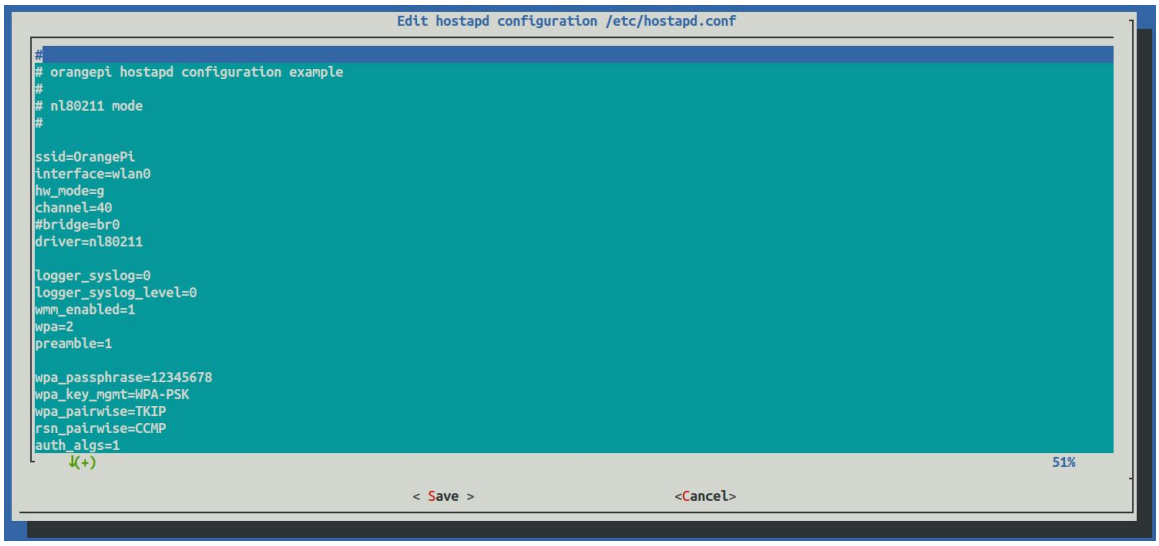


i. Select **Basic** to modify the name and password of the WIFI hotspot. After modification, select <**OK**> to save



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

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



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

3.8.4. The Way to set a static IP address

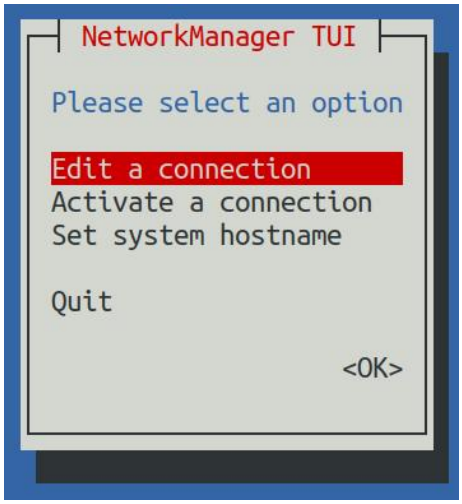
Please do not set a static IP address by modifying the /etc/network/interfaces configuration file.

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

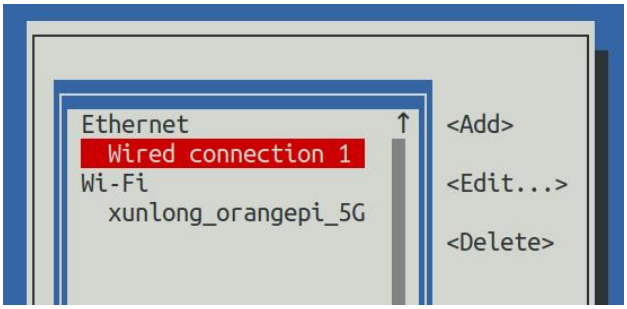
1) First run the **nmtui** command

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

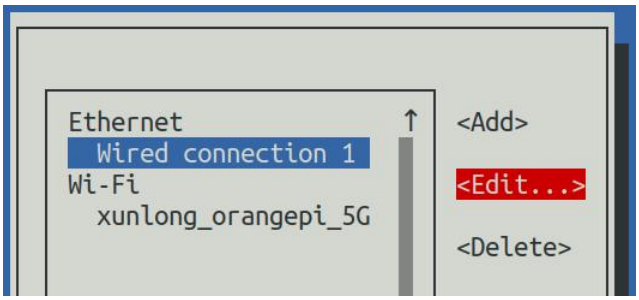
2) Then select **Edit a connection** and press enter



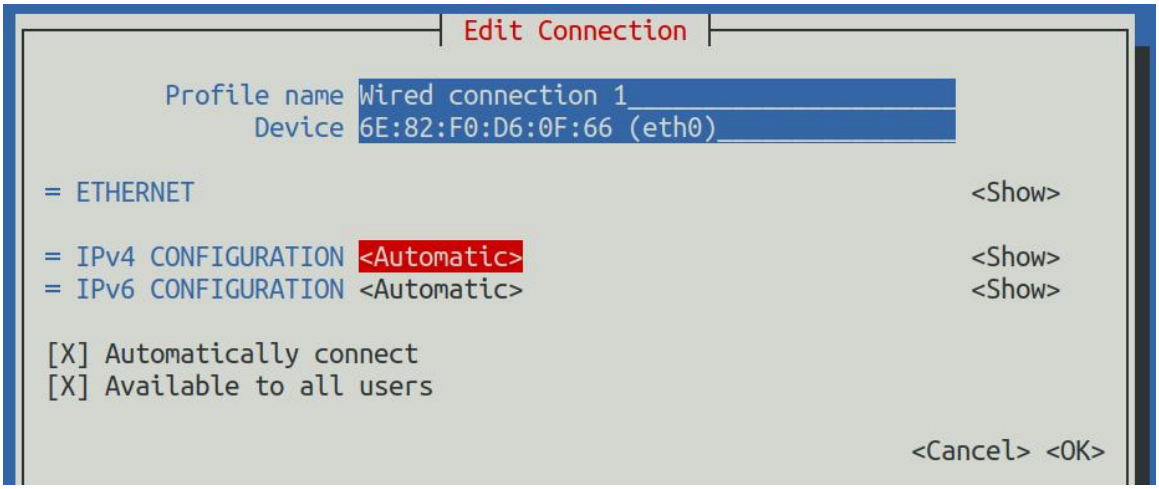
3) Then select the network interface that needs to set a static IP address, such as setting the static IP address of the **Ethernet** interface and select **Wired connection 1**.



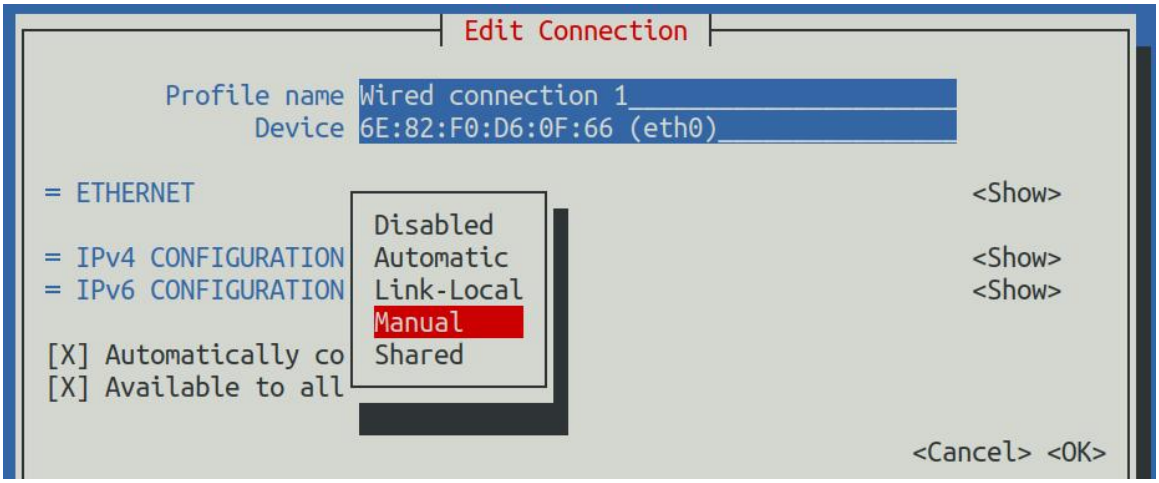
4) Then select **Edit** by **Tab** key and press Enter



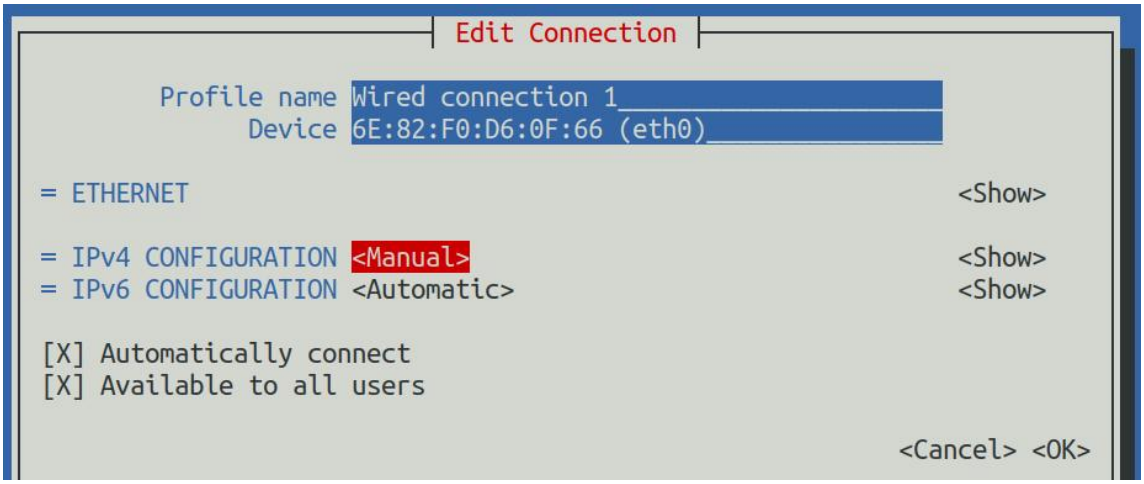
5) Then use the Tab key to move the cursor to the <**Automatic**> position shown in the figure below to configure IPv4



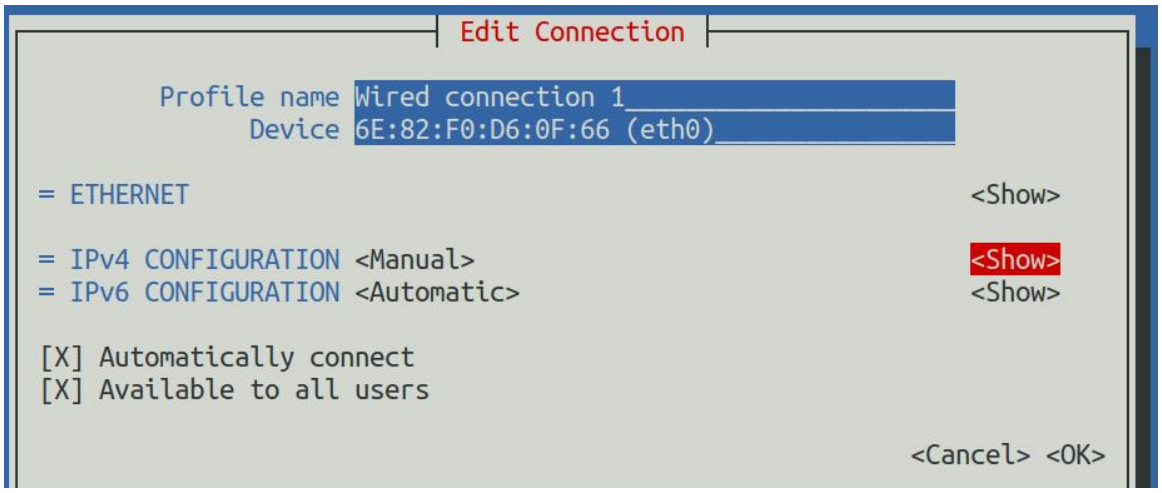
6) Then press Enter, use the up and down arrow keys to select **Manual**, then press Enter to confirm



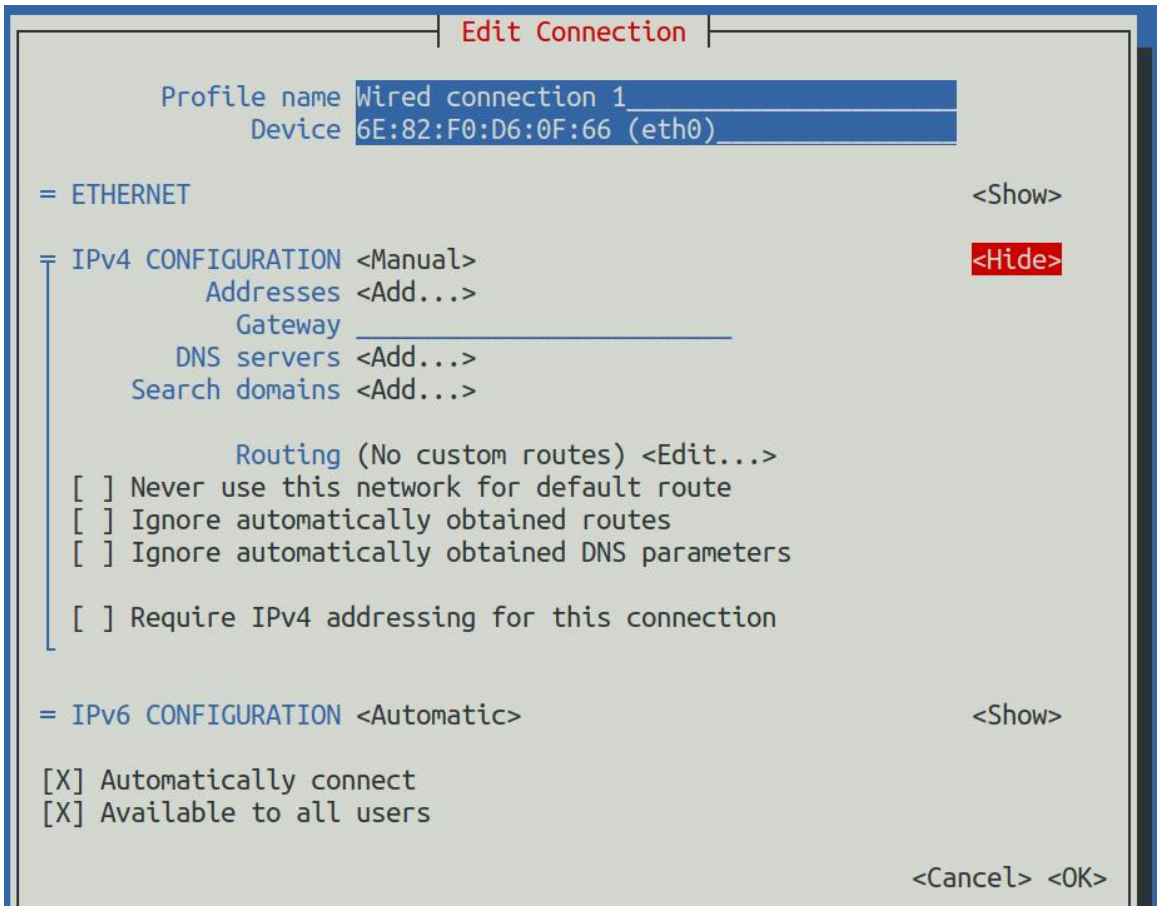
7) The display after selection is as shown below



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

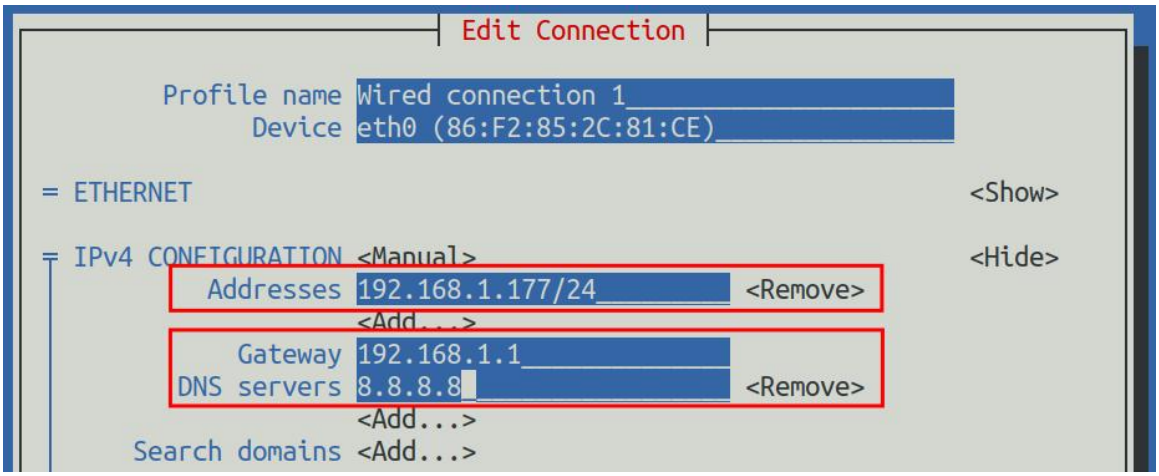


9) Then press Enter, the following setting interface will pop up after pressing Enter

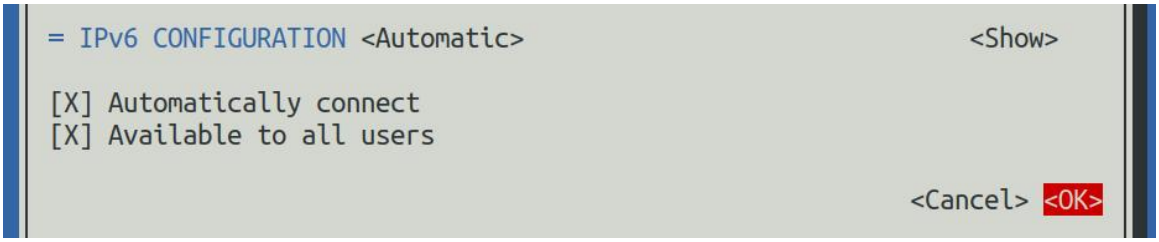


10) Then you can set the IP address (Addresses), gateway (Gateway) and DNS server addresses in the locations shown in the figure below (there are many other setting options,

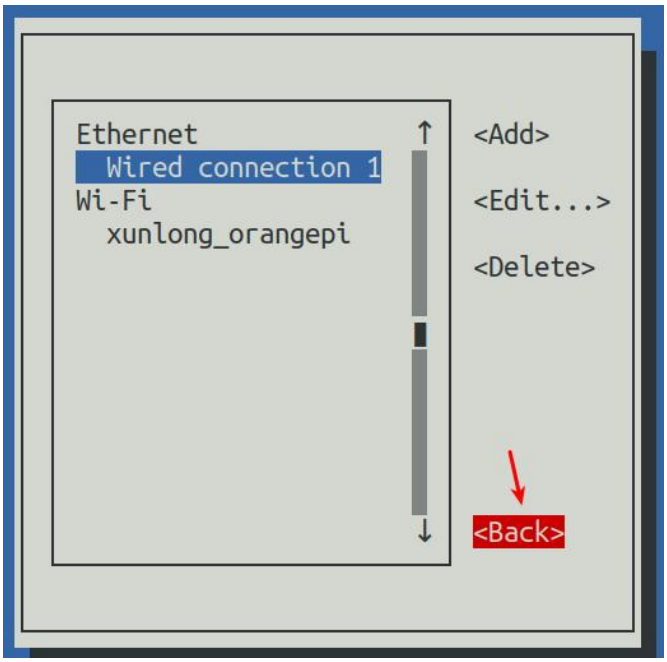
please explore by yourself)，**Please set according to your specific needs, the value set in the following figure is just an example**



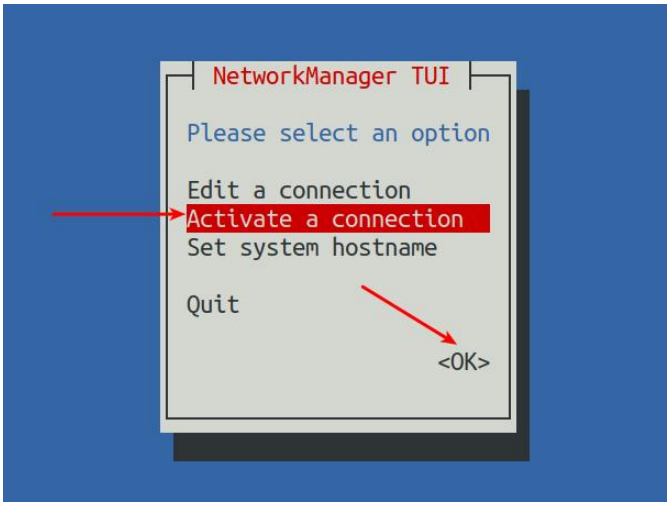
11) After setting, move the cursor to <**OK**> in the lower right corner, then press Enter to confirm



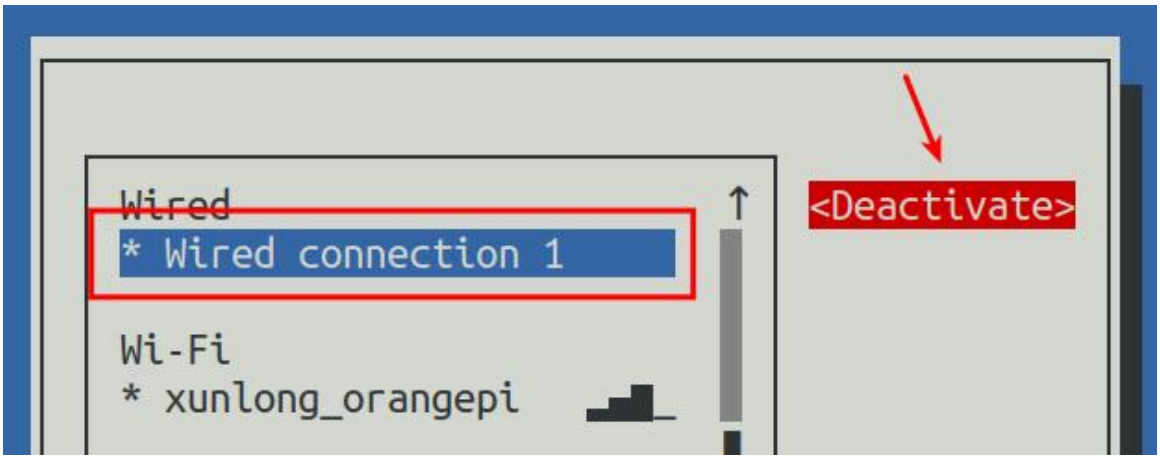
12) Then click <**Back**> to return to the previous selection interface



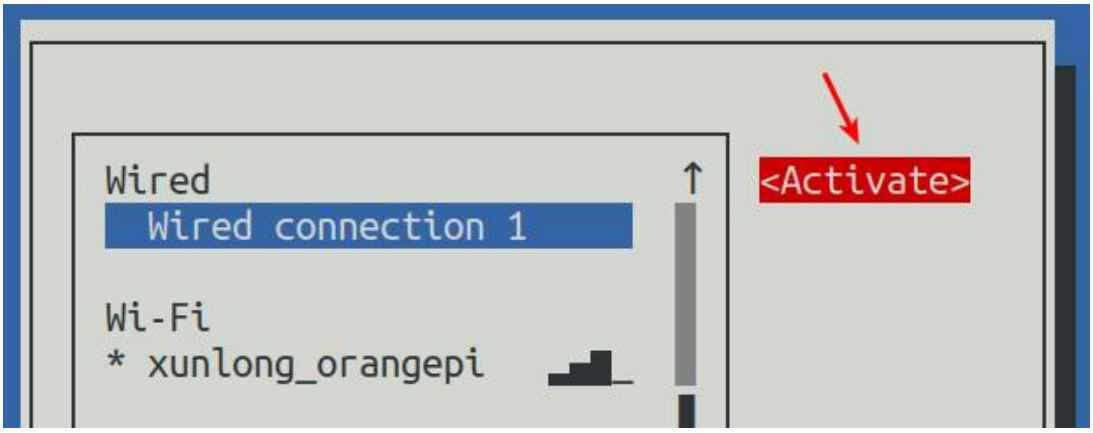
13) Then select **Activate a connection**, move the cursor to <**OK**>, and finally click Enter



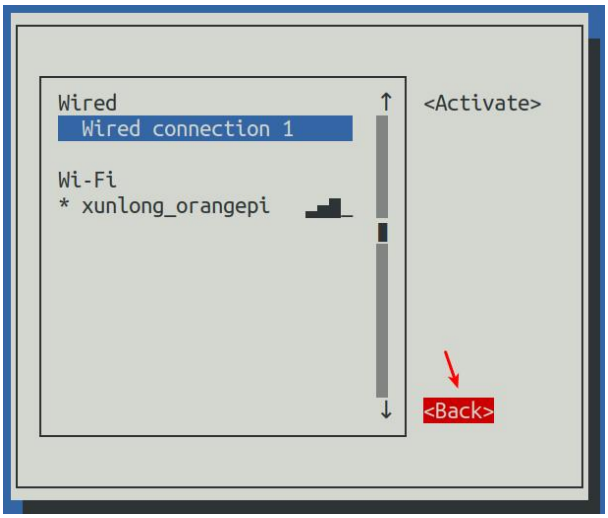
14) Then select the network interface to be set, such as **Wired connection 1**, then move the cursor to <**Deactivate**>, and press Enter to disable **Wired connection 1**

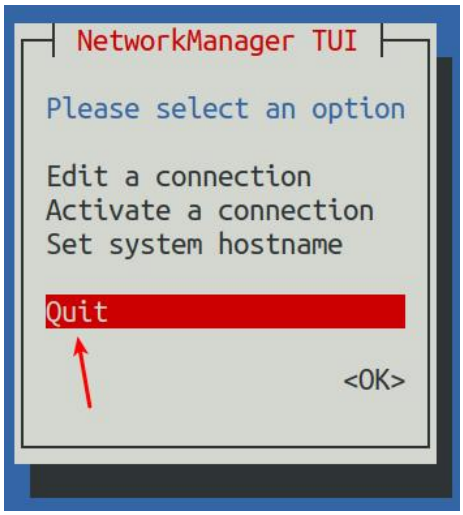


15) Then please do not move the cursor, and then press the Enter key to re-enable **Wired connection 1**, so that the static IP address set earlier will take effect



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





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

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

```
    valid_lft forever preferred_lft forever
```

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

```
orangepi@orangepi:~$ ping 192.168.1.47 -I eth0
PING 192.168.1.47 (192.168.1.47) from 192.168.1.188 eth0: 56(84) bytes of data.
64 bytes from 192.168.1.47: icmp_seq=1 ttl=64 time=0.233 ms
64 bytes from 192.168.1.47: icmp_seq=2 ttl=64 time=0.263 ms
64 bytes from 192.168.1.47: icmp_seq=3 ttl=64 time=0.273 ms
64 bytes from 192.168.1.47: icmp_seq=4 ttl=64 time=0.269 ms
64 bytes from 192.168.1.47: icmp_seq=5 ttl=64 time=0.275 ms
^C
--- 192.168.1.47 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4042ms
rtt min/avg/max/mdev = 0.233/0.262/0.275/0.015 ms
```

3.8.4.2.Using nmcli command to set static IP address

1) If you want to set the static IP address of the network port, please insert the network cable into the development board first. If you need to set the **static IP address of the WIFI, please connect the WIF**I first, and then start to set the static IP address

2) Then you can view the name of the network device through the **nmcli con show** command, as shown below

a. **Orangepi** is the name of the WIFI network interface (the names are not necessarily the same)
b. **Wired connection 1** the name of the ethernet interface

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

3) Then enter the following command,

a. **"Wired connection 1"** It means to set the static IP address of the Ethernet port. If you need to set the static IP address of the WIFI, please modify it to the name

corresponding to the WIFI network interface (which can be obtained through the **nmcli con** show command)

b. **ipv4.addresses** The following is the static IP address to be set, which can be modified to the value you want to set.

c. **ipv4.gateway** indicates the address of the gateway

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

4) Then restart the linux system

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

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

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

3.9. SSH remote login development board

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

3.9.1. SSH remote login development board under Ubuntu

1) Obtain the IP address of the development board

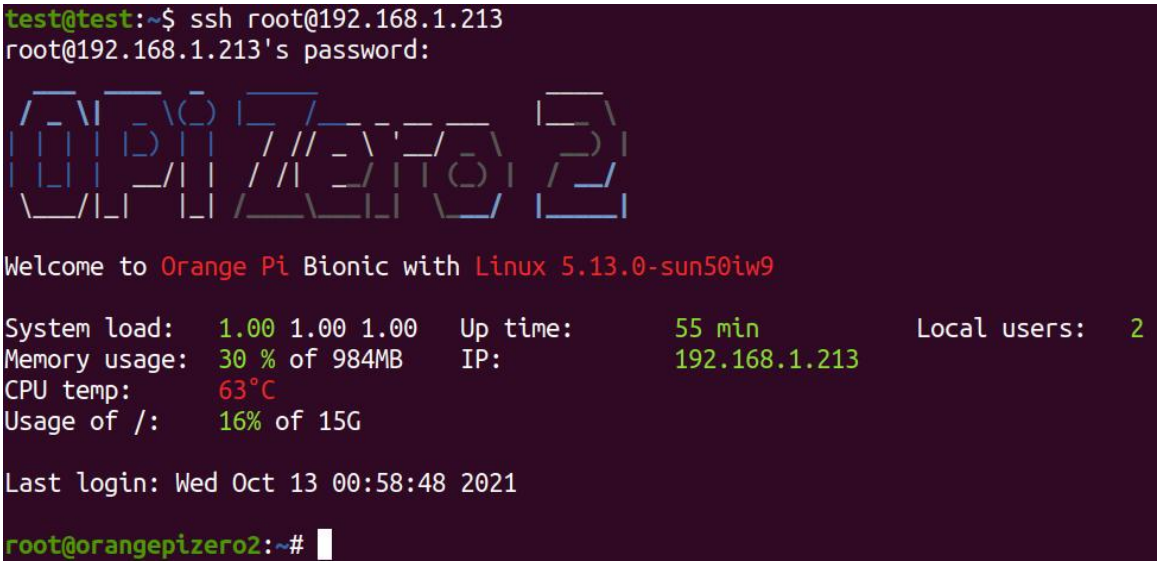
2) Then you can log in to the Linux system remotely through the ssh command

test@test:~$ **ssh root@192.168.1.xxx** (It needs to be replaced with the IP address of the development board)
root@192.168.1.xx's password: （Enter the password here, the default password is orangepi）

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

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

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



If ssh cannot log in to the linux system normally, please first check whether the IP address of the development board can be pinged. Can connect:
root@orangepi:~# **rm /etc/ssh/ssh_host_***
root@orangepi:~# **dpkg-reconfigure openssh-server**

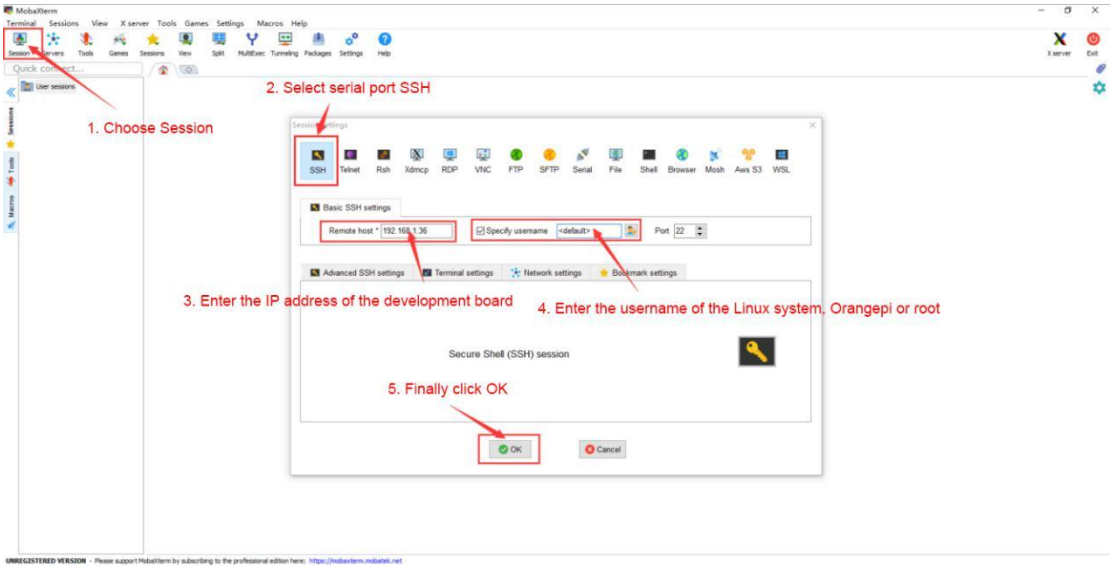
If it still doesn't work, please restart the system and try it.

3.9.2. SSH remote login development board under Windows

1) First get the IP address of the development board

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

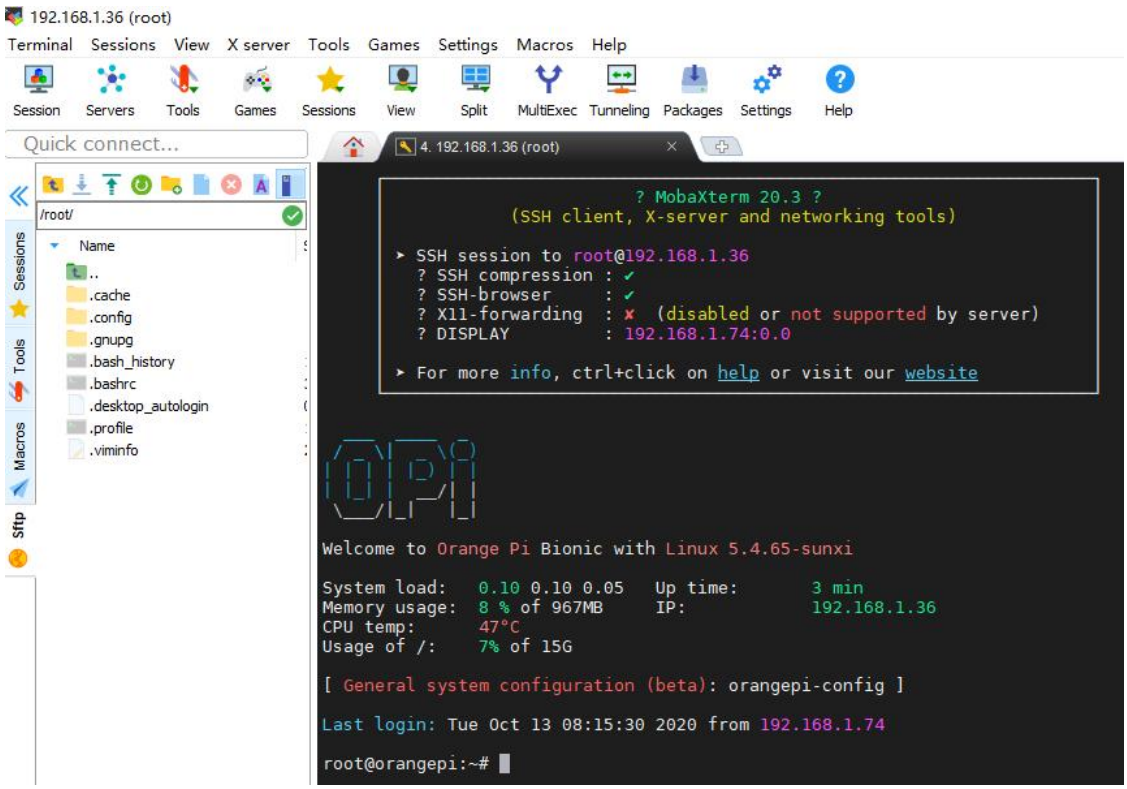
a. Open **Session**
b. Then select **SSH** in **Session Setting**
c. Then enter the IP address of the development board in **Remote host**
d. Then enter the username **root** or **orangepi** of the linux system in **Specify username**
e. Finally click **OK**



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

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

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



3.10. HDMI Test

3.10.1. HDMI Display Test

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

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

the HDMI interface is working normally

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

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

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

3.10.2. HDMI to VGA display test

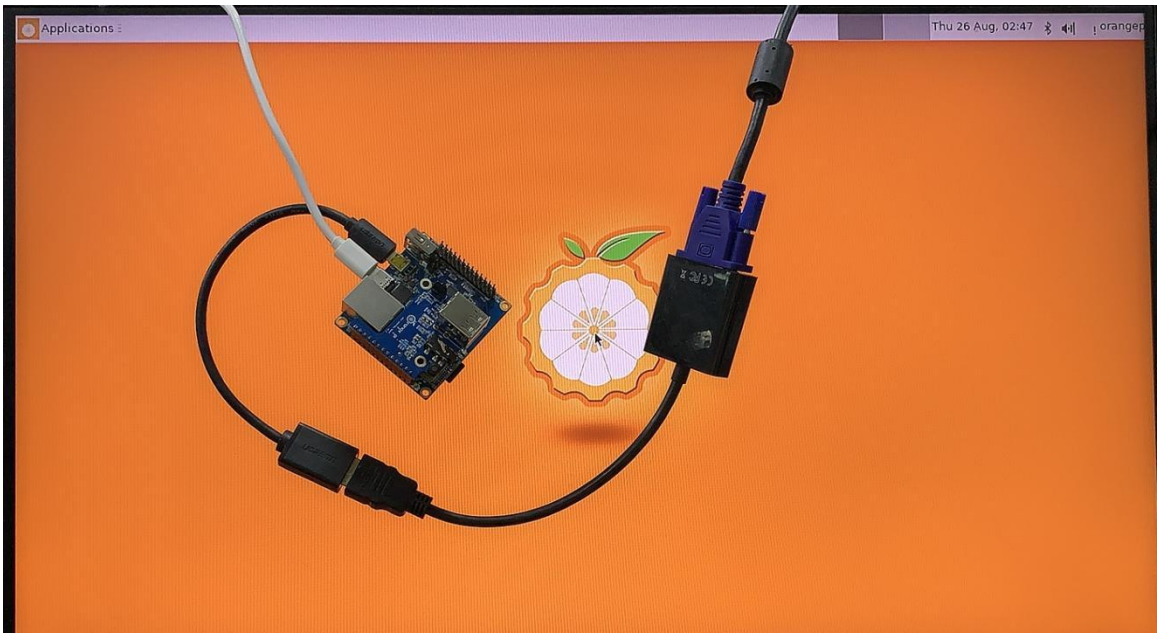
1) First you need to prepare the following accessories

a. HDMI to VGA converter

b. A VGA cable and a Micro HDMI male to HDMI female adapter cable

c. A monitor or TV that supports VGA interface

2) HDMI to VGA display test as shown below



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

3.10.3. Linux4.9 HDMI Resolution Setting

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

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

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
verbosity=1
console=both
disp_mode=1080p60
fb0_width=1920
fb0_height=1080
```

2) disp_mode the values supported by the variable are set as shown in the table below

disp_mode supported values	HDMI Resolution	HDMI refresh rate

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

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

3) Modify the value of the disp_mode variable to the resolution you want to output, then restart the system, and HDMI will output the set resolution

4) If the output resolution of HDMI is set to **2160p24, 2160p25 or 2160p30**, HDMI needs to be connected to a TV or monitor that supports 4K display to display normally

a. When connected to a 4K TV, the display is as follows

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

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

```
orangepi@orangepi:~$ sudo cat /sys/class/disp/disp/attr/sys
```

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

3.10.4. Modification of Framebuffer Width and Height

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

1) There are two variables fb0_width and fb0_height in **/boot/orangepiEnv.txt** of the linux system, which can be used to set the width and height of the Framebuffer. The linux system defaults to fb0_width=1920, fb0_height=1080

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
verbosity=1
console=both
disp_mode=1080p60
fb0_width=1920
fb0_height=1080
```

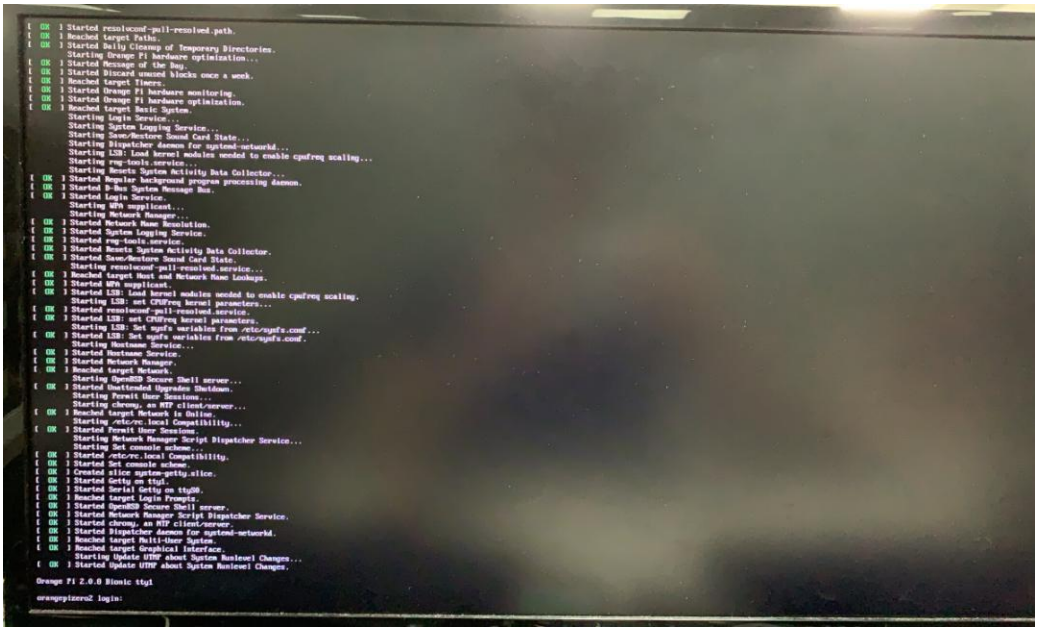
2) The reference values corresponding to different resolutions of fb0_width and fb0_height are as follows

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

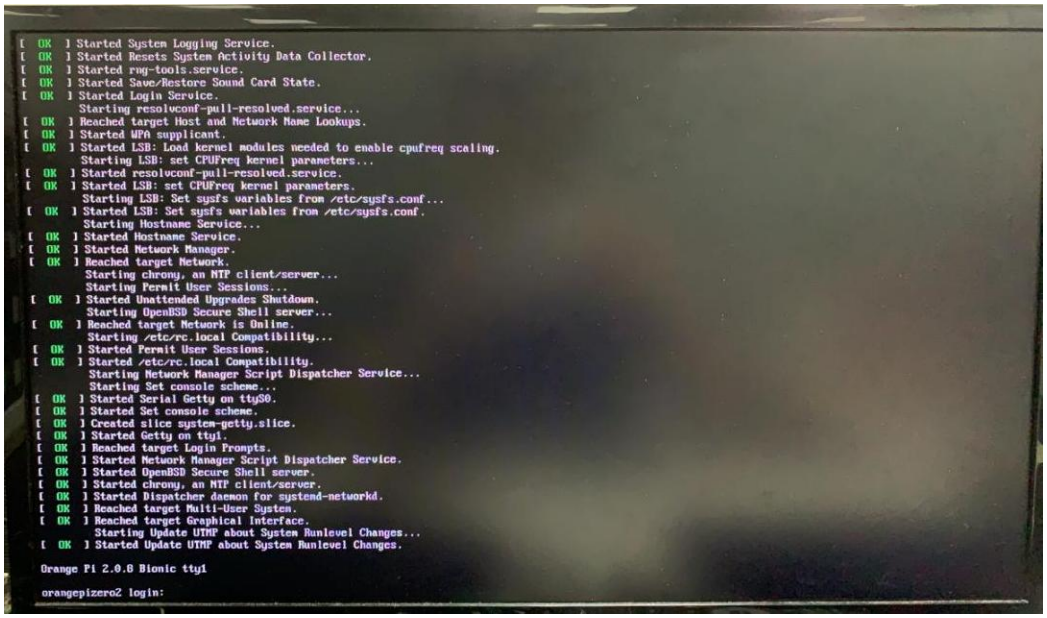
HDMI Resolution	fb0_width	fb0_height
480p	720	480
576p	720	576
720p	1280	720
1080p	1920	1080
2160p	1920	1080

3) Under the same HDMI resolution, the display of different fb0_width and fb0_height is as follows. When the value of fb0_width and fb0_height is larger, the text displayed on the screen is smaller. When the value of fb0_width and fb0_height is smaller, The larger the text displayed on the screen

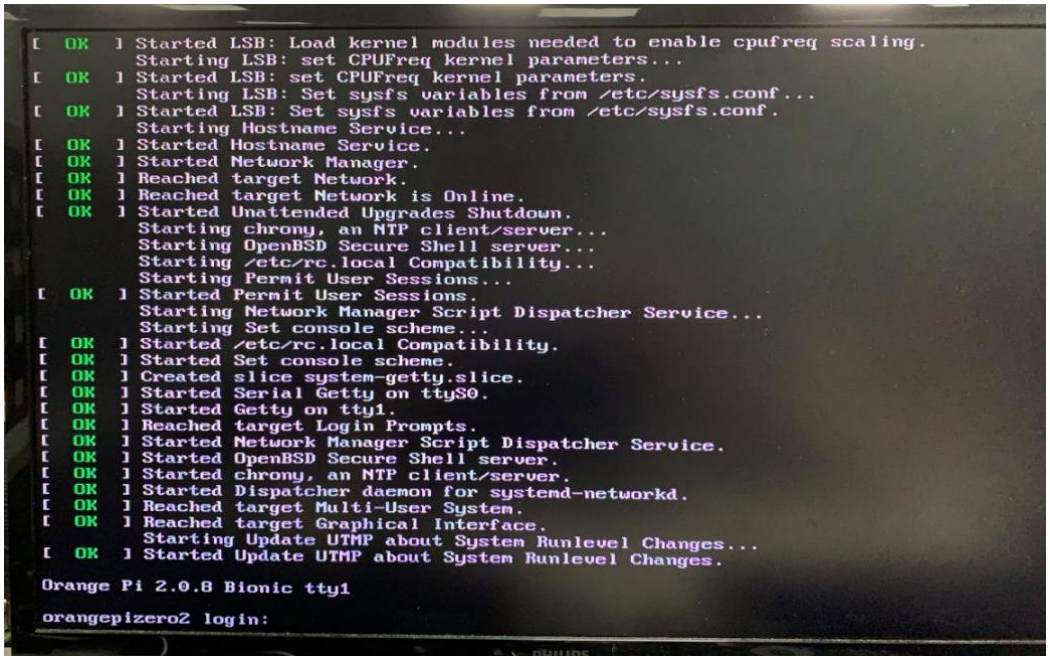
a. HDMI resolution is 1080p60, fb0_width and fb0_height are 1920x1080 display



b. HDMI resolution is 1080p60, fb0_width and fb0_height are 1280x720 display



c. The display case with HDMI resolution of 1080p60, fb0_width and fb0_height of 720x576



d. The display case with HDMI resolution of 1080p60, fb0_width and fb0_height of 720x480

3.10.5. Framebuffer cursor settings

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

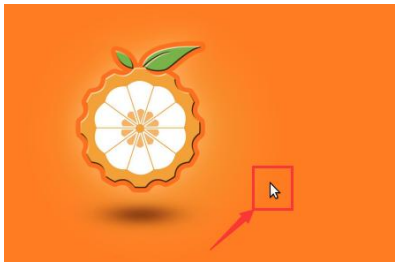
```
root@orangepi:~# echo 1 > /sys/class/graphics/fbcon/cursor_blink     #Cursor blink
root@orangepi:~# echo 0 > /sys/class/graphics/fbcon/cursor_blink     #Cursor does not blink
```

2) If you need to hide the cursor, you can use the **extraargs** variable in **/boot/orangepiEnv.txt**（The value of extraargs will be assigned to the bootargs environment variable and finally passed to the kernel） added in **vt.global_cursor_default=0**（If **vt.global_cursor_default**=1, the cursor is displayed），Then restart the system and you will see that the cursor has disappeared

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
verbosity=1
console=both
disp_mode=1080p60
fb0_width=1920
fb0_height=1080
extraargs=vt.global_cursor_default=0
```

3.10.6. The Way to hide the mouse cursor on the USB touch screen

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



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

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

3) **Then restart the Linux system**

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

3.11. Orange Pi 5-inch TFT LCD screen test

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

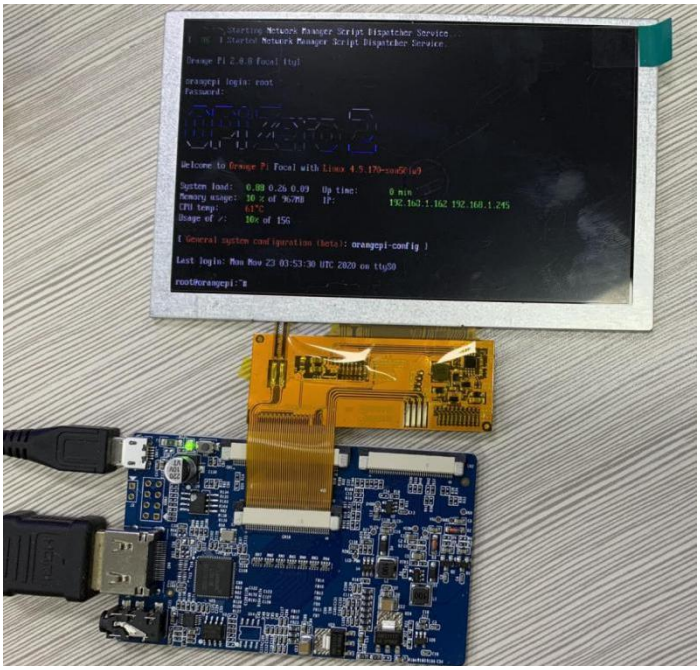
1) First prepare the 5-inch TFT LCD screen of the orange pi. The wiring method of the screen cable and the adapter board is as shown in the figure below. Please do not reverse the connection. In addition, please ensure that the screen cable and the cable socket are in place. , if the contact is not in place, the screen output will have problems

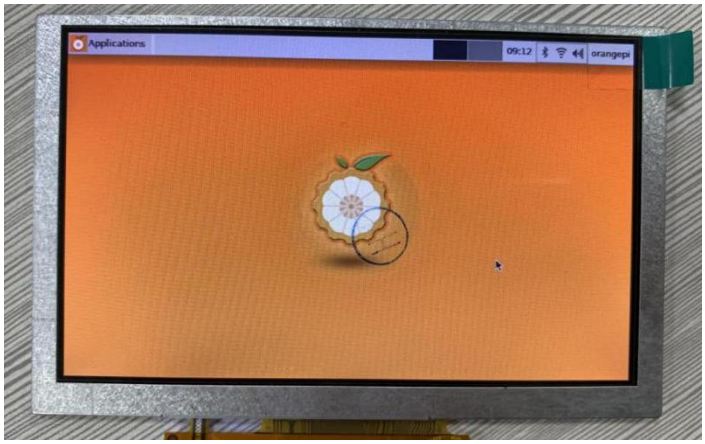
2) Then use the Micro HDMI cable to connect the Micro HDMI port of the development board to the HDMI port of the adapter board. The screen needs to be powered separately. Please insert a 5V/2A power supply into the Micro USB port of the adapter board.

3) Modify the HDMI output resolution in **/boot/orangepiEnv.txt** to 480p, the length and width of the Framebuffer can be set to 800x480, if set to 1280x720, the font displayed on the TFT LCD screen will be very small

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
disp_mode=480p
fb0_width=800
fb0_height=480
```

4) After the development board is started, the screen can see the splash screen





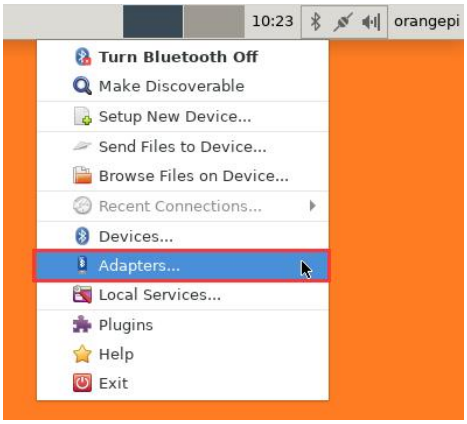
5) After turning off the power of the development board, please remember to turn off the power of the screen adapter board at the same time, and turn it on again next time you start the development board

3.12. The Way to use Bluetooth

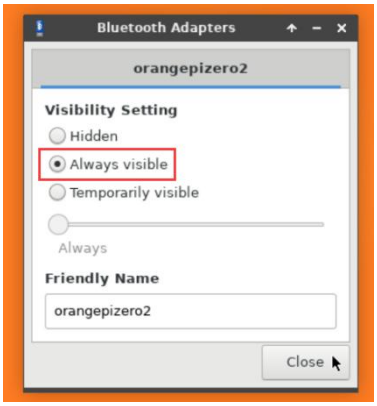
3.12.1. Test method of desktop version image

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

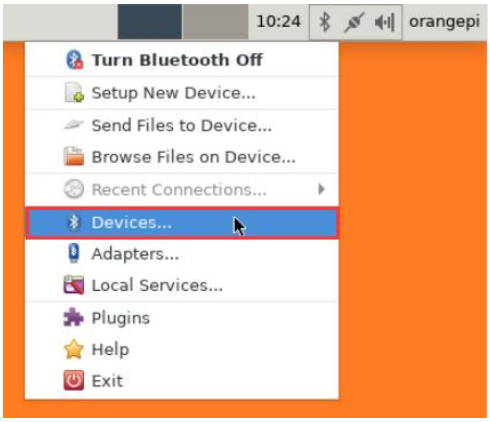
2) Then select adapter



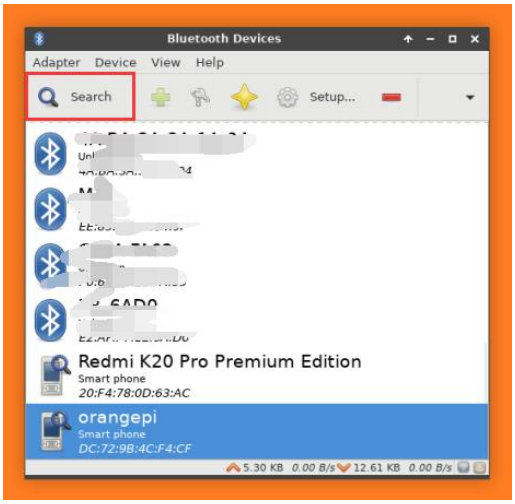
3) Set the **Visibility Setting** to **Always visible** in the Bluetooth adapter setting interface, and then click close to close



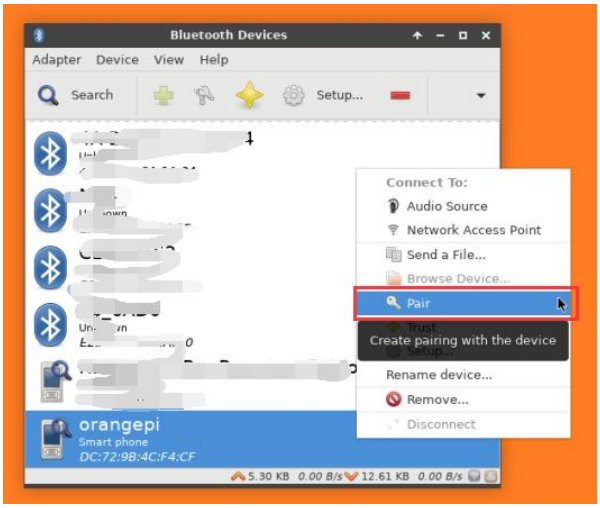
4) Then open the configuration interface of the Bluetooth device



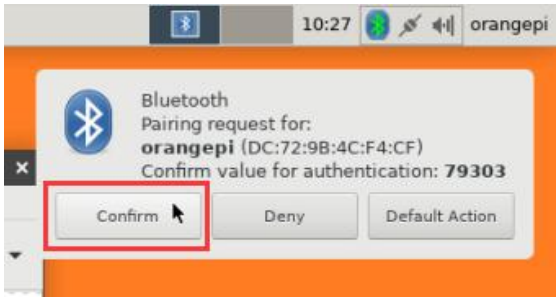
5) Click **Search** to start scanning for surrounding Bluetooth devices



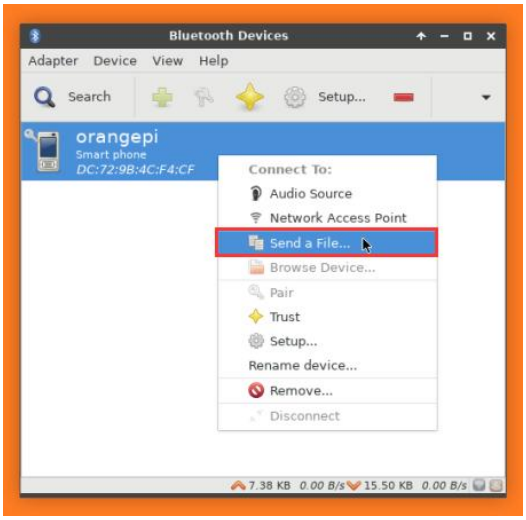
6) Then select the Bluetooth device you want to connect, and then click the right mouse button to pop up the operation interface of the Bluetooth device. Select **Pair** to start pairing. The demonstration here is pairing with an Android phone.



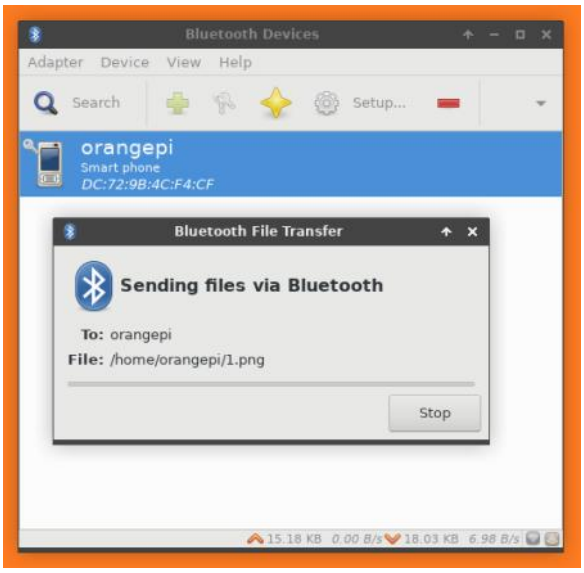
7) When pairing, a pairing confirmation box will pop up in the upper right corner of the desktop. Select **Confirm** to confirm. At this time, the mobile phone also needs to be confirmed.



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



9) The interface for sending pictures is as follows



3.12.2. The Way to use the server version image

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

```
orangepi@orangepi:~$ sudo apt update && sudo apt install bluez
orangepi@orangepi:~$ hciconfig -a
hci0:   Type: Primary   Bus: UART
        BD Address: 10:11:12:13:14:15   ACL MTU: 1021:8   SCO MTU: 240:3
        UP RUNNING
        RX bytes:646 acl:0 sco:0 events:37 errors:0
        TX bytes:2650 acl:0 sco:0 commands:37 errors:0
        Features: 0xbf 0xff 0x8d 0xfe 0xdb 0x3d 0x7b 0xc7
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy:
        Link mode: SLAVE ACCEPT
        Name: 'orangepizero2'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 5.0 (0x9)   Revision: 0x400
        LMP Version: 5.0 (0x9)   Subversion: 0x400
```

Manufacturer: Spreadtrum Communications Shanghai Ltd (492)

2) Scan for bluetooth devices using **bluetoothctl**

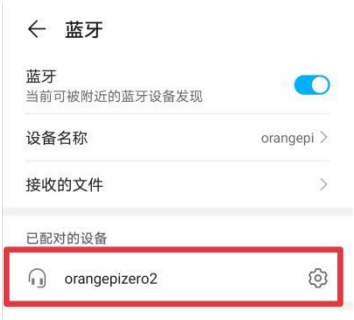
```
orangepi@orangepi:~$ sudo bluetoothctl
[NEW] Controller 10:11:12:13:14:15 orangepizero2 [default]
Agent registered
[bluetooth]# power on          #Enable controller
Changing power on succeeded
[bluetooth]# discoverable on      #Make the controller discoverable
Changing discoverable on succeeded
[CHG] Controller 10:11:12:13:14:15 Discoverable: yes
[bluetooth]# pairable on     #Set the controller to be pairable
Changing pairable on succeeded
[bluetooth]# scan on          #Start scanning for nearby bluetooth devices
Discovery started
[CHG] Controller 10:11:12:13:14:15 Discovering: yes
[NEW] Device 76:60:79:29:B9:31 76-60-79-29-B9-31
[NEW] Device 9C:2E:A1:42:71:11 Xiaomi phone
[NEW] Device DC:72:9B:4C:F4:CF orangepi
[bluetooth]# scan off          #After scanning the Bluetooth device you want to connect, you can close the scan, and then write down the MAC address of the Bluetooth device. The Bluetooth device tested here is an Android phone, the Bluetooth name is orangepi, and the corresponding MAC address is DC:72:9B:4C :F4:CF
Discovery stopped
[CHG] Controller 10:11:12:13:14:15 Discovering: no
[CHG] Device DC:72:9B:4C:F4:CF RSSI is nil
```

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

```
[bluetooth]# pair DC:72:9B:4C:F4:CF        #Use the scanned MAC address of the Bluetooth device for pairing
Attempting to pair with DC:72:9B:4C:F4:CF
[CHG] Device DC:72:9B:4C:F4:CF Connected: yes
Request confirmation
[leeb1m[agent] Confirm passkey 764475 (yes/no): yes    #Enter yes here, you also need
```

```
to confirm on the phone
[CHG] Device DC:72:9B:4C:F4:CF Modalias: bluetooth:v010Fp107Ed1436
[CHG] Device DC:72:9B:4C:F4:CF UUIDs: 0000046a-0000-1000-8000-00805f9b34fb
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: yes
[CHG] Device DC:72:9B:4C:F4:CF Paired: yes
Pairing successful      #Prompt for successful pairing
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: no
[CHG] Device DC:72:9B:4C:F4:CF Connected: no
```

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



5) Connecting to a bluetooth device requires installation **pulseaudio-module-bluetooth** software folder，then restart **pulseaudio** Service

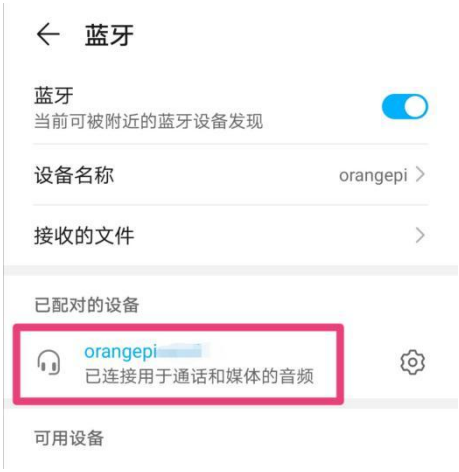
```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt -y install pulseaudio-module-bluetooth
orangepi@orangepi:~$ pulseaudio --start
```

6) The method to connect a Bluetooth device

```
orangepi@orangepi:~$ sudo bluetoothctl
Agent registered
[bluetooth]# paired-devices    #View the MAC address of a paired Bluetooth device
Device DC:72:9B:4C:F4:CF orangepi
[bluetooth]# connect DC:72:9B:4C:F4:CF      #Connect a Bluetooth device using the MAC address
Attempting to connect to DC:72:9B:4C:F4:CF
[CHG] Device DC:72:9B:4C:F4:CF Connected: yes
Connection successful
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: yes
```

```
[CHG] Controller 10:11:12:13:14:15 Discoverable: no
[orangepi]#                    #This prompt appears to indicate that the connection is successful
```

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



3.13. USB interface test

The USB interface can be connected to a USB hub to expand the number of USB interfaces.

3.13.1. Connect USB mouse or keyboard test

1) Insert the keyboard of the USB interface into the USB interface 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 normally, the USB interface is in normal use (the mouse can only be used in the desktop version of the system)

3.13.2. Connect USB storage device test

1) First, insert the U disk or USB mobile hard disk into the USB interface of the Orange Pi development board

2) Execute the following command, if you can see the output of **sdX**, it means that the U disk is successfully recognized

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

3) Use the mount command to mount the USB drive to **/mnt**, Then you can view the files in the U disk

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

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

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

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

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

3.13.3. USB Microphone Test

1) The tested USB microphone is shown below

2) First insert the USB microphone into the USB interface of the development board

3) Then use the **lsusb** command if you can see the following output, indicating that the USB microphone is recognized normally

```
orangepi@orangepi:~$ lsusb
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

```
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 007: ID 08bb:2902 Texas Instruments PCM2902 Audio Codec
```

4) Use the **arecord** -l command to check whether there is a sound card device with a USB microphone

```
orangepi@orangepi:~$ arecord -l
**** List of CAPTURE Hardware Devices ****
[...ellipsis...]
card 3: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

5) The command to record using a USB microphone is as follows,**This module only supports single-channel recording, so you need to specify the number of channels as 1**

```
orangepi@orangepi:~$ arecord -D hw:3,0 -d 10 -f cd -t wav -c 1 test.wav
```

6) After recording, if there is sound when playing the **test.wav** file, it means that the USB microphone can be used normally

3.13.4. USB wireless network card test

The **currently tested** USB wireless network cards are as follows. Please test other types of USB wireless network cards by yourself. If you cannot use them, you need to transplant the corresponding USB wireless network card driver.

serial number	model
1	RTL8723BU

3.13.4.1. RTL8723BU test

1) First insert the RTL8723BU wireless network card module into the USB interface of the development board

2) Then the linux system will automatically load the RTL8723BU Bluetooth and WIFI

related kernel modules, you can see the following output through the lsmod command

```
orangepi@orangepi:~$ lsmod
Module                  Size  Used by
rtl8xxxu              143360  0
mac80211               491520  1 rtl8xxxu
rtk_btusb              65536  0
btusb                  49152  0
btrtl                  16384  1 btusb
btbcm                  16384  1 btusb
btintel                20480  1 btusb
```

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

```
orangepi@orangepi:~$ dmesg
......
[  166.867806] sunxi-ehci 5311000.ehci3-controller: ehci_irq: highspeed device connect
[  167.090509] usb 3-1: new high-speed USB device number 2 using sunxi-ehci
[  167.228930] usb 3-1: New USB device found, idVendor=0bda, idProduct=b720
[  167.228955] usb 3-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[  167.228971] usb 3-1: Product: 802.11n WLAN Adapter
[  167.228985] usb 3-1: Manufacturer: Realtek
[  167.229000] usb 3-1: SerialNumber: 00e04c000001
[  167.336331] usbcore: registered new interface driver btusb
[  167.337532] Bluetooth: hci1: rtl: examining hci_ver=06 hci_rev=000b lmp_ver=06 lmp_subver=8723
[  167.337544] Bluetooth: hci1: rtl: loading rtl_bt/rtl8723b_config.bin
[  167.342938] Bluetooth: hci1: rtl: loading rtl_bt/rtl8723b_fw.bin
[  167.347539] Bluetooth: hci1: rom_version status=0 version=1
[  167.347629] Bluetooth: cfg_sz 68, total size 24088
[  167.352702] rtk_btusb: RTKBT_RELEASE_NAME: 20170427_TV_ANDROID_5.x
[  167.352717] rtk_btusb: Realtek Bluetooth USB driver module init, version 4.1.4
[  167.352721] rtk_btusb: Register usb char device interface for BT driver
[  167.366566] usbcore: registered new interface driver rtk_btusb
[  167.384087] usb 3-1: This Realtek USB WiFi dongle (0x0bda:0xb720) is untested!
[  167.384100] usb 3-1: Please report results to Jes.Sorensen@gmail.com
[  167.593523] usb 3-1: Vendor: Realtek
```

```
[    167.593533] usb 3-1: Product: 802.11n WLAN Adapter
......
[    167.594031] usb 3-1: RTL8723BU rev E (SMIC) 1T1R, TX queues 3, WiFi=1, BT=1, GPS=0, HI PA=0
[    167.594041] usb 3-1: RTL8723BU MAC: 00:13:ef:f4:58:ae
[    167.594052] usb 3-1: rtl8xxxu: Loading firmware rtlwifi/rtl8723bu_nic.bin
[    167.599402] usb 3-1: Firmware revision 35.0 (signature 0x5301)
[    168.488312] usbcore: registered new interface driver rtl8xxxu
```

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

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

```
orangepi@orangepi:~$ sudo apt update && sudo apt install bluez
orangepi@orangepi:~$ hciconfig
hci1:   Type: Primary   Bus: USB
        BD Address: 00:13:EF:F4:58:AE  ACL MTU: 1021:8  SCO MTU: 255:16
        UP RUNNING PSCAN ISCAN
        RX bytes:3994 acl:0 sco:0 events:206 errors:0
        TX bytes:29459 acl:0 sco:0 commands:173 errors:0

hci0:   Type: Primary  Bus: UART
        BD Address: 10:11:12:13:14:15  ACL MTU: 1021:8  SCO MTU: 240:3
        UP RUNNING
```

```
        RX bytes:2981 acl:0 sco:0 events:127 errors:0
        TX bytes:5423 acl:0 sco:0 commands:61 errors:0
```

3.13.5. USB Ethernet Card Test

1) The currently **tested** and usable USB Ethernet cards are shown below. Among them, the RTL8153 USB Gigabit network card is inserted into the USB 2.0 Host interface of the development board. The test can be used normally, but the speed cannot reach Gigabit. Please note this

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

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

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

3) Then you can see the device node of the USB network card and the automatically assigned IP address through the ifconfig command

```
orangepi@orangepi:~$ sudo ifconfig
enx00e04c362017: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.177  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::681f:d293:4bc5:e9fd  prefixlen 64  scopeid 0x20<link>
```

```
        ether 00:e0:4c:36:20:17  txqueuelen 1000  (Ethernet)
        RX packets 1849  bytes 134590 (134.5 KB)
        RX errors 0  dropped 125  overruns 0  frame 0
        TX packets 33  bytes 2834 (2.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4) The command to test network connectivity is as follows

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

3.13.6. USB camera test

1) First insert the USB camera into the USB port of the Orange Pi development board

2) Then through the lsmod command, you can see that the kernel automatically loads the following modules

```
orangepi@orangepi:~$ lsmod
Module                  Size  Used by
uvcvideo              106496  0
```

3) via v4l2-ctl（**Note that the l in v4l2 is a lowercase l, not the number 1**）The command can see that the device node information of the USB camera is /dev/video0

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y v4l-utils
orangepi@orangepi:~$ v4l2-ctl --list-devices
USB 2.0 Camera (usb-sunxi-ehci-1):
    /dev/video0
```

4) Use fswebcam to test the USB camera

a. Install fswebcam

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

b. After installing fswebcam, you can use the following command to take pictures
   a) -d Option to specify the device node for the USB camera
   b) --no-banner Watermark for removing photos
   c) -r Option to specify the resolution of the photo
   d) -S Option to set to skip previous frames
   e) ./image.jpg used to set the name and path of the generated photo

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

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

```
orangepi@orangepi:~$ scp image.jpg test@192.168.1.55:/home/test （Modify the IP address and path according to the actual situation）
```

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

5) Use motion to test the USB camera

a. Install the camera test software motion

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

b. Modify the configuration of /etc/default/motion and change start_motion_daemon=no to start_motion_daemon=yes

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

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

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

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

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

```
orangepi@orangepi:~$ vim /etc/motion/motion.conf
```

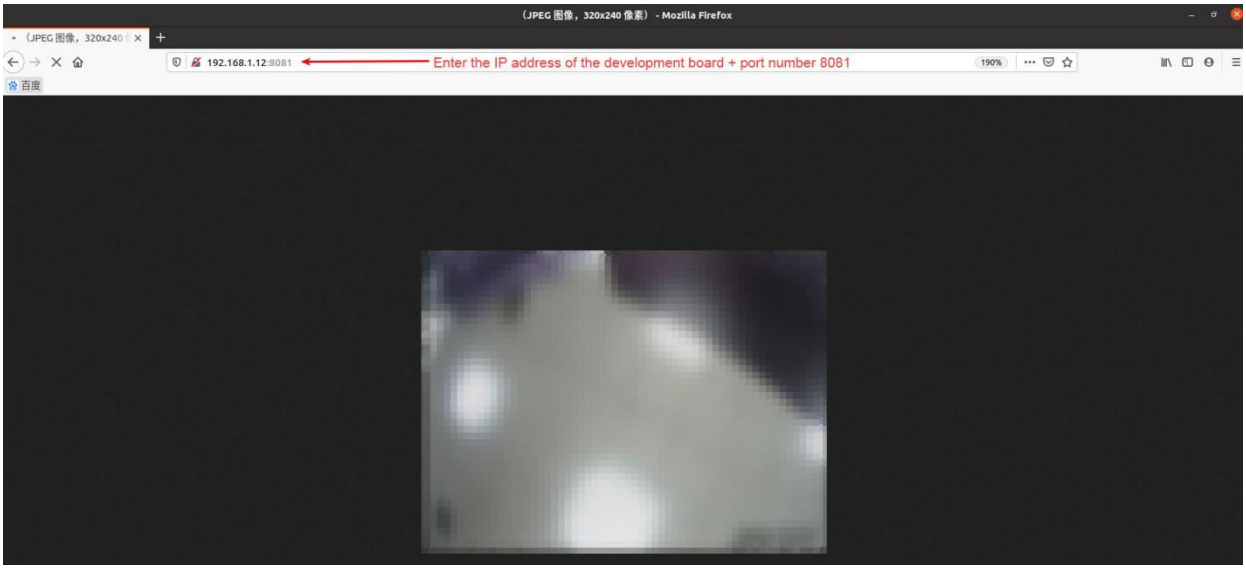
```
# Video device (e.g. /dev/video0) to be used for capturing.
videodevice /dev/video0
```

e. Then run motion

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

f. Before using motion, please make sure that the Orange Pi development board can connect to the network normally, and then obtain the IP address of the development board through the **ip addr show** command

g. Then enter **[IP address of the development board: 8081]** in the Ubuntu PC or Windows PC or the Firefox browser of the mobile phone on the same LAN as the development board to see the video output by the camera



6) Use mjpg-streamer to test the USB camera

a. Download mjpg-streamer

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

b. Install the dependent software file

a) Ubuntu system

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

b) Debian system

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

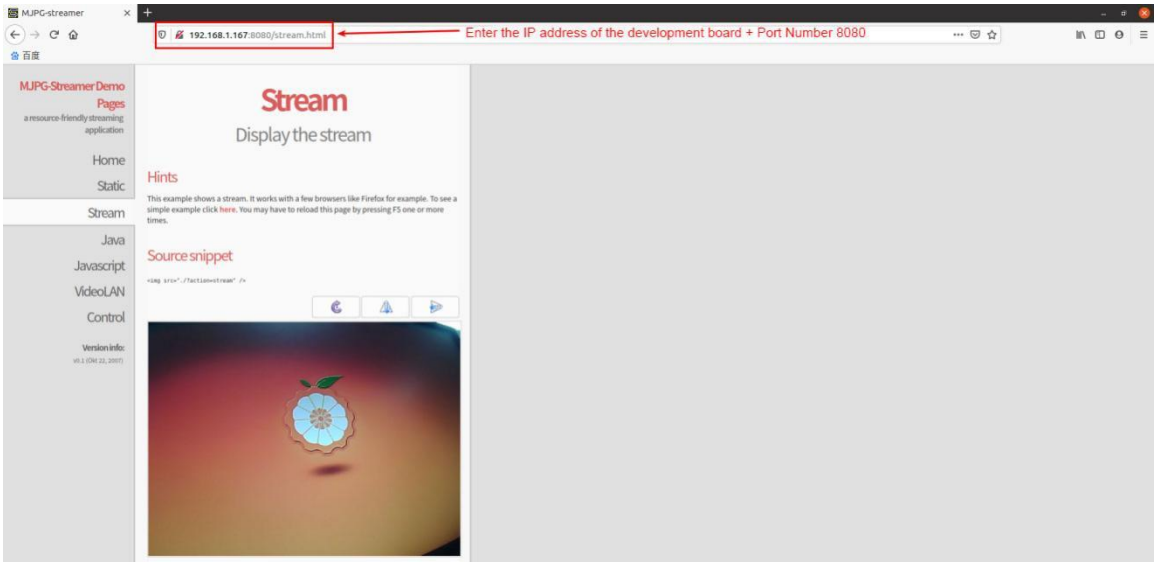
c. Compile and install mjpg-streamer

```
orangepi@orangepi:~$ cd mjpg-streamer/mjpg-streamer-experimental
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ make -j4
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ sudo make install
```

d. Then enter the following command to start mjpg_streamer

```
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ export \
LD_LIBRARY_PATH=.
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ ./mjpg_streamer \
 -i "./input_uvc.so -d /dev/video0 -u -f 30" -o "./output_http.so -w ./www"
```

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



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

3.13.7. Virtual USB network card test

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

1) First, you need to use the USB Type C cable to connect the development board to the USB interface of the computer. In this case, the development board is powered by the USB interface of the computer, **so it is necessary to ensure that the USB interface of the computer can provide enough power to drive Development board**, if there is a problem with the startup of the development board, you need to replace the USB interface or computer

2) The Linux system configures USB0 as **usb_device** mode by default. You can check the status of **otg_role** through the following command

```
orangepi@orangepi:~$ cat /sys/devices/platform/soc/usbc0/otg_role
```

```
usb_device
```

3) If otg_role is not set to usb_device mode, you can use the following command to open

```
orangepi@orangepi:~$ sudo cat /sys/devices/platform/soc/usbc0/usb_device
device_chose finished!
```

4) Then load the g_ether kernel module

```
orangepi@orangepi:~$ sudo modprobe g_ether
```

5) After the kernel module is loaded, the development board will have an additional network interface named usb0. First enable this network interface, and then set an IP address for it

```
orangepi@orangepi:~$ sudo ifconfig usb0 up
orangepi@orangepi:~$ sudo ifconfig usb0 192.168.10.10
orangepi@orangepi:~$ sudo ifconfig usb0
usb0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.10  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::180e:b0ff:fe5a:1ee4  prefixlen 64  scopeid 0x20<link>
        ether 1a:0e:b0:5a:1e:e4  txqueuelen 1000  (Ethernet)
        RX packets 47  bytes 8223 (8.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 47  bytes 8223 (8.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

6) Then go back to the Ubuntu PC, you can see the following log information through the dmesg command, from which you can know that the device name of the USB virtual network card is **enxaafd52849335**

```
test@test:~$ dmesg | tail
[33055.681514] usb 2-1.2: new high-speed USB device number 17 using ehci-pci
[33055.791512] usb 2-1.2: New USB device found, idVendor=0525, idProduct=a4a2,
bcdDevice= 4.09
[33055.791515] usb 2-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[33055.791516] usb 2-1.2: Product: RNDIS/Ethernet Gadget
[33055.791517] usb 2-1.2: Manufacturer: Linux 4.9.170-sun50iw9 with sunxi_usb_udc
[33055.792258] cdc_subset: probe of 2-1.2:1.0 failed with error -22
```

```
[33055.793063] cdc_ether 2-1.2:1.0 usb0: register 'cdc_ether' at usb-0000:00:1d.0-1.2, CDC Ethernet Device, aa:fd:52:84:93:35
[33055.862338] cdc_ether 2-1.2:1.0 enxaafd52849335: renamed from usb0
```

7) Then assign an IP address to the USB virtual network card in the Ubuntu PC. Please keep the IP address of the Ubuntu PC and the IP of the development board in the same network segment

```
test@test:~$ sudo ifconfig enxaafd52849335 192.168.10.12
test@test:~$ ifconfig enxaafd52849335
enxaafd52849335: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.12  netmask 255.255.255.0  broadcast 192.168.10.255
        ether aa:fd:52:84:93:35  txqueuelen 1000  (Ethernet)
        RX packets 56  bytes 8542 (8.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 113  bytes 21351 (21.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

8) Then you can test if the development board and the Ubuntu PC can communicate with each other

```
orangepi@orangepi:~$ sudo ping 192.168.10.12
PING 192.168.10.12 (192.168.10.12) 56(84) bytes of data.
64 bytes from 192.168.10.12: icmp_seq=1 ttl=64 time=0.376 ms
64 bytes from 192.168.10.12: icmp_seq=2 ttl=64 time=0.549 ms
64 bytes from 192.168.10.12: icmp_seq=3 ttl=64 time=0.460 ms
64 bytes from 192.168.10.12: icmp_seq=4 ttl=64 time=0.460 ms
64 bytes from 192.168.10.12: icmp_seq=5 ttl=64 time=0.493 ms
^C
--- 192.168.10.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4045ms
rtt min/avg/max/mdev = 0.376/0.467/0.549/0.056 ms
```

3.13.8. Virtual serial port test

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

1) First, you need to use the USB Type C cable to connect the development board to the

USB interface of the computer. In this case, the development board is powered by the USB interface of the computer, **so it is necessary to ensure that the USB interface of the computer can provide enough power to drive Development board,** if there is a problem with the startup of the development board, you need to replace the USB interface or computer

2) The Linux system configures USB0 as **usb_device** mode by default. You can check the status of **otg_role** through the following command

```
orangepi@orangepi:~$ cat /sys/devices/platform/soc/usbc0/otg_role
usb_device
```

3) If otg_role is not set to usb_device mode, you can use the following command to open

```
orangepi@orangepi:~$ sudo cat /sys/devices/platform/soc/usbc0/usb_device
device_chose finished!
```

4) Then load the g_serial kernel module

```
orangepi@orangepi:~$ sudo modprobe g_serial
```

5) After loading the kernel module, there will be an additional device node named ttyGS0 under /dev of the Linux system of the development board

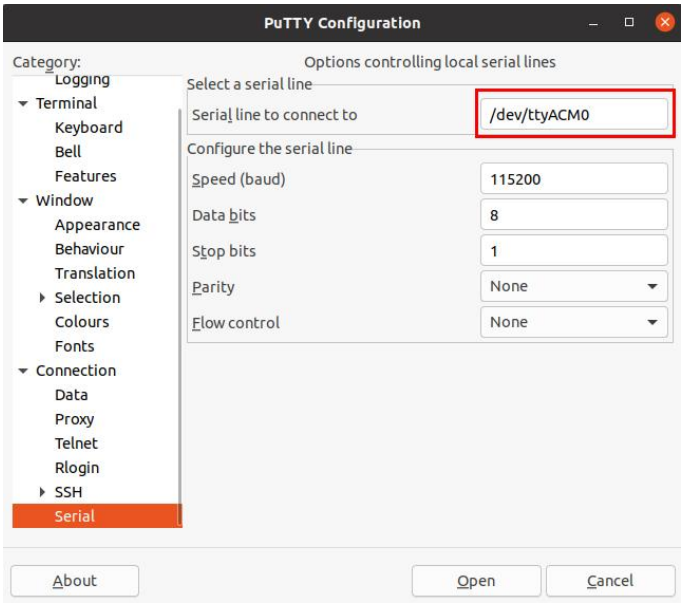
```
orangepi@orangepi:~$ ls /dev/ttyGS*
/dev/ttyGS0
```

6) Then go back to the Ubuntu PC, you can see that there will be a device node named ttyACM0 under /dev

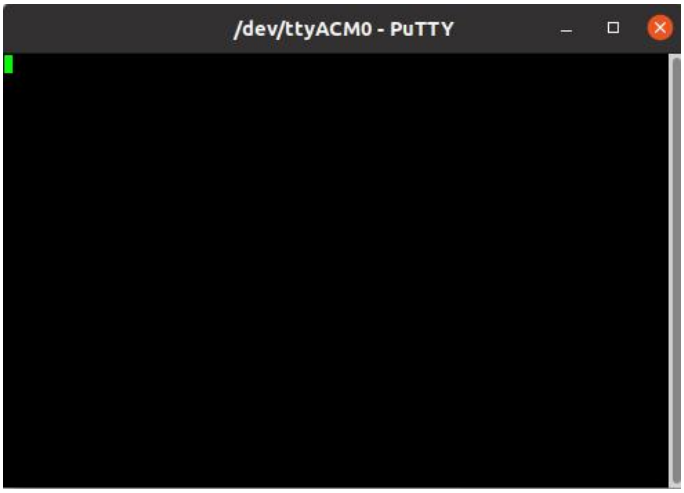
```
test@test:~$ ls /dev/ttyACM*
/dev/ttyACM0
```

7) Then open putty of Ubuntu PC and connect ttyACM0

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



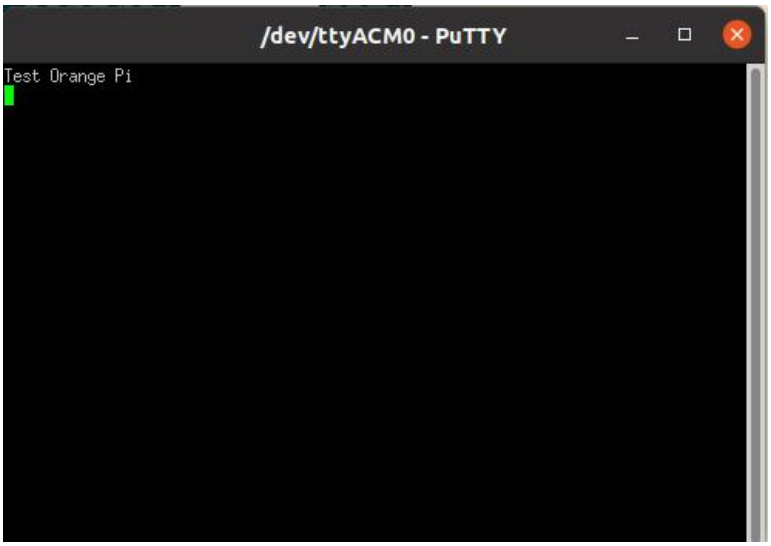
After Putty is connected to ttyACM0, the window shown in the figure below will open.



8) Then go back to the linux system of the development board and send a string of characters to /dev/ttyGS0

```
orangepi@orangepi:~$ sudo echo "Test Orange Pi" > /dev/ttyGS0
```

9) If all goes well, the Putty of the Ubuntu PC will receive the string sent from the board

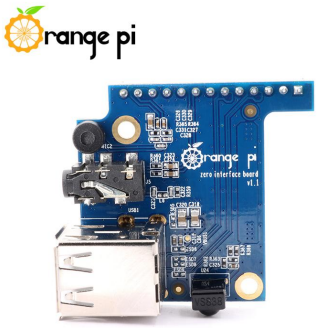


3.14. Audio Test

3.14.1. The method to use the command line to play audio

3.14.1.1. Headphone jack audio playback test

1) First, you need to insert the 13pin expansion board into the 13pin interface of the Orange Pi development board, and then insert the earphone into the audio interface



2) You can view the sound card devices supported by the Linux system through the **aplay -l** command
   a. The output of a.linux4.9 system is as follows, where **card 0: audiocodec** is the sound card device required for headset playback

```
root@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: audiocodec [audiocodec], device 0: SUNXI-CODEC sun50iw9-codec-0 []
```

```
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

b. The output of the b.linux5.16 system is as follows, in which the **H616 Audio Codec** is the sound card device required for headset playback

```
root@orangepi:~# aplay -l
card 2: Codec [H616 Audio Codec], device 0: CDC PCM Codec-0 [CDC PCM Codec-0]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

3) Then use the **aplay** command to play the audio, and the headset can hear the sound

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

```
root@orangepi:~# aplay -D hw:0,0 /usr/share/sounds/alsa/audio.wav
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

b. The playback command of Linux5.16 is as follows

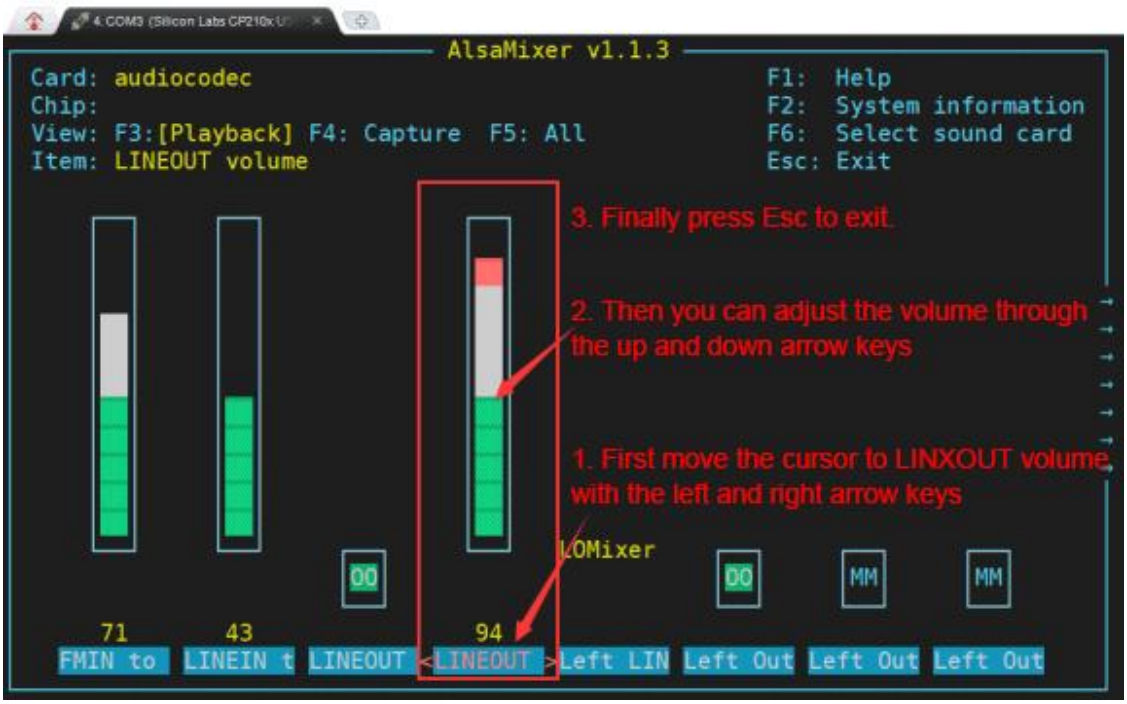
```
root@orangepi:~# aplay -D hw:2,0 /usr/share/sounds/alsa/audio.wav
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

4) The volume of the headset can be adjusted through the **alsamixer** command

a. Enter the **alsamixer** command in the terminal

```
root@orangepi:~# alsamixer
```

b. After entering the alsamixer command, the following audio setting interface will pop up, and then you can adjust the size of the audio through the up, down, left, and right direction keys. For specific steps, see the description in the figure below.



3.14.1.2. HDMI audio playback test

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

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

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

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

b. The playback command of the Linux5.16 system is as follows

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

After the Linux 5.16 system is burned for the first time, if there is no sound in the HDMI test, you can use the following command to initialize the HDMI audio device, and then use the above command to test:

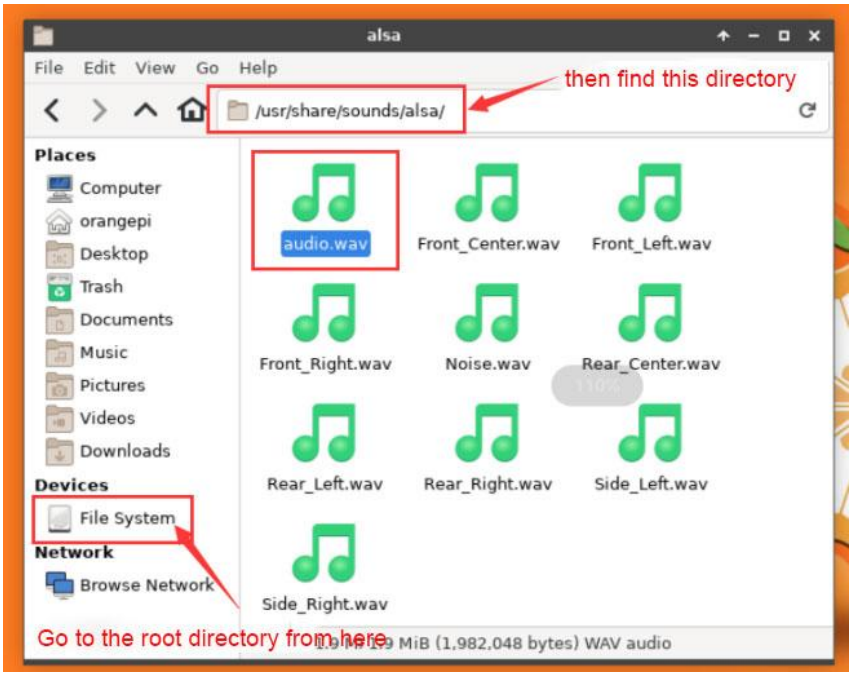
root@orangepi:~# **aplay /usr/share/sounds/alsa/audio.wav -D hw:1,0 >/dev/null 2>&1**

Or restart the system and test again.

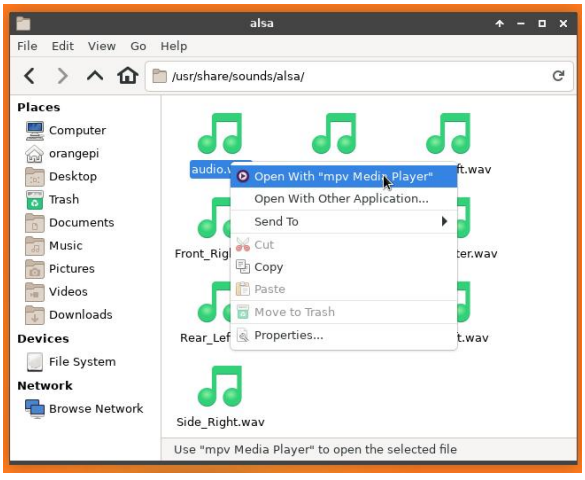
3.14.2. Test the audio method in the desktop system

1) First open the file manager

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

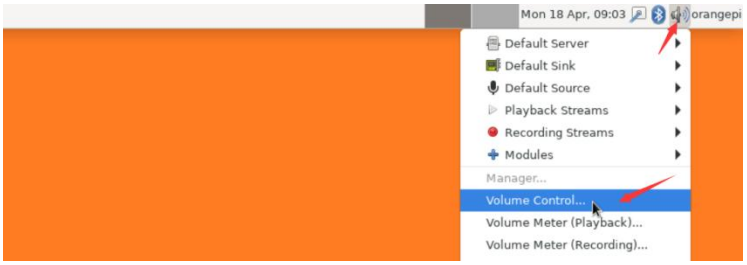


3) Then select the audio.wav file, right-click and select Open with mpv to start playing

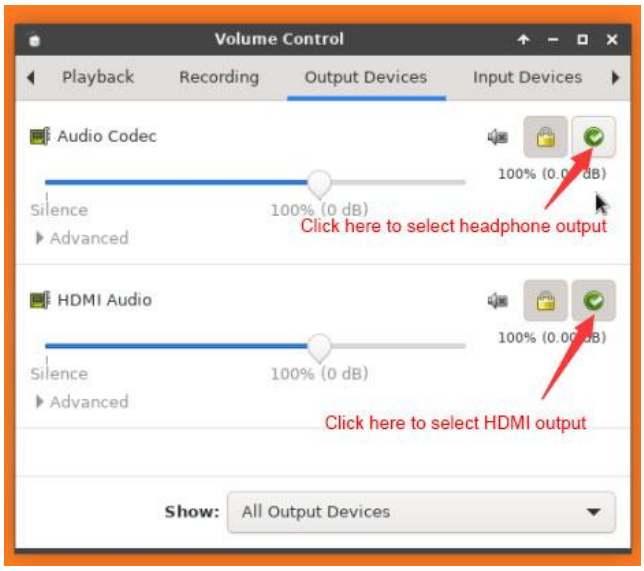


4) The Way to switch between HDMI playback and headphone playback

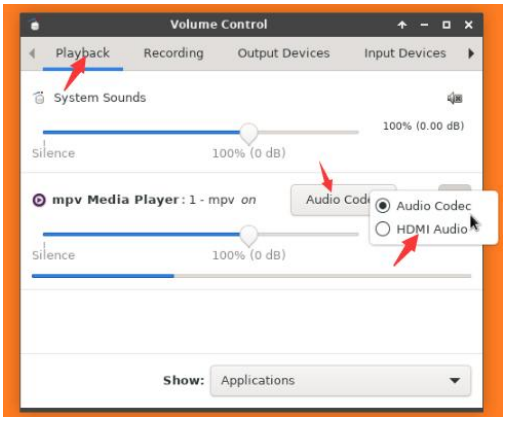
a. First open the volume control interface



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

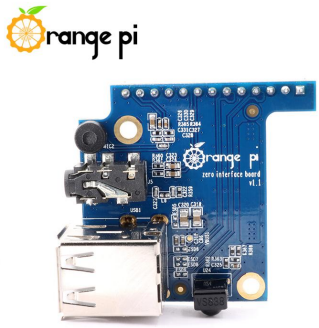


c. In addition, when playing audio, the audio setting options of the playback software will be displayed in **Playback**, as shown in the figure below, you can also set which audio device to play to here.



3.15. Infrared receiving test

1) First, you need to insert the 13pin expansion board into the 13pin interface of the Orange Pi development board. After inserting the expansion board, the Orange Pi Zero 2 can use the infrared receiving function



2) Install ir-keytable infrared test software

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

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

a. linux4.9 The system output looks like this

```
orangepi@orangepi:~$ ir-keytable
Found /sys/class/rc/rc0/ (/dev/input/event1) with:
        Driver: sunxi-rc-recv, table: rc_map_sunxi
        lirc device: /dev/lirc0
        Supported protocols: lirc nec
        Enabled protocols: lirc nec
        Name: sunxi_ir_recv
        bus: 25, vendor/product: 0001:0001, version: 0x0100
        Repeat delay = 500 ms, repeat period = 125 ms
```

b. linux5.16 The output of the system is as follows

```
orangepi@orangepi:~$ ir-keytable
Found /sys/class/rc/rc0/ with:
        Name: sunxi-ir
        Driver: sunxi-ir
        Default keymap: rc-empty
        Input device: /dev/input/event4
        LIRC device: /dev/lirc0
```

```
        Attached BPF protocols: Operation not supported
        Supported kernel protocols: lirc rc-5 rc-5-sz jvc sony nec sanyo mce_kbd rc-6 sharp xmp imon rc-mm
        Enabled kernel protocols: lirc
        bus: 25, vendor/product: 0001:0001, version: 0x0100
        Repeat delay = 500 ms, repeat period = 125 ms
```

4) Before testing the infrared receiving function, you need to prepare an infrared remote control dedicated to Orange Pi,**other remote controls do not support**

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

a. linux4.9 The system output is as follows

```
root@orangepi:~# ir-keytable -t
Testing events. Please, press CTRL-C to abort.
1598339152.260376: event type EV_MSC(0x04): scancode = 0xfb0413
1598339152.260376: event type EV_SYN(0x00).
1598339152.914715: event type EV_MSC(0x04): scancode = 0xfb0410
```

b. linux5.16 The system output looks like this

```
root@orangepi:~# ir-keytable -c -p NEC -t
Old keytable cleared
Protocols changed to nec
Testing events. Please, press CTRL-C to abort.
202.063219: lirc protocol(nec): scancode = 0x45c
202.063249: event type EV_MSC(0x04): scancode = 0x45c
```

```
202.063249: event type EV_SYN(0x00).
```

3.16. Temperature sensor

1) H616 has a total of 4 temperature sensors, the command to view the temperature is as follows：

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

a. sensor0：The temperature sensor of the CPU, the first command is used to view the type of the temperature sensor, and the second command is used to view the value of the temperature sensor

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

b. sensor1：The temperature sensor of the GPU, the first command is used to view the type of the temperature sensor, and the second command is used to view the value of the temperature sensor

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

c. sensor2：VE's temperature sensor, the first command is used to view the type of the temperature sensor, and the second command is used to view the value of the temperature sensor

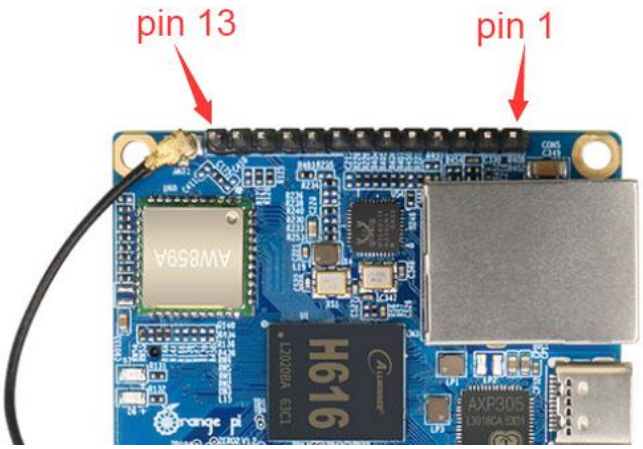
```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone2/type
ve_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone2/temp
59273
```

d. sensor3：DDR temperature sensor, the first command is used to view the type of the temperature sensor, and the second command is used to view the value of the temperature sensor

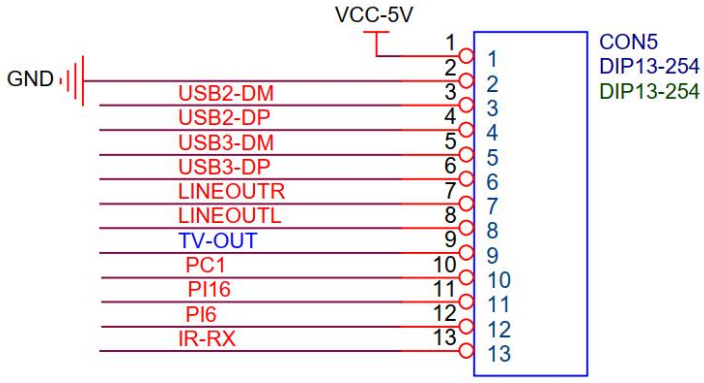
```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone3/type
ddr_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone3/temp
58949
```

3.17. 13 Pin expansion board interface pin description

1) Orange Pi Zero 2 Please refer to the following figure for the sequence of the 13 pin expansion board interface pins of the development board



2) Orange Pi Zero 2 The schematic diagram of the 13pin interface of the development board is as follows



3) The function description of the Orange Pi Zero 2 13pin expansion board interface pins of the development board is as follows

a. When the 13 pin is connected to the expansion board, it can additionally provide
   a) 2* USB 2.0 Hosts
   b) Headphone left and right channel audio output
   c) TV-OUT video output
   d) Infrared receiving function
   e) After the expansion board is connected, pins 10, 11 and 12 of the 13pin interface cannot be used.

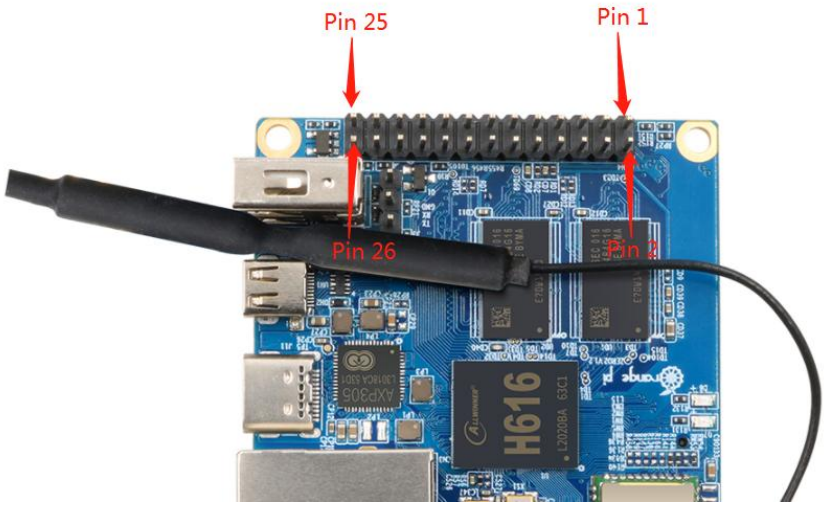
f) **Also note that the MIC on the 13pin expansion board cannot be used on the Orange Pi Zero 2**

b. When the 13pin pin is not connected to the expansion board, pins 10, 11, 12 and 13 can be used as ordinary GPIO ports

GPIO serial number	Features	pin
	5V	1
	GND	2
	USB2-DM	3
	USB2-DP	4
	USB3-DM	5
	USB3-DP	6
	LINEOUTR	7
	LINEOUTL	8
	TV-OUT	9
65	PC1	10
272	PI16	11
262	PI6	12
234	IR-RX/PH10	13

3.18. 26 Pin Interface Pin Description

1) Please refer to the figure below for the order of the 26 pin interface pins of the Orange Pi Zero 2 development board



2) The functions of the 26 pin interface pins of the Orange Pi Zero 2 development board

are shown in the table below

GPIO NO.	GPIO	Feature	Pin	Pin	Feature	GPIO	GPIO NO.
		3.3V	1	2	5V		
229	PH5	TWI3-SDA	3	4	5V		
228	PH4	TWI3-SCK	5	6	GND		
73	PC9	PC9	7	8	UART5_TX	PH2	226
		GND	9	10	UART5_RX	PH3	227
70	PC6	PC6	11	12	PC11	PC11	75
69	PC5	PC5	13	14	GND		
72	PC8	PC8	15	16	PC15	PC15	79
		3.3V	17	18	PC14	PC14	78
231	PH7	SPI1_MOSI	19	20	GND		
232	PH8	SPI1_MISO	21	22	PC7	PC7	71
230	PH6	SPI1_CLK	23	24	SPI1_CS	PH9	233
		GND	25	26	PC10	PC10	74

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

3.19. The Way to install wiringOP

1) Download the code of wiringOP

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

2) Compile and install wiringOP

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

3) The output of the test gpio readall command is as follows

a. The pins 1 to 26 correspond one-to-one with the 26 Pins on the development board

b. Pin 27 corresponds to pin 10 of 13pin on the development board

c. Pin 29 corresponds to pin 11 of 13pin on the development board

d. The 31st pin corresponds to the 12th pin of the 13pin on the development board

e. Pin 33 corresponds to pin 13 of 13pin on the development board

f. Pins 28, 30, 32, and 34 are empty, please ignore them directly

```
root@orangepizero2:~# gpio readall
 +------+-----+----------+------+---+  Zero 2  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---++----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  229 |   0 |    SDA.3 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  228 |   1 |    SCL.3 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |   73 |   2 |      PC9 |  OUT | 0 |  7 || 8  | 0 | OUT  | TXD.5    | 3   | 226  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OUT  | RXD.5    | 4   | 227  |
 |   70 |   5 |      PC6 |  OUT | 0 | 11 || 12 | 0 | OUT  | PC11     | 6   | 75   |
 |   69 |   7 |      PC5 |  OUT | 0 | 13 || 14 |   |      | GND      |     |      |
 |   72 |   8 |      PC8 |  OUT | 1 | 15 || 16 | 0 | OUT  | PC15     | 9   | 79   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | OUT  | PC14     | 10  | 78   |
 |  231 |  11 |   MOSI.1 |  OUT | 0 | 19 || 20 |   |      | GND      |     |      |
 |  232 |  12 |   MISO.1 |  OUT | 0 | 21 || 22 | 0 | OUT  | PC7      | 13  | 71   |
 |  230 |  14 |   SCLK.1 |  OUT | 0 | 23 || 24 | 0 | OUT  | CE.1     | 15  | 233  |
 |      |     |      GND |      |   | 25 || 26 | 0 | OUT  | PC10     | 16  | 74   |
 |   65 |  17 |      PC1 |  OUT | 0 | 27 || 28 |   |      |          |     |      |
 |  272 |  18 |     PI16 |  OUT | 0 | 29 || 30 |   |      |          |     |      |
 |  262 |  19 |      PI6 |  OUT | 0 | 31 || 32 |   |      |          |     |      |
 |  234 |  20 |     PH10 |  OUT | 0 | 33 || 34 |   |      |          |     |      |
 +------+-----+----------+------+---++----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+  Zero 2  +---+------+----------+-----+------+
```

3.20. 26pin interface GPIO, I2C, UART, SPI and PWM test

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

wiringOP。Before starting the test, make sure you have compiled and installed wiringOP by referring to the section Installing wiringOP.

3.20.1. 26pin GPIO port test

1) The following is an example of how to set the high and low levels of the GPIO port by taking pin No. 7—the corresponding GPIO is PC9—the corresponding wPi serial number is 2

```
root@orangepizero2:~/wiringOP# gpio readall
 +------+-----+----------+------+---+  Zero 2  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---++----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  229 |   0 |    SDA.3 |  OUT | 1 |  3 || 4  |   |      | 5V       |     |      |
 |  228 |   1 |    SCL.3 |  OUT | 1 |  5 || 6  |   |      | GND      |     |      |
 |   73 |   2 |      PC9 |  OUT | 1 |  7 || 8  | 1 | OUT  | TXD.5    | 3   | 226  |
 |      |     |      GND |      |   |  9 || 10 | 1 | OUT  | RXD.5    | 4   | 227  |
 |   70 |   5 |      PC6 |  OUT | 1 | 11 || 12 | 1 | OUT  | PC11     | 6   | 75   |
```

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

```
root@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 successfully set.

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

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

```
root@orangepizero2:~# gpio readall
 +------+-----+----------+------+---+  Zero 2  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  229 |   0 |    SDA.3 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  228 |   1 |    SCL.3 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |   73 |   2 |      PC9 |  OUT | 0 |  7 || 8  | 0 | ALT2 | TXD.5    | 3   | 226  |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT2 | RXD.5    | 4   | 227  |
 |   70 |   5 |      PC6 | ALT5 | 0 | 11 || 12 | 0 | OFF  | PC11     | 6   | 75   |
```

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

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

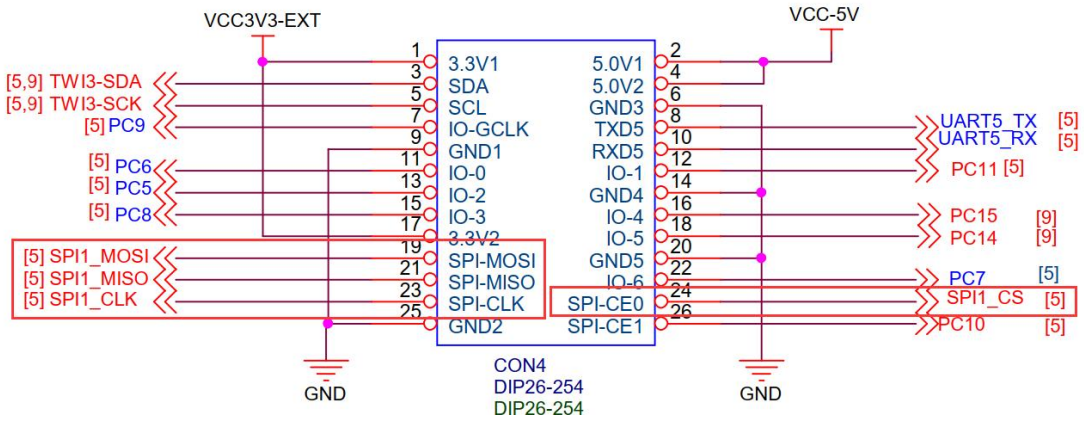
Use gpio readall to see that the value of pin 7 (V) has changed to 1

```
root@orangepizero2:~# gpio readall
 +------+-----+----------+------+---+  Zero 2  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  229 |   0 |    SDA.3 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  228 |   1 |    SCL.3 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |   73 |   2 |      PC9 |  OUT | 1 |  7 || 8  | 0 | ALT2 | TXD.5    | 3   | 226  |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT2 | RXD.5    | 4   | 227  |
 |   70 |   5 |      PC6 | ALT5 | 0 | 11 || 12 | 0 | OFF  | PC11     | 6   | 75   |
```

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

3.20.2. 26pin SPI test

1) According to the schematic diagram of the 26pin interface, the spi available for Orange Pi Zero 2 is spi1



If you are using a Linux 5.16 kernel system, spi1 is turned off by default and needs to be manually turned on to use it.

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

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=spi-spidev
param_spidev_spi_bus=1
param_spidev_spi_cs=1

2) First check whether there is a device node of **spidev1.1** in the Linux system. If it exists, it means that SPI1 has been set and can be used directly

```
root@orangepi:~/wiringOP/examples# ls /dev/spidev1*
/dev/spidev1.1
```

3) Compile the spidev_test test program in the examples of wiringOP

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

4) Do not short the mosi and miso pins of SPI1 first. The output result of running

spidev_test is as follows. You can see that the data of TX and RX are inconsistent

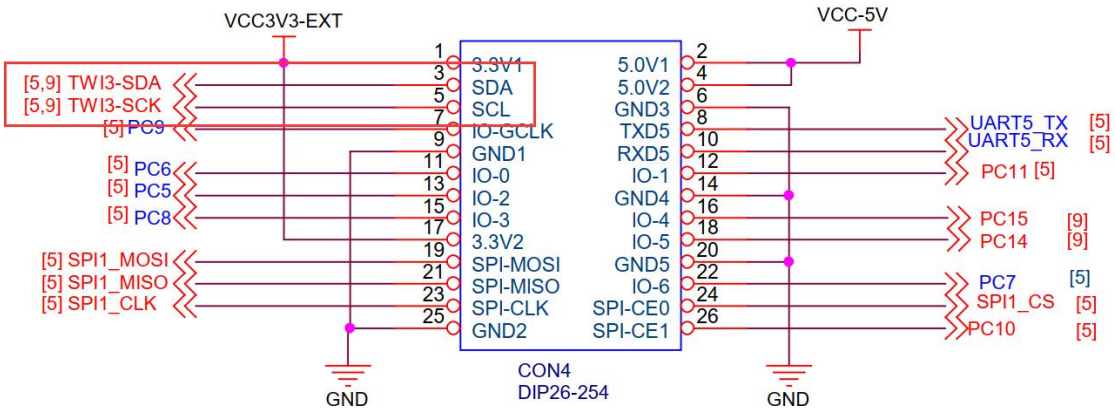
```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.1
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.....................▒.
RX | FF FF FF FF FF FF FF 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    | ................................
```

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

```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.1
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    | ......@....▒.....................▒.
```

3.20.3. 26pin I2C test

1) According to the schematic diagram of 26pin, the i2c available for Orange Pi Zero 2 is i2c3



If you are using a Linux 5.16 kernel system, i2c3 is disabled by default and needs to be manually opened before it can be used.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then restart the Linux system to open i2c3.

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=i2c3

2) After starting the linux system, first confirm that there is an i2c3 device node under /dev

```
root@orangepi:~# ls /dev/i2c-*
/dev/i2c-3    /dev/i2c-5
```

3) Then start testing i2c, first install i2c-tools

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y i2c-tools
```

4) Then connect an i2c device to the i2c3 pin of the 26pin header

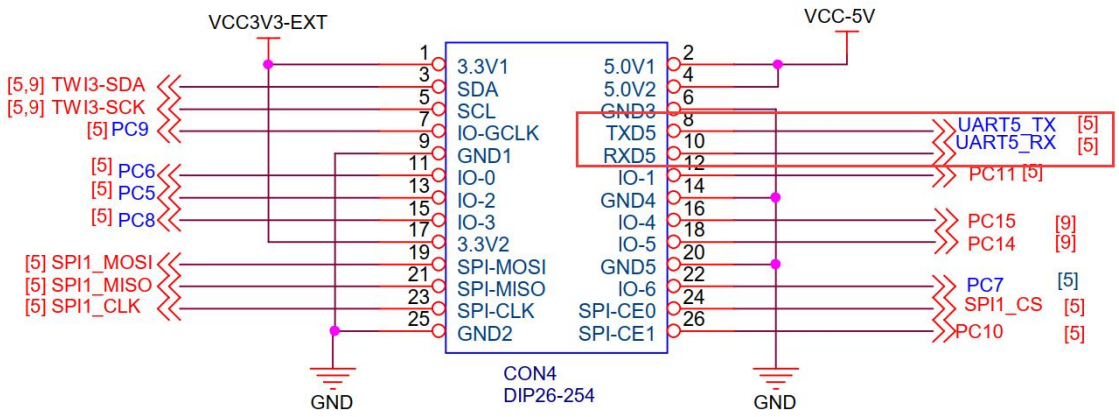
	i2c3
sda pin	To 3 pin
sck pin	To 5 pin
vcc pin	To 1 pin
gnd pin	To 6 pin

5) Then use the **i2cdetect -y 3** command if the address of the connected i2c device can be detected, it means that i2c can be used normally

```
root@orangepi:~# i2cdetect -y 3
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

3.20.4. 26pin UART test

1) From the schematic diagram of the 26pin interface, the uart available for Orange Pi Zero 2 is uart5



If you are using a Linux 5.16 kernel system, uart5 is disabled by default and needs to be manually opened before it can be used.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then restart the Linux system to open uart5.

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=uart5

2) After entering the linux system, first confirm whether there is a uart5 device node under **/dev**

```
root@orangepi:~# ls /dev/ttyS5
```

```
/dev/ttyS5
```

3) Then start to test the uart5 interface, first use the DuPont line to short-circuit the rx and tx of the uart5 interface to be tested

	uart5
tx pin	To 8 pin
rx pin	To 10 pin

4) Use the **gpio** command in wiringOP to test the loopback function of the serial port as shown below. If you can see the following print, it means that the serial port communication is normal

```
orangepi@orangepi:~$ gpio serial /dev/ttyS5

Out:    0:  ->  0
Out:    1:  ->  1
Out:    2:  ->  2
Out:    3:  ->  3^C
```

5) You can also use the serialTest.c program in wiringOP to test the loopback function of the serial port. The specific steps are as follows：

a. First modify the serial device node name opened by the serial test program serialTest in wiringOP to **/dev/ttyS5**

```
root@orangepi:~/wiringOP/examples# vim serialTest.c
```

```
int main ()
{
  int fd ;
  int count ;
  unsigned int nextTime ;

  if ((fd = serialOpen ("/dev/ttyS5", 115200)) < 0)
  {
    fprintf (stderr, "Unable to open serial device: %s\n", strerror (errno)) ;
    return 1 ;
  }
```

b. Then compile the serial test program serialTest in wiringOP

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

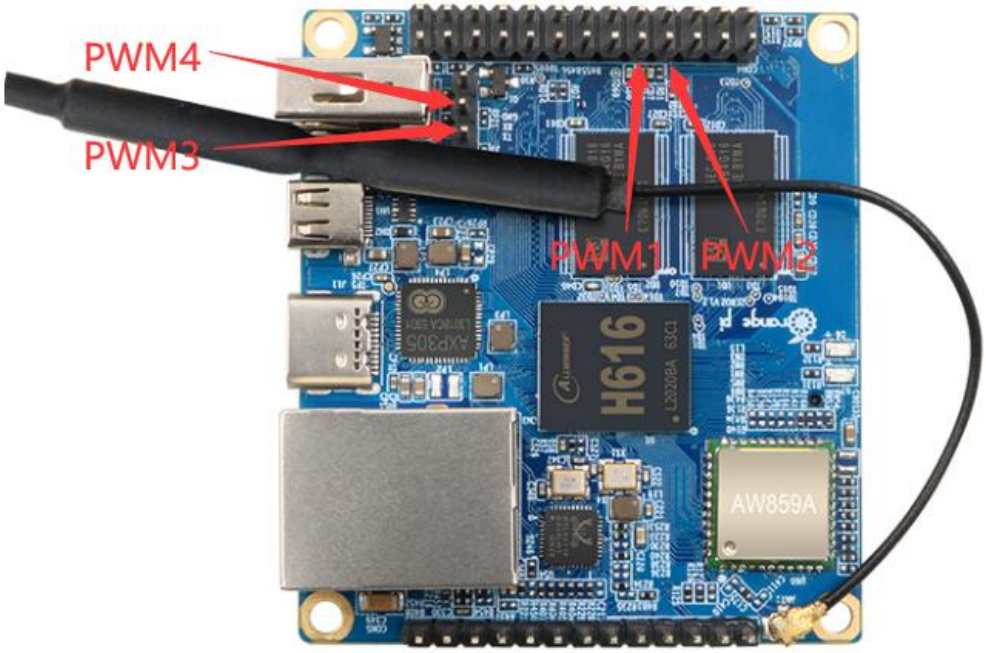
```
root@orangepi:~/wiringOP/examples#
```

c. Finally run serialTest, if you can see the following print, it means the serial communication is normal

```
root@orangepi:~/wiringOP/examples# ./serialTest

Out:    0:  ->    0
Out:    1:  ->    1
Out:    2:  ->    2
Out:    3:  ->    3^C
```

3.20.5. PWM test method

The Orange Pi Zero 2 can use up to 4 channels of PWM, and their pins are located as shown in the picture below:



The RX and TX pins of UART5 in the PWM1, PWM2 and 26pin interfaces are multiplexed, so before using PWM1 and PWM2, you need to turn off the configuration of UART5 in dts.

PWM3, PWM4 and the TX and RX pins in the debugging serial port are multiplexed, so before using PWM3 and PWM4, you need to turn off the configuration of UART0 in dts, and the debugging serial port cannot be used at this time.

3.20.5.1. Linux5.16 system PWM test method

1) PWM1 and PWM2 test methods are as follows

a. First, you need to open the configuration of PWM12, please add the configuration in the red font part below to **/boot/orangepiEnv.txt**

Note that if pwm12 is turned on, uart5 cannot be turned on at the same time, you can only choose one of the two.

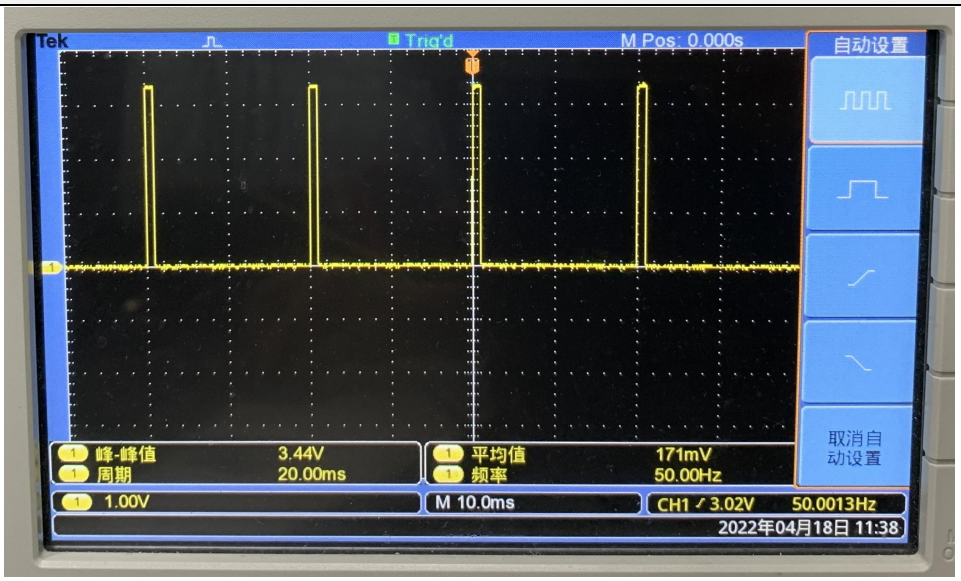
```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=pwm12
```

b. Then restart the Linux system. If you can see the following printing information in the log output from the serial port, it means that pwm12 is successfully opened.

```
Applying kernel provided DT overlay sun50i-h616-pwm12.dtbo
4203 bytes read in 5 ms (820.3 KiB/s)
Applying kernel provided DT fixup script (sun50i-h616-fixup.scr)
```

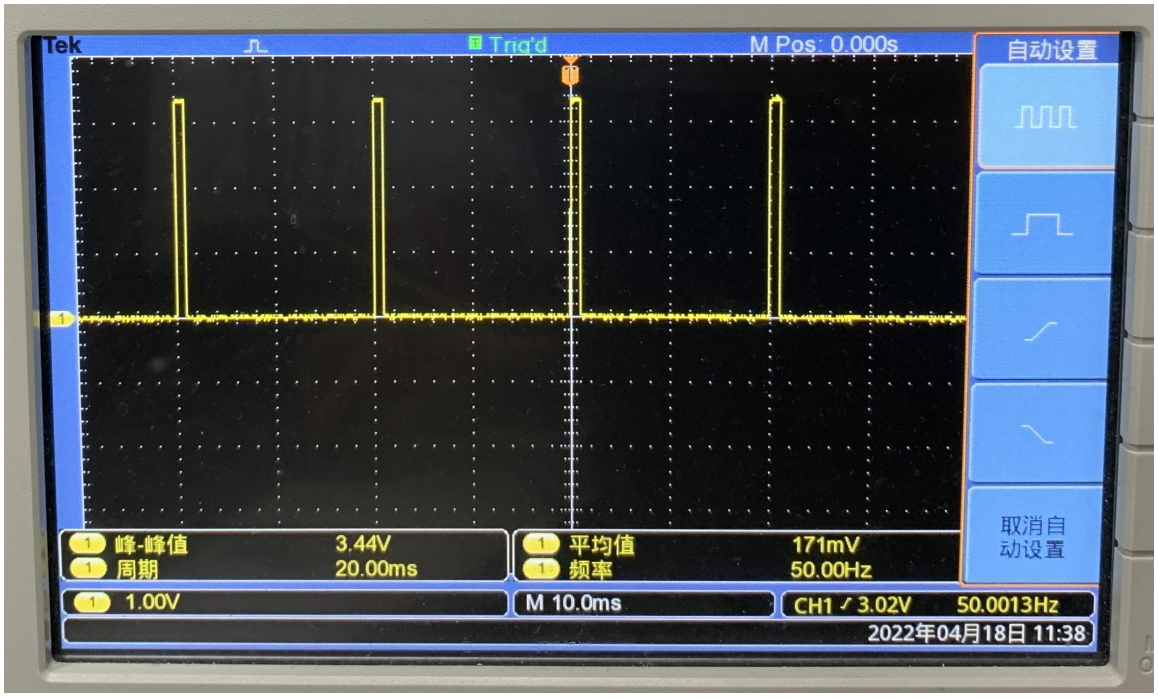
c. Then you can start the PWM test, enter the following command in the system to make pwm1 output a 50Hz rectangular wave

```
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm1/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm1/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm1/enable
```



d. Enter the following command in the system to make pwm2 output a 50Hz square wave

```
root@orangepi:~# echo 2 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm2/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm2/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm2/enable
```



2) PWM3 and PWM4 test methods are as follows

a. First, you need to open the configuration of PWM34, please add the configuration in the red font part below to **/boot/orangepiEnv.txt**

Note that if pwm34 is turned on, uart0 will be turned off at the same time, and the debugging serial port will not be available at this time.

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=pwm34
```

b. Then restart the Linux system. If you can see the following print information in the log output from the serial port, it means that pwm34 is successfully opened.

```
Applying kernel provided DT overlay sun50i-h616-pwm34.dtbo
4203 bytes read in 5 ms (820.3 KiB/s)
Applying kernel provided DT fixup script (sun50i-h616-fixup.scr)
## Executing script at 45000000
Warning: Disabling ttyS0 console due to enabled PWM3 and PWM4 overlay
10735700 bytes read in 446 ms (23 MiB/s)
```

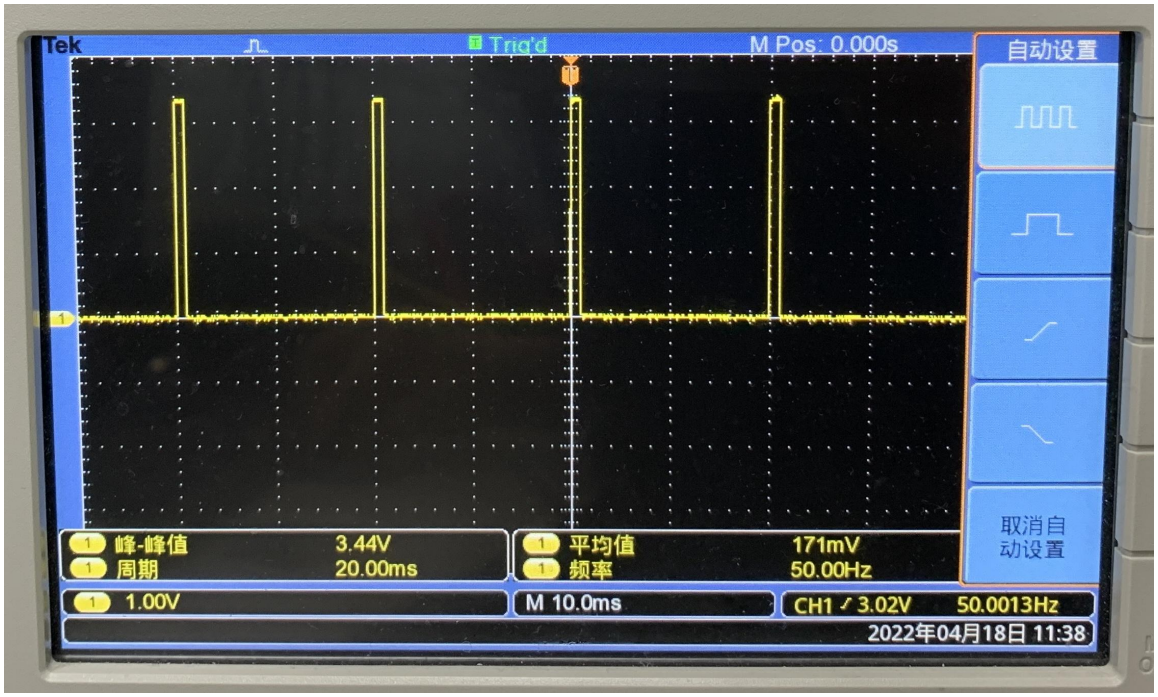
```
22259720 bytes read in 922 ms (23 MiB/s)
```

c. Note that the debugging serial port will be stuck in the following place at this time, and there will be no more output or input.

```
## Flattened Device Tree blob at 4fa00000
   Booting using the fdt blob at 0x4fa00000
   Loading Ramdisk to 495c2000, end 49fff014 ... OK
   Loading Device Tree to 0000000049550000, end 00000000495c1fff ... OK

Starting kernel ...
```

d. Then you can start the PWM test, enter the following command in the system to make pwm3 output a 50Hz rectangular wave

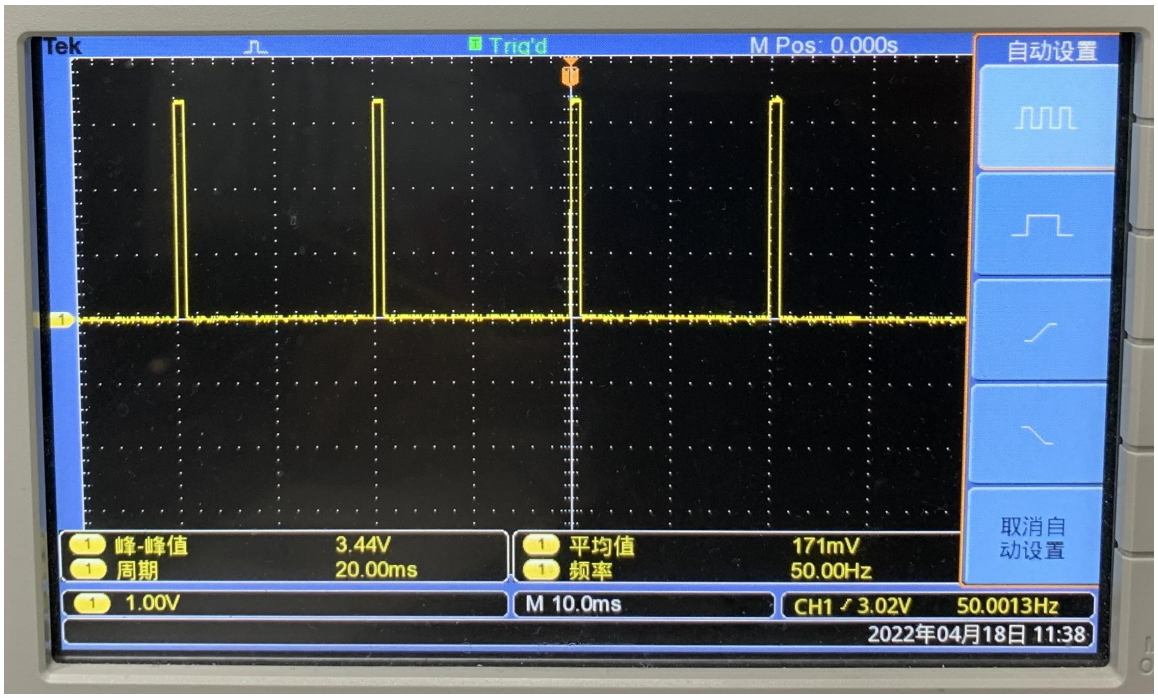
```
root@orangepi:~# echo 3 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm3/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm3/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm3/enable
```



e. Enter the following command in the system to make pwm4 output a 50Hz square wave

```
root@orangepi:~# echo 4 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm4/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm4/duty_cycle
```

```
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm4/enable
```



3) The method to turn on PWM12 and PWM34 at the same time is as follows:

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=pwm12 pwm34
```

3.20.5.2. Linux4.9 system PWM test method

1) PWM1 and PWM2 test methods are as follows

a. First, you need to turn off the configuration of UART5 in dts. A script named **orangepi-add-overlay** is pre-installed in the linux system. Through this script, we can use DT overlay to dynamically modify the configuration in dts. First write the uart5_disable.dts file, the content is as follows

```
orangepi@orangepi:~$ vim uart5_disable.dts
/dts-v1/;
/plugin/;

/ {
        compatible = "allwinner,h616", "arm,sun50iw9p1";
```

```
        fragment@0 {
                target = <&uart5>;
                __overlay__ {
                        status = "disabled";
                };
        };
};
```

b. Then you can use **orangepi-add-overlay** to compile uart5_disable.dts into uart5_disable.dtbo

```
orangepi@orangepi:~$ sudo orangepi-add-overlay uart5_disable.dts
Compiling the overlay
Copying the compiled overlay file to /boot/overlay-user/
Reboot is required to apply the changes
```

c. Then restart the Linux system, you can see the following printing information in the log output from the serial port, indicating that uart5_disable.dtbo is loaded successfully

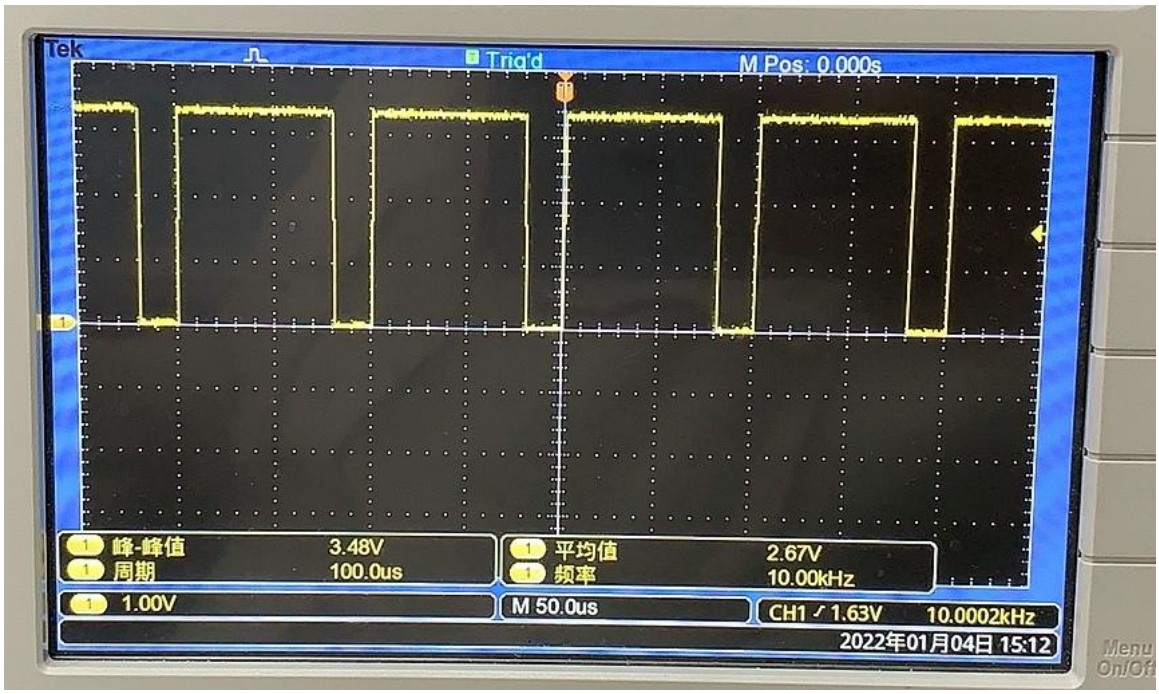
```
216 bytes read in 7 ms (29.3 KiB/s)
283 bytes read in 11 ms (24.4 KiB/s)
Applying user provided DT overlay uart5_disable.dtbo
8472451 bytes read in 367 ms (22 MiB/s)
24125512 bytes read in 1020 ms (22.6 MiB/s)
```

d. After entering the Linux system, the device node ttyS5 cannot be seen under **/dev.**

```
orangepi@orangepi:~$ ls /dev/ttyS*
/dev/ttyS0    /dev/ttyS1
```

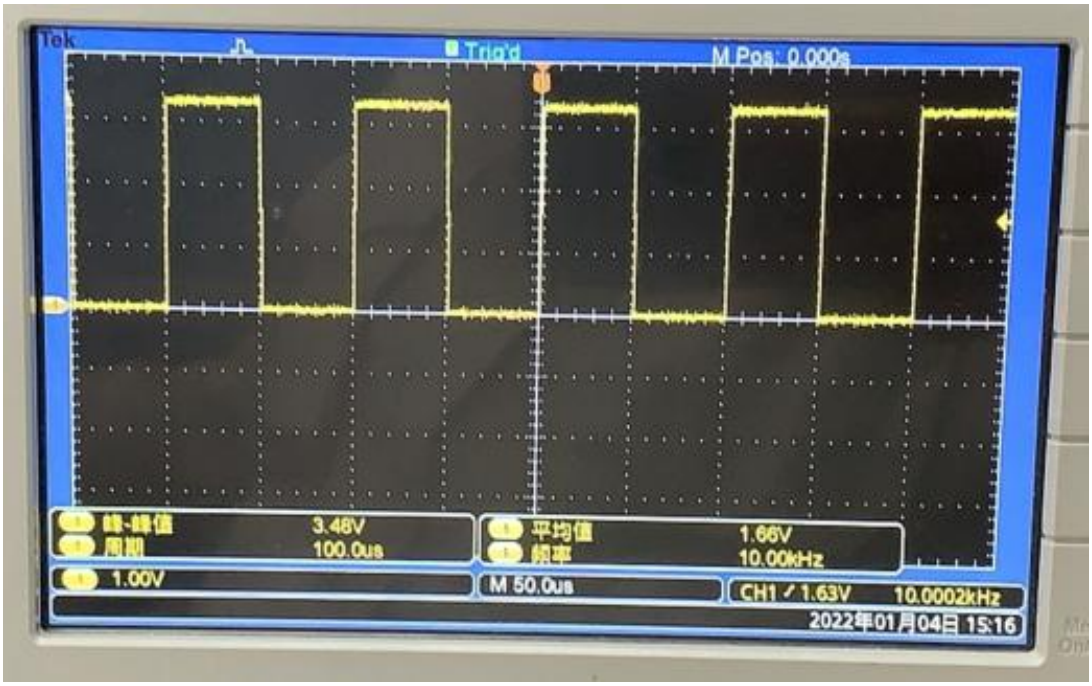
e. Then you can start the PWM test, enter the following command in the Linux4.9 system to make pwm1 output a 10kHz rectangular wave

```
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 100000 > /sys/class/pwm/pwmchip0/pwm1/period
root@orangepi:~# echo 20000 > /sys/class/pwm/pwmchip0/pwm1/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm1/enable
```



f. Enter the following command in the Linux4.9 system to make pwm2 output a 10kHz square wave

```
root@orangepi:~# echo 2 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 100000 > /sys/class/pwm/pwmchip0/pwm2/period
root@orangepi:~# echo 50000 > /sys/class/pwm/pwmchip0/pwm2/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm2/enable
```



2) PWM3 and PWM4 test methods are as follows

a. First, you need to turn off the configuration of UART0 in dts. A script named **orangepi-add-overlay** is pre-installed in the linux system. Through this script, we can use DT overlay to dynamically modify the configuration in dts. First write the uart0_disable.dts file, the content is as follows

```
orangepi@orangepi:~$ vim uart0_disable.dts
/dts-v1/;
/plugin/;

/ {
        compatible = "allwinner,h616", "arm,sun50iw9p1";

        fragment@0 {
                target = <&uart0>;
                __overlay__ {
                        status = "disabled";
                };
        };
};
```

b. Then you can use **orangepi-add-overlay** to compile uart0_disable.dts into uart0_disable.dtbo

```
orangepi@orangepi:~$ sudo orangepi-add-overlay uart0_disable.dts
Compiling the overlay
Copying the compiled overlay file to /boot/overlay-user/
Reboot is required to apply the changes
```

c. Then restart the Linux system, you can see the following printing information in the log output from the serial port, indicating that uart0_disable.dtbo is loaded successfully

```
216 bytes read in 7 ms (29.3 KiB/s)
283 bytes read in 12 ms (22.5 KiB/s)
Applying user provided DT overlay uart0_disable.dtbo
8472451 bytes read in 367 ms (22 MiB/s)
24125512 bytes read in 1020 ms (22.6 MiB/s)
```

d. After entering the Linux system, the device node ttyS0 cannot be seen under

/dev.

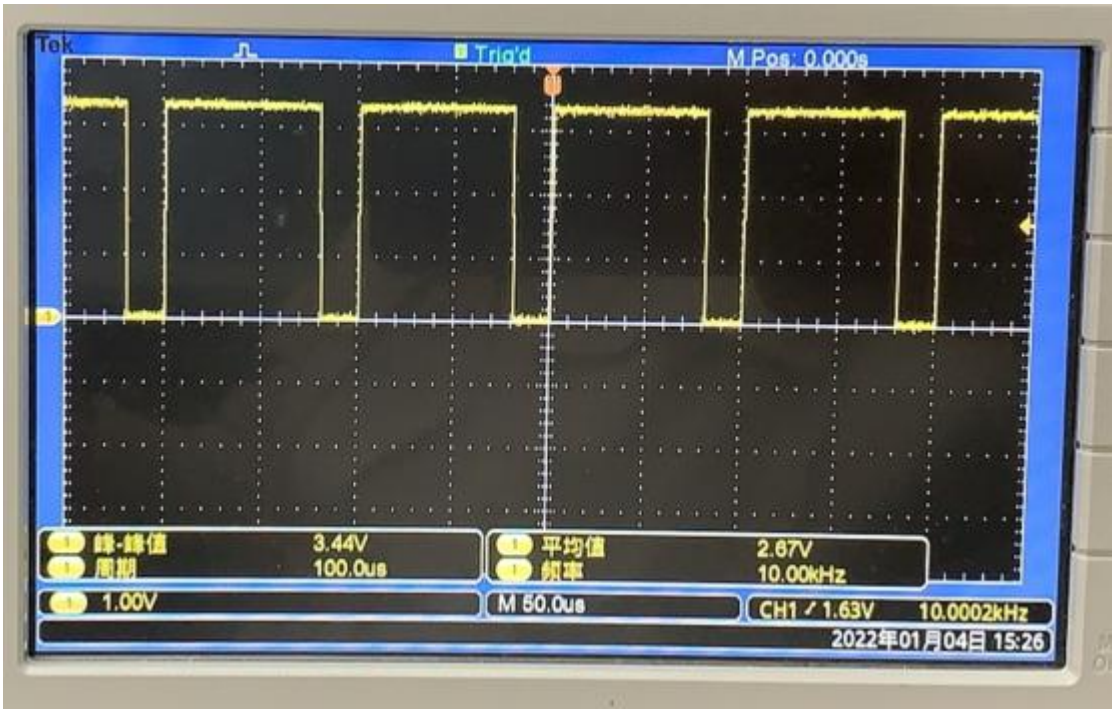
```
orangepi@orangepi:~$ ls /dev/ttyS*
/dev/ttyS1    /dev/ttyS5
```

e. Note that the debugging serial port will be stuck in the following place, and there will be no more output and no input.

```
## Loading init Ramdisk from Legacy Image at 43300000 ...
   Image Name:   uInitrd
   Image Type:   ARM Linux RAMDisk Image (gzip compressed)
   Data Size:    8472387 Bytes = 8.1 MiB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
   Loading Ramdisk to 497eb000, end 49fff743 ... OK
   reserving fdt memory region: addr=48000000 size=1000000
## Linux machid: 00000000, FDT addr: 7be88d60

Starting kernel ...
```

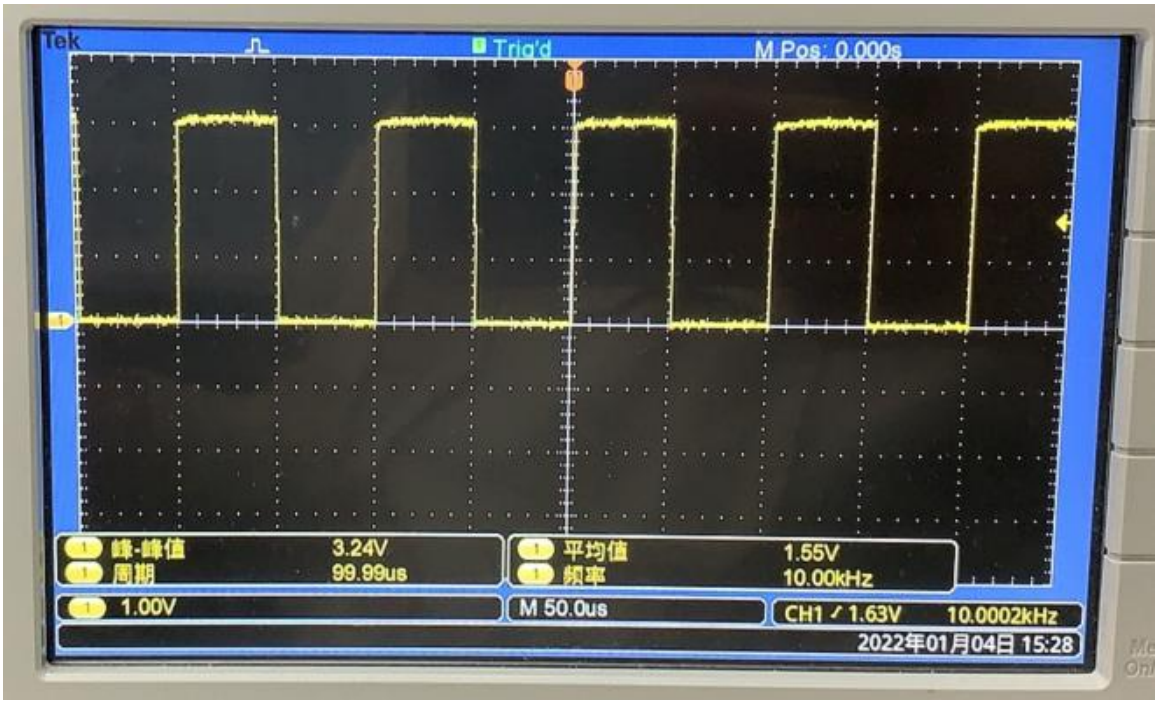
f. Then you can start the PWM test, enter the following command in the Linux4.9 system to make pwm3 output a 10kHz rectangular wave

```
root@orangepi:~# echo 3 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 100000 > /sys/class/pwm/pwmchip0/pwm3/period
root@orangepi:~# echo 20000 > /sys/class/pwm/pwmchip0/pwm3/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm3/enable
```



g. Enter the following command in the Linux4.9 system to make pwm4 output a 10kHz square wave

```
root@orangepi:~# echo 4 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 100000 > /sys/class/pwm/pwmchip0/pwm4/period
root@orangepi:~# echo 50000 > /sys/class/pwm/pwmchip0/pwm4/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm4/enable
```



3.21. The Way to use SPI LCD display

Note: This method is only applicable to the system with the linux4.9 kernel, and the system with the Linux5.13 kernel is not adapted

3.21.1. 2.4 inch SPI LCD display

1) The details page of the tested LCD display is linked as follows

http://www.lcdwiki.com/2.4inch_SPI_Module_ILI9341_SKU:MSP2402

2) The wiring method of the LCD display and the development board is as follows

TFT SPI Module pins	The corresponding pin of the development board 26pin	GPIO -- GPIO num
VCC	1 pin	
GND	6 pin	
CS	24 pin	

RESET	7 pin	PC9 -- 73
D/C	11 pin	PC6 -- 70
SDI(MOSI)	19 pin	
SCK	23 pin	
LED	13 pin	PC5 -- 69
SDO(MISO)	21 pin	

3) After connecting the display to the development board, use the following command to load the **fbtft_device** kernel module

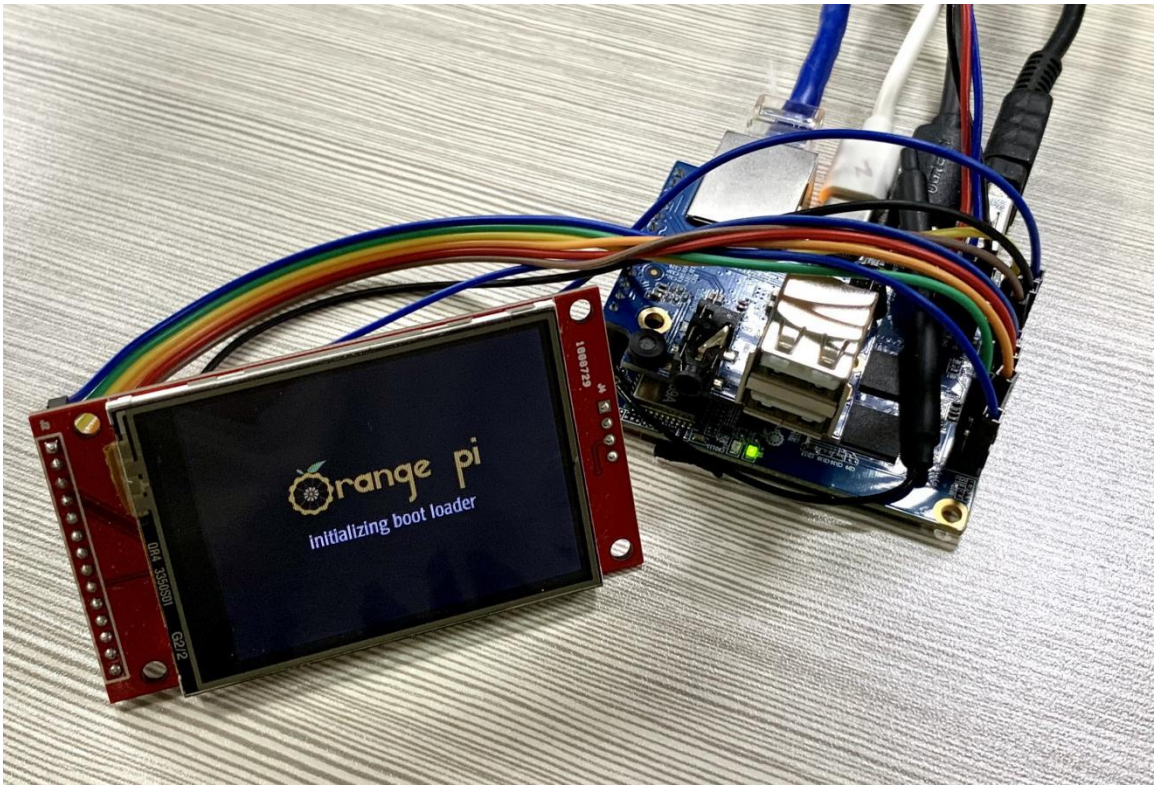
```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341    busnum=1 cs=1 gpios=reset:73,dc:70,led:69 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is as follows, and the log can know that the framebuffer used by the LCD display is **fb1**

```
root@orangepi:~# dmesg | tail
[    391.862343] spidev spi1.1: dh2228fv spi1.1 16777kHz 8 bits mode=0x00
[    391.862773] spidev spi1.1: Deleting spi1.1
[    391.864506] fbtft_device: GPIOS used by 'fb_ili9341':
[    391.864529] fbtft_device: 'reset' = GPIO73
[    391.864540] fbtft_device: 'dc' = GPIO70
[    391.864550] fbtft_device: 'led' = GPIO69
[    391.864579] spidev spi0.0: dh2228fv spi0.0 16777kHz 8 bits mode=0x00
[    391.864598] spi spi1.1: fb_ili9341 spi1.1 65000kHz 8 bits mode=0x00
[    391.883881] fb_ili9341: module is from the staging directory, the quality is unknown, you have been warned.
[    392.159982] graphics fb1: fb_ili9341 frame buffer, 320x240, 150 KiB video memory, 64 KiB buffer memory, fps=20, spi1.1 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo image on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb1 /boot/boot.bmp
```



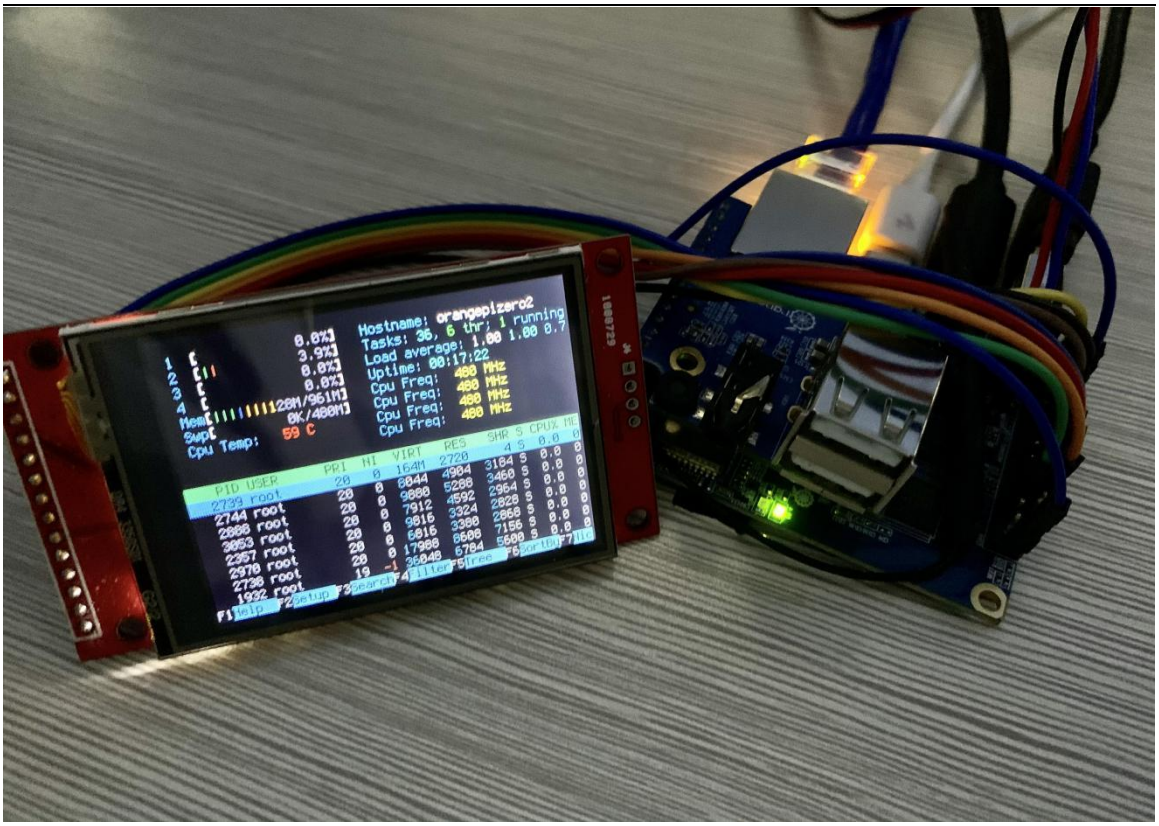
6) You can also map the output of tty1 to the fb device of the LCD display screen - **fb1**. After the mapping, there will be no more image output from HDMI

```
root@orangepi:~# con2fbmap 1 1
```

If you want to switch back to HDMI display use below command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command

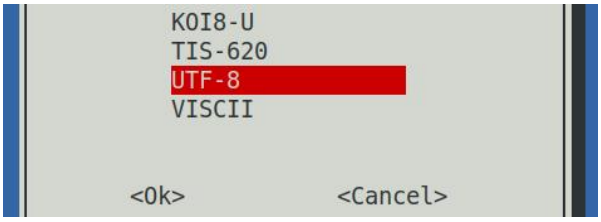


7) Since the default terminal font is too large, the display screen cannot display too much content. The following method can be used to reduce the font of the terminal

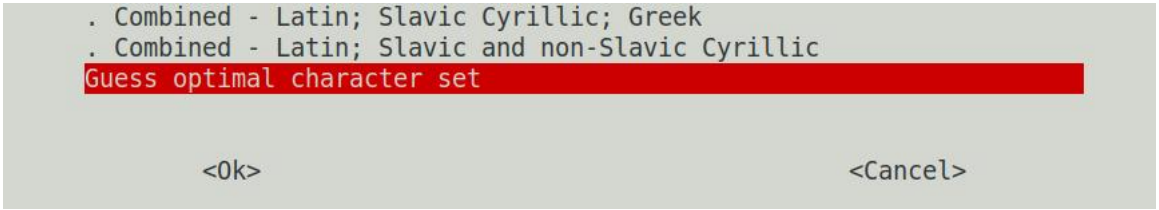
a. first run **dpkg-reconfigure console-setup**

```
root@orangepi:~# apt-get update
root@orangepi:~# apt-get install kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

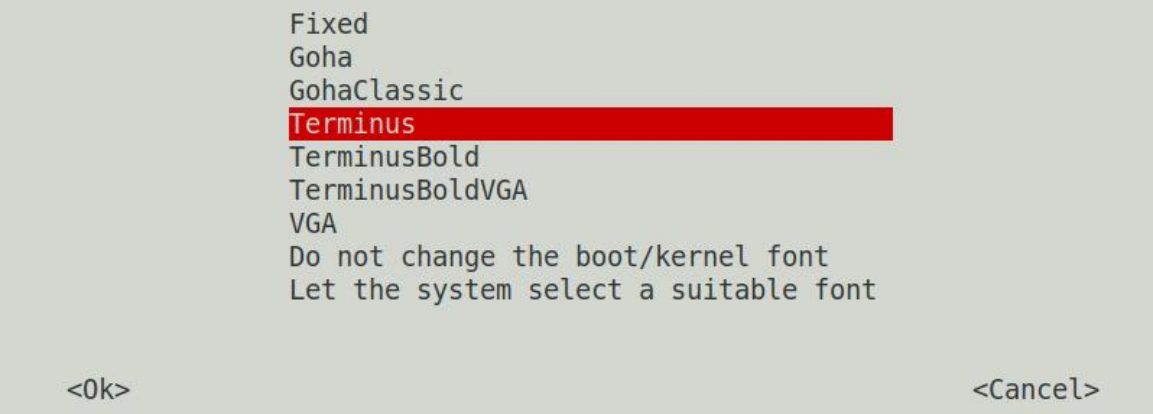
b. Terminal code selection **UTF-8**



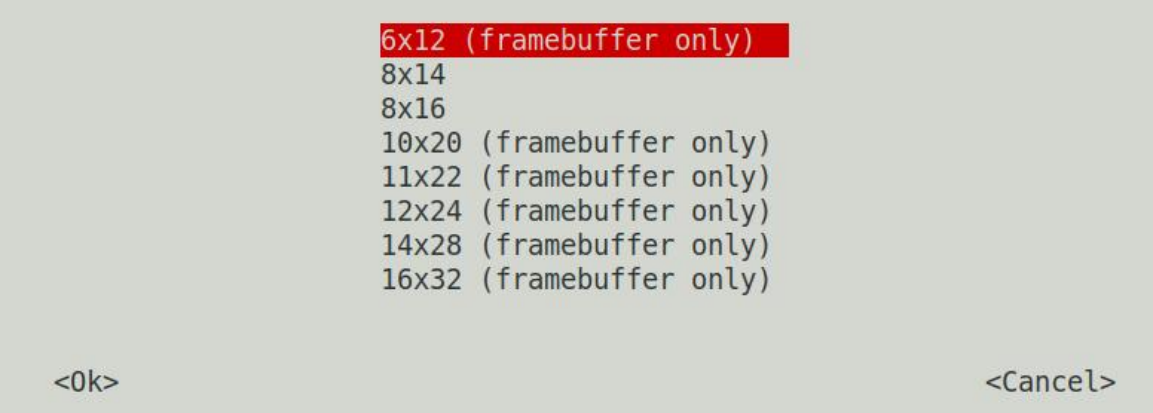
c. then select **Guess optimal character set**



d. then select **Terminus**



e. Finally select the font size as 6x12



f. After setting, you can see that the font on the LCD display becomes smaller

8) Set the method to automatically load the fbtft_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9341    busnum=1 cs=1
gpios=reset:73,dc:70,led:69 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system to see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD display

after booting, please add the following configuration to **/boot/orangepiEnv.txt**, and then restart the system to see the LCD display output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=fbcon=map:1
```

3.21.2. 3.2 inch RPi SPI LCD display

1) The details page of the tested LCD display is linked as follows

```
http://www.lcdwiki.com/3.2inch_RPi_Display
```

2) The wiring method of LCD display and development board is as follows

3) After connecting the LCD display to the development board, use the following command to load the **fbtft_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341    busnum=1 cs=1
gpios=reset:69,dc:72 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

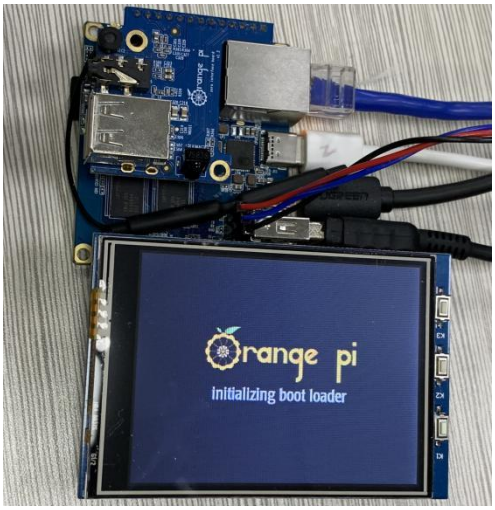
4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is as follows, and the log can know that the framebuffer used by the LCD screen is **fb1**

```
root@orangepi:~# dmesg | tail
[    271.924571] spidev spi0.0: dh2228fv spi0.0 16777kHz 8 bits mode=0x00
[    271.924598] spidev spi1.1: dh2228fv spi1.1 16777kHz 8 bits mode=0x00
[    271.925034] spidev spi1.1: Deleting spi1.1
[    271.926925] fbtft_device: GPIOS used by 'fb_ili9341':
[    271.926957] fbtft_device: 'reset' = GPIO69
[    271.926968] fbtft_device: 'dc' = GPIO72
[    271.926997] spidev spi0.0: dh2228fv spi0.0 16777kHz 8 bits mode=0x00
[    271.927016] spi spi1.1: fb_ili9341 spi1.1 65000kHz 8 bits mode=0x00
```

```
[    271.946173] fb_ili9341: module is from the staging directory, the quality is unknown, you have been warned.
[    272.220982] graphics fb1: fb_ili9341 frame buffer, 320x240, 150 KiB video memory, 64 KiB buffer memory, fps=20, spi1.1 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo image on the LCD screen

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb1 /boot/boot.bmp
```



6) You can also map the output of tty1 to the fb device of the LCD screen - **fb1**. After the mapping, there will be no more image output from HDMI

```
root@orangepi:~# con2fbmap 1 1
```

If you want to switch back to HDMI display use below command

```
root@orangepi:~# con2fbmap 1 0
```

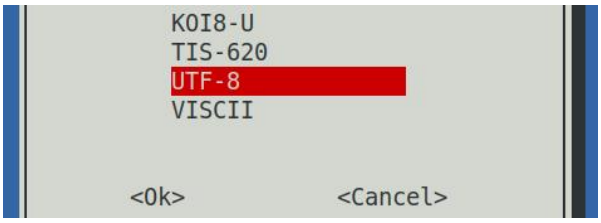
Below is the output of running the htop command

7) Since the default terminal font is too large, the screen cannot display too much content. You can reduce the terminal font by the following method

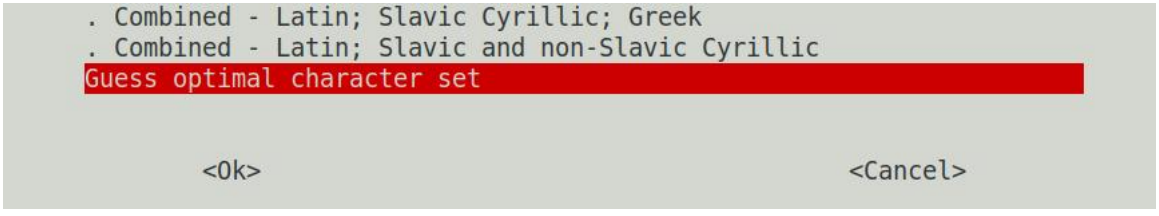
a. first run **dpkg-reconfigure console-setup**

```
root@orangepi:~# apt-get update
root@orangepi:~# apt-get -y install kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

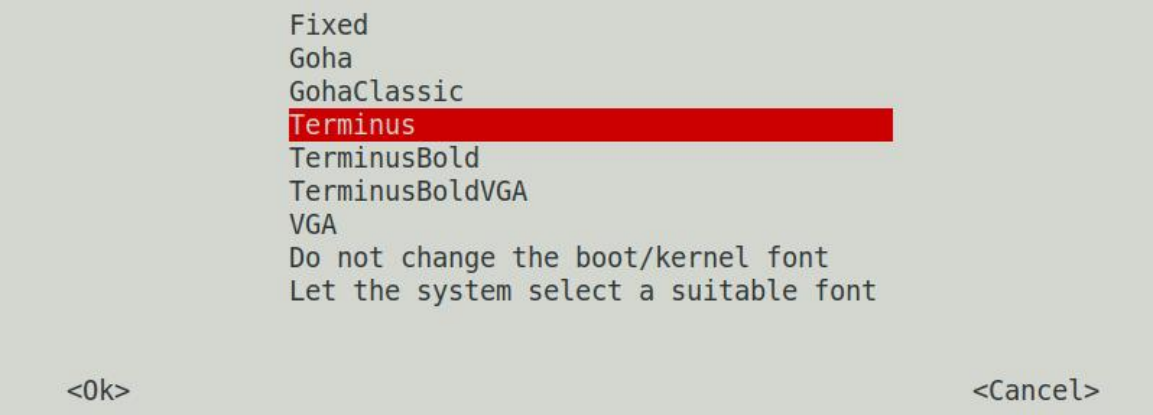
b. Terminal code selection **UTF-8**



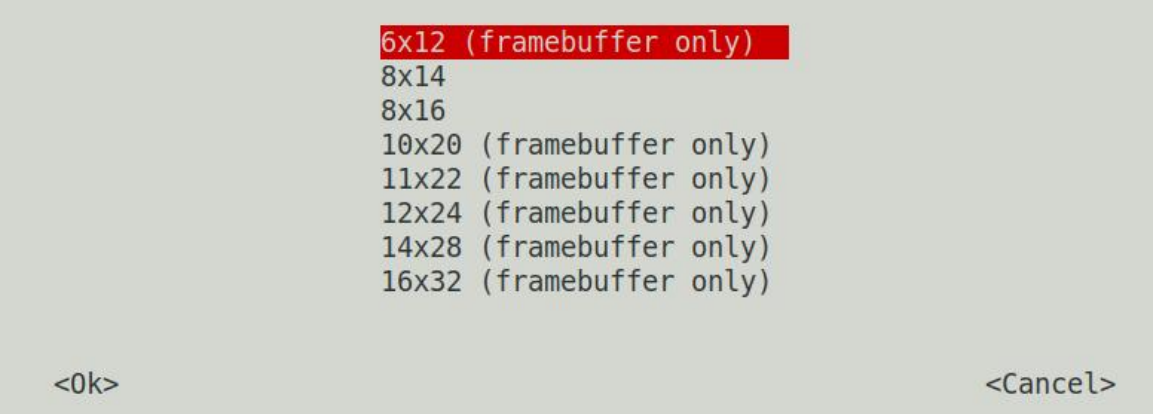
c. then select **Guess optimal character set**



d. then select **Terminus**



e. Finally select the font size as 6x12



f. After setting, you can see that the font on the LCD screen has become smaller

8) Set the method to automatically load the fbtft_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new /etc/modprobe.d/fbtft.conf configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9341  busnum=1 cs=1  gpios=reset:69,dc:72
rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system to see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD screen after booting, please add the following configuration to **/boot/orangepiEnv.txt**, and then

restart the system to see the LCD screen output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=fbcon=map:1
```

3.21.3. 3.5 inch SPI LCD display

1) The details page of the tested LCD display is linked as follows

```
http://www.lcdwiki.com/3.5inch_SPI_Module_ILI9488_SKU:MSP3520
```

2) The wiring method of the LCD display and the development board is as follows

TFT SPI Module pins	The corresponding pin of the development board 26pin	GPIO -- GPIO num
VCC	1 pin	
GND	6 pin	
CS	24 pin	
RESET	7 pin	PC9 -- 73
DC/RS	11 pin	PC6 -- 70
SDI(MOSI)	19 pin	
SCK	23 pin	
LED	13 pin	PC5 -- 69
SDO(MISO)	21 pin	

3) After connecting the display to the development board, use the following command to load the **fbtft_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9488    busnum=1 cs=1
gpios=reset:73,dc:70,led:69 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is as follows, and the log can know that the framebuffer used by the LCD display is **fb1**

```
root@orangepi:~# dmesg | tail
[    378.953595] spidev spi1.1: dh2228fv spi1.1 16777kHz 8 bits mode=0x00
[    378.953952] spidev spi1.1: Deleting spi1.1
[    378.955865] fbtft_device: GPIOS used by 'fb_ili9488':
[    378.955881] fbtft_device: 'reset' = GPIO73
[    378.955890] fbtft_device: 'dc' = GPIO70
[    378.955898] fbtft_device: 'led' = GPIO69
```

```
[    378.955924] spidev spi0.0: dh2228fv spi0.0 16777kHz 8 bits mode=0x00
[    378.955939] spi spi1.1: fb_ili9488 spi1.1 65000kHz 8 bits mode=0x00
[    378.971754] fb_ili9488: module is from the staging directory, the quality is unknown, you have been warned.
[    379.318032] graphics fb1: fb_ili9488 frame buffer, 480x320, 300 KiB video memory, 64 KiB buffer memory, fps=60, spi1.1 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo image on the LCD display

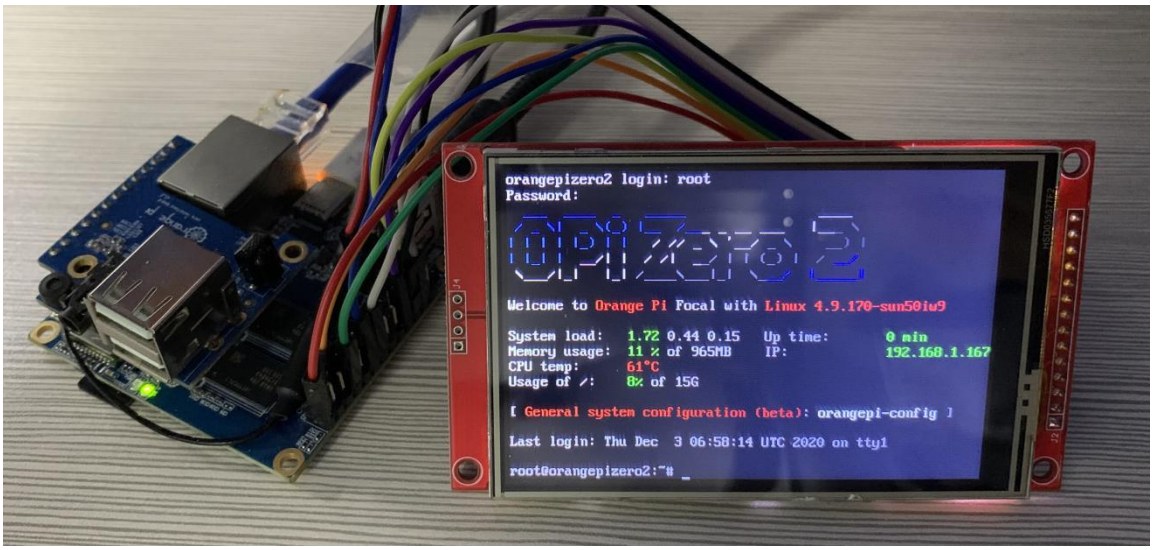
```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb1 /boot/boot.bmp
```

6) You can also map the output of tty1 to the fb device of the LCD display screen - **fb1**. After mapping, the LCD screen will display the output of the terminal, and there will be no more image output from HDMI

```
root@orangepi:~# con2fbmap 1 1
```

If you want to switch back to HDMI display use below command

```
root@orangepi:~# con2fbmap 1 0
```



7) Set the method to automatically load the fbtft_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9488  busnum=1 cs=1
gpios=reset:73,dc:70,led:69 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

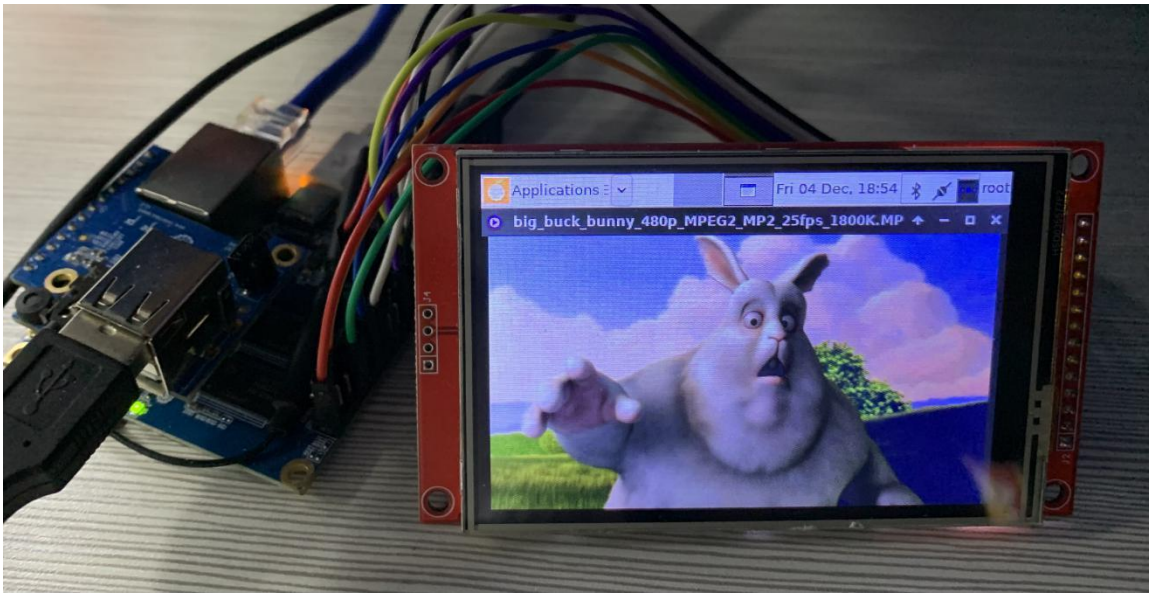
c. Then restart the linux system to see that the kernel modules related to fbtft_device have been automatically loaded

8) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to **/boot/orangepiEnv.txt**, and then restart the system to see that the LCD display has output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=fbcon=map:1
```

9) If you need to display the desktop version system to the LCD screen, you can execute the following command, after a few seconds, the LCD screen can see the desktop of the Linux system

```
root@orangepi:~# FRAMEBUFFER=/dev/fb1 startx
```



10) If you want to automatically display the desktop to the LCD display after the linux system starts, please add the following configuration file to the linux system, and then restart the system to see that the LCD display has display output

```
root@orangepi:~# cat /usr/share/X11/xorg.conf.d/99-fbdev.conf
Section "Device"
    Identifier "myfb"
    Driver "fbdev"
    Option "fbdev" "/dev/fb1"
```

```
EndSection
```

3.22. The method of outputting the kernel print information to the 26pin serial port

The kernel console is output to ttyS0 by default, which is the 3pin debug serial port on the development board. We can also set the kernel console output to be redirected to UART5 in the 26pin interface. Please refer to the following steps for the specific method.

If you are using a Linux 5.16 kernel system, uart5 is disabled by default and needs to be manually opened before it can be used.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then restart the Linux system to open uart5.

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=uart5

1) Modify console=ttyS0 in /boot/boot.cmd to console=ttyS5

rangepi@orangepi:~$ **sudo vim /boot/boot.cmd**

```
if test "${console}" = "display" || test "${console}" = "both"; then setenv consoleargs "console=ttyS5 115200 console=tty1"; fi
if test "${console}" = "serial"; then setenv consoleargs "console=ttyS5 115200"; fi
if test "${bootlogo}" = "true"; then setenv consoleargs "bootsplash.bootfile=bootsplash.orangepi ${consoleargs}"; fi
```

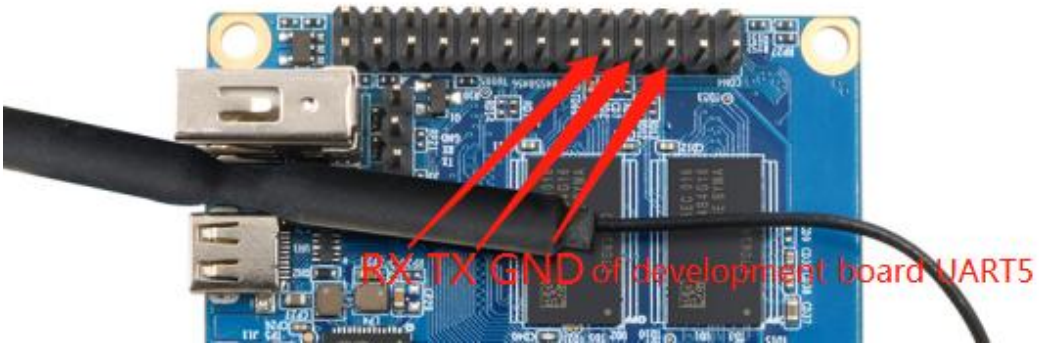
2) Then recompile /boot/boot.cmd to /boot/boot.scr (operate in the linux system of the development board)

rangepi@orangepi:~$ **sudo mkimage -C none -A arm **
-T script -d /boot/boot.cmd /boot/boot.scr
Image Name:
Created: Tue Nov 3 01:45:17 2020
Image Type: ARM Linux Script (uncompressed)
Data Size: 2247 Bytes = 2.19 KiB = 0.00 MiB
Load Address: 00000000
Entry Point: 00000000
Contents:

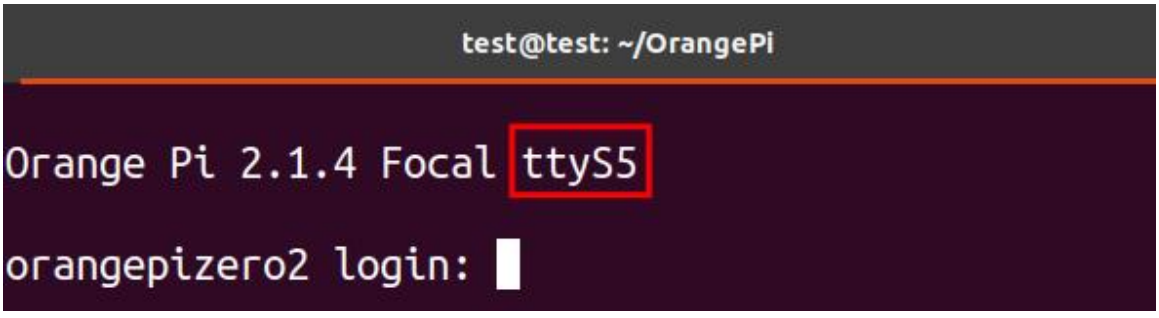
```
Image 0: 2239 Bytes = 2.19 KiB = 0.00 MiB
```

3) Then connect the USB to TTL module to the UART5 pin of the 26pin interface through the DuPont line

a. The GND of the USB to TTL module is connected to the GND of the 26pin interface of the development board
b. **The RX of the USB to TTL module is connected to the TX** of the UART5 of the development board
c. **The TX of the USB to TTL module is connected to the RX** of the UART5 of the development board



4) Then restart the development board, you can see that the kernel console is output to ttyS5 by default. Note that the output log of u-boot is still output to ttyS0 at this time, and will not be output to ttyS5



3.23. The Way to use 0.96-inch OLED module with I2C interface

If you are using a Linux 5.16 kernel system, i2c3 is disabled by default and needs to be manually opened before it can be used.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then

restart the Linux system to open i2c3.

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=i2c3

1) The 0.96-inch OLED module of Orange pi is shown in the figure below, and its 7-bit i2c slave address is 0x3c

2) First, connect the 0.96-inch OLED module to the 26pin interface of the Orange Pi development board through the DuPont cable. The wiring method is as follows

OLED Module pins	Description	Development board 26pin interface corresponding pin
GND	power ground	6 pin
VCC	5V	2 pin
SCL	I2C clock line	5 pin
SDA	I2C data line	3 pin
RST	Connect 3.3V	1 pin
DC	Connect GND	9 pin
CS	Connect GND	25 pin

3) After connecting the OLED module to the development board, first use the i2c-tools tool to check whether the address of the OLED module can be scanned

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y i2c-tools
orangepi@orangepi:~$ sudo i2cdetect -y 3
```

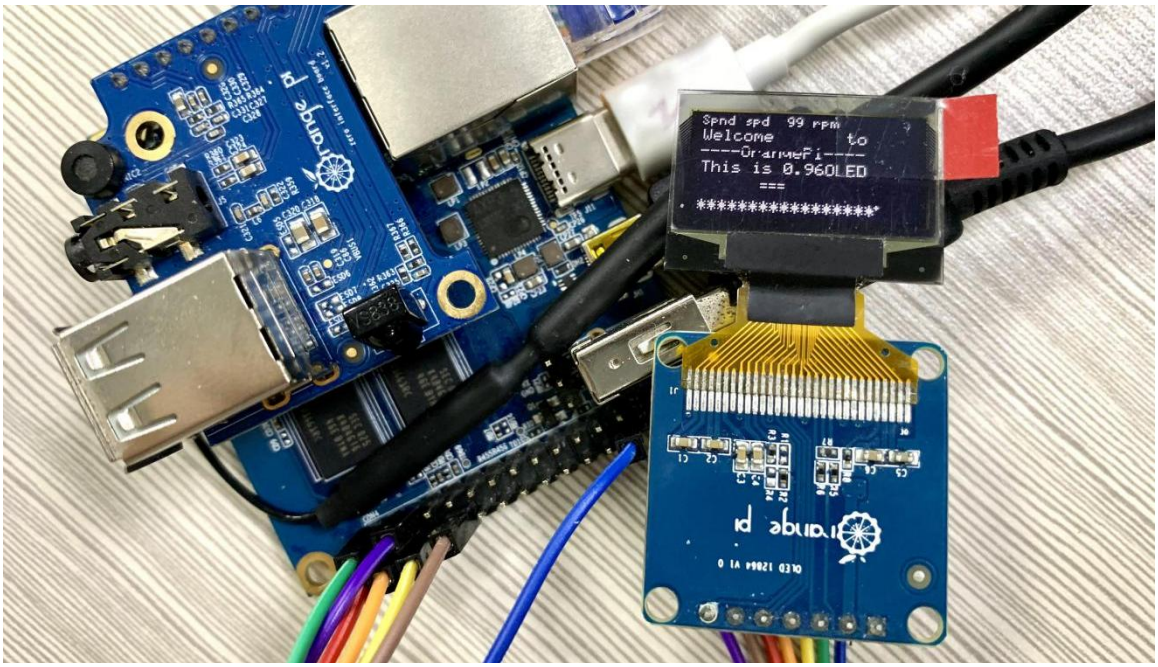
```
root@orangepizero2:~# i2cdetect -y 3
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@orangepizero2:~#
```

4) Then you can use the **oled_demo** in wiringOP to test the OLED module. The test steps are as follows

```
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

```
root@orangepi:~# cd wiringOP
root@orangepi:~/wiringOP# ./build clean && ./build
root@orangepi:~/wiringOP# cd examples
root@orangepi:~/wiringOP/examples# make oled_demo
root@orangepi:~/wiringOP/examples# ./oled_demo /dev/i2c-3
---------start--------
----------end---------
```

5) After running oled_demo, you can see the following output on the OLED screen



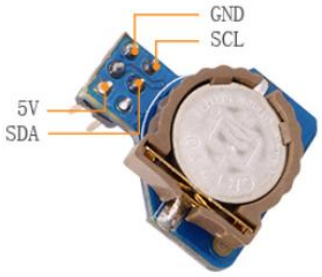
3.24. The method to use the orange pi DS1307 RTC clock module

If you are using a Linux 5.16 kernel system, i2c3 is disabled by default and needs to be manually opened before it can be used.

Add the configuration in the red font part below to /boot/orangepiEnv.txt, and then restart the Linux system to open i2c3.

orangepi@orangepi:~$ **sudo vim /boot/orangepiEnv.txt**
overlays=i2c3

1) The orange pi DS1307 RTC clock module is shown in the figure below. It uses the i2c interface to communicate with the development board, and the i2c device address is 0x68. The RTC module is not equipped with a battery by default, and a button battery needs to be prepared before use



2) First connect the RTC module to the 26pin of the development board, the wiring method is as follows

RTC Module pins	The corresponding pin of the development board 26pin
5V	2 pin
GND	6 pin
SDA	3 pin
SCL	5 pin

3) After connecting the RTC module, first use the i2cdetect command to check whether the device address of the RTC module can be detected

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y i2c-tools
orangepi@orangepi:~$ sudo i2cdetect -y 3
```

```
root@orangepizero2:~# i2cdetect -y 3
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

4) Then add the configuration of the **rtc-ds1307** module in dts. A script called **orangepi-add-overlay** is pre-installed in the linux system. Through this script, we can use DT overlay to dynamically add some functions that are not available in dts. First write the dts file of the **rtc-ds1307** module, the content is as follows

a. linux4.9 system rtc-ds1307 module dts file content

```
orangepi@orangepi:~$ vim i2c-ds1307.dts
/dts-v1/;
/plugin/;

/ {
    compatible = "allwinner,h616", "arm,sun50iw9p1";

    fragment@1 {
        target = <&twi3>;
        __overlay__ {
            #address-cells = <1>;
            #size-cells = <0>;
            ds1307@68 {
                compatible = "dallas,ds1307";
                reg = <0x68>;
                status = "okay";
            };
        };
    };
};
```

b. The content of the rtc-ds1307 module dts file of the linux5.16 system

```
orangepi@orangepi:~$ vim i2c-ds1307.dts
/dts-v1/;
/plugin/;

/ {
        compatible = "xunlong,orangepi-zero2", "allwinner,sun50i-h616";

        fragment@1 {
                target = <&i2c3>;
                __overlay__ {
```

```
                        #address-cells = <1>;
                        #size-cells = <0>;
                        ds1307@68 {
                                compatible = "dallas,ds1307";
                                reg = <0x68>;
                                status = "okay";
                        };
                };
        };
};
```

c. Then use **orangepi-add-overlay** to compile i2c-ds1307.dts into i2c-ds1307.dtbo, and set the relevant startup variables

```
orangepi@orangepi:~$ sudo orangepi-add-overlay i2c-ds1307.dts
Compiling the overlay
Copying the compiled overlay file to /boot/overlay-user/
Reboot is required to apply the changes
```

d. i2c-ds1307.dtbo will be copied to **/boot/overlay-user.** After running **orangepi-add-overlay,** you can check whether the file i2c-ds1307.dtbo exists in **/boot/overlay-user**

```
orangepi@orangepi:~$ cd /boot/overlay-user/
orangepi@orangepi:/boot/overlay-user$ ls
i2c-ds1307.dtbo
```

e. **orangepi-add-overlay** will also add the user_overlays variable in **/boot/orangepiEnv** and set the value to i2c-ssd1307

```
orangepi@orangepi:~$ cat /boot/orangepiEnv.txt | grep "user"
user_overlays=i2c-ds1307
```

f. Then restart the linux system. When starting, you can see the DT overlay related output in the u-boot log

```
U-boot loaded from SD
Boot script loaded from mmc
214 bytes read in 8 ms (25.4 KiB/s)
645 bytes read in 13 ms (47.9 KiB/s)
Applying user provided DT overlay i2c-ds1307.dtbo
8482593 bytes read in 369 ms (21.9 MiB/s)
23638088 bytes read in 1005 ms (22.4 MiB/s)
```

```
## Booting kernel from Legacy Image at 41000000 ...
```

5) After restarting, you can see the loading information of the ds1307 module from the log output by dmesg. The device node corresponding to ds1307 is **rtc0（linux5.16 is rtc1）**

```
orangepi@orangepi:~$ dmesg | grep "rtc"
[    2.131445] rtc-ds1307 3-0068: rtc core: registered ds1307 as rtc0
[    2.131470] rtc-ds1307 3-0068: 56 bytes nvram
[    2.132256] sunxi-rtc rtc: rtc core: registered sunxi-rtc as rtc1
[    2.132329] sunxi-rtc rtc: RTC enabled
[    2.307120] rtc-ds1307 3-0068: setting system clock to 2000-00-00 06:33:46 UTC (1607063626)
```

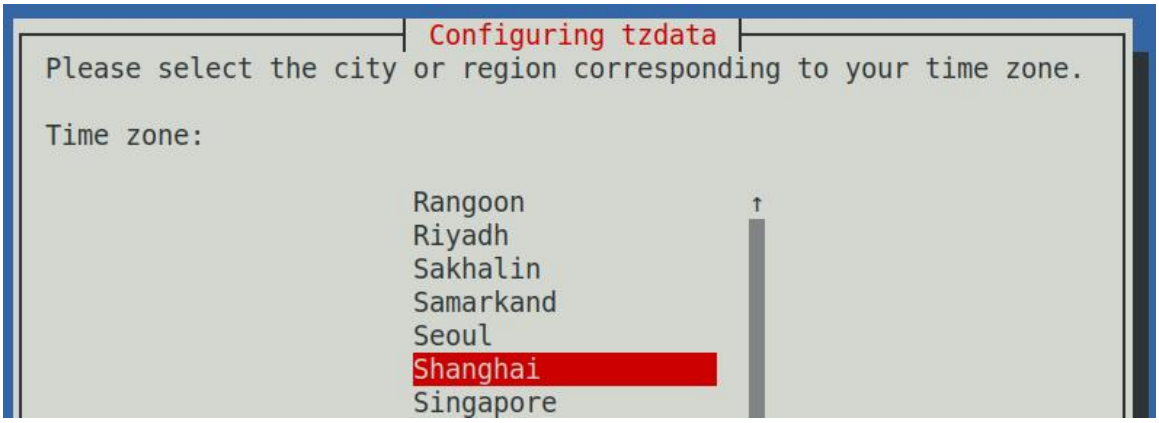
6) When the Linux system starts, if the development board is connected to the network, the Linux system will automatically synchronize the system time to the correct time through the network. The default time of the Linux system is UTC. In China, the time zone needs to be changed to **Asia/Shanghai**. The time obtained by using the data command is correct, the method is as follows

a. Execute the following command

```
orangepi@orangepi:~$ sudo dpkg-reconfigure tzdata
```

b. Then select the geographic area as **Asia**

c. Then select the time zone as **Shanghai**



d. After the configuration is completed, use the date command to view the time and it will be normal

```
orangepi@orangepi:~$ sudo date
```

7) If the current time of the system is incorrect, please connect to the network first, and then use the following command to synchronize the time. The reason why the system time is set correctly is to prepare for the synchronization of the time of the RTC module later.

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt install -y ntpdate
orangepi@orangepi:~$ sudo ntpdate 0.cn.pool.ntp.org
```

8) The command to view the current time of the RTC module is as follows

a. Linux4.9

```
orangepi@orangepi:~$ sudo hwclock -r
```

b. Linux5.16

```
orangepi@orangepi:~$ sudo hwclock -r -f /dev/rtc1
```

9) The time read by the RTC module for the first time is definitely wrong. The current time of the system can be synchronized to the RTC module through the following command. Before synchronization, it is necessary to ensure that the current time of the system is correct

a. Linux4.9

```
orangepi@orangepi:~$ date              #First make sure the current system time is correct
orangepi@orangepi:~$ sudo hwclock -w   #Then write the system time to the RTC module
```

```
orangepi@orangepi:~$ sudo hwclock -r  #Finally read the time of the RTC module to confirm that the settings are correct
```

b. Linux5.16

```
orangepi@orangepi:~$ date
orangepi@orangepi:~$ sudo hwclock -w -f /dev/rtc1
orangepi@orangepi:~$ sudo hwclock -r -f /dev/rtc1
```

10) At this point, you can disconnect all network connections of the development board, wait for a few minutes, restart the system, and then check the system time to find that even if there is no network, the system time is correct

3.25. Hardware watchdog test

1) Download the code of wiringOP

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

2) Compile the watchdog test program

```
orangepi@orangepi:~$ cd wiringOP/examples/
orangepi@orangepi:~/wiringOP/examples$ gcc watchdog.c -o watchdog
```

3) Run the watchdog test program

a. The second parameter 10 represents the counting time of the watchdog. If the dog is not fed within this time, the system will restart

b. We can feed the dog by pressing any key on the keyboard (except ESC), after feeding the dog, the program will print a line of keep alive to indicate that the dog was fed successfully

```
orangepi@orangepi:~/wiringOP/examples$ sudo ./watchdog 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
```

```
put_usr return,if 0,success:0
keep alive
keep alive
keep alive
```

3.26. Set up Chinese environment and install Chinese input method

Note that before installing the Chinese input method, please make sure that the Linux system used by the development board is the desktop version system.

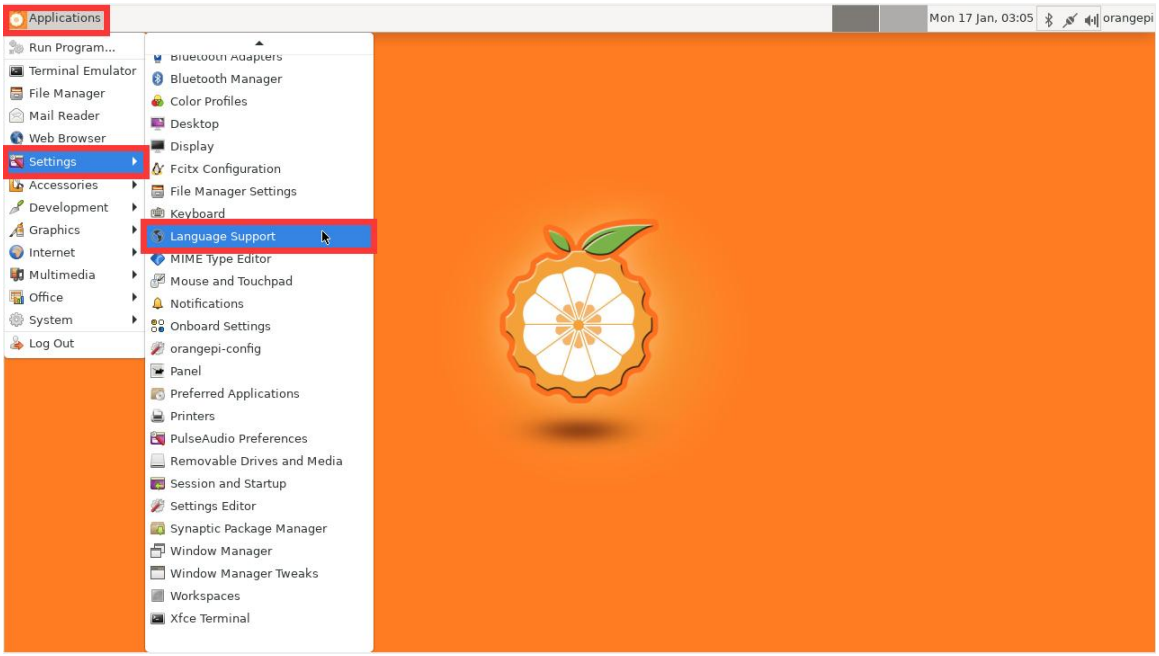
3.26.1. Installation method of Ubuntu system

1) First update the software source of the system

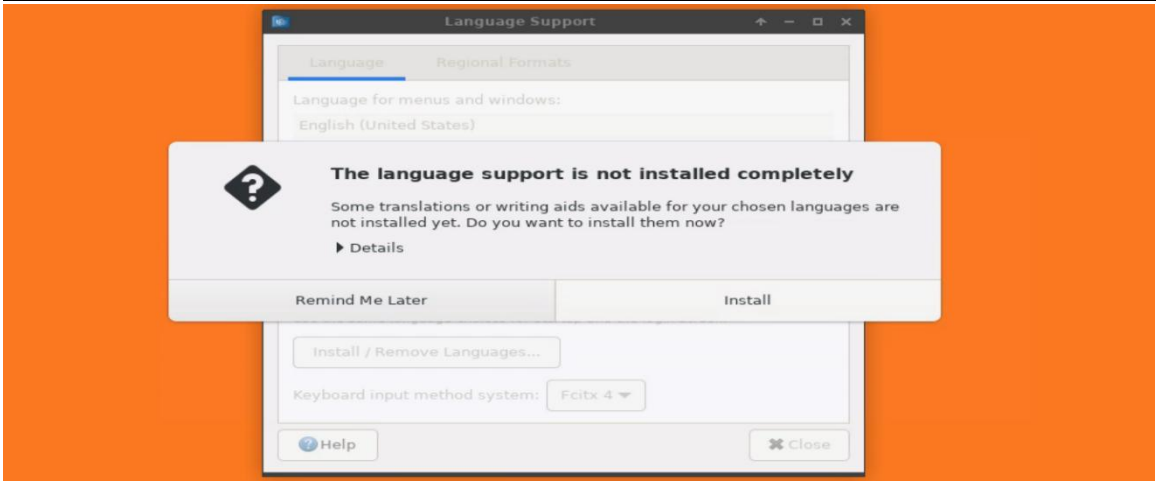
Note that an error may be reported during the installation process without executing the following command, so please do not ignore this step.

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

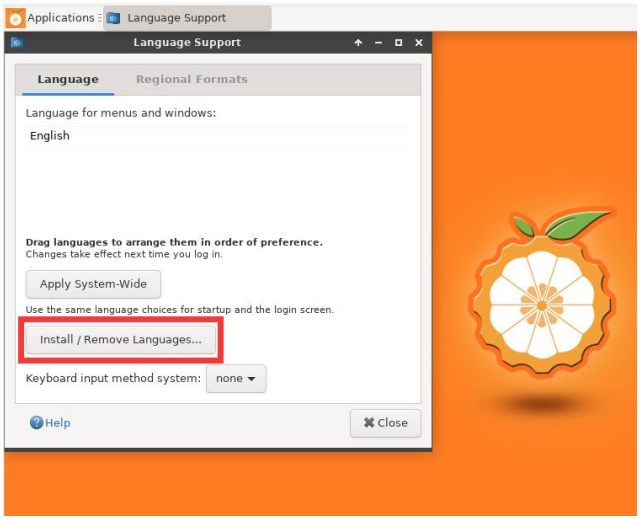
2) Then open **Language Support**



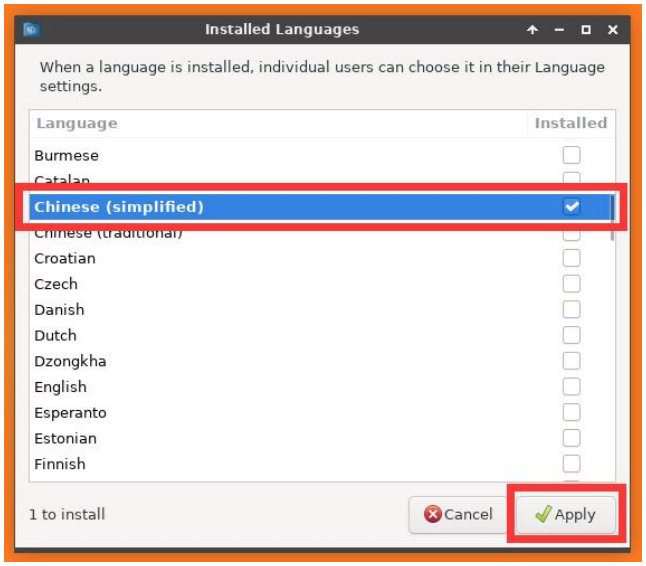
3) If the following information is prompted, please click **Install** to repair it. Some systems will not have the following prompt, just skip it.



4) Then open **Install/Remove Languages...**

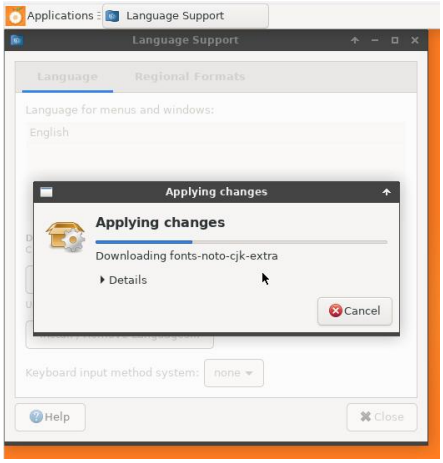


5) Then find **Chinese (simplified)**, click the box on the right to select it, and then click **Apply** in the lower right corner



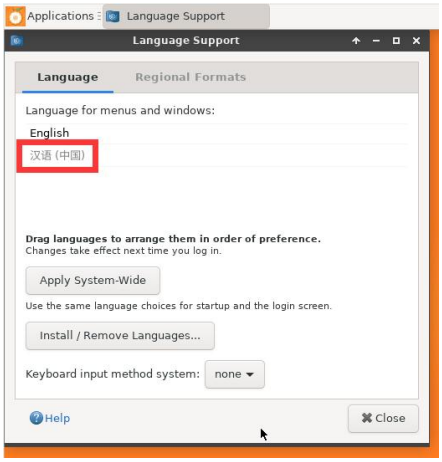
6) Then enter the password of the Linux system in the pop-up password input interface, the default is **orangepi**

7) Then it will start to install the required software packages. At this time, wait patiently for the installation to complete.

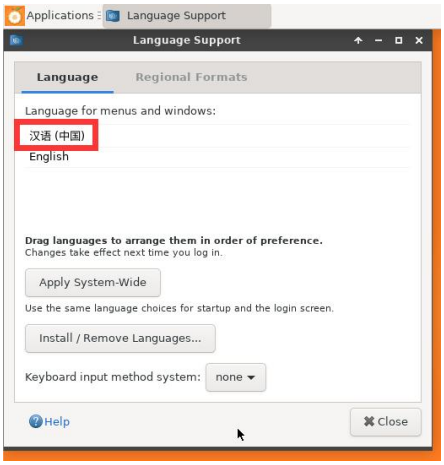


If the software installation fails, it is generally because the apt update command was not executed at the beginning. If the apt update command is executed and the prompt fails, it is generally because an error occurred when the apt update command was executed, but it was not executed successfully.

8) After the installation is complete, you can see the **Chinese (China)** option

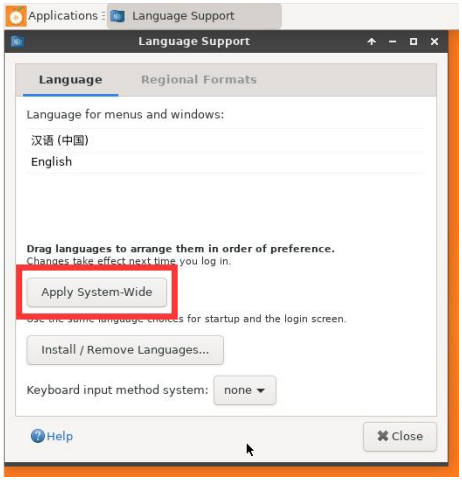


9) Then please use the left mouse button to select **Chinese (China)** and hold it down, then drag it up to the first position, the display after dragging is as shown below:

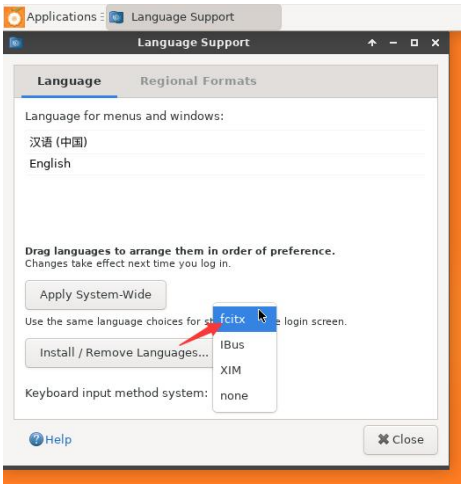


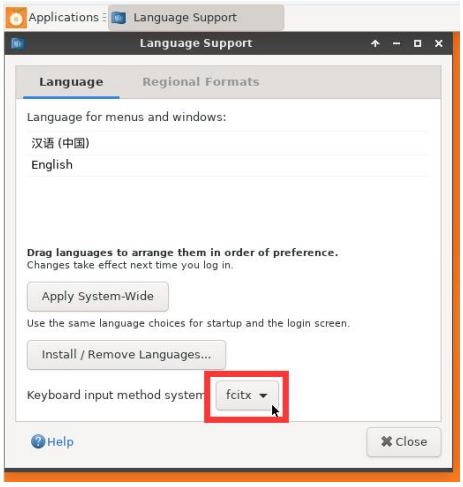
Note that this step is not very easy to drag, please be patient and try several times.

10) Then select **Apply System-Wide** to apply Chinese settings to the entire system



11) Then select **Keyboard input method system** as **fcitx**

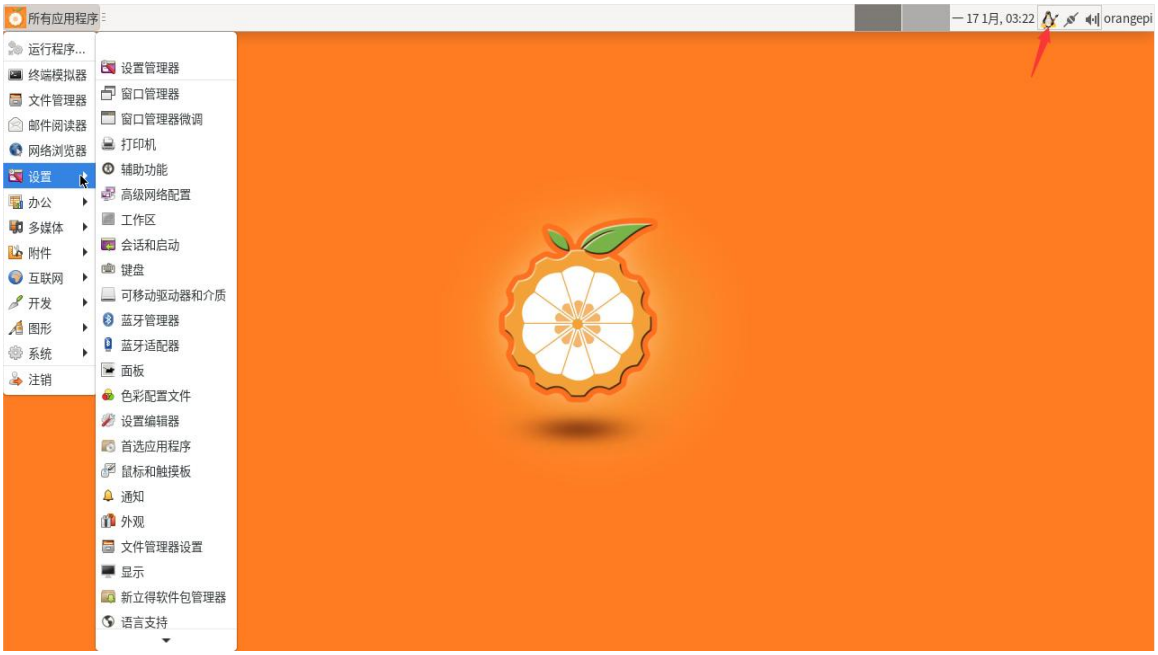




12) **Then restart the Linux system to make the configuration take effect**

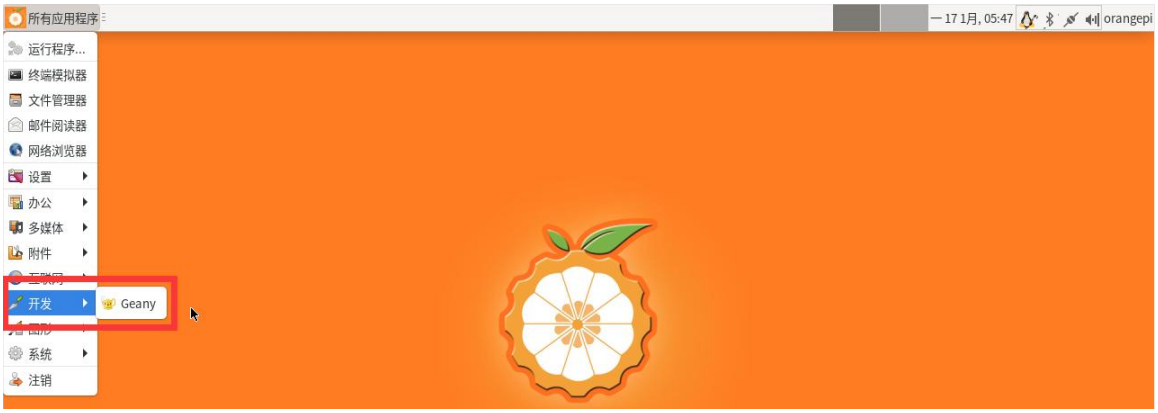
13) After re-entering the system, you can see that the desktop is displayed in Chinese, and you can also see a penguin in the upper right corner of the system

Note that Ubuntu 22.04 shows a black keyboard icon in the upper right corner.

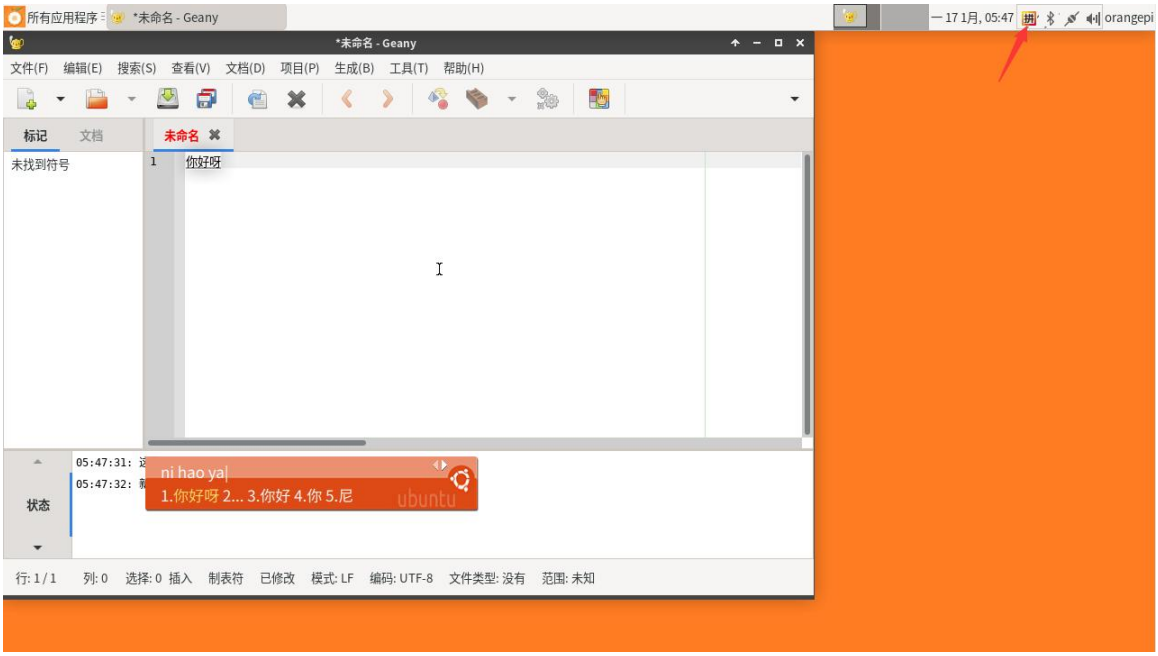


14) Then we can open the Chinese input method under **Geany** test, the opening method is as shown in the figure below

Note that if Geany is not installed on the system, feel free to open another application for testing.



15) After opening **Geany**, the default is English input method, we can switch to Chinese input method by **Ctrl+Space** shortcut key, and then we can input Chinese



16) In addition, you can place the mouse on the penguin in the upper right corner of the desktop, and then click the **right mouse button** to view all input methods supported by the system, or select the input method to be used.

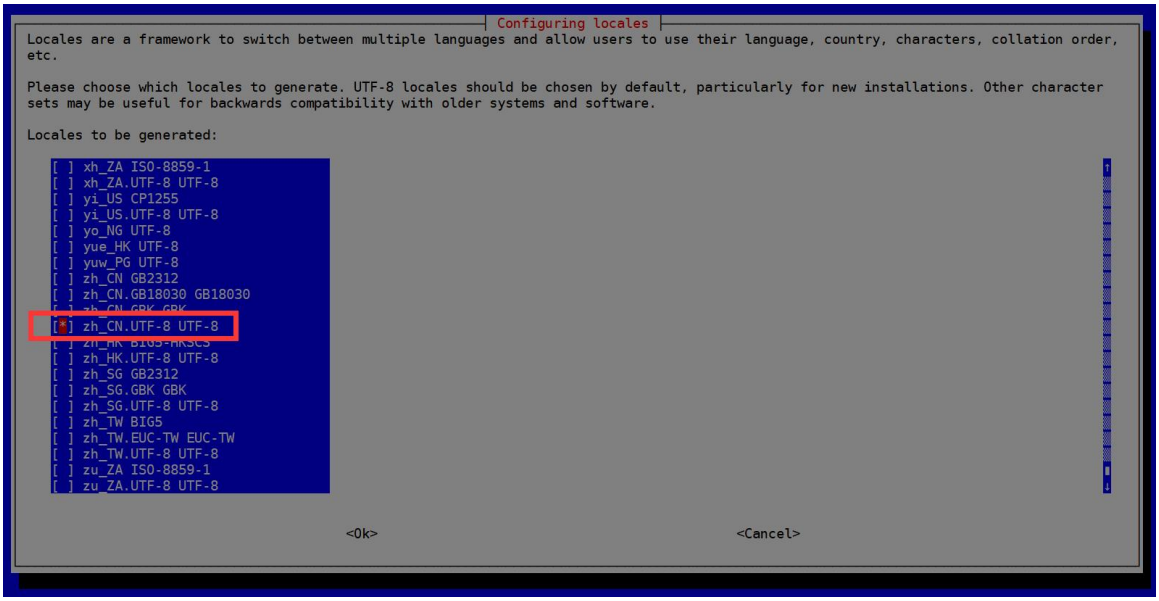
3.26.2. Installation method of Debian system

1) First set the default **locale** to Chinese

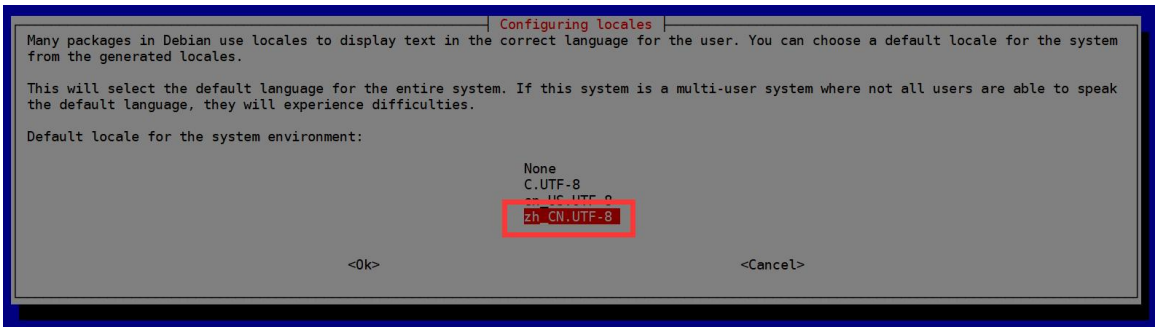
a. Enter the following command to start configuring **locale**

```
orangepi@orangepi:~$ sudo dpkg-reconfigure locales
```

b. Then select **zh_CN.UTF-8 UTF-8** in the pop-up interface (move up and down through the up and down direction keys on the keyboard, select through the space bar, and finally use the Tab key to move the cursor to **<OK>**, and then return to car)



c. Then set the default **locale** to **zh_CN.UTF-8**



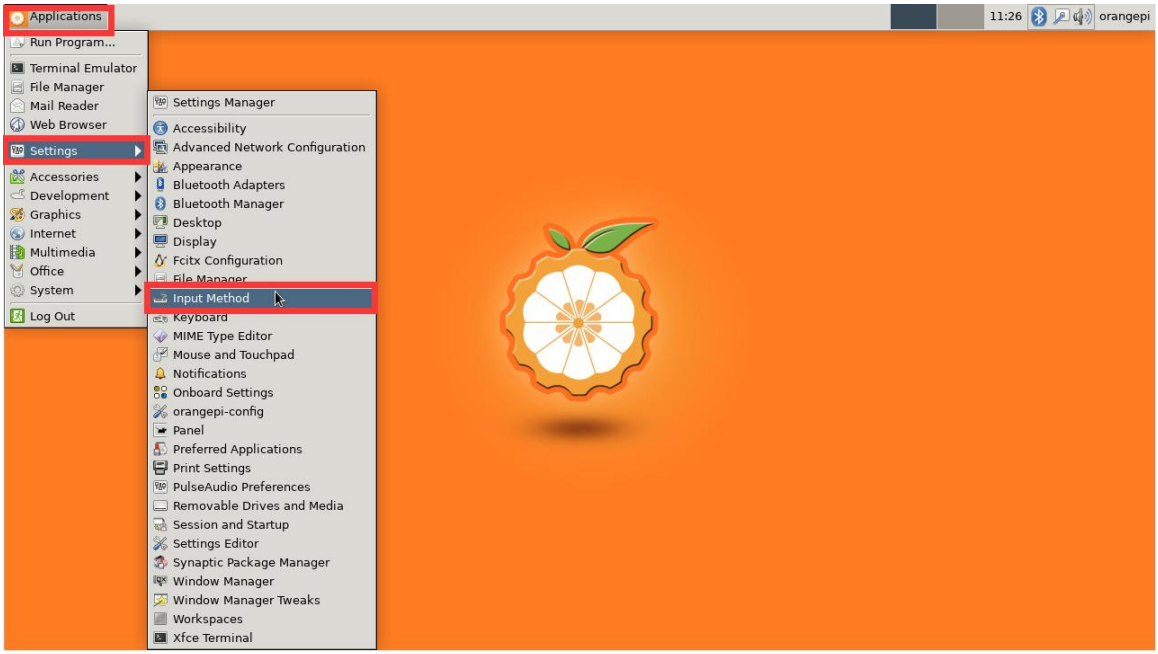
d. After exiting the interface, the **locale** setting will start, and the output displayed on the command line is as follows

```
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.
```

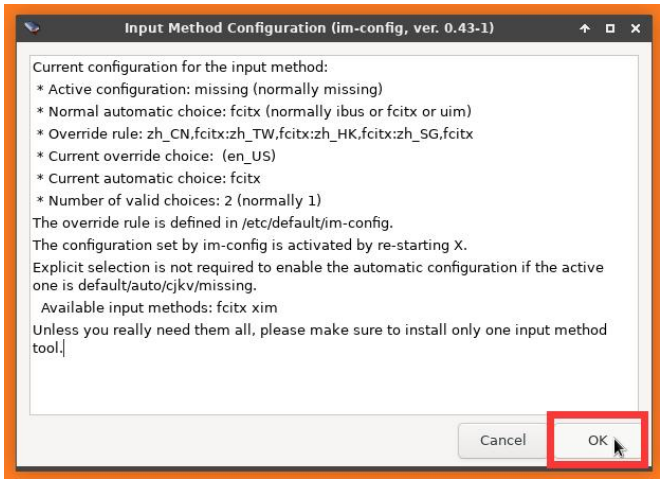
2) Then install the following packages

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y fonts-arphic-bsmi00lp   \
fonts-arphic-gbsn00lp fonts-arphic-gkai00mp fcitx fcitx-table*    \
fcitx-frontend-gtk* fcitx-frontend-qt* fcitx-config-gtk* im-config   \
fcitx-googlepinyin fcitx-ui* zenity geany
```

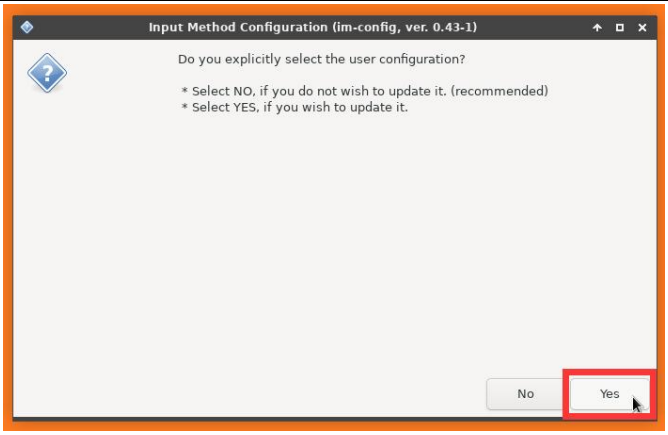
3) Then open **Input Method**



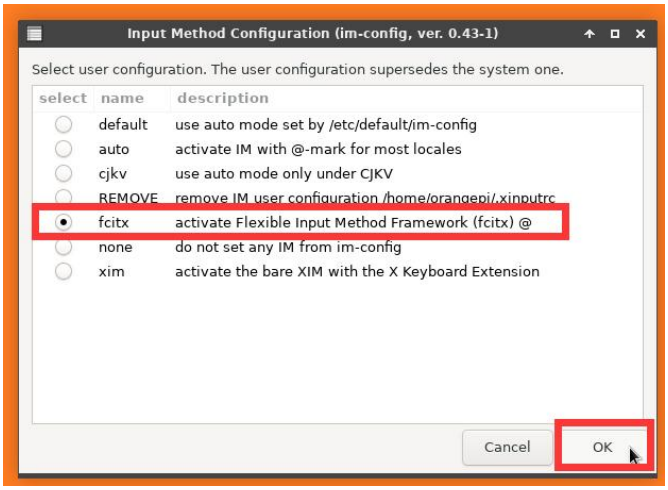
4) Then select**OK**



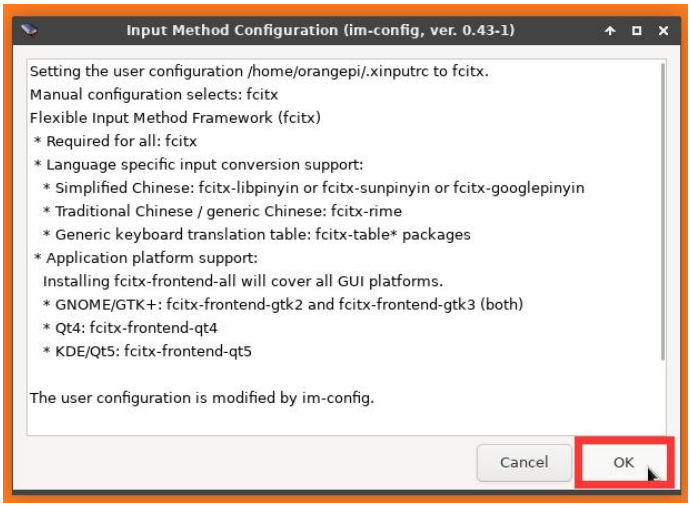
5) Then select **Yes**



6) Then select **fcitx**

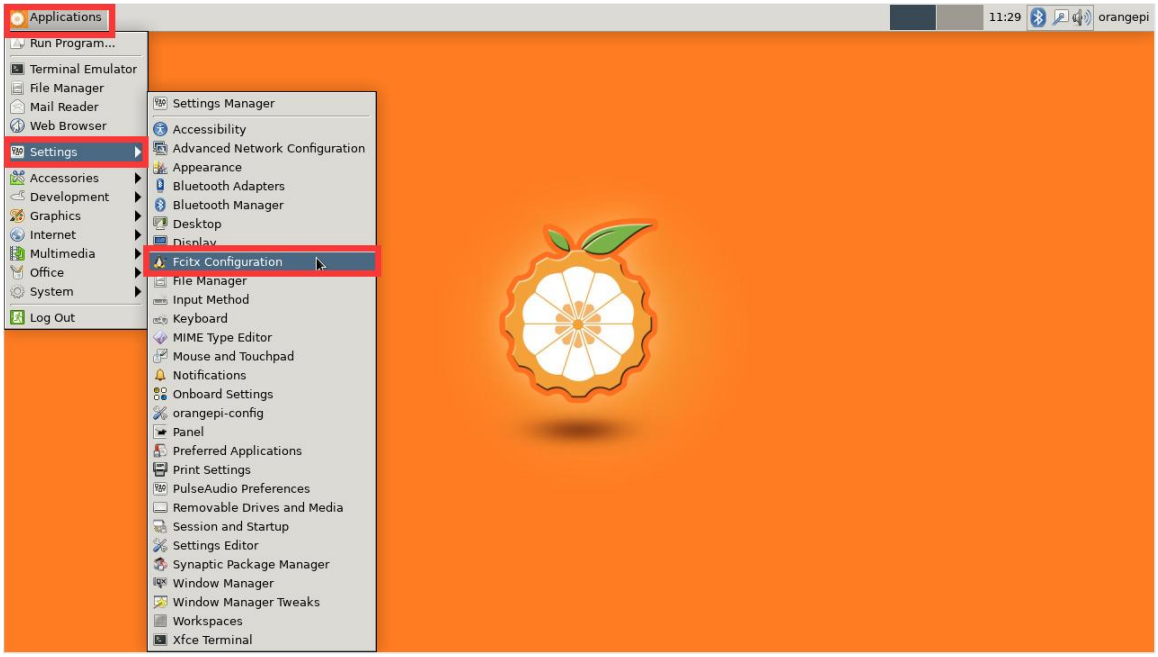


7) Then select **OK**

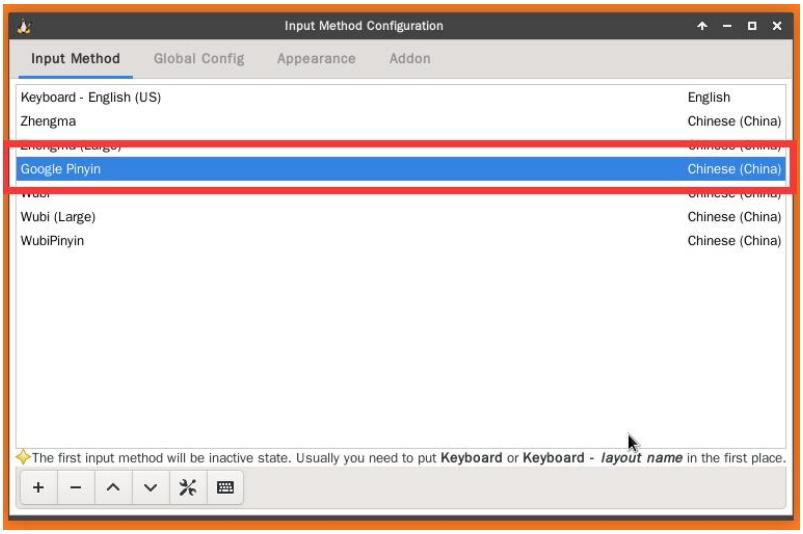


8) **Then restart the Linux system for the configuration to take effect**

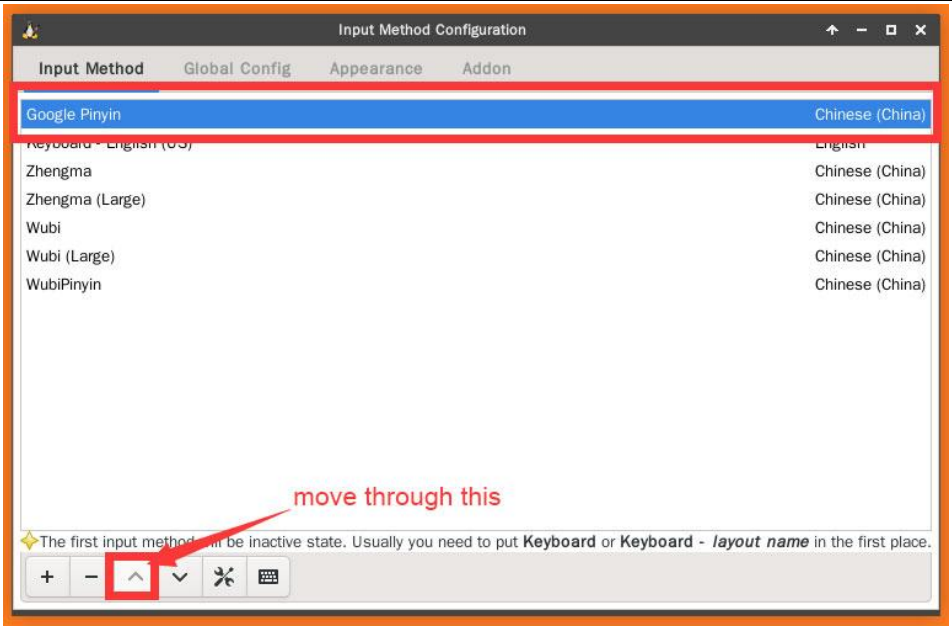
9) Then open **Fcitx configuration**



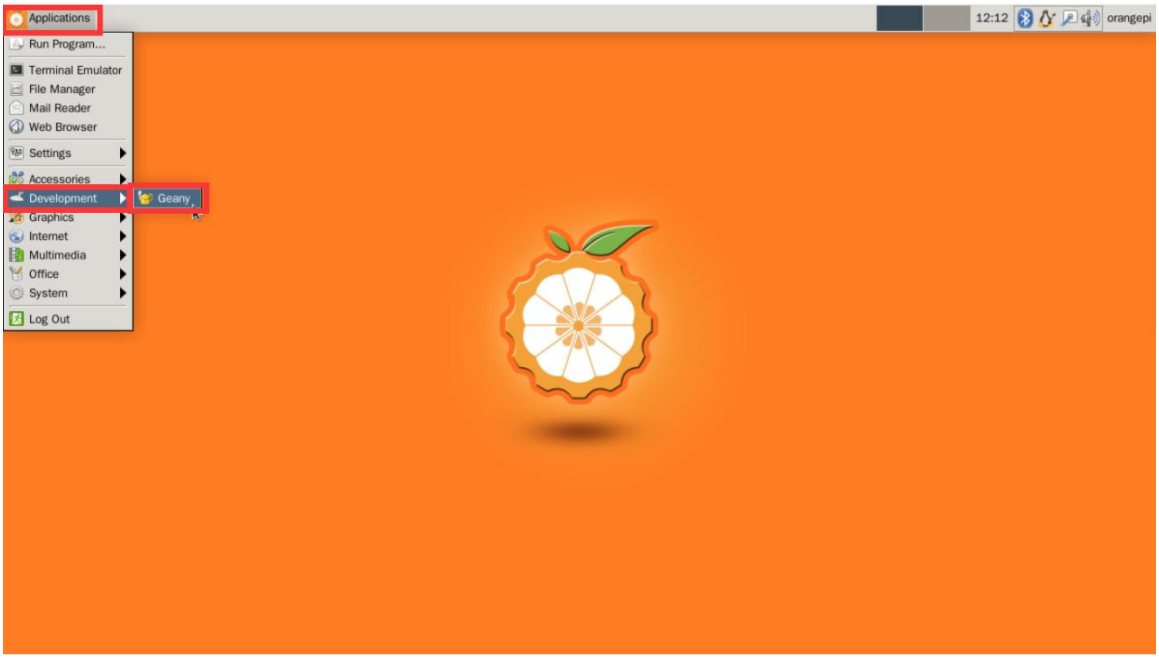
10) Then open **Google Pinyin**



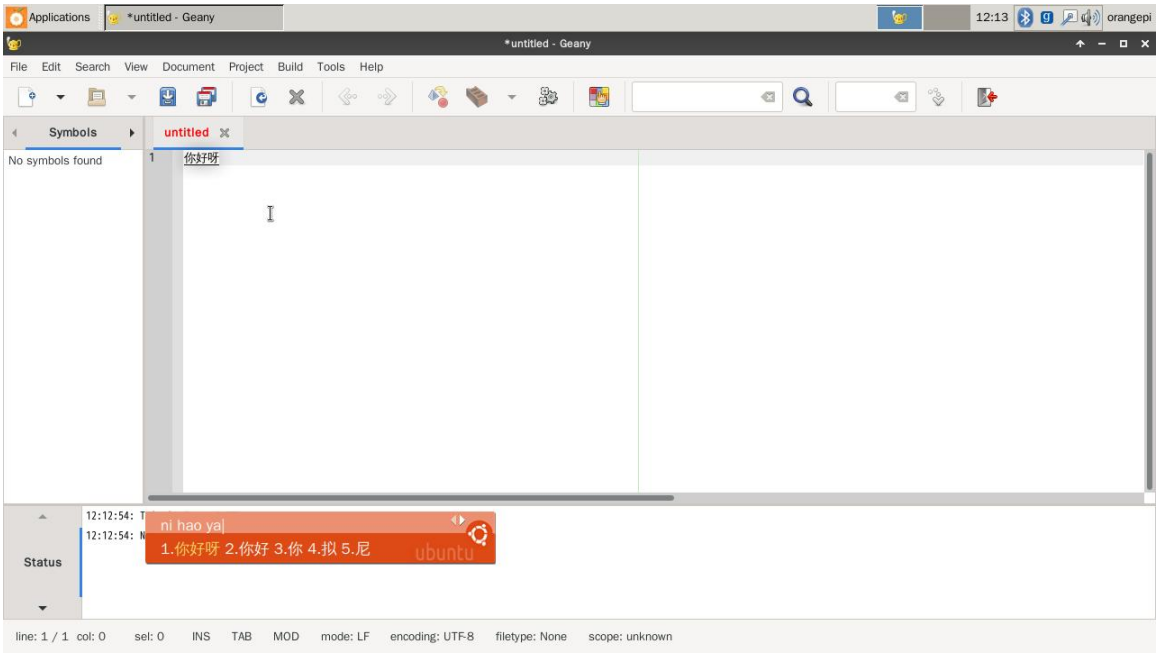
11) Then bring **Google Pinyin** to the front, and then close the configuration interface in the upper right corner



12) Then open the **Geany** editor to test the Chinese input method



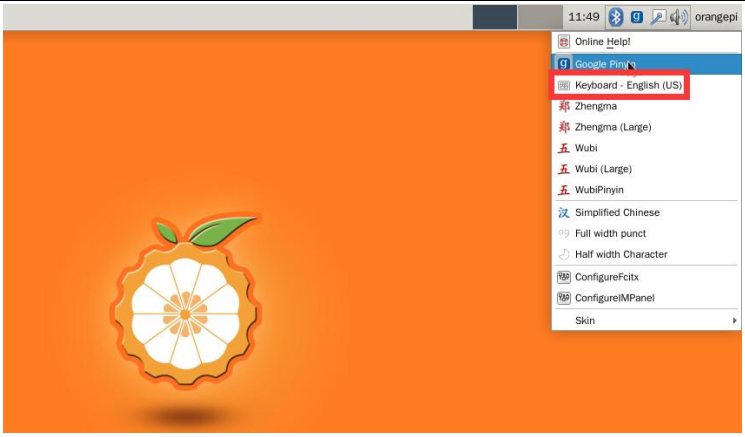
13) The Chinese input test is as follows



14) You can switch back to English input in the upper right corner of the desktop

a. First place the mouse cursor on the penguin position as shown in the figure below

b. Then click the **right** mouse button to see the following options, and then select **Keyboard-English (US)** to switch back to English input

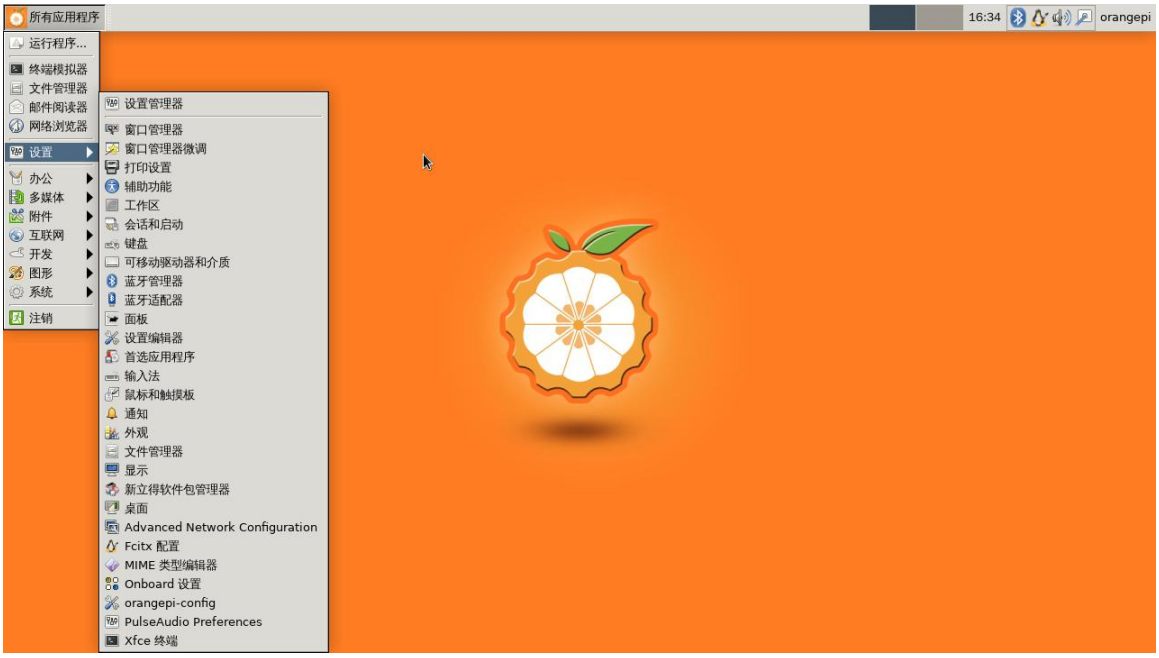


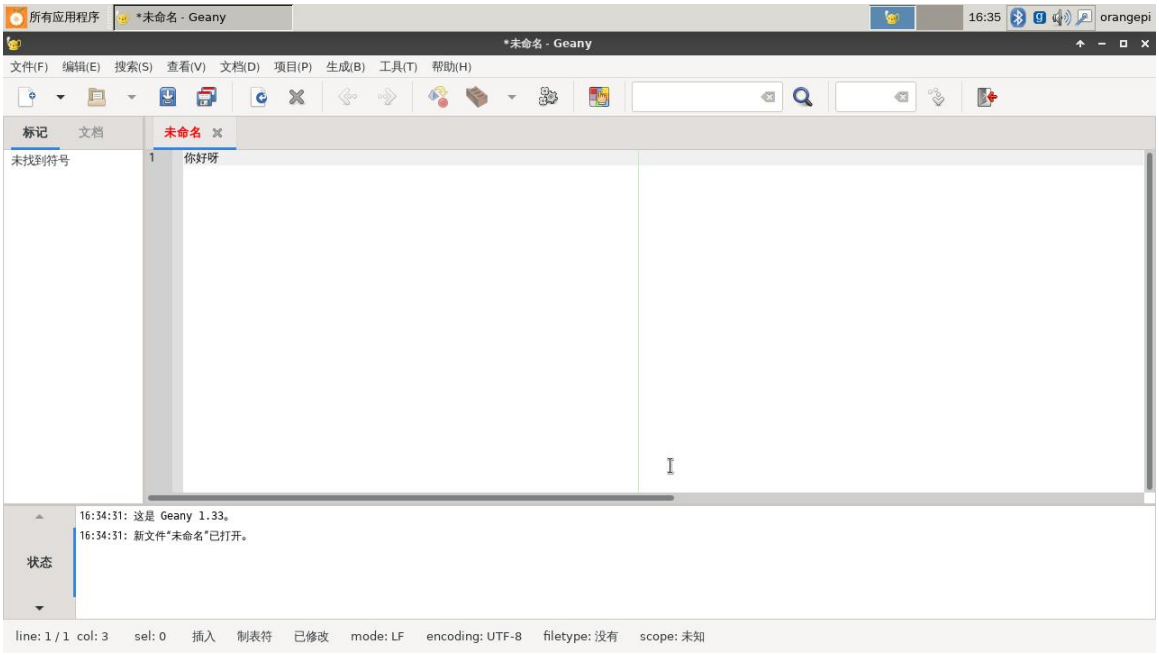
15) At this point, you can switch between Chinese and English input methods through the **Ctrl+Space** shortcut key

16) If you need the entire system to be displayed in Chinese, you can set the variables in **/etc/default/locale** to **zh_CN.UTF-8**

```
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
```

17) Then **restart the system** to see that the system is displayed in Chinese





3.27. View the chipid of the H616 chip

The command to view the chipid of the h616 chip is as follows, the chipid of each chip is different, so you can use the chipid to distinguish multiple development boards

```
orangepi@orangepi:~$ cat /sys/class/sunxi_info/sys_info | grep "chipid"
sunxi_chipid            : 32c05000dc00480801411a64298e1dce
```

3.28. The method of modifying the pictures displayed in the startup phase of the Linux 4.9 system

1) The following picture will be displayed on the HDMI display during the U-boot startup phase of the Linux 4.9 system

2) The location of this picture in the linux system is as follows

```
orangepi@orangepi:~$ ls /boot/boot.bmp
/boot/boot.bmp
```

3) The location in the source code of orangepi-build is as follows. When compiling the linux system, the script in orangepi-build will copy orangepi-u-boot.bmp to the **/boot** directory of the final linux system and rename it to **boot.bmp**

orangepi-build/external/packages/blobs/splash/orangepi-u-boot.bmp

4) The relevant properties of the boot.bmp picture are as follows. If you need to replace the boot.bmp, you only need to make the corresponding picture according to the following property values, and then replace the boot.bmp in the linux system in the TF card. If it is To compile the linux system by yourself, you only need to replace orangepi-u-boot.bmp in orangepi-build

3.29. Usage of TF card storage space and memory after Linux system starts

1) For the server version image, the storage space and memory usage viewed after the system is started are as follows

```
root@orangepizero2:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            424M     0  424M   0% /dev
tmpfs            99M  3.2M   96M   4% /run
/dev/mmcblk1p1   15G  1.2G   14G   8% /
tmpfs           493M     0  493M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           493M     0  493M   0% /sys/fs/cgroup
tmpfs           493M  4.0K  493M   1% /tmp
/dev/zram0       49M  1.4M   44M   4% /var/log
tmpfs            99M     0   99M   0% /run/user/0
root@orangepizero2:~#
root@orangepizero2:~# free -h
              total        used        free      shared  buff/cache   available
Mem:          984Mi       111Mi       768Mi       3.0Mi       104Mi       803Mi
Swap:         492Mi          0B       492Mi
root@orangepizero2:~#
root@orangepizero2:~# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.3 LTS
Release:        20.04
Codename:       focal
root@orangepizero2:~#
root@orangepizero2:~# uname -r
5.13.0-sun50iw9
root@orangepizero2:~#
```

2) For the desktop version image, the storage space and memory usage viewed after the system is started are as follows

```
root@orangepizero2:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            424M     0  424M   0% /dev
tmpfs            99M  3.1M   96M   4% /run
/dev/mmcblk1p1   15G  2.1G   13G  15% /
tmpfs           493M     0  493M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           493M     0  493M   0% /sys/fs/cgroup
tmpfs           493M   12K  493M   1% /tmp
/dev/zram0       49M  1.9M   44M   5% /var/log
tmpfs            99M   12K   99M   1% /run/user/1000
tmpfs            99M     0   99M   0% /run/user/0
root@orangepizero2:~#
root@orangepizero2:~# free -h
              total        used        free      shared  buff/cache   available
Mem:           984M        273M        436M        3.6M        273M        635M
Swap:          492M          0B        492M
root@orangepizero2:~#
root@orangepizero2:~# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.5 LTS
Release:        18.04
Codename:       bionic
root@orangepizero2:~#
root@orangepizero2:~# uname -r
5.13.0-sun50iw9
root@orangepizero2:~#
```

3.30. The method to install Klipper firmware host computer using Kiauh

Klipper is a 3D printer firmware. It can combine the functionality of the Orange Pi development board with one or more microcontrollers. For more information about Klipper, please refer to the official documentation of Klipper.

Kiauh is the abbreviation of Klipper Installation And Update Helper, which provides a very intuitive selection interface, and can complete the installation of Klipper and other software through simple selection. For more information about Kiauh, please refer to the README document in the Kiauh source code.

This section only demonstrates the process of using Kiauh to install the Klipper firmware host computer in the Linux system of the Orange Pi development board, and does not involve how to use the microcontroller or 3D printer.

Please make sure that the Linux system used by the development board is Ubuntu Focal or Debian Buster. Ubuntu Bionic will have problems because the default Python3 version does not meet the requirements.

If there are no special requirements, it is recommended to use the following images, which will encounter the least problems:

If you have never used a Linux system, before installing Klipper, please read the contents of the linux command format description section of this manual carefully, and figure out which commands you really need to enter before starting the installation.

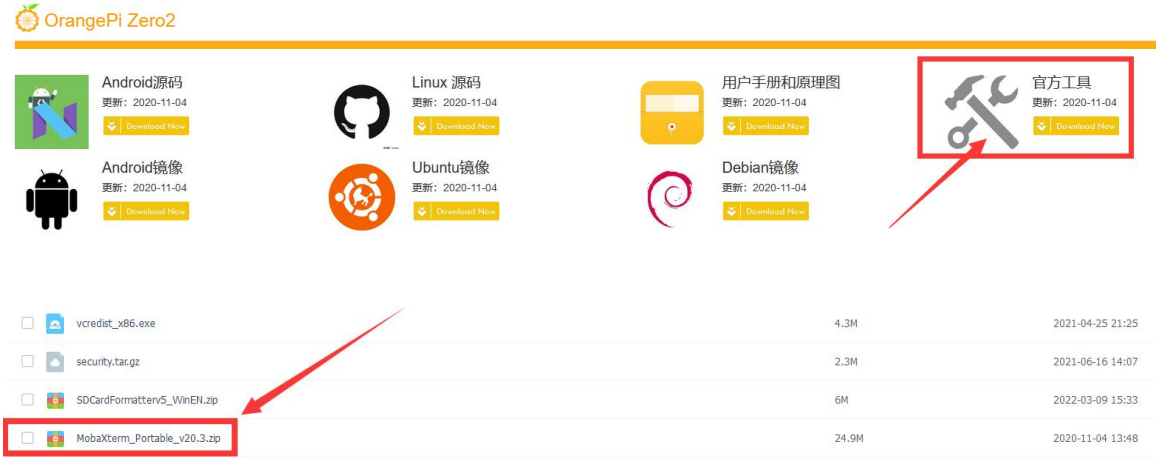
The default apt software source of the Linux system provided by Orange Pi has been set to Tsinghua source, so there is no need to replace the software source.

1) First, please make sure that the Linux system of the development board has been connected to WIFI or wired network, and then log in to the Linux system remotely through ssh. In Windows system, it is recommended to use the software MobaXterm to remotely log in to the Linux system of the development board.

a. First download MobaXterm, this software can be downloaded in the **official tool**

to

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



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

c. Then create a new ssh session in MobaXterm

a) Open **Session**

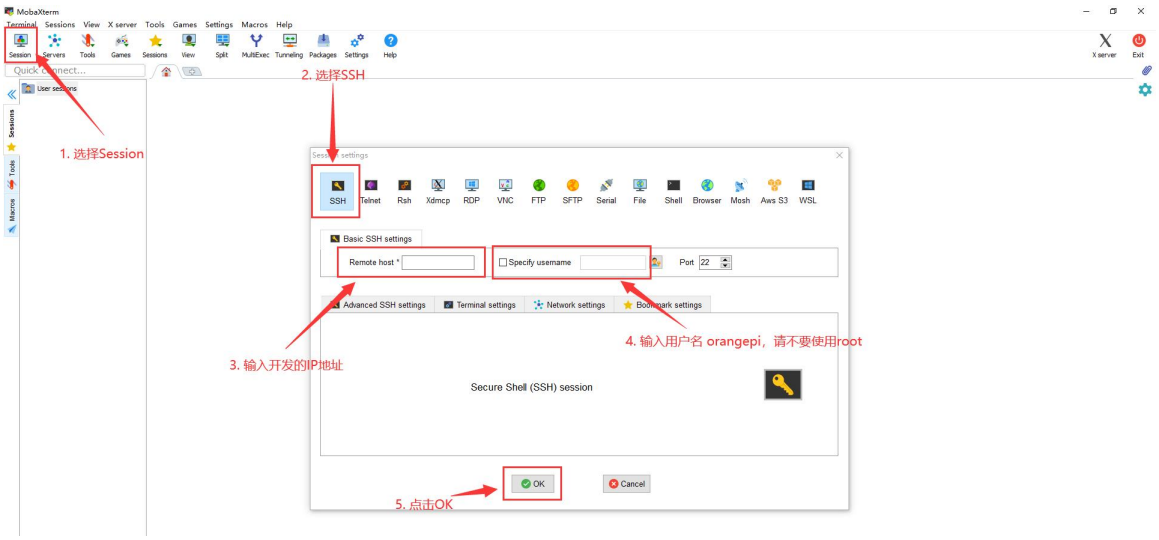
b) Then select **SSH** in **Session Setting**

c) Then enter the IP address of the development board in **Remote host**

d) Then enter the username **orangepi** of the Linux system of the development board in **Specify username,Please do not use root user**

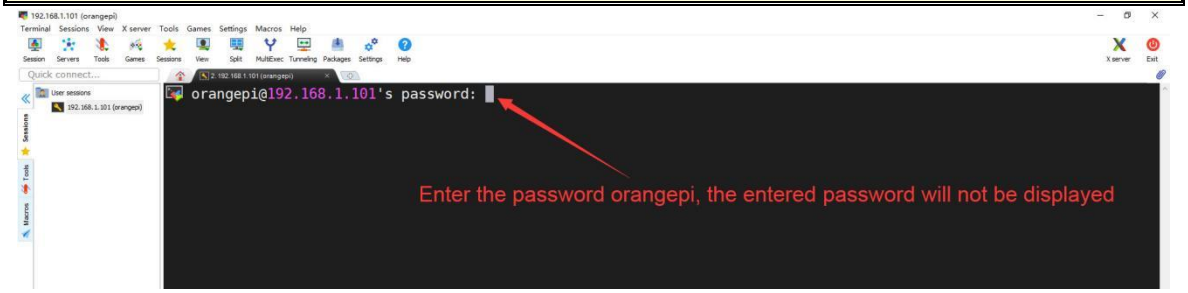
e) Finally click **OK**

Kiauh does not support the root user, so please use the orangepi user to log in to the Linux system.

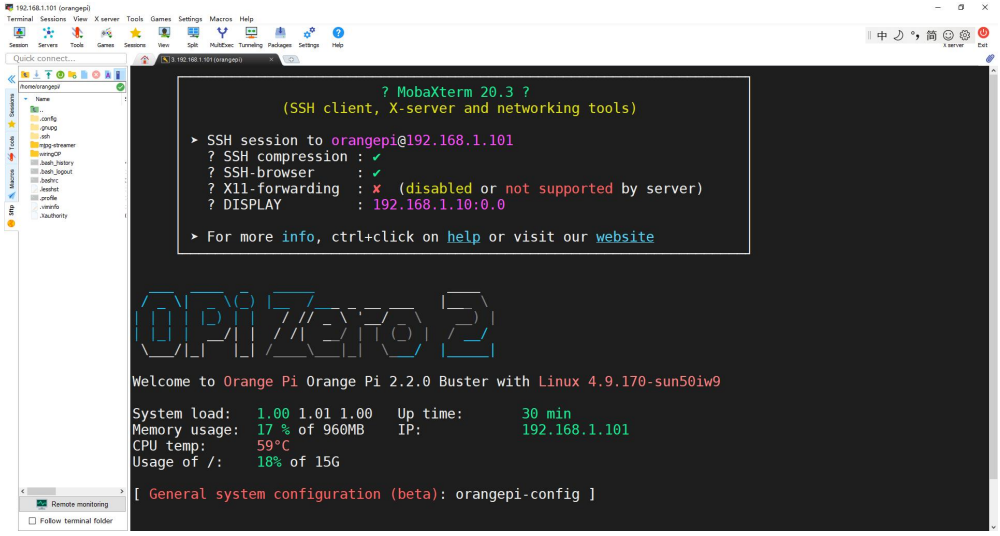


d. Then you will be prompted to enter a password. The default password of the orangepi user in the Linux system of the development board is **orangepi**

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



e. The display after successfully logging in to the system is as shown below



2) Then download the source code of Kiauh

Here are two ways to download the kiauh source code:

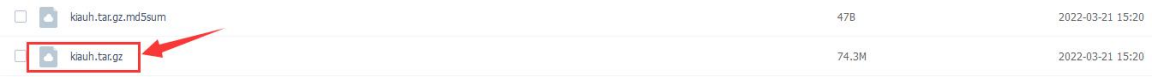
The first is to download the kiauh source code compressed package provided by Orange Pi from Google Drive for testing. The kiauh source code compression package provided by Orange Pi solves the problem of installation failure due to inaccessibility of github, and also fixes the problem of KlipperScreen installation failure.

The second is to download from kiauh's github repository. However, using this method to download the source code, if it does not solve the problem of accessing github from the development board linux system, it is basically difficult to install klipper successfully, and the installation speed is extremely slow.

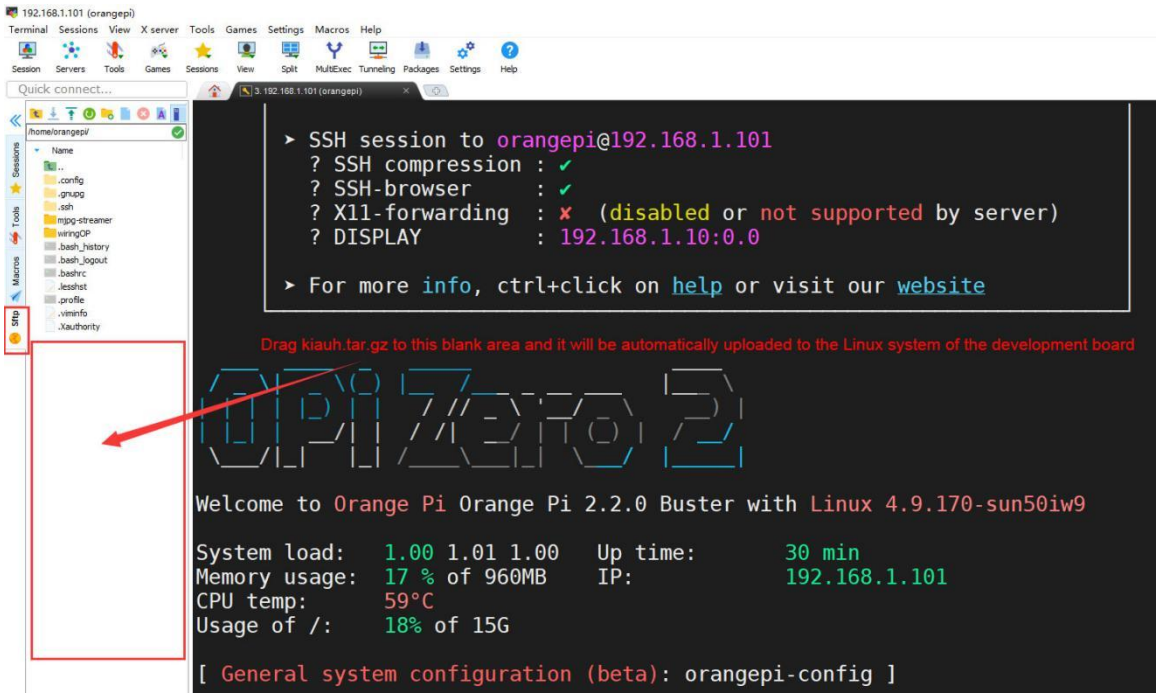
a. The first: the method of downloading the source code compressed package of kiauh provided by Orange Pi from Google Drive（**Recommended method**）

a) From the Google Drive link below, you can download the kiauh source code compressed package provided by Orange Pi. You can see it in the **kiauh_klipper** folder

https://drive.google.com/drive/folders/1fie2GcF-6zfWMqBDufnLUV3B9qMnbUkO?usp=sharing



b) After downloading the **kiauh.tar.gz** compressed package, first upload **kiauh.tar.gz** to the **/home/orangepi** directory of the Linux system of the development board



c) Then extract **kiauh.tar.gz** using the following command

```
orangepi@orangepi:~$ tar zxf kiauh.tar.gz
```

d) The difference between the compressed package of Kiauh source code provided by Orange Pi and the Kiauh source code downloaded from github is as follows:

If you don't understand what you are saying about the modification instructions of the source code in the following part, please ignore it directly, and it will not affect the subsequent installation.

i. There is a github_src folder in the Kiauh source code provided by i.Orange Pi, which caches the source code and compressed package required during the installation of Klipper and other software, as well as the configuration files required for the use of the USB camera

```
orangepi@orangepi:~/kiauh$ ls github_src/
DuetWebControl-SD.zip  fluidd.zip  klipper  mainsail.zip  moonraker
pgcode  webcam.txt  dwc2-for-klipper-socket  KlipperScreen mjpg-streamer
moonraker-telegram-bot  webcamd
```

ii. Modified part of the kiauh script code, you can see all the modified files through the **git status .** command

```
orangepi@orangepi:~/kiauh$ git status .
```

iii. The functions implemented by the modified code mainly include:

i) Disable the automatic update of Kiauh code. Every time **kiauh.sh** runs, it will automatically synchronize the code with github. If there is a problem with the network, it will cause it to get stuck.

ii) Change the installation source of pip to **https://pypi.tuna.tsinghua.edu.cn/simple** to speed up the installation of the python library

iii) Since the source code and other files that need to be downloaded from github are cached in the **github_src** folder, part of the code in the Kiauh script that needs to download the source code from github is blocked, and modified to use the cp command to copy from **github_src** to the corresponding location. In this way, there is no need to download the source code needed to install Klipper and other software from github through the network.

b. The second: the command to download the source code of kiauh from github is as follows（**If you have no experience in installing klipper in kiauh, you will not solve the network problem of github. Please do not use this method to download the source code of kiauh. It is difficult to install successfully.**）

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y git
orangepi@orangepi:~$ git clone https://github.com/th33xitus/kiauh.git
```

3) Enter the source code directory of Kiauh, you can see the files and folders it contains are as follows

a. kiauh.sh：kiauh's startup script

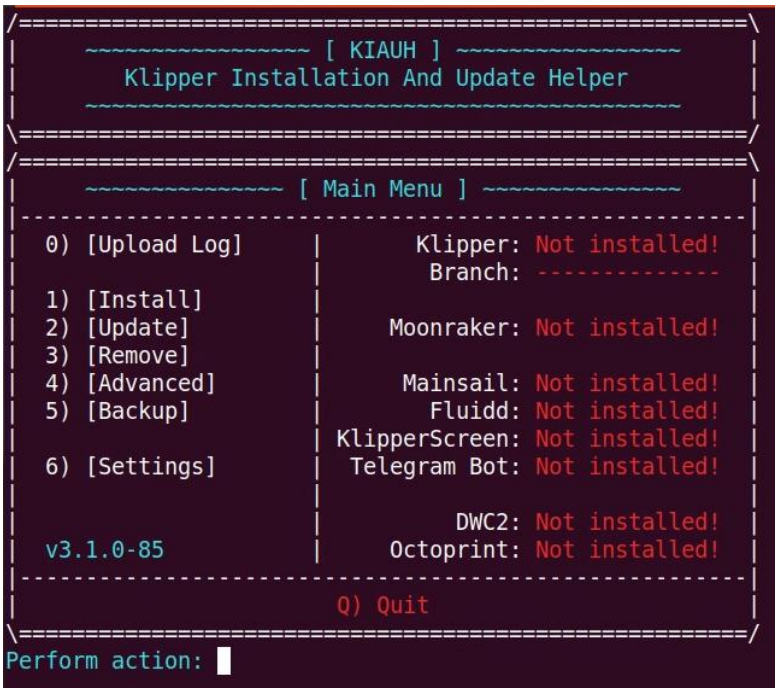
b. scripts：Contains installation scripts for software such as klipper

```
orangepi@orangepi:~$ cd kiauh/
orangepi@orangepi:~/kiauh$ ls
docs  github_src  kiauh.sh  LICENSE  README.md  resources  scripts
```

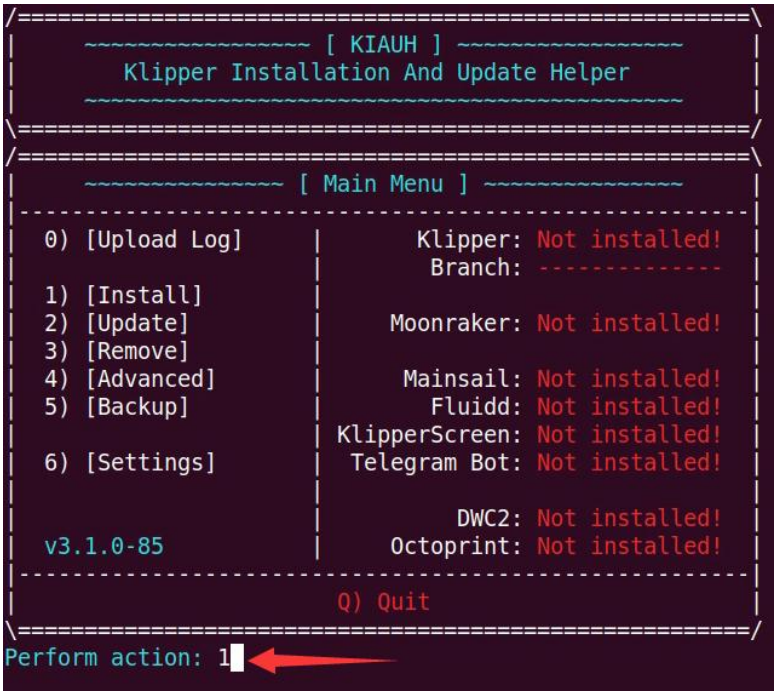
4) Then you can run the kiauh.sh script in the source directory of Kiauh

```
orangepi@orangepi:~/kiauh$ ./kiauh.sh
```

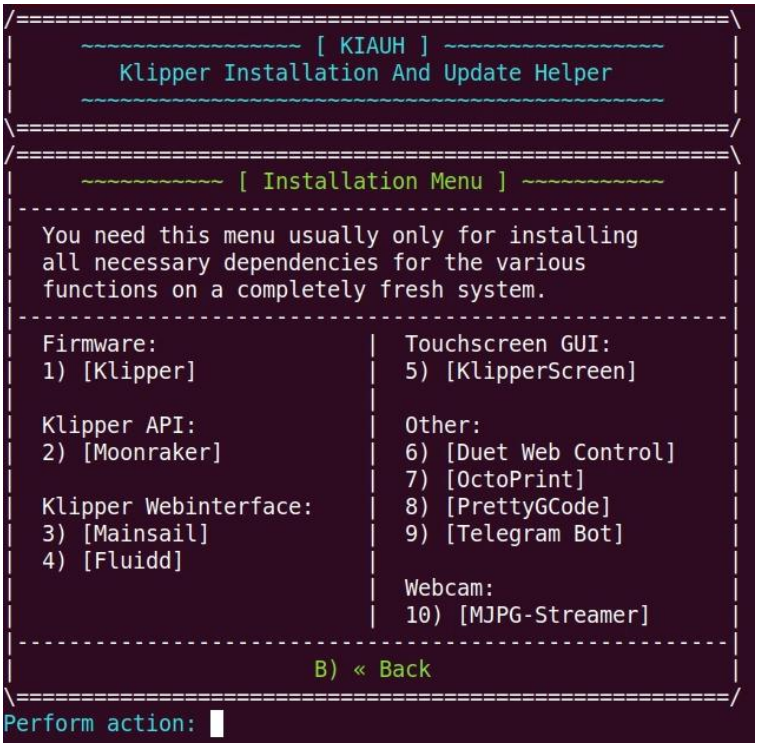
5) After running kiauh.sh, the following selection interface will pop up. You can see that Kiauh provides many operation options, and can also display the installation status of Klipper and other related software



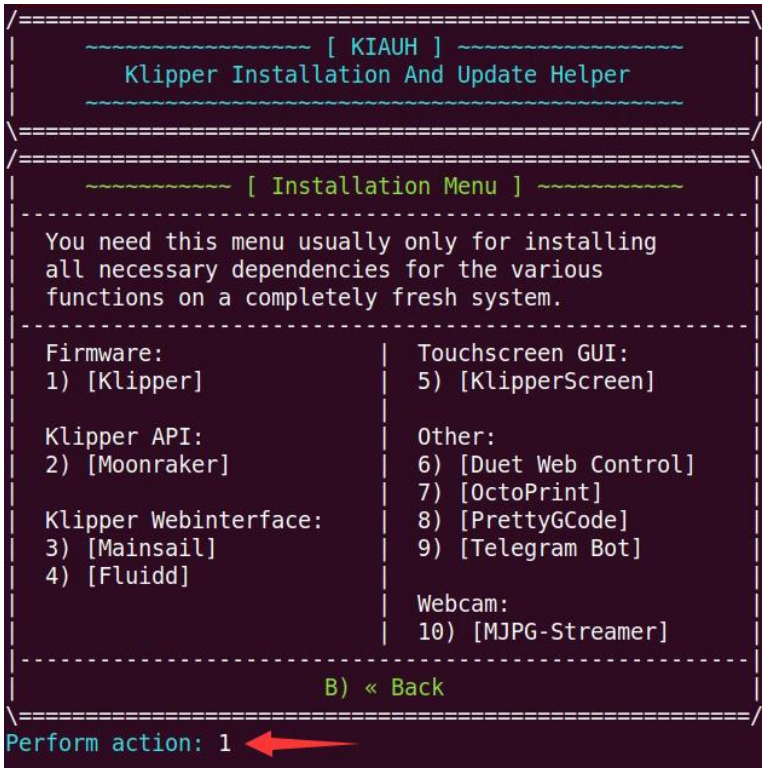
6) Then you can enter **1** after **Perform action**: and press Enter, select 1) **[Install]**



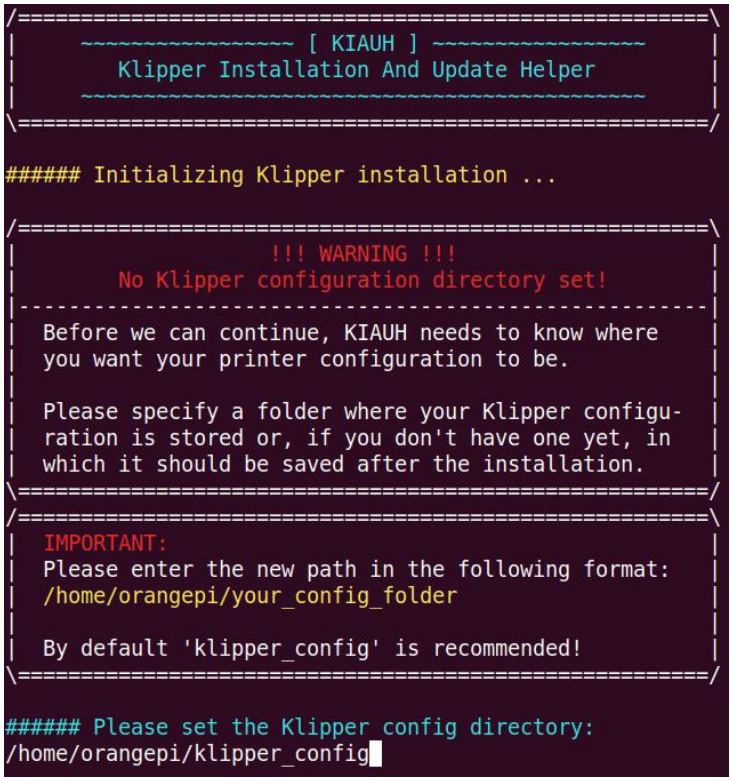
7) Then the following installation selection interface will pop up



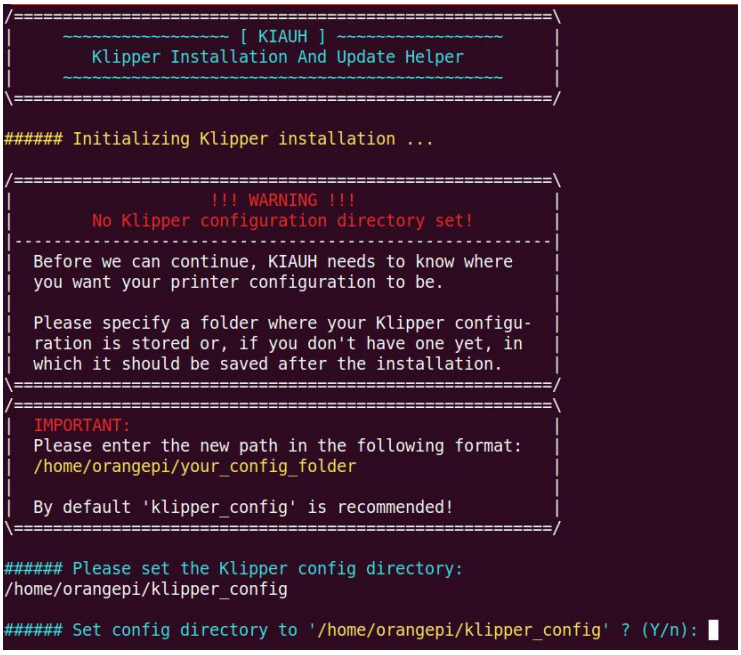
8) We first enter **1** after **Perform action**: and press Enter to start the process of installing Klipper



9) After installing Klipper, you will first be reminded to set the path of the folder where the Klipper configuration file is located. The default is **/home/orangepi/klipper_config.** If you do not need to modify it, just press Enter.



Then the following prompt message will pop up, the default is **Y**, just press Enter



10) Then you will be prompted to enter the password of the Linux system, the default is orangepi

```
###### Please set the Klipper config directory:
/home/orangepi/klipper_config

###### Set config directory to '/home/orangepi/klipper_config' ? (Y/n):
###### > Yes

###### Create KIAUH backup directory ...
>>>>>> Directory created!
>>>>>> No config directory found! Skipping backup ...


###### Directory set to '/home/orangepi/klipper_config'!
[sudo] password for orangepi:
```

Enter the password of the Linux system here: orangepi

11) Then you will be prompted to set the required number of Klipper instances, here we can enter **1**

```
###### Directory set to '/home/orangepi/klipper_config'!
[sudo] password for orangepi:

>>>>>> Config directory changed!
/=======================================================\
| Please select the number of Klipper instances to set  |
| up. The number of Klipper instances will determine    |
| the amount of printers you can run from this machine. |
|                                                       |
| WARNING: There is no limit on the number of instances |
| you can set up with this script.                      |
\=======================================================/
###### Number of Klipper instances to set up: 1
```

Then continue to press Enter to confirm

```
>>>>>> Config directory changed!
/=======================================================\
| Please select the number of Klipper instances to set  |
| up. The number of Klipper instances will determine    |
| the amount of printers you can run from this machine. |
|                                                       |
| WARNING: There is no limit on the number of instances |
| you can set up with this script.                      |
\=======================================================/
###### Number of Klipper instances to set up: 1

###### Install 1 instance(s)? (Y/n):
```

12) Then the Klipper installation process will officially begin

a. The script in Kiauh will first automatically download the source code of Klipper from GitHub. This step is very easy to download failure due to network problems. Please pay special attention to the output log and the following figure, and no error is reported.（**Using the kiauh source code provided by Orange Pi will not have the process of downloading the code, so there will be no errors due to network problems.**）

```
###### Installing 1 Klipper instance(s) ...

###### Checking for the following dependencies:
● git
>>>>>> Dependencies already met! Continue...

###### Downloading Klipper ...
Cloning into 'klipper'...
remote: Enumerating objects: 28306, done.
remote: Counting objects: 100% (76/76), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 28306 (delta 49), reused 67 (delta 44), pack-reused 28230
Receiving objects: 100% (28306/28306), 20.32 MiB | 396.00 KiB/s, done.
Resolving deltas: 100% (21377/21377), done.
Updating files: 100% (1526/1526), done.

###### Download complete!
```

b. Then the dependency package will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons

```
###### Installing packages...
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'python-dev-is-python2' instead of 'python-dev'
Note, selecting 'libusb-1.0-0' for regex 'libusb-1.0'
Note, selecting 'libusb-1.0-0-dev' for regex 'libusb-1.0'
Note, selecting 'libusb-1.0-doc' for regex 'libusb-1.0'
libusb-1.0-0 is already the newest version (2:1.0.23-2build1).
libusb-1.0-0 set to manually installed.
build-essential is already the newest version (12.8ubuntu1.1).
```

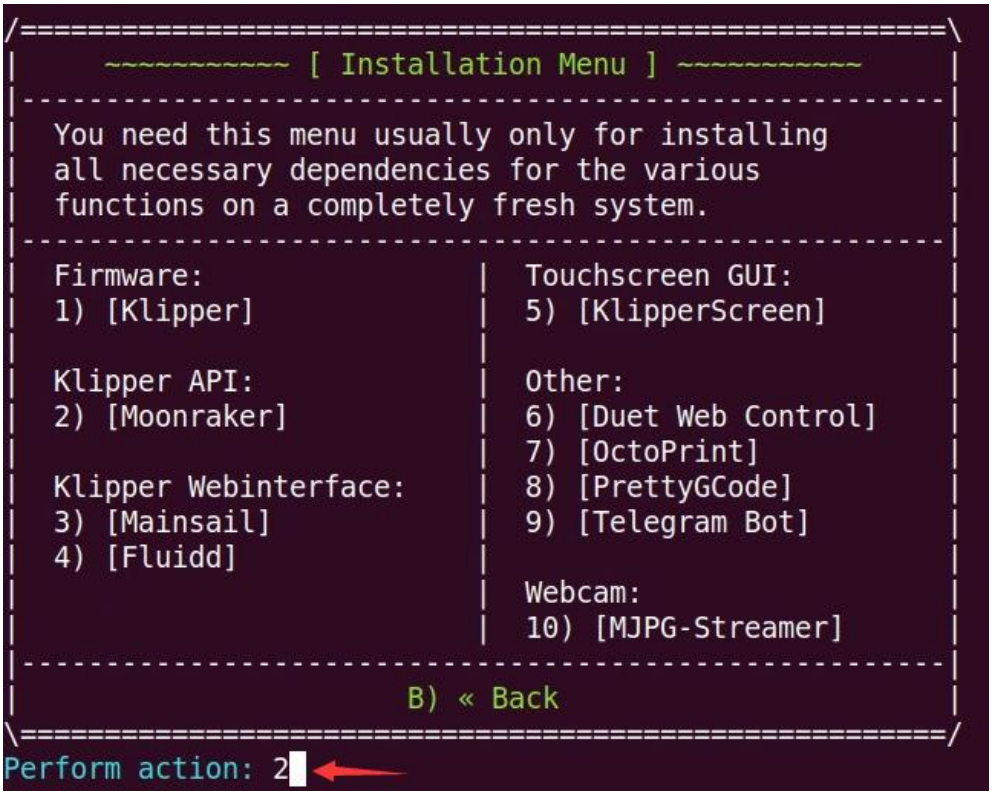
c. Then the virtual environment of Python will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons

```
###### Installing python virtual environment...
created virtual environment CPython2.7.18.final.0-64 in 1637ms
  creator CPython2Posix(dest=/home/orangepi/klippy-env, clear=False, global=
  seeder FromAppData(download=False, pip=latest, setuptools=latest, wheel=la
eed-app-data/v1.0.1.debian.1)
  activators BashActivator,CShellActivator,FishActivator,PowerShellActivator
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Pl
l drop support for Python 2.7. More details about Python 2 support in pip, c
Collecting cffi==1.14.6
  Downloading cffi-1.14.6.tar.gz (475 kB)
     |████████████████████████████████| 475 kB 401 kB/s
Collecting pyserial==3.4
  Downloading pyserial-3.4-py2.py3-none-any.whl (193 kB)
     |████████████████████████████████| 193 kB 328 kB/s
```

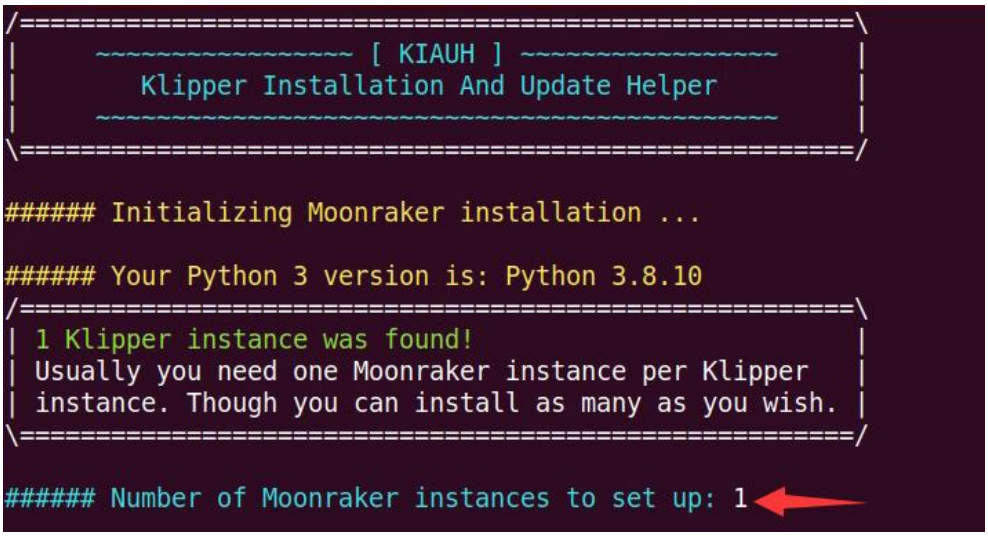
d. After the installation is complete, the following information will be displayed, and it will automatically return to the Kiauh installation interface

```
###### Creating Klipper Service  ...
Created symlink /etc/systemd/system/multi-user.target.wants/klipper.service → /etc/systemd/system/klipper.service.
>>>>>> Single Klipper instance created!

###### Launching Klipper instance ...

#######################################################
 Klipper has been set up!
#######################################################

/=======================================================\
|     ~~~~~~~~~~ [ Installation Menu ] ~~~~~~~~~~      |
|-------------------------------------------------------|
|  You need this menu usually only for installing       |
|  all necessary dependencies for the various           |
|  functions on a completely fresh system.              |
|-------------------------------------------------------|
|  Firmware:                |  Touchscreen GUI:         |
|  1) [Klipper]             |  5) [KlipperScreen]       |
|                           |                           |
|  Klipper API:             |  Other:                   |
|  2) [Moonraker]           |  6) [Duet Web Control]    |
|                           |  7) [OctoPrint]           |
|  Klipper Webinterface:    |  8) [PrettyGCode]         |
|  3) [Mainsail]            |  9) [Telegram Bot]        |
|  4) [Fluidd]              |                           |
|                           |  Webcam:                  |
|                           |  10) [MJPG-Streamer]      |
|-------------------------------------------------------|
|                    B) « Back                          |
\=======================================================/
Perform action:
```

13) Then start to install Klipper API - Moonraker, enter 2 after **Perform action**: and press Enter

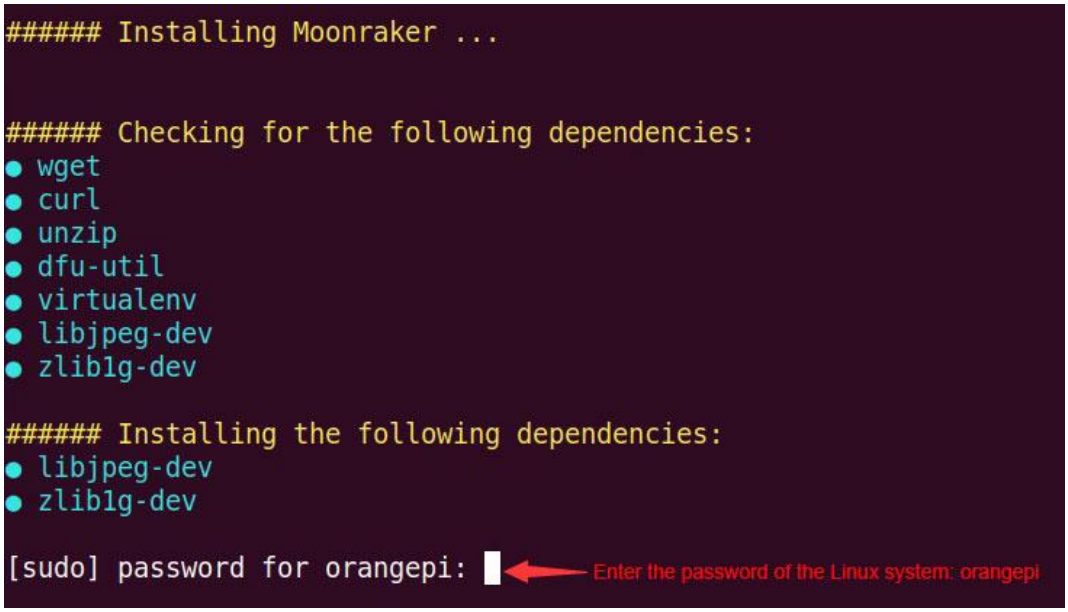


14) Then you will be prompted to set the number of Moonraker instances, which needs to correspond to the number of Klipper instances set earlier. Here we set it to **1** and press Enter.



Then continue to press Enter to confirm

Then enter the default password orangepi of the Linux system to start the official installation process



15) The specific installation process of Moonraker is as follows

a. The script in Kiauh will automatically install the dependency package first, please make sure that the download process will not fail due to network reasons

```
###### Checking for the following dependencies:
● wget
● curl
● unzip
● dfu-util
● virtualenv
● libjpeg-dev
● zlib1g-dev

###### Installing the following dependencies:
● libjpeg-dev
● zlib1g-dev

[sudo] password for orangepi:
Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libjpeg-turbo8-dev libjpeg8-dev
The following NEW packages will be installed:
  libjpeg-dev libjpeg-turbo8-dev libjpeg8-dev zlib1g-dev
0 upgraded, 4 newly installed, 0 to remove and 124 not upgraded.
```

b. Then the source code of Moonraker will be automatically downloaded from GitHub. This step is very easy to download failure due to network problems. Please pay special attention to the following output log and the following figure, and no error is reported.（**Using the kiauh source code provided by Orange Pi will not have the process of downloading the code, so there will be no errors due to network problems.**）

```
###### Downloading Moonraker ...
Cloning into 'moonraker'...
remote: Enumerating objects: 5413, done.
remote: Counting objects: 100% (188/188), done.
remote: Compressing objects: 100% (86/86), done.
remote: Total 5413 (delta 118), reused 144 (delta 102), pack-reused 5225
Receiving objects: 100% (5413/5413), 1.65 MiB | 1.38 MiB/s, done.
Resolving deltas: 100% (3955/3955), done.
>>>>>> Download complete!
```

c. The download error is as follows

```
###### Downloading Moonraker ...
Cloning into 'moonraker'...
fatal: unable to access 'https://github.com/Arksine/moonraker.git/': GnuTLS recv error (-110): The TLS connection was non-properly terminated.
>>>>>> Download complete!
```

d. Then it will continue to install the dependent package, please make sure that the process of downloading and installing will not fail due to network reasons

```
###### Installing dependencies ...

###### Reading dependencies...
python3-virtualenv
python3-dev
libopenjp2-7
python3-libgpiod
curl
libcurl4-openssl-dev
libssl-dev
liblmdb-dev
libsodium-dev
zlib1g-dev
libjpeg-dev

###### Running apt-get update...
[sudo] password for orangepi:
Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease
Reading package lists... Done

###### Installing packages...
```

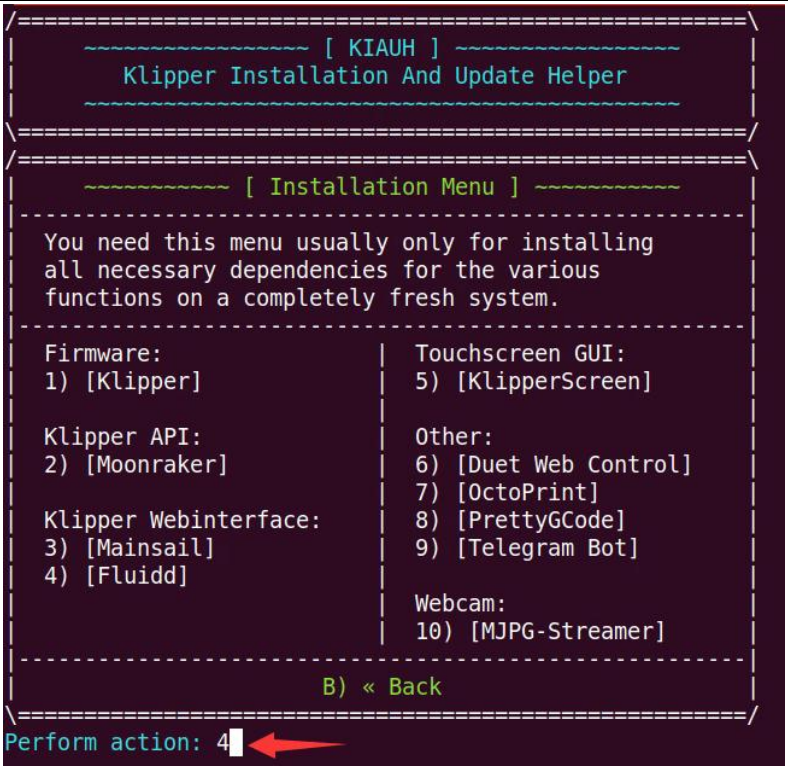
e. Then the virtual environment of Python will be installed automatically, please make sure that the download and installation process will not go wrong due to network reasons

```
###### Installing python virtual environment...
created virtual environment CPython3.8.10.final.0-64 in 1075ms
  creator CPython3Posix(dest=/home/orangepi/moonraker-env, clear=False, global=False)
  seeder FromAppData(download=False, pip=latest, setuptools=latest, wheel=latest, pkg_resources
eed-app-data/v1.0.1.debian.1)
  activators BashActivator,CShellActivator,FishActivator,PowerShellActivator,PythonActivator,Xo
Collecting tornado==6.1.0
  Downloading tornado-6.1-cp38-cp38-manylinux2014_aarch64.whl (427 kB)
     |████████████████████████████████| 427 kB 609 kB/s
Collecting pyserial==3.4
  Using cached pyserial-3.4-py2.py3-none-any.whl (193 kB)
Collecting pillow==8.3.2
  Downloading Pillow-8.3.2-cp38-cp38-manylinux_2_17_aarch64.manylinux2014_aarch64.whl (3.0 MB)
     |████████████████████████████████| 3.0 MB 7.9 MB/s
```

f. After the installation is complete, the following information will be displayed, and it will automatically return to the Kiauh installation interface

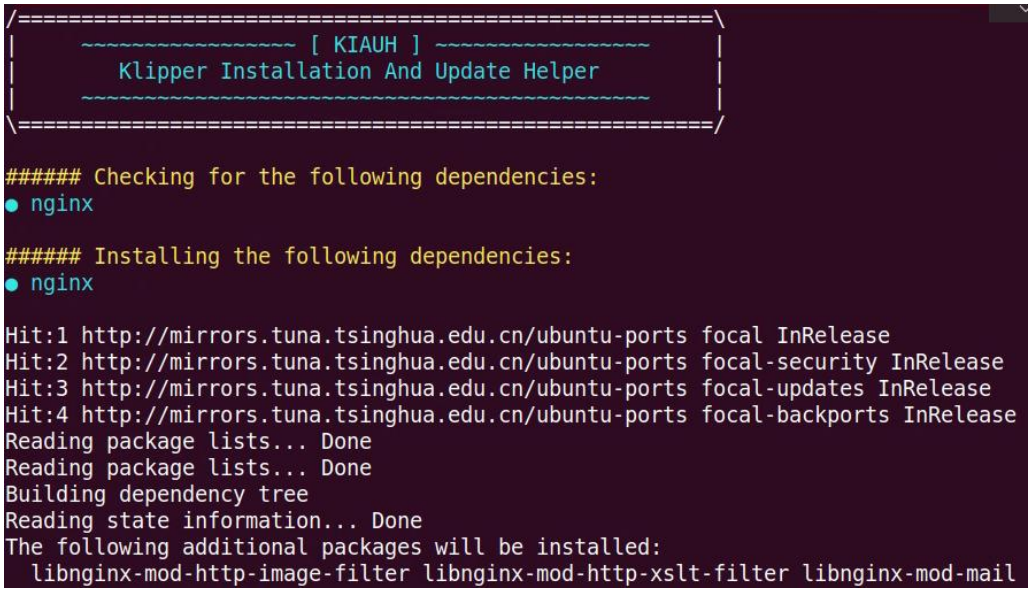
```
###### Enabling moonraker.service ...
Created symlink /etc/systemd/system/multi-user.target.wants/moonraker.service → /etc/systemd/system/moonraker.service.
>>>>>> moonraker.service enabled!
>>>>>> Single Moonraker instance created!

###### Starting moonraker.service ...
>>>>>> moonraker.service started!

#########################################################
 Moonraker has been set up!
#########################################################

      ● Instance 1: 192.168.1.11:7125
```

16) Then start to install Klipper's web interface, the default options are **3) [Mainsail]** and **4) [Fluidd]**, etc. Here we test the process of installing **4) [Fluidd]**, to install **4) [Fluidd]** in Perform action: Enter **4** after that and press Enter



17) Klipper's Web interface - the specific installation process of **Fluidd** is as follows

a. The script in Kiauh will automatically download and install the dependency package first, please make sure that the download process will not fail due to network reasons



b. Then it will prompt whether you need to install MJGP-Streamer to display the output video of the camera. If you need to press Enter directly, if not, you can

enter **n** in the place shown in the figure below and press Enter。

```
###### Initializing Fluidd installation ...

/=======================================================\
|  Install MJGP-Streamer for webcam support?            |
\=======================================================/
###### Install MJPG-Streamer? (Y/n):
```

c. Here we choose to install MJGP-Streamer, and then the following information will be prompted, just press Enter

```
/=======================================================\
|  Install MJGP-Streamer for webcam support?            |
\=======================================================/
###### Install MJPG-Streamer? (Y/n):
###### > Yes

/=======================================================\
| It is recommended to have some important macros set   |
| up in your printer configuration to have Fluidd       |
| fully functional and working.                         |
|                                                       |
| Those macros are:                                     |
| ● [gcode_macro PAUSE]                                 |
| ● [gcode_macro RESUME]                                |
| ● [gcode_macro CANCEL_PRINT]                          |
|                                                       |
| If you already have these macros in your config file  |
| you can skip this step and choose 'no'.               |
| Otherwise you should consider to answer with 'yes' to |
| add the recommended example macros to your config.    |
\=======================================================/
###### Add the recommended macros? (Y/n):
```

d. Then it will download **fluidd.zip** from GitHub and decompress and install it. This step is easy to fail to download due to network problems. Please pay special attention to the output log that is consistent with the figure below, and no error is reported.（**Using the kiauh source code provided by Orange Pi will not have the process of downloading fluidd.zip, so there will be no errors due to network problems.**）

```
###### Downloading Fluidd ...
--              13:33:33--  https://github.com/fluidd-core/fluidd/releases/download/v1.16.2/fluidd.zip
Resolving github.com (github.com)... 140.82.112.4
Connecting to github.com (github.com)|140.82.112.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/295836951/5d248200-e0dd-11eb-9d39-b4a76797fd0a?X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220101%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220101T133334Z&X-Amz-Expires=300&X-Amz-Signature=76e8c06f5c7a375dc51fd370
56c9b0f5c07b37e6266d2334b1a412361e2a74&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=295836951&response-content-disposition=attachment%3B%20filename%3Dfluidd.
zip&response-content-type=application%2Foctet-stream [following]
--              13:33:34--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/295836951/5d248200-e0dd-11eb-9d39-b4a76797fd0a?X-Amz-Algorithm=AW
S4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220101%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220101T133334Z&X-Amz-Expires=300&X-Amz-Signature=76e8c06f5
c7a375dc51fd37056c9b0f5c07b37e6266d2334b1a412361e2a74&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=295836951&response-content-disposition=attachment%3B%20fil
ename%3Dfluidd.zip&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9518655 (9.1M) [application/octet-stream]
Saving to: 'fluidd.zip'

fluidd.zip                          100%[=============================================================================>]   9.08M  15.3KB/s    in 10m 54s

              13:44:30 (14.2 KB/s) - 'fluidd.zip' saved [9518655/9518655]

>>>>>> Download complete!

###### Extracting archive ...
>>>>>> Done!

###### Remove downloaded archive ...
>>>>>> Done!
```

e. Then the dependency package will be installed, please make sure that the download process will not fail due to network reasons

```
###### Checking for the following dependencies:
● git
● cmake
● build-essential
● imagemagick
● libv4l-dev
● ffmpeg
● libjpeg8-dev

###### Installing the following dependencies:
● cmake
● imagemagick
● libv4l-dev
● ffmpeg

Hit:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal InRelease
Hit:2 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-security InRelease
Hit:3 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-updates InRelease
Hit:4 http://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports focal-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
```

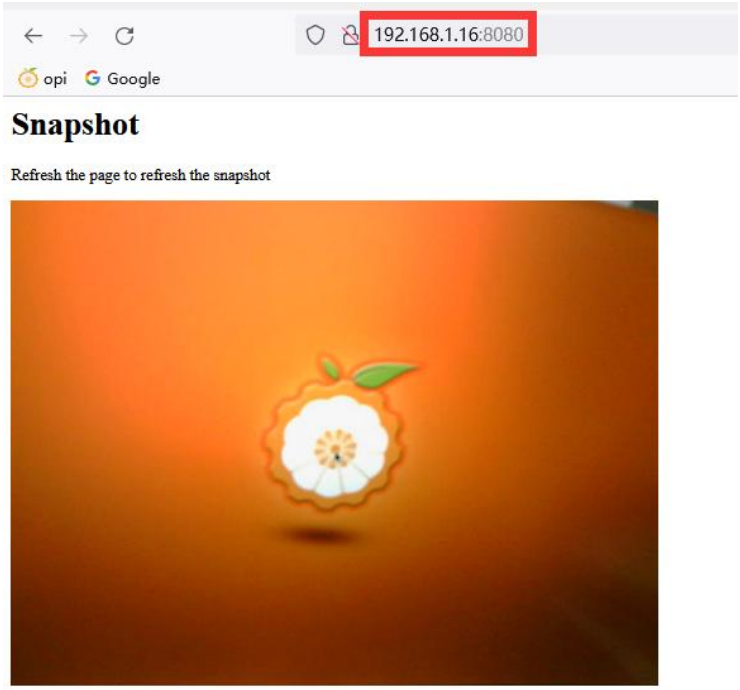
f. Then it will download, compile and install MJPG-Streamer. This step is very easy to download failure due to network problems. Please pay special attention to the output log and the following figure, and no error is reported.（**Using the kiauh source code provided by Orange Pi will not have the process of downloading the code, so there will be no errors due to network problems.**）

```
###### Downloading MJPG-Streamer ...
Cloning into 'mjpg-streamer'...
remote: Enumerating objects: 2964, done.
remote: Total 2964 (delta 0), reused 0 (delta 0), pack-reused 2964
Receiving objects: 100% (2964/2964), 3.48 MiB | 1.96 MiB/s, done.
Resolving deltas: 100% (1885/1885), done.
>>>>>> Download complete!

###### Compiling MJPG-Streamer ...
[ -d _build ] || mkdir _build
[ -f _build/Makefile ] || (cd _build && cmake  -DCMAKE_BUILD_TYPE=Release ..)
-- The C compiler identification is GNU 9.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
```

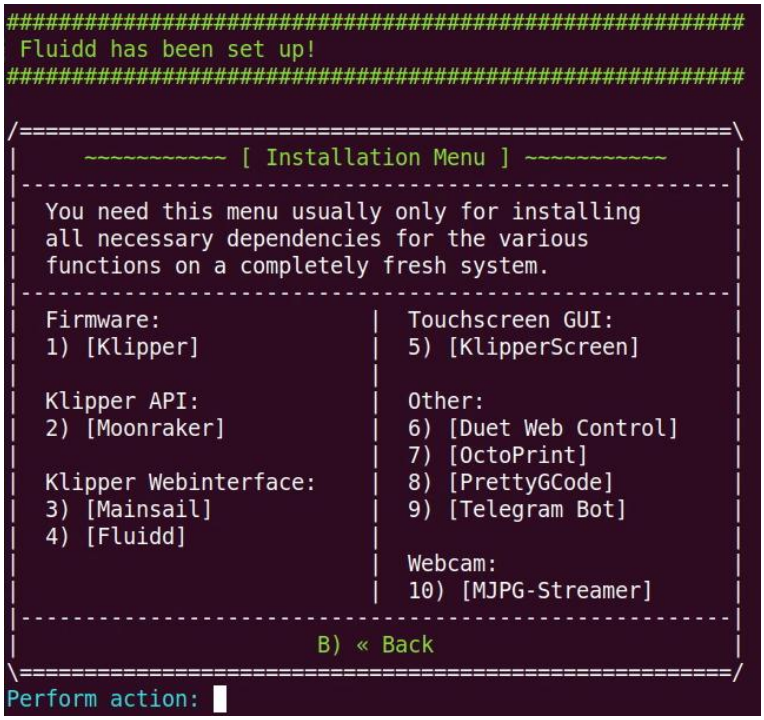
g. The access address of g.MJPG-Streamer is as follows. If the development board is connected to a USB camera, enter the following address in the browser to see the video output of the USB camera.

```
#########################################################
 MJPG-Streamer has been set up!
#########################################################

 ● Webcam URL: http://192.168.1.11:8080/?action=stream
 ● Webcam URL: http://192.168.1.11/webcam/?action=stream
```

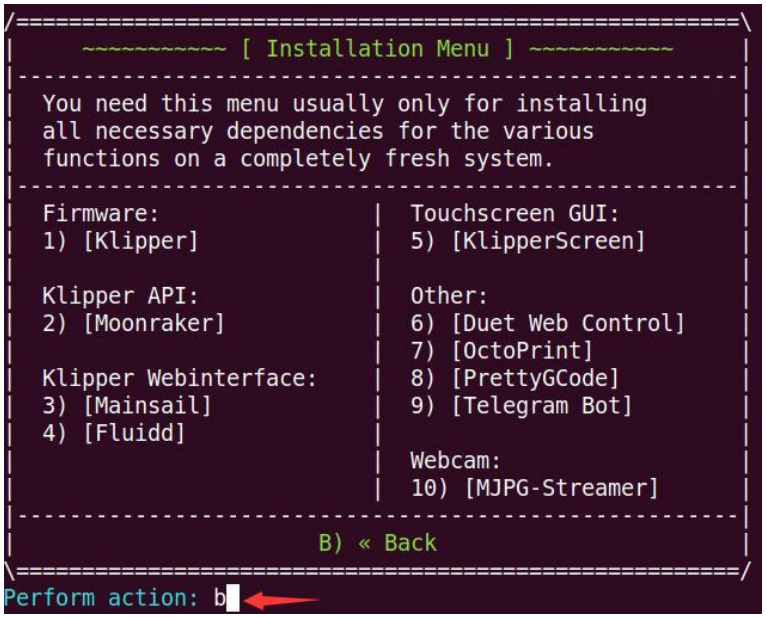
The output of the camera looks like this

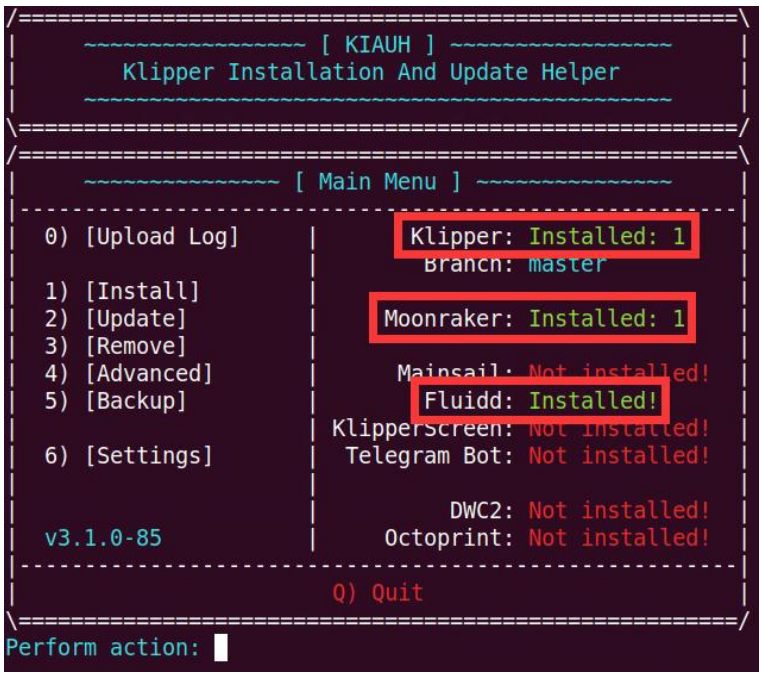


h. The output after installation is as shown below



18) Then enter **B** after **Perform action**: to return to the main interface of the installation

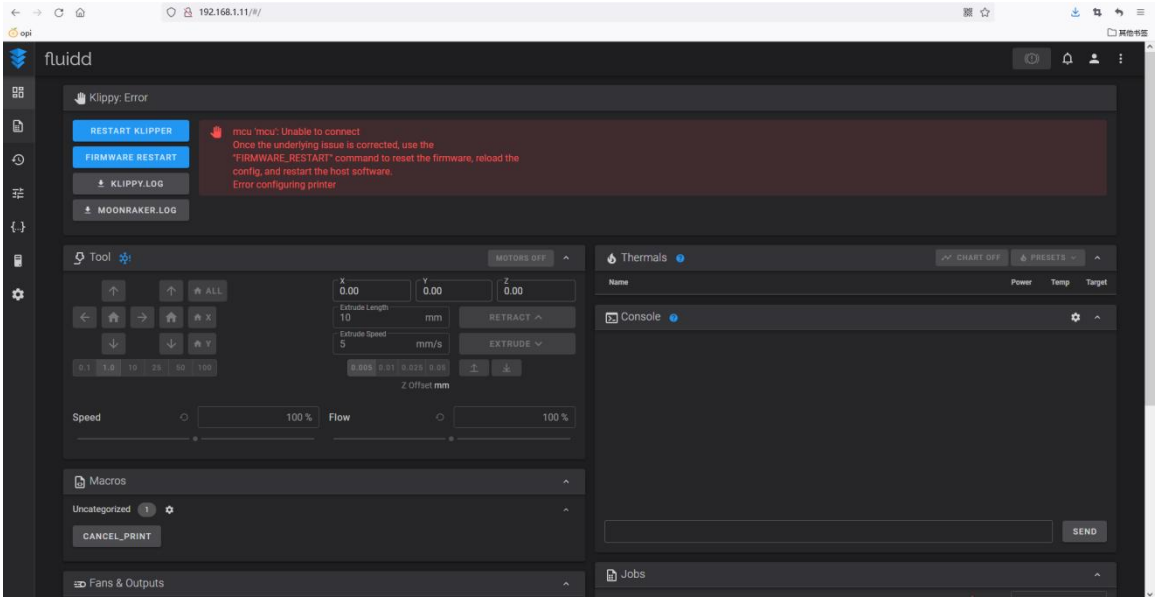




After returning to the main interface, you can see that Klipper, Moonraker and Fluidd are all displayed as Installed. However, it should be noted that even if Klipper, Moonraker and Fluidd are all displayed as Installed, it does not ensure that all software has been installed successfully. During the test, it was found that even after the code download failed due to network problems, when you return to the main interface, you can see that Klipper, Moonraker, and Fluidd are all displayed as

Installed. So the best way to ensure that the installation is no problem is to ensure that the output log of the installation process does not report an error that the source code download fails due to network problems.

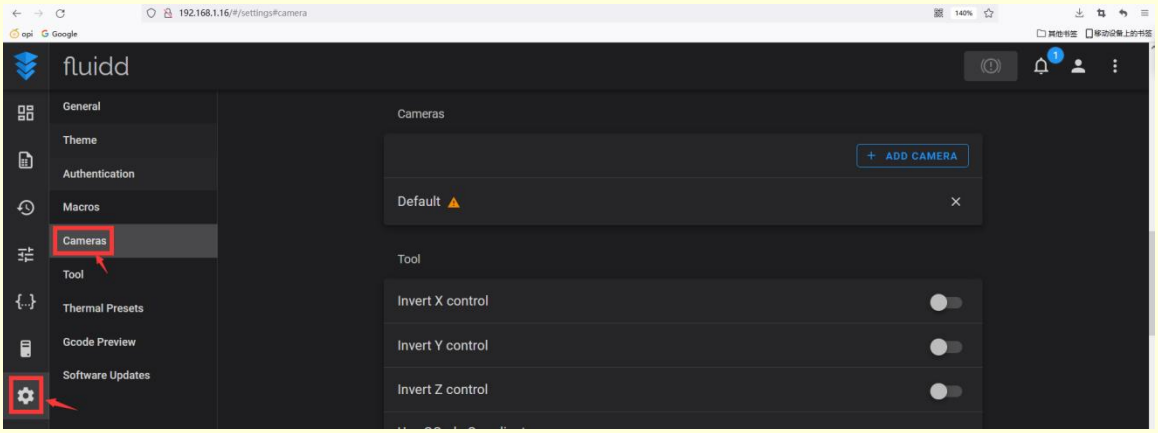
19) Finally, enter the IP address of the development board in the browser to see the web control interface of fluidd



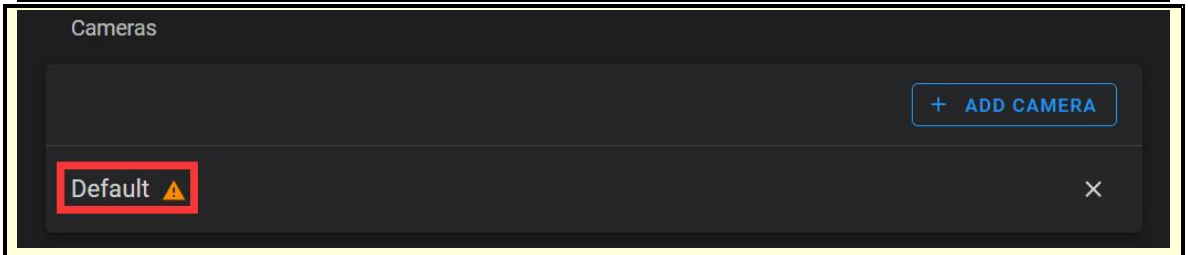
The steps to set the camera in the web control interface of fluidd are as follows:

1. First of all, please make sure that MJPG-Streamer has been installed normally, and you can open the camera in the browser to see the output screen of the camera

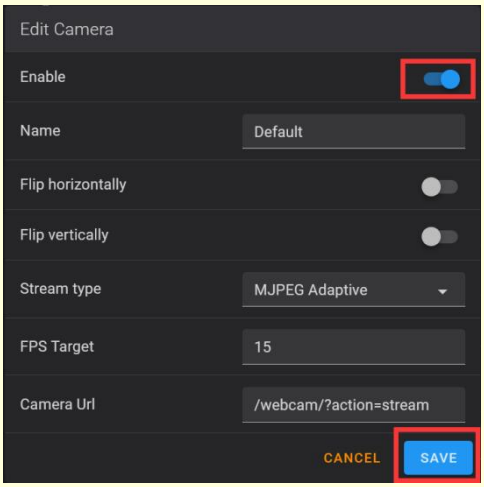
2. Then find the setting interface of the camera in Fluidd



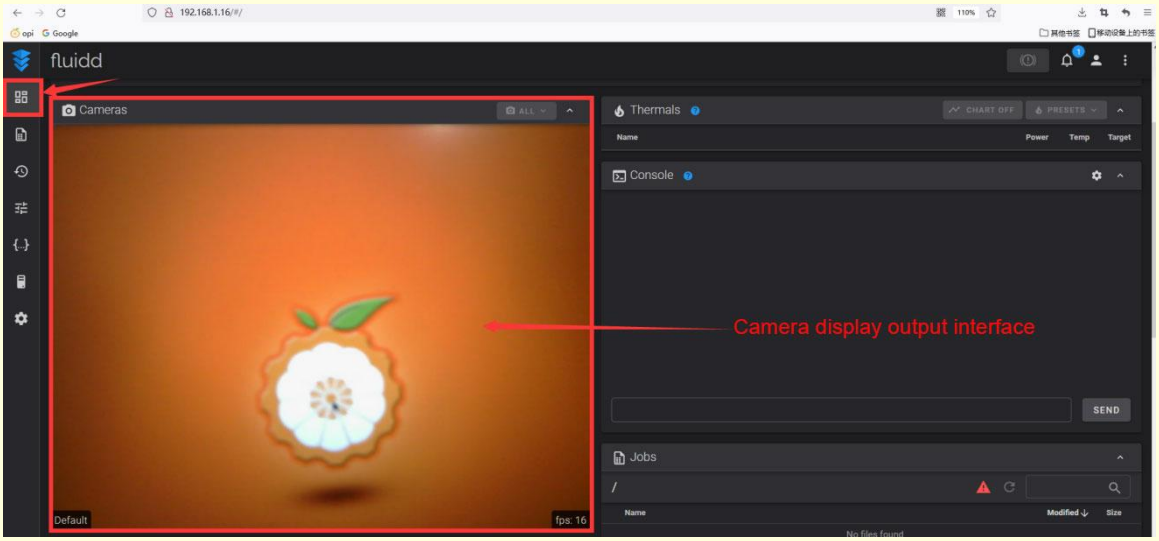
3. Then click Default to open the camera settings interface



4. Then Enable the camera, and then click Save to save



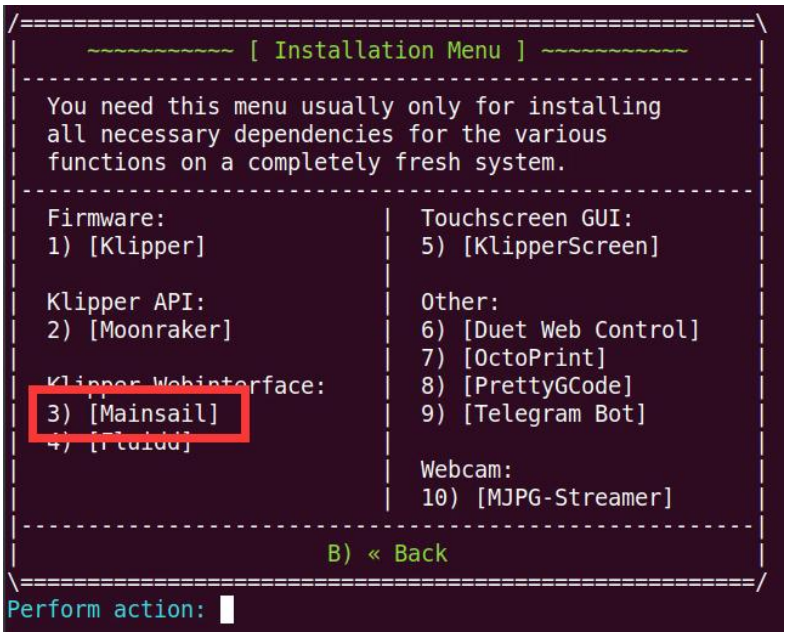
5. Then go back to the main interface to see the output screen of the camera



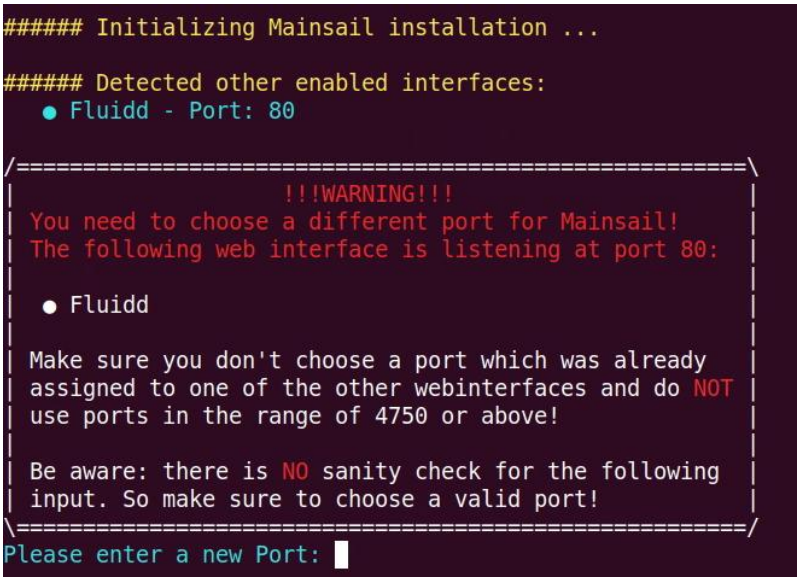
20) Install the Mainsail example using the Kiauh source code archive provided by Orange Pi

a. First please install **1) [Klipper]** and **2) [Moonraker]**

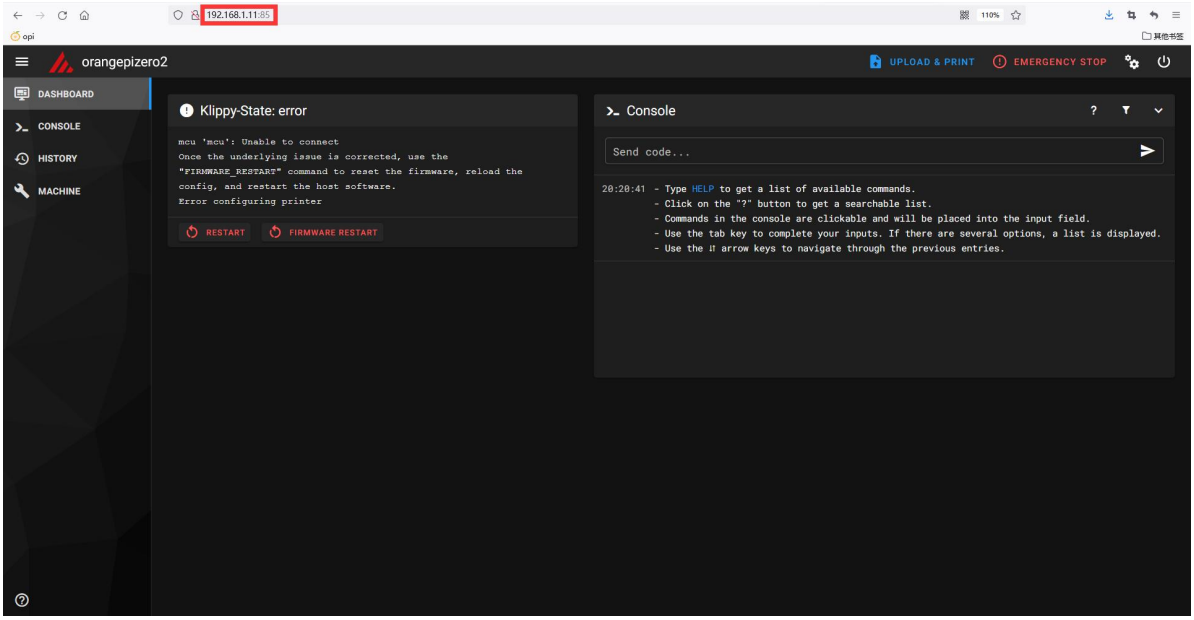
b. Install **3) The [Mainsail]** option is as follows



c. If Fluuid has been installed before, port 80 is already occupied. First, you need to set the port number, for example, set it to 85



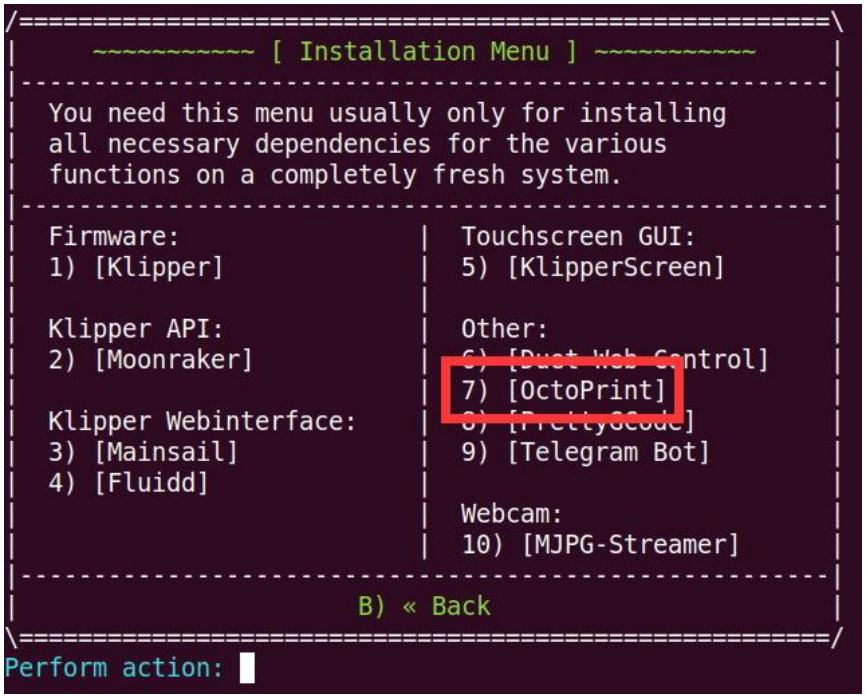
d. After installation, enter the **IP address of the development board: port number** in the browser to see the Mainsail web control interface



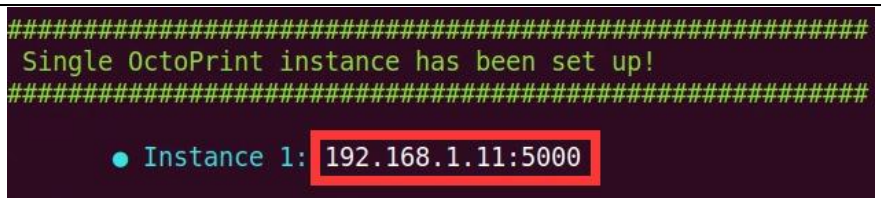
21) Install the OctoPrint example using the Kiauh source code archive provided by Orange Pi

a. First please install **1) [Klipper]** and **2) [Moonraker]**

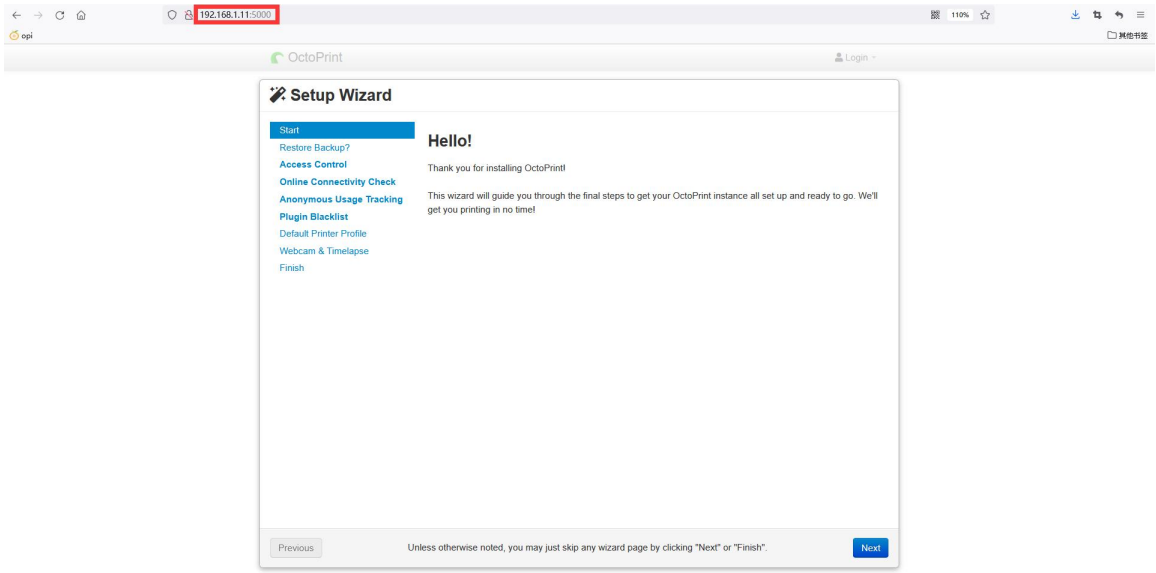
b. Install **7) The [OctoPrint]** option is shown below



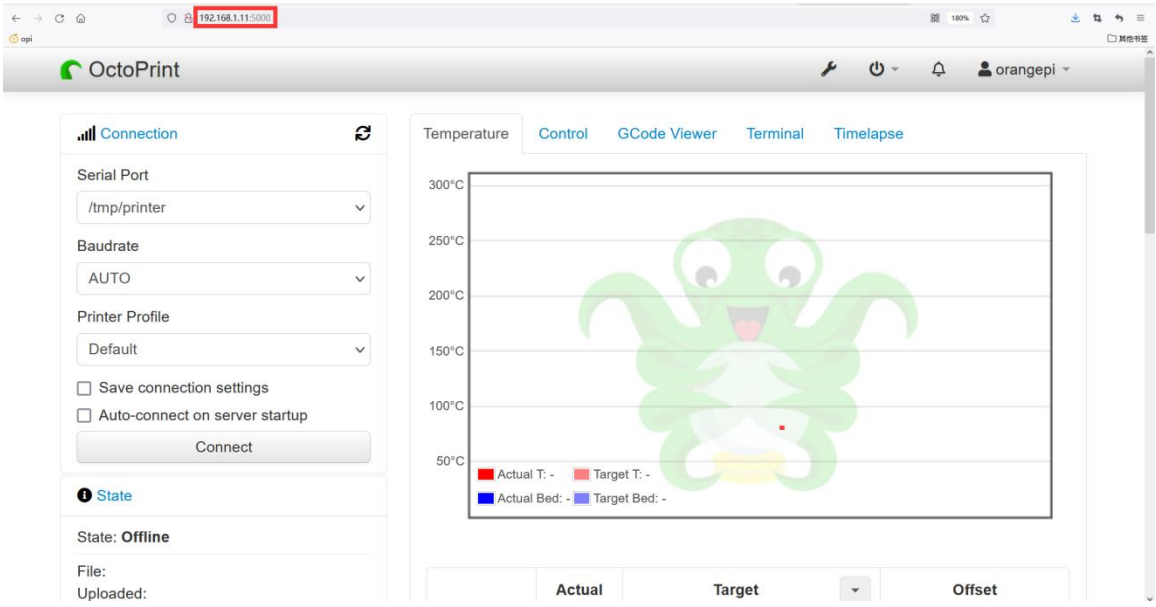
c. After installation, you can see the access address of OctoPrint's web interface, the port number is 5000



d. Then enter the address shown above in the browser to see the OctoPrint web control interface



e. The interface after setting according to the OctoPrint installation wizard is as follows



22) Install the KlipperScreen example using the Kiauh source code archive provided by Orange Pi

If you need to use kiauh to install KlipperScreen, please make sure that the Ubuntu or Debian system used is the linux4.9 server version system, and the kiauh source code used is the Kiauh source code compressed package downloaded from the Google Drive of Orange Pi. Installing KlipperScreen from the kiauh source code downloaded from github will not work properly.

a. Orange Pi's modification to the KlipperScreen source code can be viewed using the following command

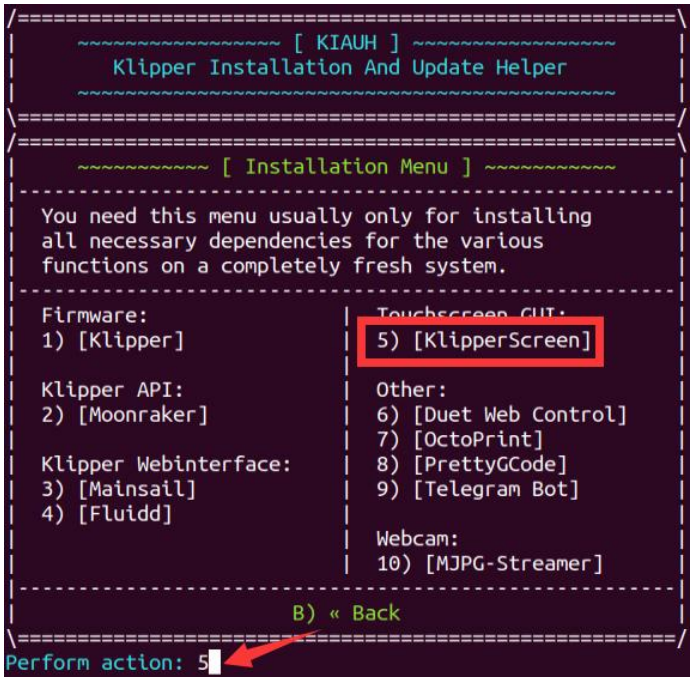
If you are not interested in the source code, please skip it directly, it will not affect the installation and use of KlipperScreen.

```
orangepi@orangepi:~$ cd kiauh/github_src/KlipperScreen
orangepi@orangepi:~/kiauh/github_src/KlipperScreen$ git status .
orangepi@orangepi:~/kiauh/github_src/KlipperScreen$ git diff .
```

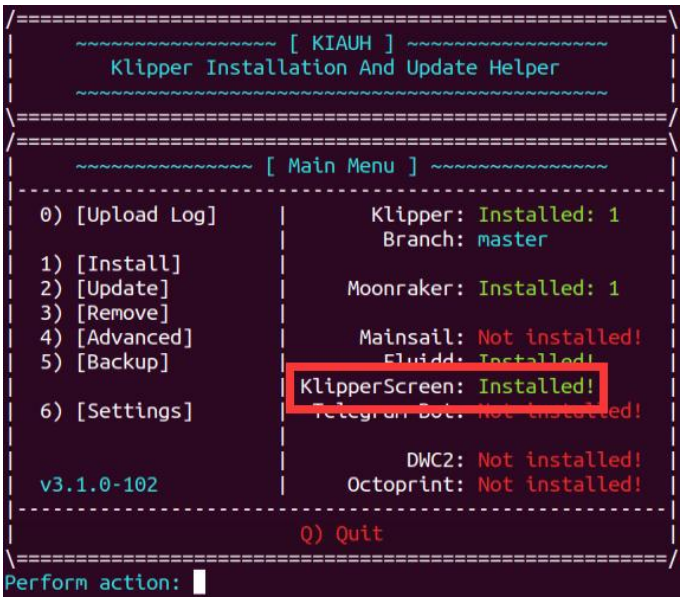
b. Before installing KlipperScreen, please connect the development board to the HDMI display, and ensure that the HDMI can be displayed normally

c. If you must use the **desktop version** of Ubuntu or Debian, first close the desktop according to **the instructions in the section on how to disable the desktop in the Linux desktop version**

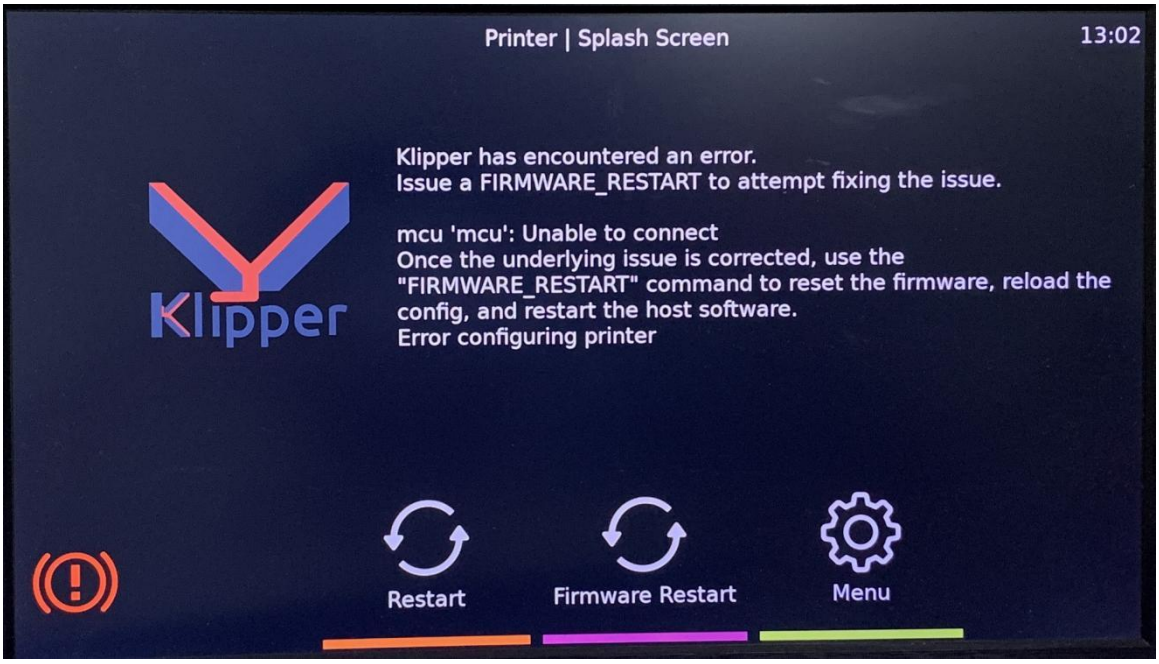
d. Then select **5) [KlipperScreen]** in Kiauh to install KlipperScreen,**Please carefully check the printing information output during the installation process to ensure that no errors are reported**



e. After e.KlipperScreen is installed, Kiauh is displayed as shown below. If there is a problem, please uninstall and reinstall



f. After KlipperScreen is installed, the HDMI monitor can see the interface shown in the figure below (if you can't see it, you can restart it and try it)



23) Related information

Klipper source code：**https://github.com/Klipper3d/klipper**
Klipper official documentation：**http://www.klipper3d.org**

Kiauh source code：**https://github.com/th33xitus/kiauh**

How to connect the Klipper to the microcontroller and the use of the 3D printer cannot be tested in this section. Please find the relevant information or video for testing and debugging.

3.31. Python related instructions

3.31.1. Method for compiling and installing Python source code

If the Python version in the Ubuntu or Debian system software repository does not meet the development requirements, and you want to use the latest version of Python, you can use the following method to download the Python source package to compile and install the latest version of Python.

The following demonstration is to compile and install the latest version of Python 3.9. If you want to compile and install other versions of Python, the method is the same (you need to download the source code corresponding to the Python you want to install)

1) First install the dependencies needed to compile Python

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y build-essential zlib1g-dev  \
libncurses5-dev libgdbm-dev libnss3-dev libssl-dev libsqlite3-dev  \
libreadline-dev libffi-dev curl libbz2-dev
```

2) Then download the latest version of Python3.9 source code and unzip it

```
orangepi@orangepi:~$ wget  \
https://www.python.org/ftp/python/3.9.10/Python-3.9.10.tgz
orangepi@orangepi:~$ tar xvf Python-3.9.10.tgz
```

3) Then run the configure command

```
orangepi@orangepi:~$ cd Python-3.9.10
orangepi@orangepi:~$ ./configure --enable-optimizations
```

4) Then compile and install Python3.9, the compilation time will take about half an hour

```
orangepi@orangepi:~$ make -j4
```

```
orangepi@orangepi:~$ sudo make altinstall
```

5) After installation, you can use the following command to check the version number of the Python you just installed

```
orangepi@orangepi:~$ python3.9 --version
Python 3.9.10
```

6) Then update pip

```
orangepi@orangepi:~$ /usr/local/bin/python3.9 -m pip install --upgrade pip
```

3.31.2. The Method to replace pip source in Python

The default source used by Linux system pip is the official source of Python, but the speed of accessing the official source of Python in China is very slow, and the installation of Python packages often fails due to network reasons. So when using pip to install the Python library, please remember to replace the pip source.

1) First install **python3-pip**

```
orangepi@orangepi:~$ sudo apt install -y python3-pip
```

2) The method of permanently replacing the pip source under Linux

a. First create a new **~/.pip** directory, then add the **pip.conf** configuration file, and set the source of pip to Tsinghua source in it

```
orangepi@orangepi:~$ mkdir -p ~/.pip
orangepi@orangepi:~$ cat <<EOF > ~/.pip/pip.conf
[global]
timeout = 6000
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
trusted-host = pypi.tuna.tsinghua.edu.cn
EOF
```

b. Then using pip3 to install the Python library will be very fast

3) The method of temporarily replacing the pip source under Linux, where **<packagename>** needs to be replaced with a specific package name

```
orangepi@orangepi:~$ pip3 install <packagename> -i  \
https://pypi.tuna.tsinghua.edu.cn/simple --trusted-host pypi.tuna.tsinghua.edu.cn
```

3.32. The method to install Docker

The official installation documentation link provided by Docker is as follows:

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

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

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

2) Then add the official docker software repository

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

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt-get install -y ca-certificates curl gnupg lsb-release
orangepi@orangepi:~$ curl -fsSL https://download.docker.com/linux/debian/gpg |  \
sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
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
```

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

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

3) Then install Docker Engine

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

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

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

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

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

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

Note: You need to log out and log in to the system to take effect, and you can also restart the system

5) Verify the status of docker

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

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

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

```
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

7) Method of setting docker warehouse as domestic source

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

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

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

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

3.33. The way to install Home Assistant

Note that only the method of installing Home Assistant in Ubuntu or Debian system is provided here. For detailed usage of Home Assistant, please refer to the official documentation or corresponding books.

3.33.1. Installation via docker

1) Please install docker first, and make sure that docker can run normally. For the installation steps of Docker, please refer to the instructions in the section How to **Install Docker**.

2) Then you can search for the docker image of Home Assistant

```
orangepi@orangepi:~$ docker search homeassistant
```

3) Then use the following command to download the docker image of Home Assistant to the local, the image size is about 1GB, the download time will be longer, please wait patiently for the download to complete

```
orangepi@orangepi:~$ docker pull homeassistant/home-assistant
Using default tag: latest
```

```
latest: Pulling from homeassistant/home-assistant
be307f383ecc: Downloading
5fbc4c07ac88: Download complete
...... (omit part of the output)
3cc6a1510c9f: Pull complete
7a4e4d5b979f: Pull complete
Digest:
sha256:81d381f5008c082a37da97d8b08dd8b358dae7ecf49e62ce3ef1eeaefc4381bb
Status: Downloaded newer image for homeassistant/home-assistant:latest
docker.io/homeassistant/home-assistant:latest
```

If the network connected to the development board is relatively fast, but the download of the docker image is very slow, please check if you forgot to configure the download address of the docker image as a domestic source. The configuration method is described in the section on how to install Docker.

4) Then you can use the following command to view the docker image of Home Assistant you just downloaded

```
orangepi@orangepi:~$ docker images homeassistant/home-assistant
REPOSITORY                  TAG       IMAGE ID       CREATED        SIZE
homeassistant/home-assistant  latest    bfa0ab9e1cf5   2 months ago   1.17GB
```

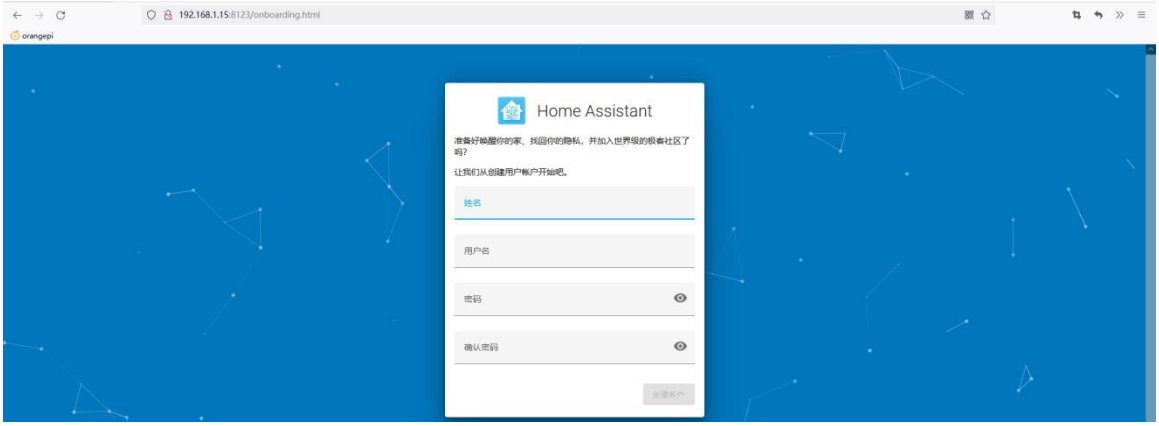
5) At this point, you can run the docker container of Home Assistant

```
orangepi@orangepi:~$ docker run -d \
  --name homeassistant \
  --privileged \
  --restart=unless-stopped \
  -e TZ=Asia/Shanghai \
  -v /home/orangepi/home-assistant:/config \
  --network=host \
  homeassistant/home-assistant:latest
```

6) Then enter **[IP address of the development board: 8123]** in the browser to see the interface of Home Assistant

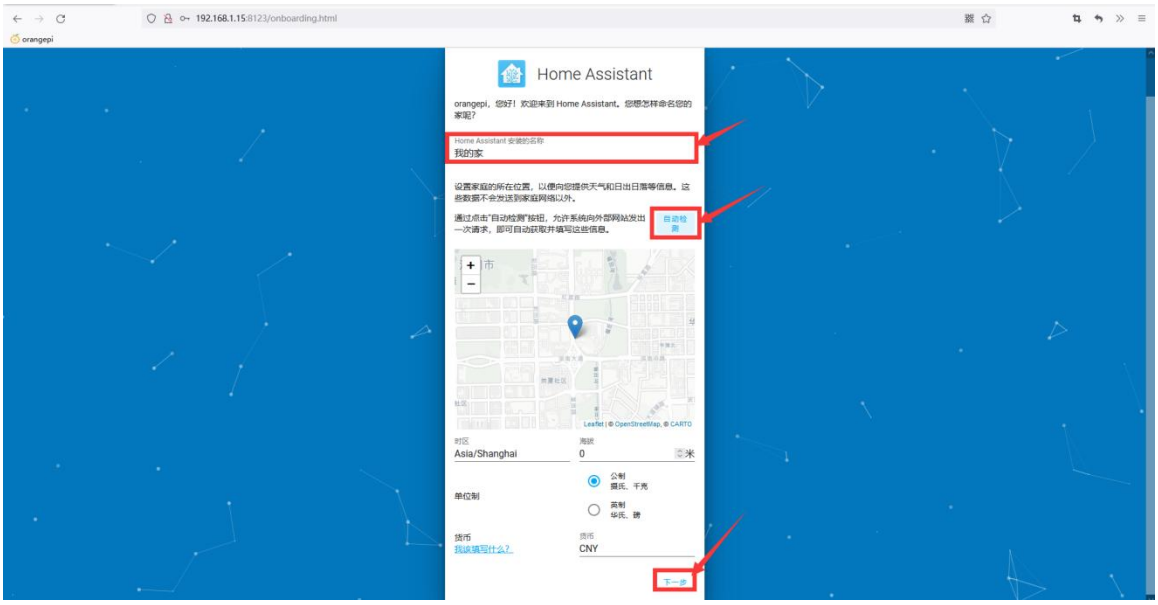
The startup of the Home Assistant container takes a while, if the interface below

does not appear normally, please wait a few seconds before refreshing. If the following interface is not displayed after waiting for more than a minute, it means that there is a problem with the Home Assistant installation. At this time, you need to check whether there is a problem with the previous installation and setting process.

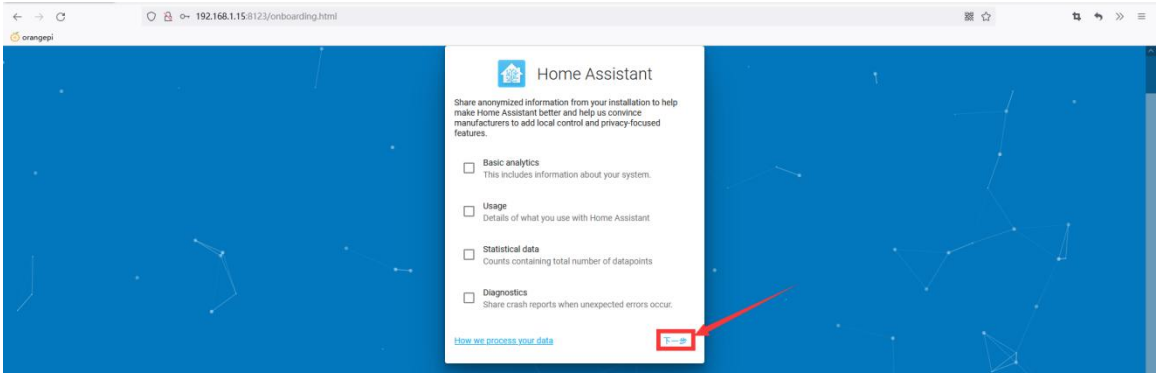


7) Then enter your name, **username** and **password** and **click Create Account**

8) Then follow the interface prompts to set according to your own preferences, and then click Next

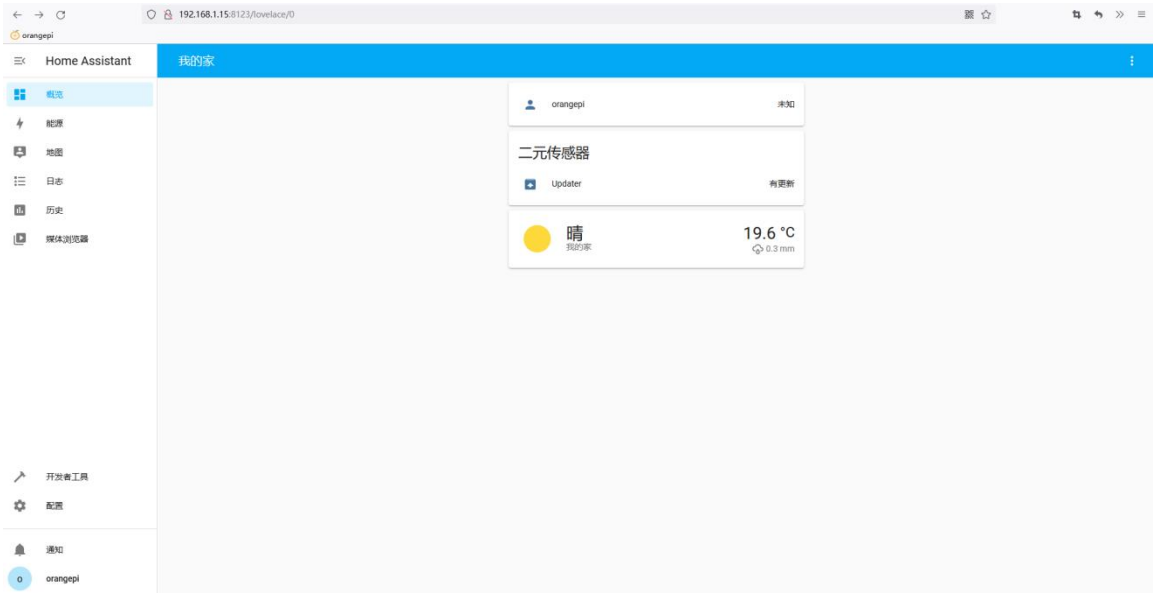


9) Then click Next



10) Then click Finish

11) The main interface finally displayed by Home Assistant is shown in the figure below



12) Ways to stop the Home Assistant container

a. The command to view the docker container is as follows

```
orangepi@orangepi:~$ docker ps -a
```

b. The command to stop the Home Assistant container is as follows

```
orangepi@orangepi:~$ docker stop homeassistant
```

c. The command to delete the Home Assistant container is as follows

```
orangepi@orangepi:~$ docker rm homeassistant
```

3.33.2. Install via python

Before installation, please change the source of pip to a domestic source to speed up the installation of Python packages. For the configuration method, see the description in the section How to replace pip source in Python.

1) First install the dependency package

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt-get install -y python3 python3-dev python3-venv \
python3-pip libffi-dev libssl-dev libjpeg-dev zlib1g-dev autoconf build-essential \
libopenjp2-7 libtiff5 libturbojpeg0-dev tzdata
```

2) Then you need to compile and install Python 3.9. For the method, please refer to the section on compiling and installing Python source code.

The default Python version of Debian Bullseye is Python3.9, so there is no need

to compile and install.

The default Python version of Ubuntu Jammy is Python3.10, so there is no need to compile and install.

3) Then create a Python virtual environment

```
orangepi@orangepi:~$ sudo mkdir /srv/homeassistant
orangepi@orangepi:~$ sudo chown orangepi:orangepi /srv/homeassistant
orangepi@orangepi:~$ cd /srv/homeassistant
orangepi@orangepi:~$ python3.9 -m venv .
orangepi@orangepi:~$ source bin/activate
(homeassistant) orangepi@orangepi:/srv/homeassistant$
```

4) Then install the required Python packages

```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ python3 -m pip install wheel
```

5) Then you can install Home Assistant Core

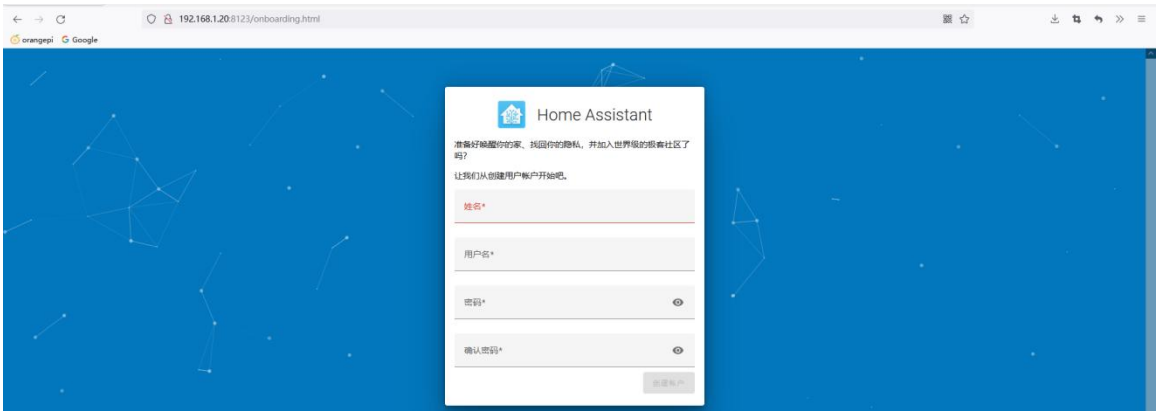
```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ pip3 install homeassistant
```

6) Then enter the following command to run Home Assistant Core

```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ hass
```

7) Then enter **[IP address of the development board: 8123]** in the browser to see the interface of Home Assistant

When you run the hass command for the first time, it will download, install and cache some necessary libraries and dependencies. This process may take a few minutes. Note that the interface of Home Assistant cannot be seen in the browser at this time, please wait for a while before refreshing.



3.34. Installation method of OpenCV

3.34.1. Using apt to install OpenCV

1) The installation command is as follows

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y libopencv-dev python3-opencv
```

2) Then use the following command to print the version number of OpenCV and the output is normal, indicating that the installation of OpenCV is successful

a. The version of OpenCV in Ubuntu22.04 is as follows：

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
4.5.4
```

b. The version of OpenCV in Ubuntu20.04 is as follows：

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
4.2.0
```

c. The version of OpenCV in Ubuntu18.04 is as follows：

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
3.2.0
```

d. The version of OpenCV in Debian10 is as follows：

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
3.2.0
```

e. The version of OpenCV in Debian11 is as follows：

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
4.5.1
```

3.35. Installation method of pagoda Linux panel

BT(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 BT**(baota) **official website)**

1) The recommended order for BT(baota) Linux system compatibility is

```
Debian10 > Ubuntu 20.04 > Ubuntu 18.04 > Other system
```

2) First, you need to expand the size of the **/tmp** space. After setting, you need to **restart the linux system** of the development board. The command is as follows

```
orangepi@orangepi:~$ sudo sed -i 's/nosuid/&,size=2G/' /etc/fstab
orangepi@orangepi:~$ sudo reboot
```

3) After restarting, you can see that the size of the **/tmp** space has become 2G

```
orangepi@orangepi:~$ df -h | grep "/tmp"
tmpfs                   2.0G     12K  2.0G      1% /tmp
```

4) Then enter the following command in the linux system to start the installation of the pagoda

```
orangepi@orangepi:~$ wget -O install.sh    \
http://download.bt.cn/install/install-ubuntu_6.0.sh && sudo bash install.sh
```

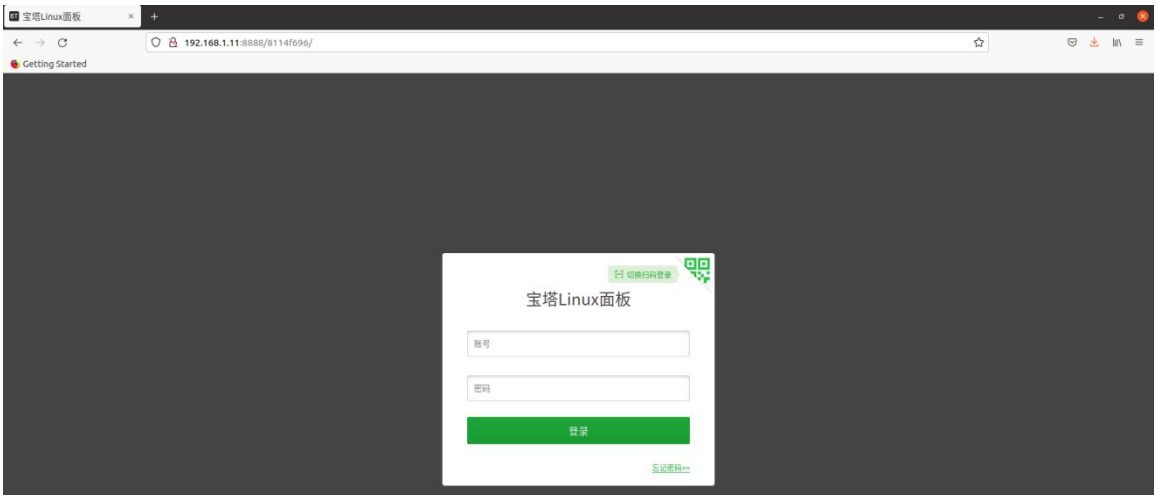
5) Then the pagoda installer will remind whether to install **Bt-Panel** to the **/www** folder, then enter **y**

```
+----------------------------------------------------------------------
| Bt-WebPanel FOR CentOS/Ubuntu/Debian
+----------------------------------------------------------------------
| Copyright © 2015-2099 BT-SOFT(http://www.bt.cn) All rights reserved.
+----------------------------------------------------------------------
| The WebPanel URL will be http://SERVER_IP:8888 when installed.
+----------------------------------------------------------------------

Do you want to install Bt-Panel to the /www directory now?(y/n): y
```

6) Then all you have to do is wait patiently. When you see the following print information output from the terminal, it means that the pagoda has been installed. The whole installation process takes about 59 minutes, and there may be some differences depending on the network speed.

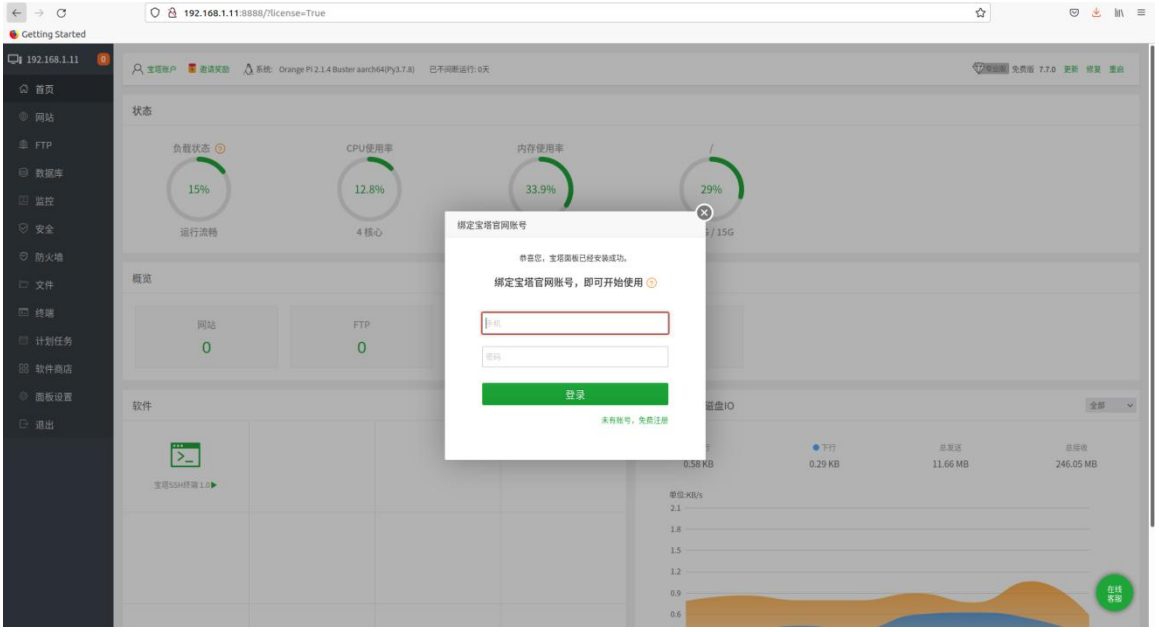
```
==================================================================
Congratulations! Installed successfully!
==================================================================
外网面板地址: http://[240e:3b7:3240:c3a0:d81f:613c:1739:3d6a]:8888/8114f696
内网面板地址: http://192.168.1.11:8888/8114f696
username: klpyxjy6
password: ae287263
If you cannot access the panel,
release the following panel port [8888] in the security group
若无法访问面板，请检查防火墙/安全组是否有放行面板[8888]端口
==================================================================
Time consumed: 59 Minute!
orangepi@orangepizero2:~$
```

7) At this time, enter the **panel address** shown above in the browser to open the login interface of the BT(Baota) Linux panel, and then enter the **username** and **password** shown in the above figure in the corresponding position to log in to the pagoda

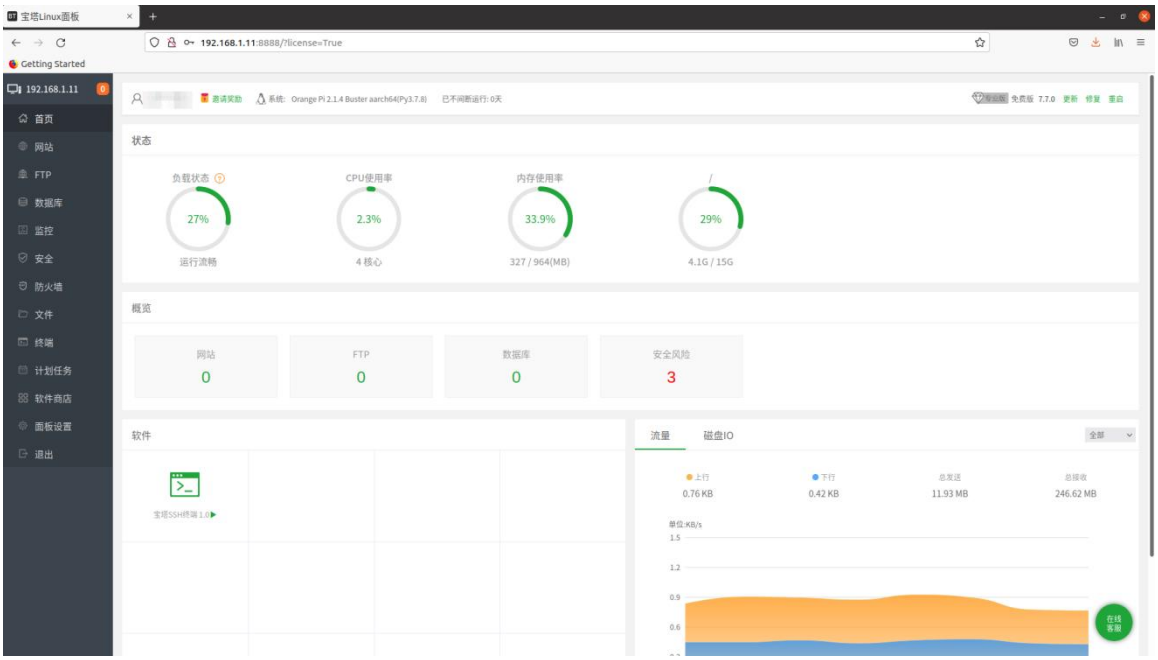


8) After successfully logging in to the BT(baota), the following welcome interface will pop up. First, please read the user instructions in the middle and drag it to the bottom, then you can select "I have agreed and read the "User Agreement", and then click "Enter the panel" You can enter the BT(baota)

9) After entering the BT(baota), you will first be prompted to bind an account on the official website of the BT(baota). If you do not have an account, you can go to the official website of the BT(baota) (**https://www.bt.cn**) to register one

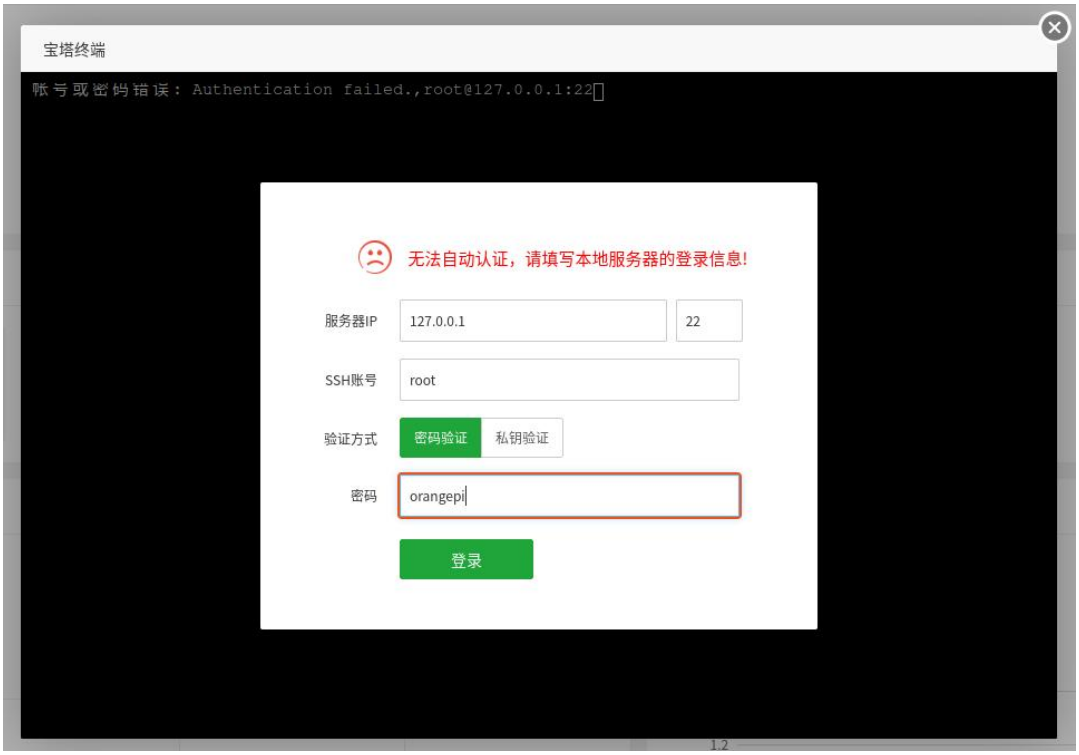


10) The final displayed interface is shown in the figure below. You can intuitively see some status information of the Linux system on the development board, such as load status, CPU usage, memory usage, and storage space usage, etc.

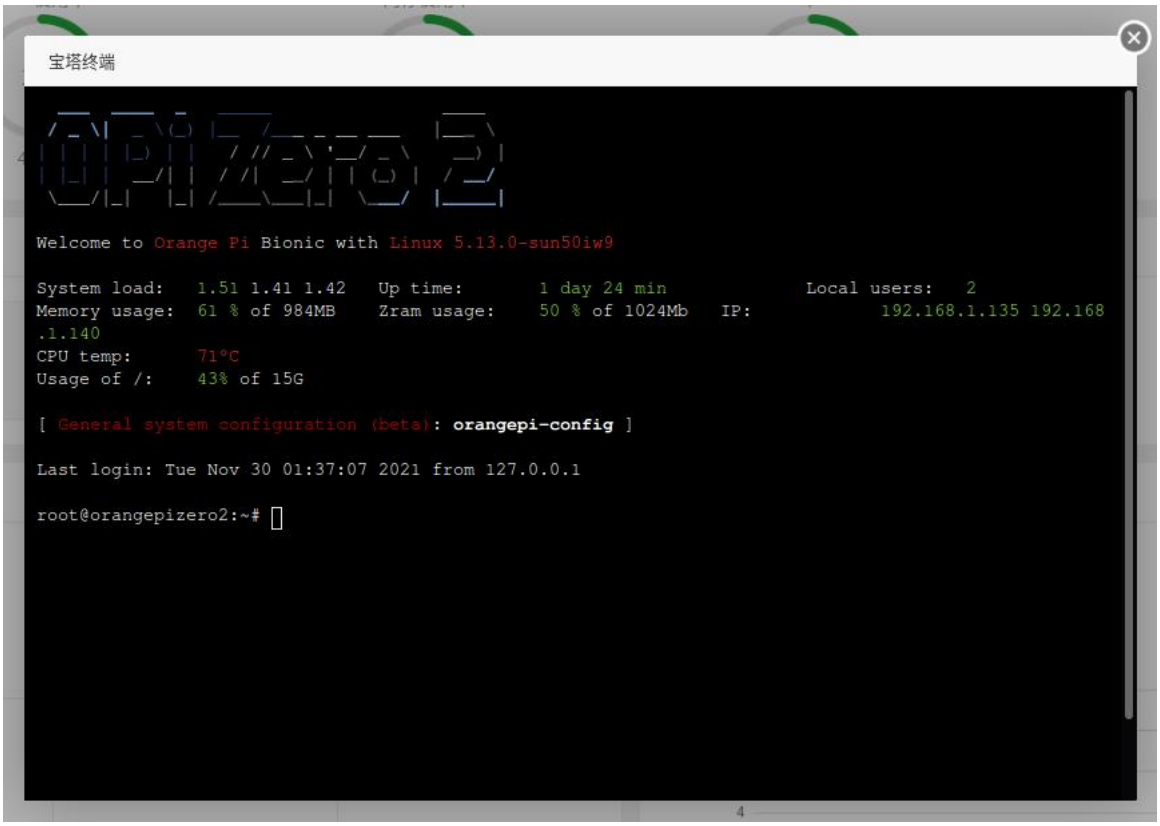


11) Test the SSH terminal login of the pagoda

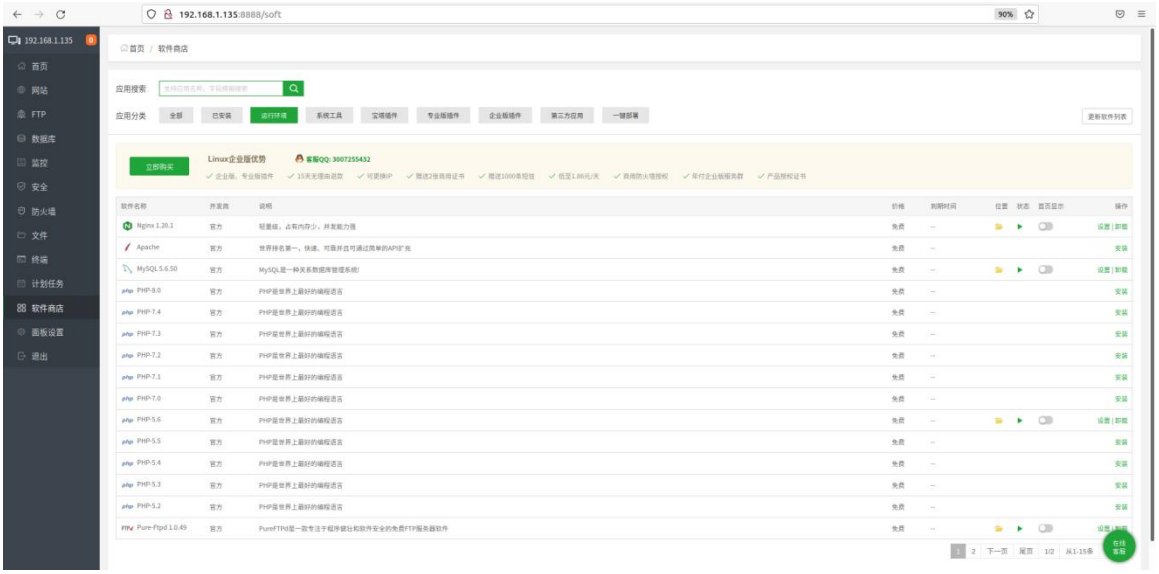
a. After opening the **SSH** terminal of the pagoda, you will first be prompted to enter the password of the development board system. At this time, enter **orangepi** in the password box (the default password, if there is any modification, please fill in the modified one)



b. The display after successful login is as shown below



12) Software such as Apache, MySQL and PHP can be installed in the software store of the BT(baota), and various applications can also be deployed with one click. Please explore these functions by yourself, and I will not demonstrate them one by one here.



13) Pagoda command line tool test

```
root@orangepizero2:~# bt
===============宝塔面板命令行==================
(1) 重启面板服务              (8) 改面板端口
(2) 停止面板服务              (9) 清除面板缓存
(3) 启动面板服务              (10) 清除登录限制
(4) 重载面板服务              (11) 取消入口限制
(5) 修改面板密码              (12) 取消域名绑定限制
(6) 修改面板用户名            (13) 取消IP访问限制
(7) 强制修改MySQL密码         (14) 查看面板默认信息
(22) 显示面板错误日志         (15) 清理系统垃圾
(23) 关闭BasicAuth认证        (16) 修复面板(检查错误并更新面板文件到最新版)
(24) 关闭谷歌认证             (17) 设置日志切割是否压缩
(25) 设置是否保存文件历史副本  (18) 设置是否自动备份面板
(0) 取消
==========================================
请输入命令编号: 14
==========================================
正在执行(14)...
==========================================
==================================================================
BT-Panel default info!
==================================================================
外网面板地址: http://[240e:3b7:3240:c3a0:d81f:613c:1739:3d6a]:8888/8114f696
内网面板地址: http://192.168.1.11:8888/8114f696
*以下仅为初始默认账户密码，若无法登录请执行bt命令重置账户/密码登录
username: klpyxjy6
password: ae287263
If you cannot access the panel,
release the following panel port [8888] in the security group
若无法访问面板，请检查防火墙/安全组是否有放行面板[8888]端口
==================================================================
root@orangepizero2:~#
```

14) For more functions of the BT(baota), you can refer to the following information to explore by yourself

Manual：**http://docs.bt.cn**
Forum address：**https://www.bt.cn/bbs**
GitHub Link：**https://github.com/aaPanel/BaoTa**

3.36. The method of remotely logging in to the Linux system desktop

Compared with VNC, it is more recommended to use NoMachine to remotely log in to the Linux system desktop

3.36.1. Remote login using NoMachine

Please make sure that the Ubuntu or Debian system installed on the development board is the desktop version. In addition, NoMachine also provides detailed usage documentation. It is strongly recommended to read through this documentation to familiarize yourself with the usage of NoMachine. The documentation links are as follows:

https://knowledgebase.nomachine.com/DT10R00166

NoMachine supports Windows, Mac, Linux, iOS and Android platforms, so we can remotely log in to control the Orange Pi development board through NoMachine on a variety of devices. The following demonstrates how to remotely log in to the Linux system desktop of the Orange Pi development board through NoMachine in Windows. For installation methods on other platforms, please refer to the official documentation of NoMachine.

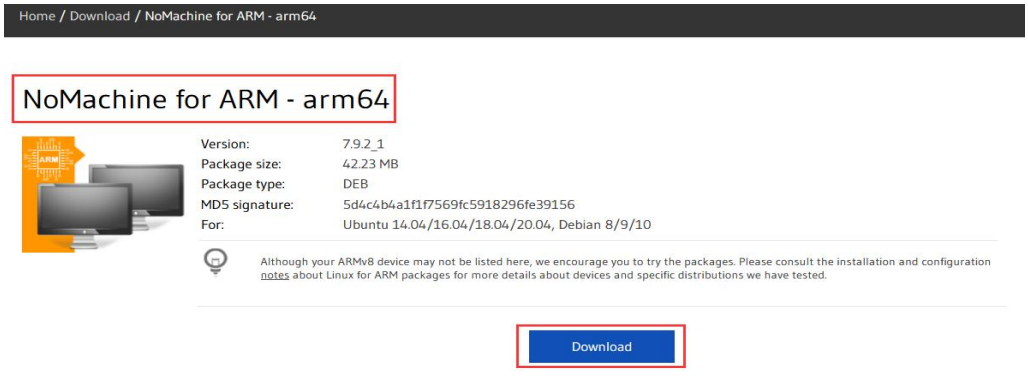
Before operation, please make sure that the Windwos computer and the development board are in the same local area network, and can log in to the Ubuntu or Debian system of the development board normally through ssh.

1) First download the installation package of the Linux **arm64** deb version of the NoMachine software, and then install it into the Linux system of the development board

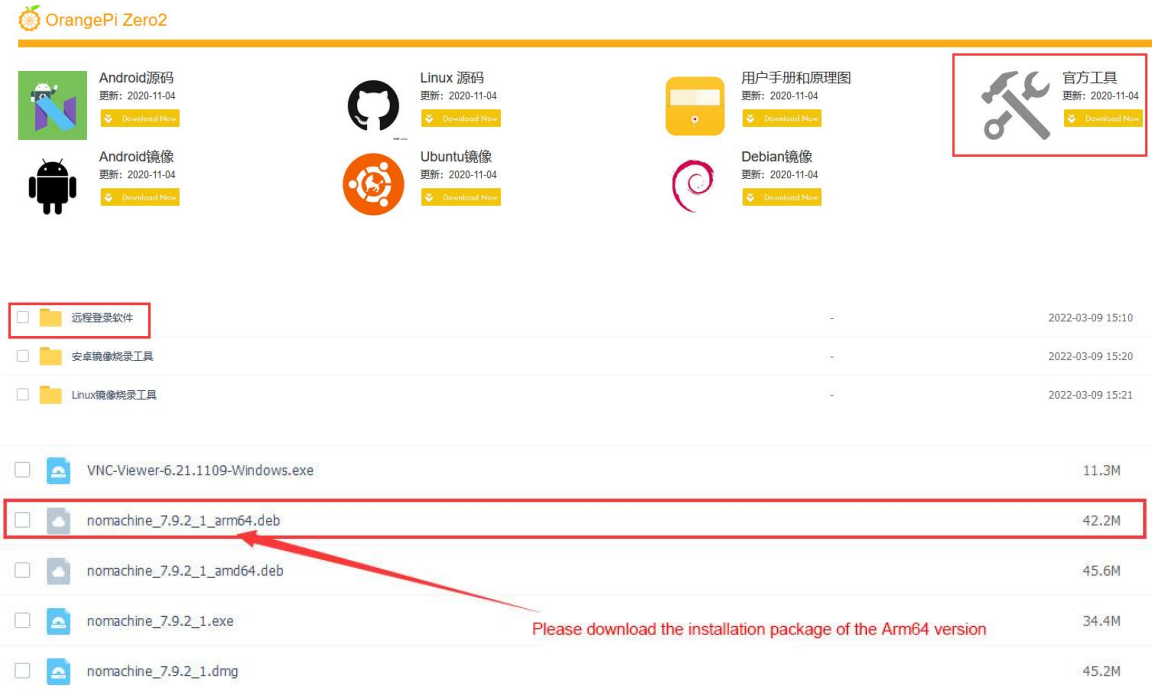
a. Since H616 is an SOC of ARMv8 architecture, the system we use is Ubuntu or Debian, so here we need to download the **NoMachine for ARM ARMv8 DEB** installation package. The download link is as follows:

Note that this download link may change, please look for the Armv8/Arm64 version of the deb package.

https://www.nomachine.com/download/download&id=116&s=ARM



b. In addition, the **NoMachine** installation package can also be downloaded in the **official tool**



c. Then upload the downloaded **nomachine_7.9.2_1_arm64.deb** to the Linux system of the development board

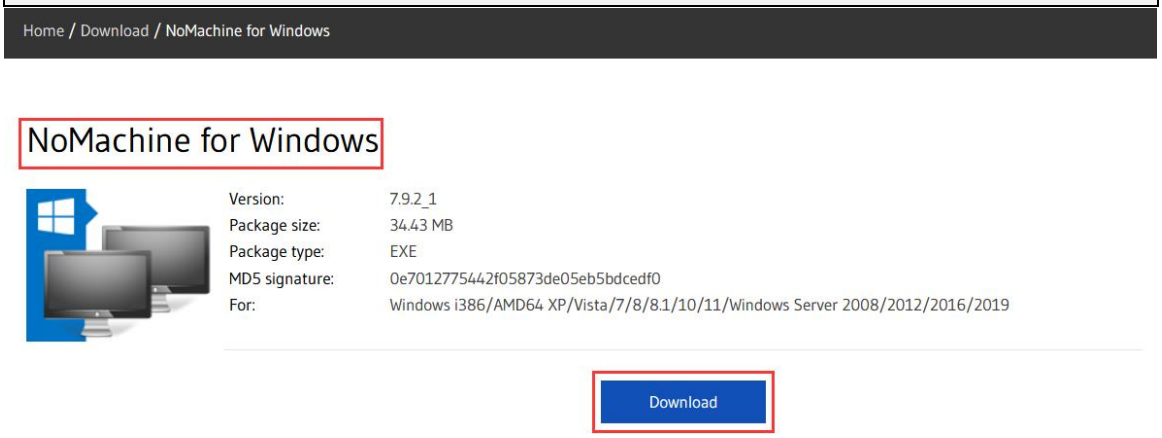
d. Then use the following command to install **NoMachine** in the Linux system of the development board

```
orangepi@orangepi:~$ sudo dpkg -i nomachine_7.9.2_1_arm64.deb
[sudo] password for orangepi:
Selecting previously unselected package nomachine.
(Reading database ... 182635 files and directories currently installed.)
Preparing to unpack nomachine_7.9.2_1_arm64.deb ...
Unpacking nomachine (7.9.2-1) ...
Setting up nomachine (7.9.2-1) ...
NX> 700 Starting install at: Sun Apr 17 10:52:07 2022.
NX> 700 Installing: nxclient version: 7.9.2.
NX> 700 Using installation profile: Debian.
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
NX> 700 Compiling the USB module.
NX> 700 Installing: nxplayer version: 7.9.2.
NX> 700 Using installation profile: Debian.
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
```

```
NX> 700 To connect the remote printer to the local desktop,
NX> 700 the user account must be a member of the CUPS System Group: lpadmin.
NX> 700 Installing: nxnode version: 7.9.2.
NX> 700 Using installation profile: Debian.
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
NX> 700 Creating configuration in: /usr/NX/etc/node.cfg.
NX> 700 Installing: nxserver version: 7.9.2.
NX> 700 Using installation profile: Debian.
NX> 700 Install log is: /usr/NX/var/log/nxinstall.log.
NX> 700 Creating configuration in: /usr/NX/etc/server.cfg.
NX> 700 Install completed at: Sun Apr 17 10:53:00 2022.
NX> 700 NoMachine was configured to run the following services:
NX> 700 NX service on port: 4000
```

2) Then download the installation package of the Windows version of the NoMachine software, the download address is as follows

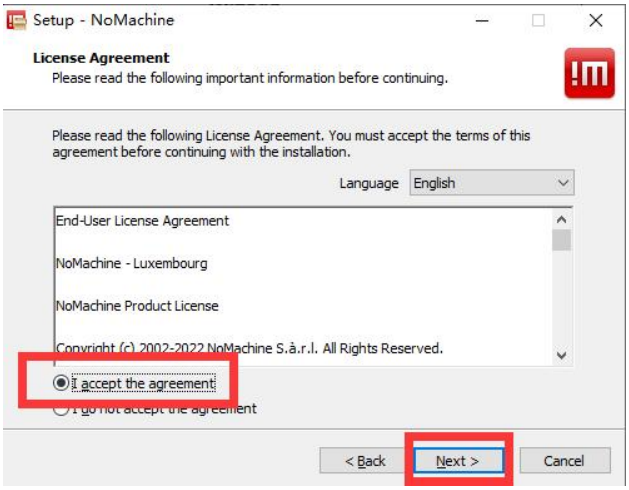
https://www.nomachine.com/download/download&id=8



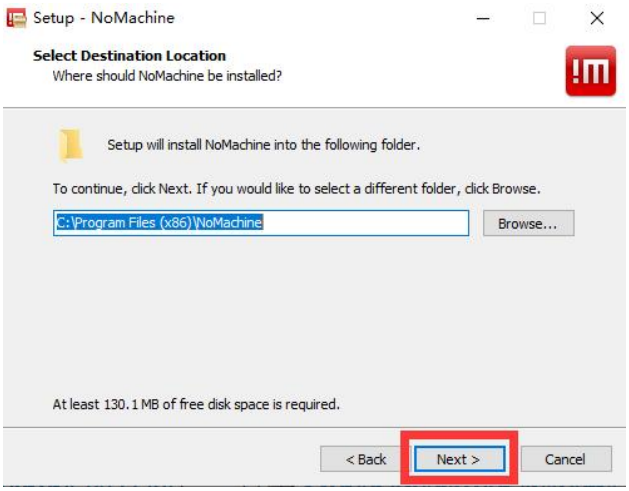
3) Then install NoMachine in Windows

a. Double-click the NoMachine installation package in Windows to start the NoMachine installation, and then select **Next**.

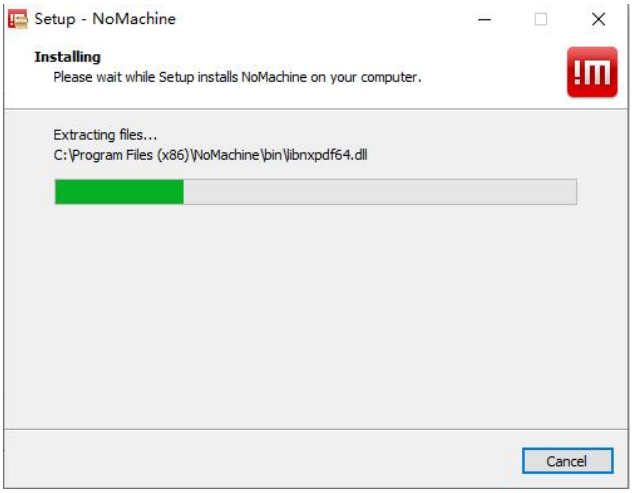
b. Then select **I accept the agreement**, and then select **Next**



c. Then click **Next**

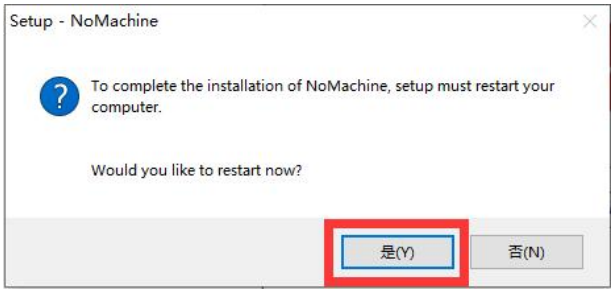


d. The installation process will then begin



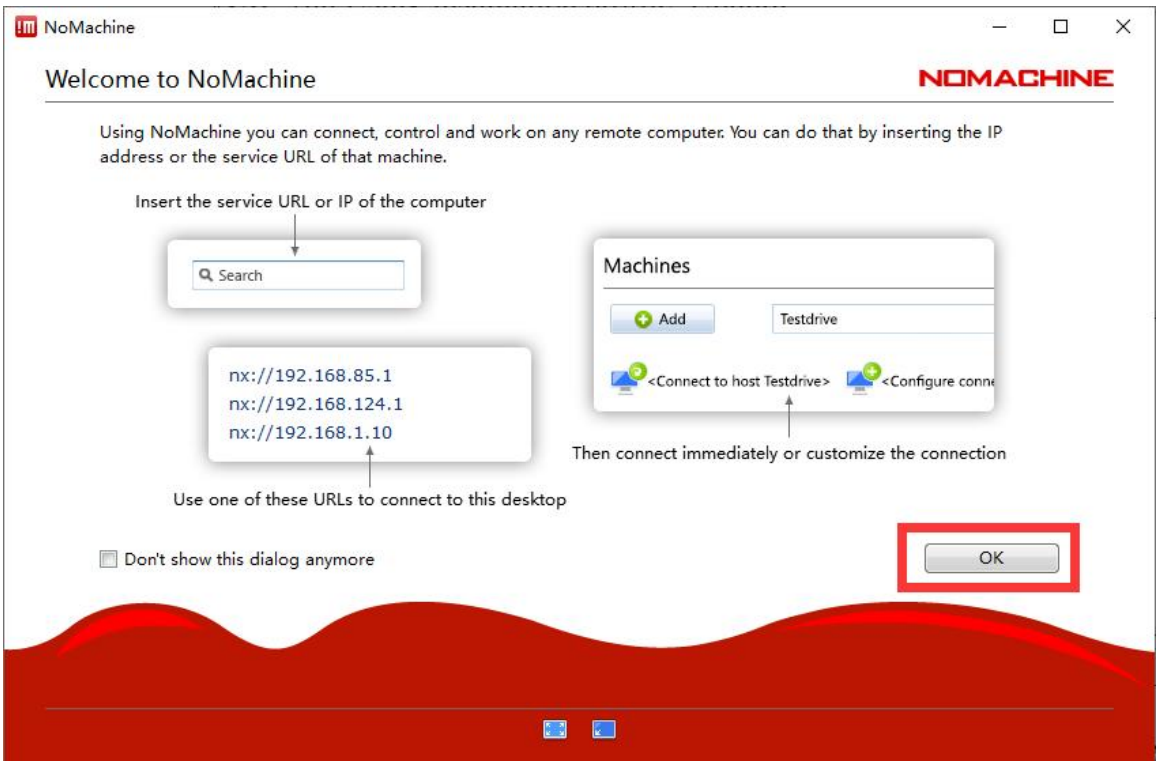
e. After the installation is complete, the display is as shown in the figure below, and then click **Finish**.

f. Then NoMachine will prompt you to restart to complete the installation, here we choose **Yes (Y)** to restart the computer

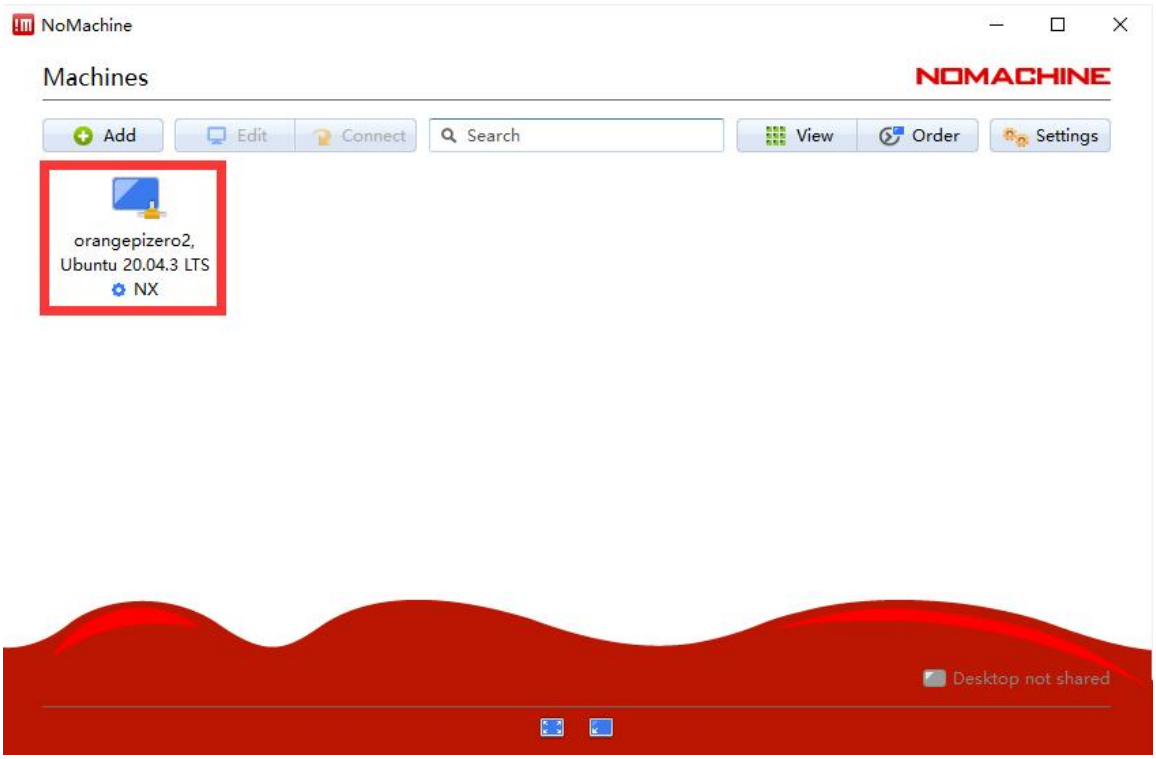


4) Then double click open **NoMachine** in Window

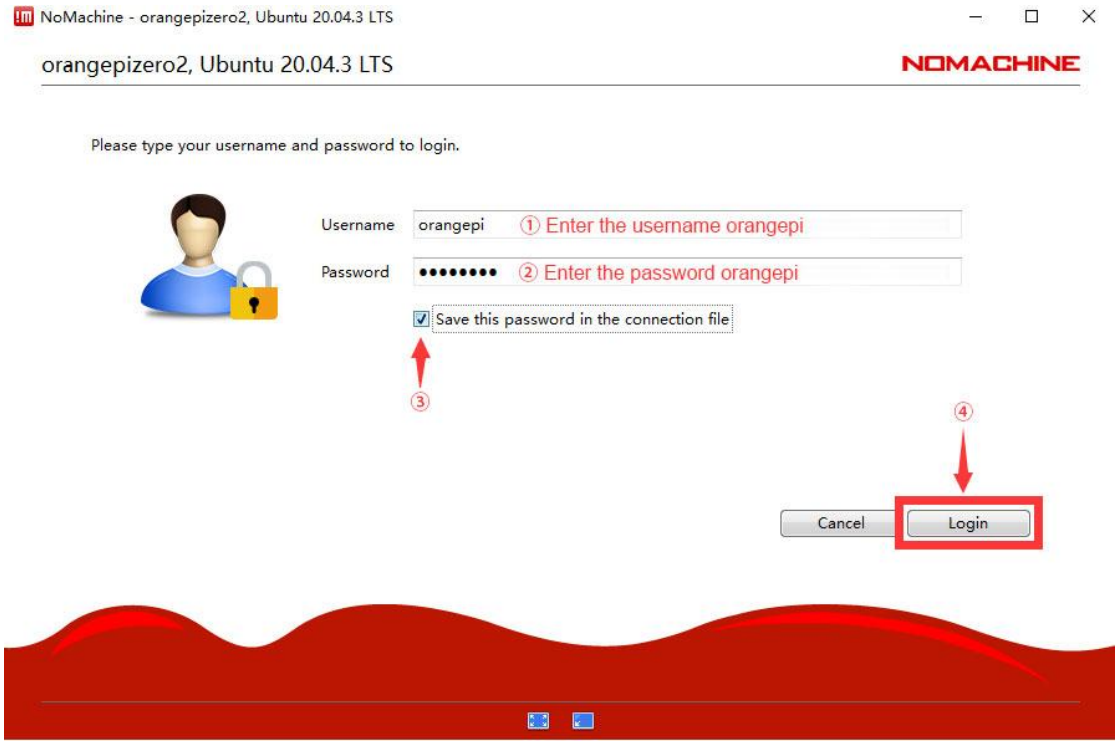
5) Then click**OK**



6) After NoMachine is started, it will automatically scan other devices with NoMachine installed in the local area network. After entering the main interface of NoMachine, you can see that the development board is already in the list of connectable devices, and then click the position shown in the red box in the figure below. You can start to log in to the Linux system desktop of the development board

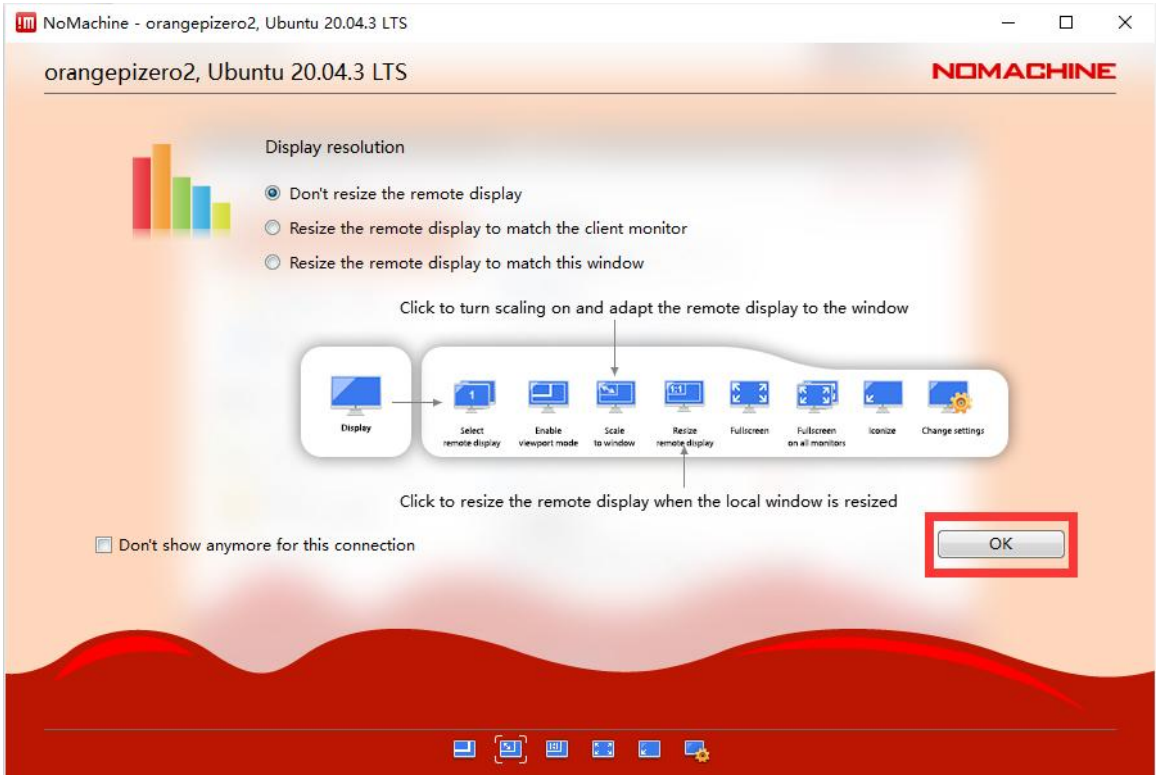


7) Then enter the user name and password of the Linux system of the development board in the corresponding position in the figure below, and then click **Login** to start logging in



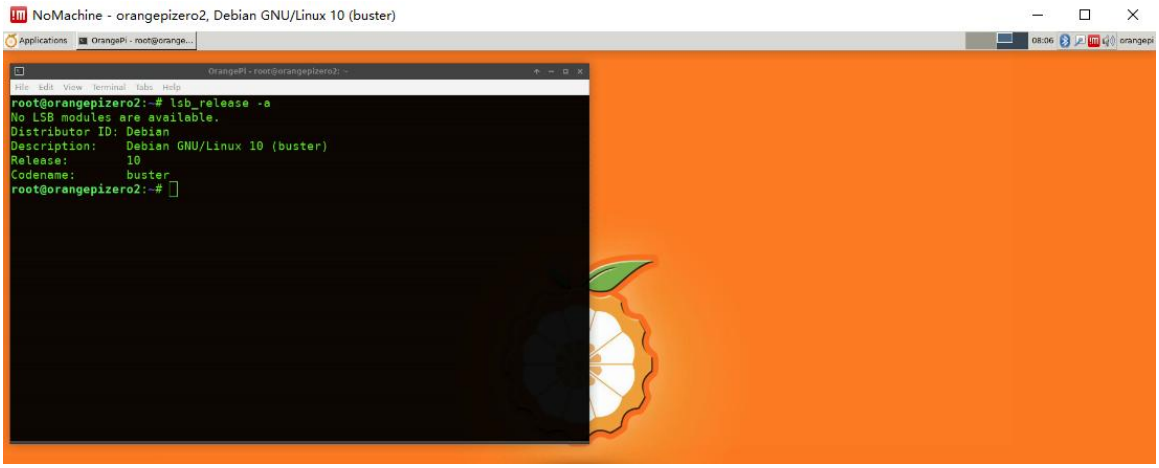
8) Then click OK

9) Then you can set the display resolution, select it as needed, and then click OK

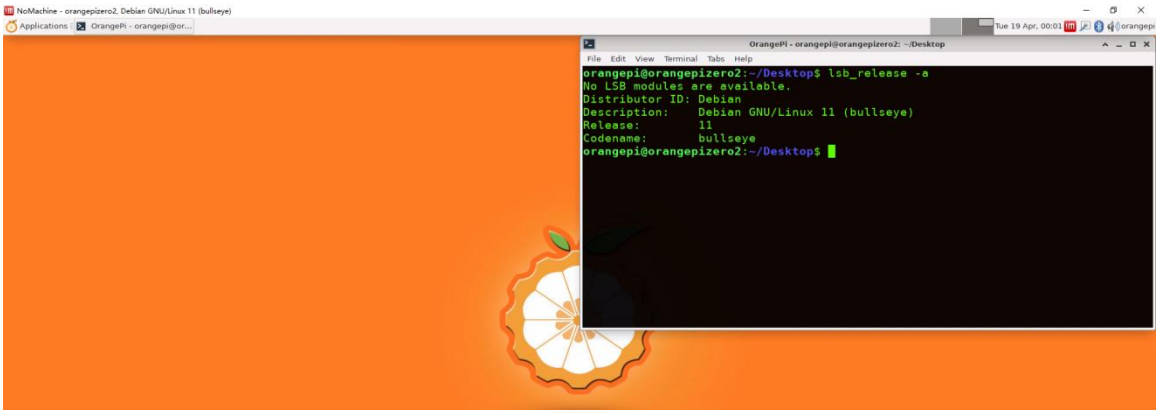


10) Finally, you can see the desktop of the Linux system of the development board

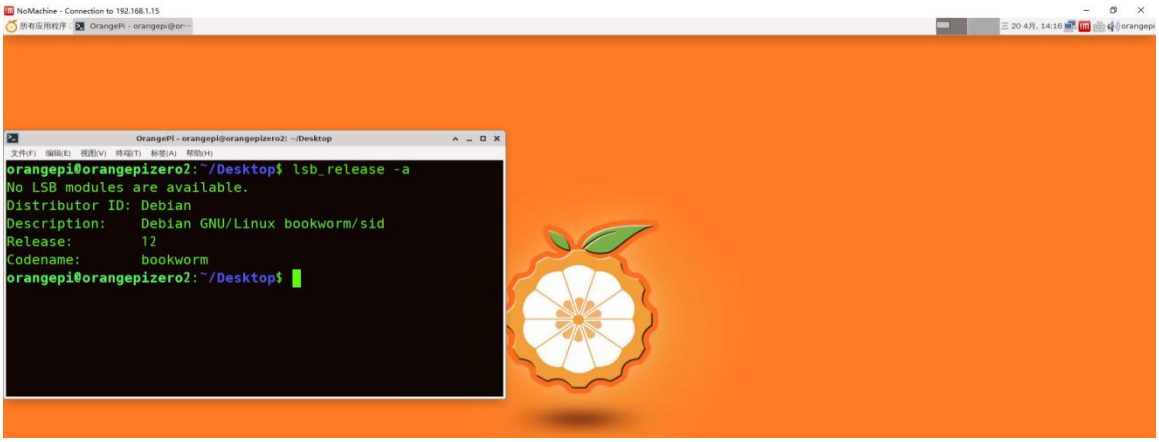
a. Debian10



b. Debian11



c. Debian12



3.36.2. Remote login using VNC

Before operation, please make sure that the Windwos computer and the development board are in the same local area network, and can log in to the Ubuntu or Debian system of the development board normally through ssh

1) First execute the following commands in the Linux system of the development board to install **tightvncserver** and **xrdp**

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt -y install tightvncserver xrdp
```

Debian12 does not have the xrdp package, so just install tightvncserver

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

2) Then run the **vncserver** command in the Linux system of the development board to set the password (**the password is 6 to 8 characters**) and create the initial configuration file

```
orangepi@orangepi:~$ vncserver

You will require a password to access your desktops.

Password:
Verify:
Would you like to enter a view-only password (y/n)? n

New 'X' desktop is orangepi:1

Creating default startup script /home/orangepi/.vnc/xstartup
Starting applications specified in /home/orangepi/.vnc/xstartup
Log file is /home/orangepi/.vnc/orangepi:1.log
```

3) Then set the xstartup configuration file of vnc

a. First deactivate the vnc server instance

```
orangepi@orangepi:~$ vncserver -kill :1
Killing Xtightvnc process ID 5337
```

b. Then back up the xstartup configuration file

```
orangepi@orangepi:~$ mv ~/.vnc/xstartup ~/.vnc/xstartup.bak
```

c. Then create a new xstartup configuration file and enter the following content in it

```
orangepi@orangepi:~$ vim ~/.vnc/xstartup
#!/bin/bash
xrdb $HOME/.Xresources
startxfce4 &
```

d. Then add executable permissions to xstartup

```
orangepi@orangepi:~$ chmod +x ~/.vnc/xstartup
```

e. Finally restart the vncserver server, then you can log in to the Linux desktop remotely through the vnc client

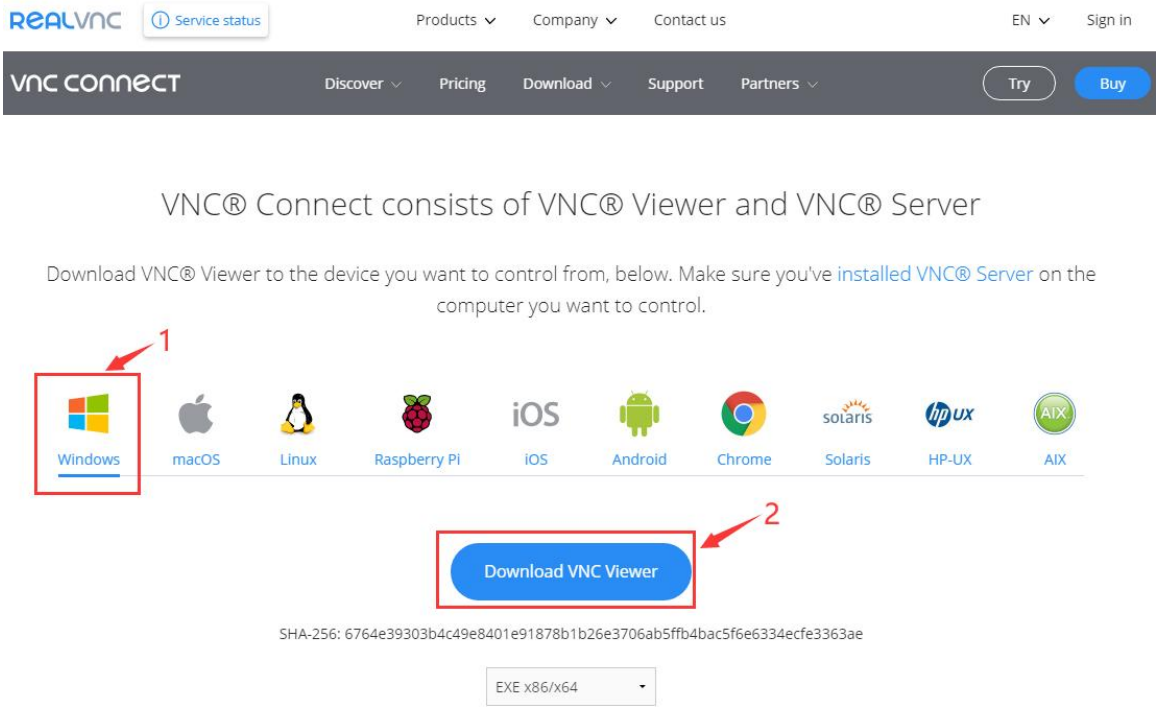
```
orangepi@orangepi:~$ vncserver
```

```
New 'X' desktop is orangepizero2:1

Starting applications specified in /home/orangepi/.vnc/xstartup
Log file is /home/orangepi/.vnc/orangepizero2:1.log
```

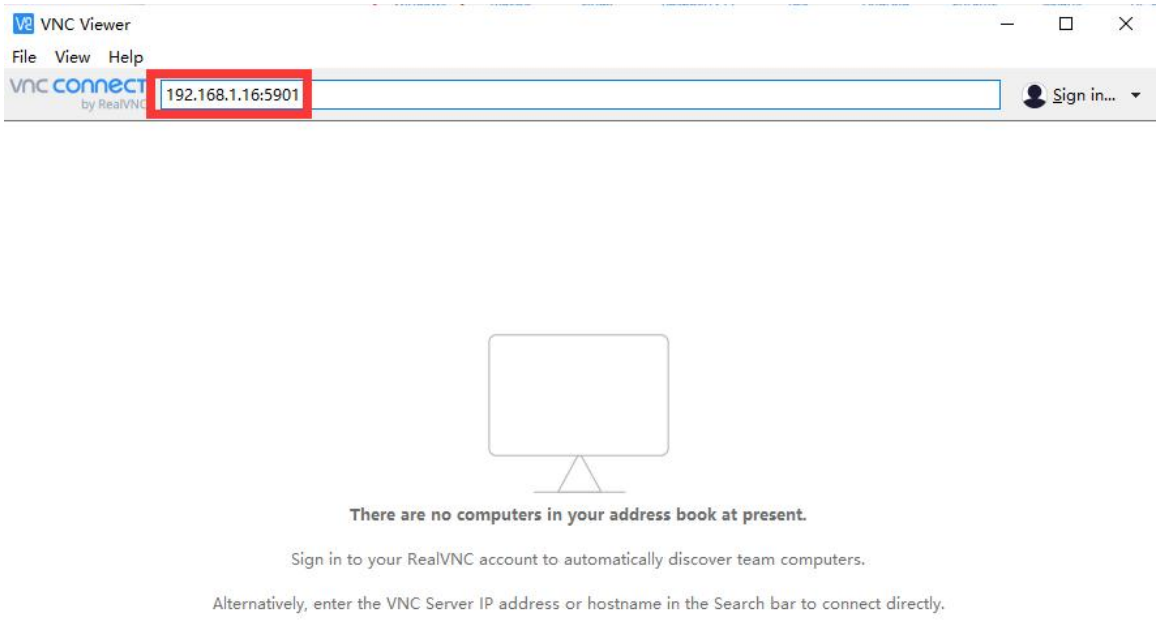
4) The steps to use the VNC Viewer client to connect to the Linux system desktop of the development board are as follows

a. First download and install the VNC Viewer client on the Windows PC, the download link is as follows

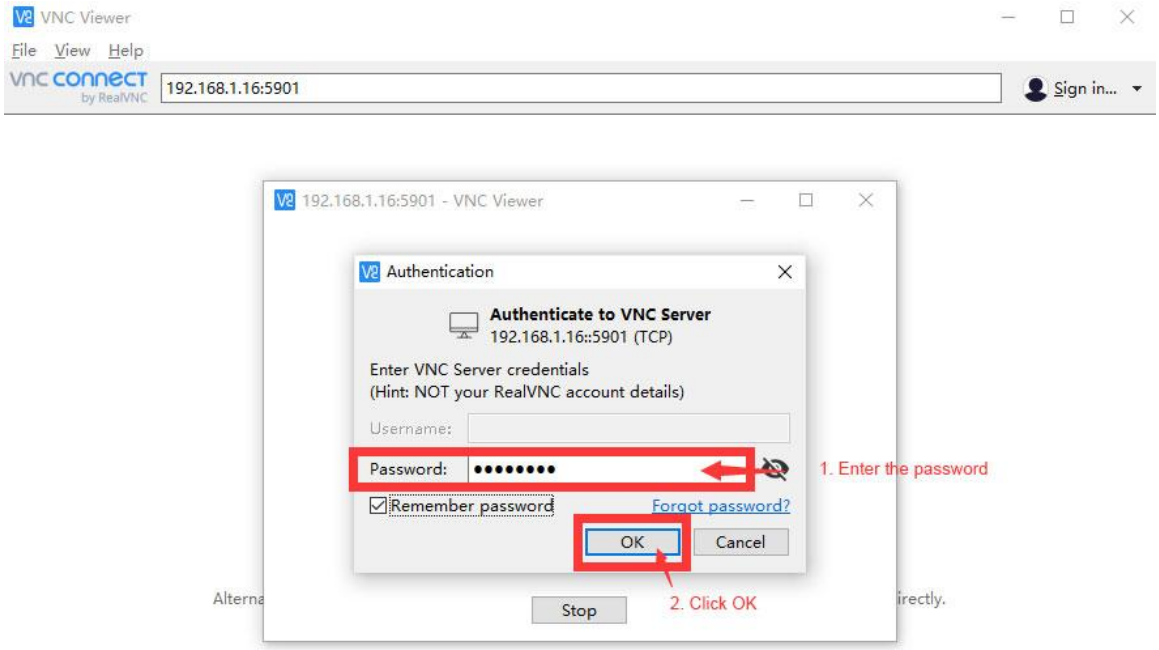
https://www.realvnc.com/en/connect/download/viewer/



b. After installing the VNC Viewer client on the Windows PC, open the **VNC Viewer,** and then enter **[the IP address of the development board: 5901]** in the search bar of the **VNC Viewer**

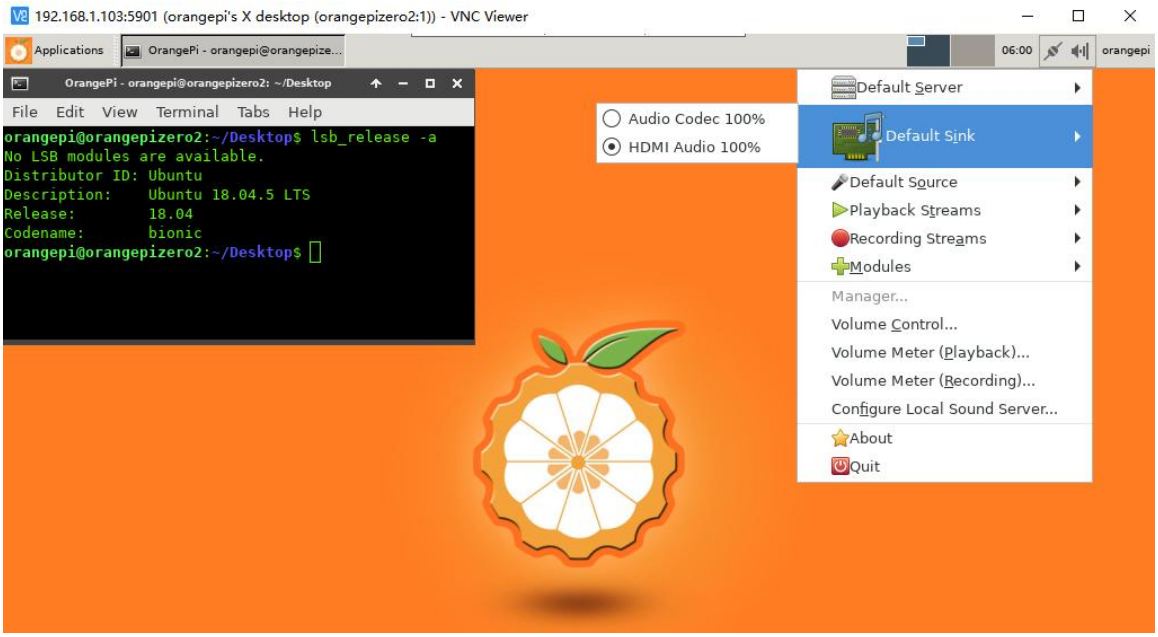


c. Then enter the **password** set when running the vncserver command in step 2 in the Passowrd column, and then click **OK** to remotely log in to the Linux system desktop of the development board

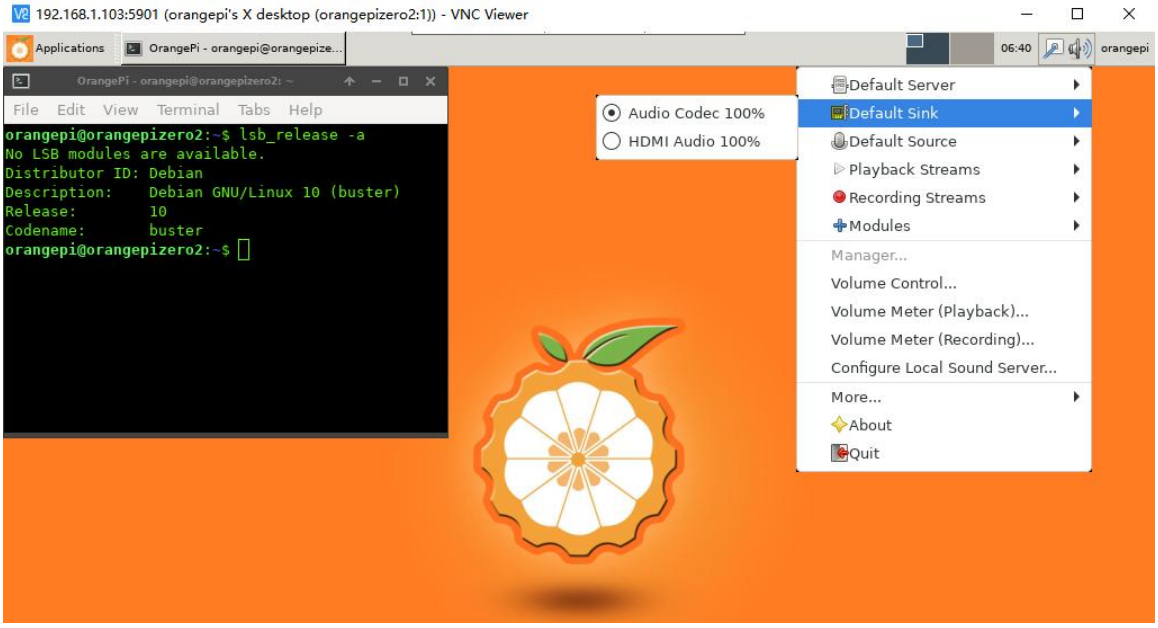


d. After the login is successful, the interface is displayed as shown in the figure below, and then the desktop of the Linux system of the development board can be remotely operated.

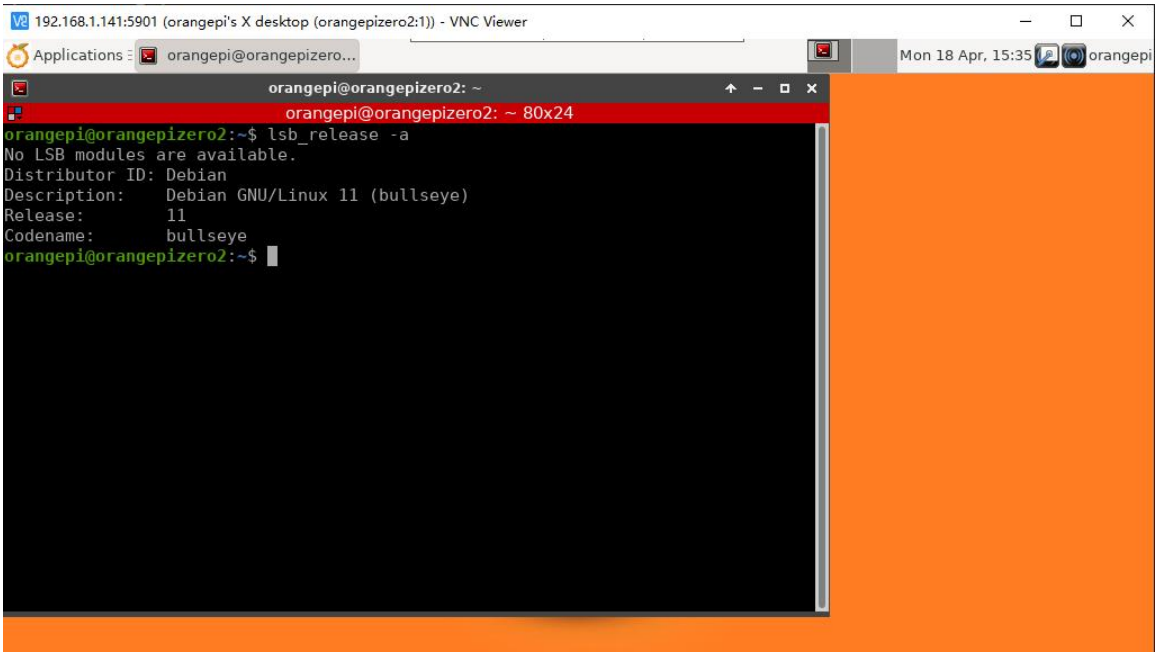
a) Ubuntu18.04 login display as shown below



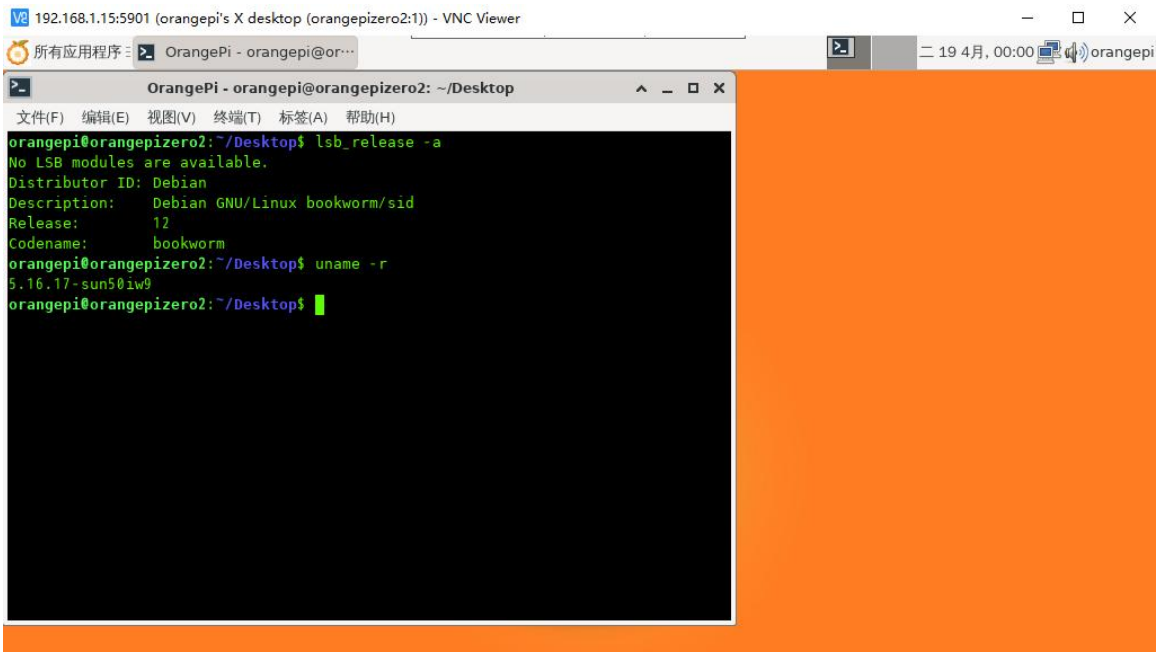
b) Debian10 login shows as below



c) Debian11 login shows as below



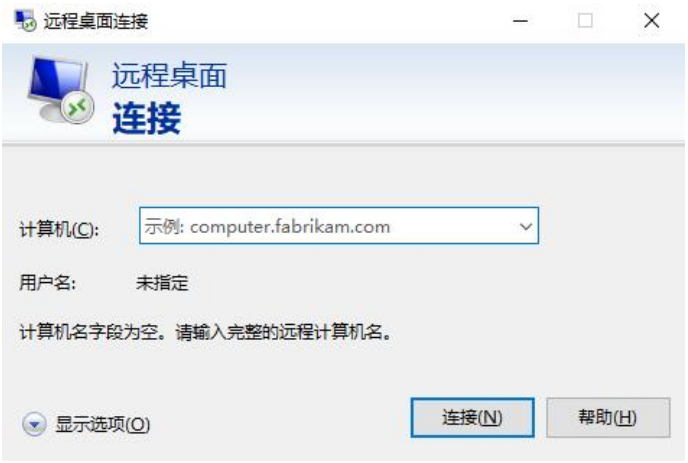
d) Debian12 login shows as below



e) Ubuntu20.04 currently has some problems in testing, please use NoMachine first

5) The steps to use the **remote desktop connection** application that comes with Windows to log in to the Linux system desktop of the development board are:

a. First open the **remote desktop connection** that comes with Windows



b. Then enter the IP address of the development board

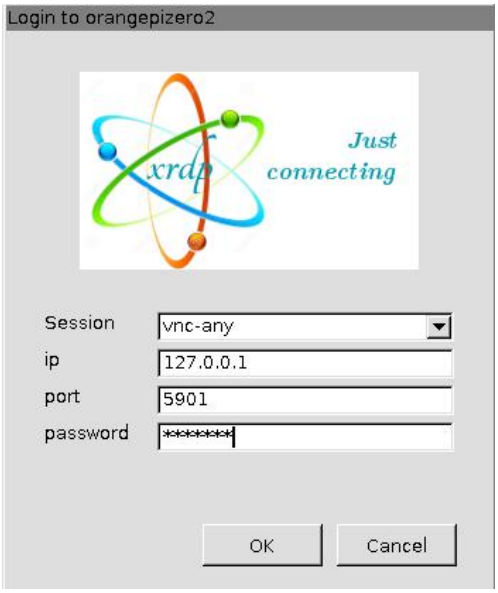
c. Then set the connection information according to the following figure

a) **Session：** Need to select vnc-any

b) **ip：** You can enter 127.0.0.1 or the IP address of the development board

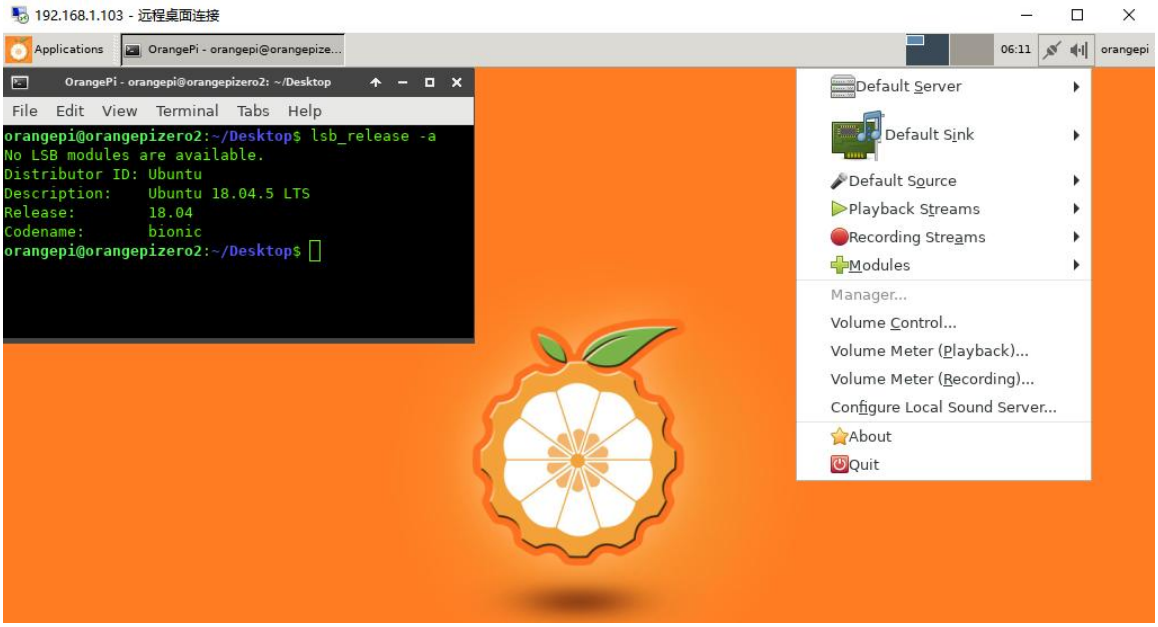
c) **port：** Usually 5901

d) **password:** You need to enter the password you set when you ran the vncserver command in step 2

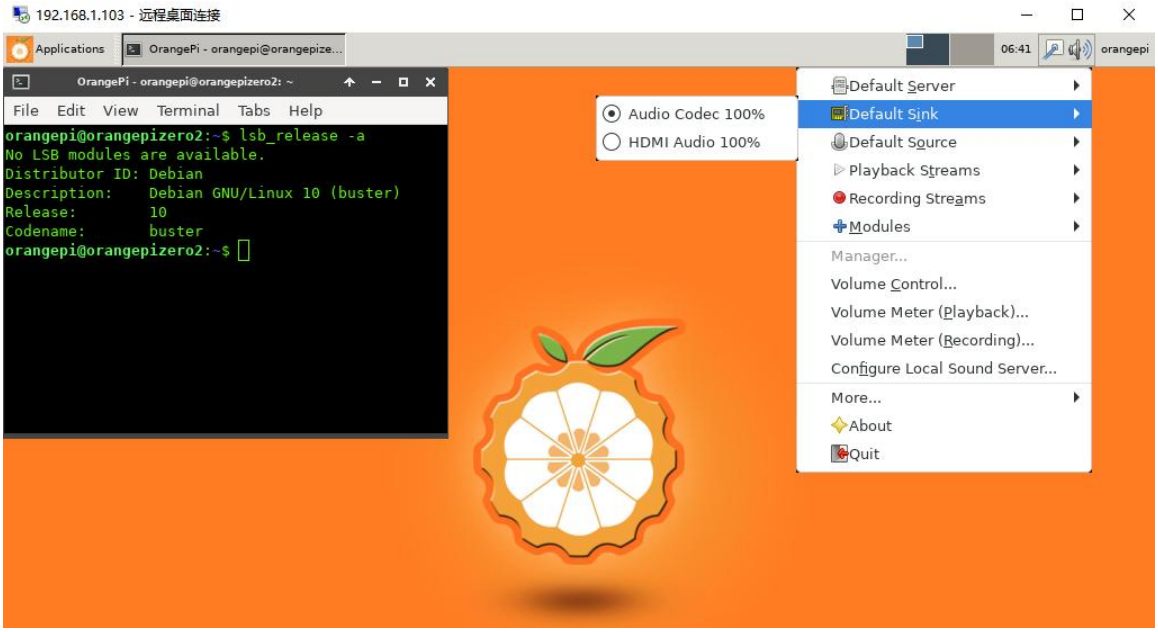


d. The display of successfully logging in to the Linux system desktop of the development board is shown in the following figure

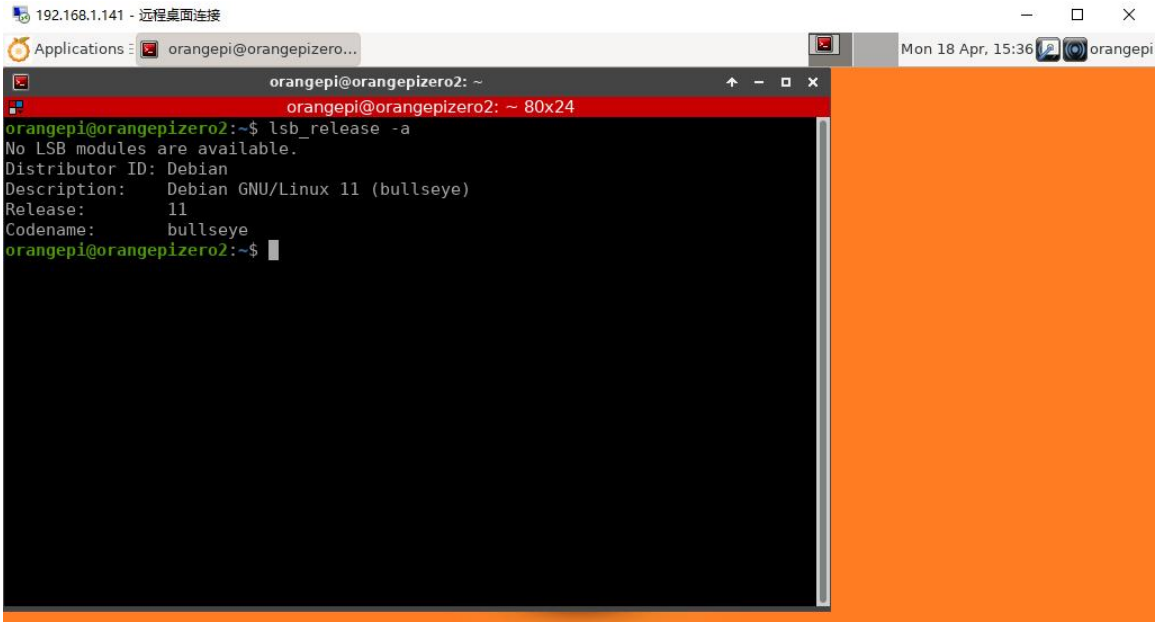
a) Ubuntu18.04 login display as shown below



b) Debian10 login shows as below



c) Debian11 login shows as below



3.37. Tencent ncnn high-performance neural network forward computing framework test

If you don't know what ncnn is, you can read the introduction of ncnn in the README on github.

ncnn github introduction address is：https://github.com/Tencent/ncnn

ncnn

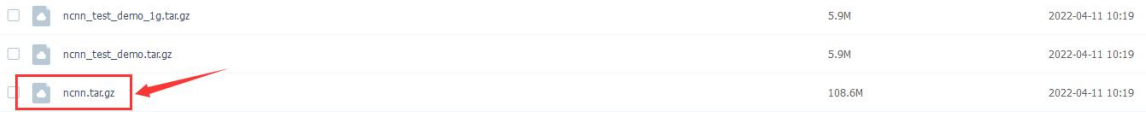
license BSD-3-Clause | downloads 89k | codecov 93% | code quality: c/c++ A+

ncnn is a high-performance neural network inference computing framework optimized for mobile platforms. ncnn is deeply considerate about deployment and uses on mobile phones from the beginning of design. ncnn does not have third party dependencies. It is cross-platform, and runs faster than all known open source frameworks on mobile phone cpu. Developers can easily deploy deep learning algorithm models to the mobile platform by using efficient ncnn implementation, create intelligent APPs, and bring the artificial intelligence to your fingertips. ncnn is currently being used in many Tencent applications, such as QQ, Qzone, WeChat, Pitu and so on.

ncnn 是一个为手机端极致优化的高性能神经网络前向计算框架。ncnn 从设计之初深刻考虑手机端的部署和使用。无第三方依赖，跨平台，手机端 cpu 的速度快于目前所有已知的开源框架。基于 ncnn，开发者能够将深度学习算法轻松移植到手机端高效执行，开发出人工智能 APP，将 AI 带到你的指尖。ncnn 目前已在腾讯多款应用中使用，如 QQ，Qzone，微信，天天P图等。

1) Tencent ncnn source code download command is as follows
   a. The first method: Download the **ncnn.tar.gz** compressed package provided in the Orang Pi Google Drive
      a) From the Google Drive link below, you can download the **ncnn.tar.gz** source code compressed package. Go to the ncnn folder and you can see

https://drive.google.com/drive/folders/1JAj7xT1ViZKH51BZ-dUqM9bBySQILpZ1?usp=sharing



      b) After downloading the **ncnn.tar.gz** compressed package, first upload **ncnn.tar.gz** to the Linux system of the development board
      c) Then extract **ncnn.tar.gz** using the following command

```
orangepi@orangepi:~$ tar zxf ncnn.tar.gz
orangepi@orangepi:~$ ls
ncnn    ncnn.tar.gz
```

   b. The second method: use the git command to download the source code directly, but if the problem of accessing github from the development board is not solved, it is difficult to download successfully. If there is no problem accessing github, it is recommended to use this method, because this method can ensure that the

code is up-to-date.

```
orangepi@orangepi:~$ git clone https://github.com/Tencent/ncnn.git
orangepi@orangepi:~$ cd ncnn
orangepi@orangepi:~/ncnn$ git submodule update --init
```

2) Then install the dependency package

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y build-essential git cmake    \
libprotobuf-dev protobuf-compiler libopencv-dev
```

3) Then open the **benchmark/benchncnn.cpp** file in the source code of ncnn, and remove the test item **vgg16** from it. Since the memory of the development board is only 1GB, when the test reaches vgg16, the benchncnn test will be interrupted and exited due to insufficient memory.

```
orangepi@orangepi:~$ cd ncnn
orangepi@orangepi:~/ncnn$ vim benchmark/benchncnn.cpp

After opening benchmark/benchncnn.cpp, find the following line and comment it out

//benchmark("vgg16", ncnn::Mat(224, 224, 3), opt);
```

If you are not interested in benchmark testing, you can do it without modifying it.

This problem is only in the development board with 1G memory, in the development board with 2G memory, if it is normal in the Orange Pi 3 LTS.

4) Then start compiling, the ncnn compilation command is as follows

```
orangepi@orangepi:~/ncnn$ mkdir build
orangepi@orangepi:~/ncnn$ cd build
orangepi@orangepi:~/ncnn/build$ cmake    \
-DCMAKE_TOOLCHAIN_FILE=../toolchains/aarch64-linux-gnu.toolchain.cmake \
-DNCNN_SIMPLEOCV=ON -DNCNN_BUILD_EXAMPLES=ON ..
orangepi@orangepi:~/ncnn/build$ make -j$(nproc)
[  0%] Built target ncnn-generate-spirv
```

```
[   1%] Generating source absval_arm_arm82.h
[   1%] Generating source convolution1d_arm_arm82.cpp
......
[ 99%] Linking CXX executable squeezenetssd
[ 99%] Built target squeezenetssd
[100%] Linking CXX executable shufflenetv2
[100%] Built target shufflenetv2
```

Without any cooling measures, it takes about 20~25 minutes to compile ncnn directly on the development board, please wait patiently for the compilation to complete.

5) There are some test examples in ncnn, such as **squeezenet** test commands and results are as follows

```
orangepi@orangepi:~/ncnn/build$ cd ../examples
orangepi@orangepi:~/ncnn/examples$ ../build/examples/squeezenet   \
../images/256-ncnn.png
532 = 0.165950
920 = 0.094098
716 = 0.062193
```

6) **benchcnnn** can be used to test the reasoning performance of the neural network. The test method is as follows

a. The **benchncnn** executable file generated by compilation is in the following path. **Note that the execution path of the following command is the top-level directory of the ncnn source code**

```
orangepi@orangepi:~/ncnn$ ls build/benchmark/
benchncnn  CMakeFiles  cmake_install.cmake  Makefile
```

b. First, you need to copy **benchmarkcnn** to the **benchmark** directory

```
orangepi@orangepi:~/ncnn$ cp build/benchmark/benchncnn benchmark
```

c. The usage of **c.benchcnnn** is as follows

```
./benchncnn [loop count] [num threads] [powersave] [gpu device] [cooling down]
```

Parameter

param	options	default
loop count	1~N	4
num threads	1~N	max_cpu_count
powersave	0=all cores, 1=little cores only, 2=big cores only	0
gpu device	-1=cpu-only, 0=gpu0, 1=gpu1 ...	-1
cooling down	0=disable, 1=enable	1

https://github.com/Tencent/ncnn/blob/9d0c36358cec2d1da471574064d4abd8787b45a8/benchmark/README.md

d. **benchncnn** uses the cpu test command and the results are as follows

```
orangepi@orangepi:~/ncnn$ cd benchmark
orangepi@orangepi:~/ncnn/benchmark$ ./benchncnn 4 $(nproc) 0 -1
```

a) Debian Bookworm Linux 5.16 server version system test results

```
orangepi@orangepizero2:~/ncnn/benchmark$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
          squeezenet  min =   85.25  max =   88.18  avg =   86.30
     squeezenet_int8  min =   68.47  max =   68.75  avg =   68.62
           mobilenet  min =  109.89  max =  112.64  avg =  111.55
      mobilenet_int8  min =   60.99  max =   61.78  avg =   61.44
        mobilenet_v2  min =  105.97  max =  107.50  avg =  106.86
        mobilenet_v3  min =  101.99  max =  102.32  avg =  102.16
          shufflenet  min =   59.98  max =   83.20  avg =   66.64
       shufflenet_v2  min =   55.81  max =   57.12  avg =   56.29
             mnasnet  min =   93.07  max =   95.41  avg =   94.18
     proxylessnasnet  min =  100.65  max =  103.87  avg =  101.92
     efficientnet_b0  min =  156.15  max =  157.41  avg =  156.82
   efficientnetv2_b0  min =  162.43  max =  178.05  avg =  173.62
        regnety_400m  min =  111.62  max =  113.63  avg =  112.50
           blazeface  min =   19.83  max =   20.21  avg =   20.03
           googlenet  min =  252.25  max =  277.35  avg =  262.28
      googlenet_int8  min =  200.10  max =  200.59  avg =  200.32
            resnet18  min =  280.52  max =  282.21  avg =  281.05
       resnet18_int8  min =  164.09  max =  164.72  avg =  164.34
             alexnet  min =  231.92  max =  237.94  avg =  234.99
          vgg16_int8  min =  705.68  max =  714.29  avg =  708.92
            resnet50  min =  592.31  max =  595.78  avg =  593.96
       resnet50_int8  min =  411.82  max =  487.82  avg =  431.32
      squeezenet_ssd  min =  295.52  max =  298.63  avg =  297.11
 squeezenet_ssd_int8  min =  191.61  max =  263.06  avg =  214.73
       mobilenet_ssd  min =  246.86  max =  251.50  avg =  249.71
  mobilenet_ssd_int8  min =  145.98  max =  146.35  avg =  146.19
      mobilenet_yolo  min =  500.29  max =  665.17  avg =  559.03
  mobilenetv2_yolov3  min =  347.81  max =  356.19  avg =  350.40
         yolov4-tiny  min =  452.98  max =  458.24  avg =  455.48
           nanodet_m  min =  138.41  max =  146.72  avg =  144.06
    yolo-fastest-1.1  min =   77.42  max =   78.10  avg =   77.68
      yolo-fastestv2  min =   60.28  max =   60.83  avg =   60.57
orangepi@orangepizero2:~/ncnn/benchmark$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux bookworm/sid
Release:        12
Codename:       bookworm
orangepi@orangepizero2:~/ncnn/benchmark$ uname -r
5.16.17-sun50iw9
orangepi@orangepizero2:~/ncnn/benchmark$
```

b) Debian Bullseye Linux 5.16 server version system test results

```
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
          squeezenet  min =   84.73  max =   85.78  avg =   85.05
     squeezenet_int8  min =   68.34  max =   68.71  avg =   68.54
           mobilenet  min =  111.59  max =  117.68  avg =  113.43
      mobilenet_int8  min =   62.66  max =   63.37  avg =   63.07
        mobilenet_v2  min =  105.77  max =  107.41  avg =  106.59
        mobilenet_v3  min =   87.12  max =   88.33  avg =   87.62
          shufflenet  min =   58.86  max =   59.51  avg =   59.20
       shufflenet_v2  min =   55.06  max =   55.81  avg =   55.43
             mnasnet  min =   92.46  max =   94.67  avg =   94.06
     proxylessnasnet  min =   98.56  max =  103.01  avg =  100.81
     efficientnet_b0  min =  156.78  max =  157.07  avg =  156.97
   efficientnetv2_b0  min =  177.06  max =  178.41  avg =  177.72
        regnety_400m  min =  119.82  max =  125.51  avg =  123.79
           blazeface  min =   19.50  max =   23.61  avg =   21.59
           googlenet  min =  257.38  max =  260.14  avg =  258.51
      googlenet_int8  min =  196.01  max =  248.73  avg =  209.44
           resnet18  min =  274.66  max =  286.09  avg =  278.35
       resnet18_int8  min =  167.24  max =  167.51  avg =  167.37
            alexnet  min =  245.09  max =  247.36  avg =  246.20
          vgg16_int8  min =  707.20  max =  709.31  avg =  708.32
           resnet50  min =  590.21  max =  615.26  avg =  602.39
       resnet50_int8  min =  407.66  max =  428.18  avg =  413.04
      squeezenet_ssd  min =  300.67  max =  327.00  avg =  309.63
 squeezenet_ssd_int8  min =  198.15  max =  221.46  avg =  204.54
       mobilenet_ssd  min =  250.40  max =  275.33  avg =  261.94
  mobilenet_ssd_int8  min =  166.05  max =  166.33  avg =  166.16
      mobilenet_yolo  min =  496.32  max =  524.59  avg =  506.88
  mobilenetv2_yolov3  min =  348.41  max =  376.18  avg =  356.31
         yolov4-tiny  min =  457.43  max =  491.69  avg =  466.93
           nanodet_m  min =  139.55  max =  168.32  avg =  152.27
    yolo-fastest-1.1  min =   80.87  max =   81.34  avg =   81.14
      yolo-fastestv2  min =   59.47  max =   62.13  avg =   60.89
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux 11 (bullseye)
Release:        11
Codename:       bullseye
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ uname -r
5.16.17-sun50iw9
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$
```

c) Debian Buster Linux 4.9 server version system test results

```
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
          squeezenet  min =  101.21  max =  102.54  avg =  101.86
     squeezenet_int8  min =   73.11  max =   73.34  avg =   73.23
           mobilenet  min =  129.88  max =  141.39  avg =  133.65
      mobilenet_int8  min =   65.91  max =   74.71  avg =   68.30
         mobilenet_v2  min =  119.89  max =  120.54  avg =  120.27
         mobilenet_v3  min =   98.65  max =   99.71  avg =   99.10
           shufflenet  min =   65.46  max =   66.61  avg =   66.07
        shufflenet_v2  min =   60.29  max =   62.68  avg =   61.16
              mnasnet  min =  107.85  max =  109.11  avg =  108.53
     proxylessnasnet  min =  110.94  max =  112.82  avg =  111.96
     efficientnet_b0  min =  159.31  max =  160.83  avg =  159.90
   efficientnetv2_b0  min =  185.99  max =  189.42  avg =  188.07
        regnety_400m  min =  124.11  max =  125.88  avg =  125.16
           blazeface  min =   22.18  max =   23.21  avg =   22.86
           googlenet  min =  273.08  max =  282.78  avg =  277.98
      googlenet_int8  min =  203.28  max =  209.00  avg =  205.70
            resnet18  min =  315.77  max =  318.84  avg =  317.48
       resnet18_int8  min =  183.04  max =  187.42  avg =  184.66
             alexnet  min =  270.63  max =  275.00  avg =  272.14
          vgg16_int8  min =  785.36  max =  818.18  avg =  795.80
            resnet50  min =  644.28  max =  659.93  avg =  650.18
       resnet50_int8  min =  412.37  max =  414.93  avg =  413.83
      squeezenet_ssd  min =  329.54  max =  336.00  avg =  331.96
 squeezenet_ssd_int8  min =  211.66  max =  214.66  avg =  212.80
       mobilenet_ssd  min =  264.88  max =  281.44  avg =  270.00
  mobilenet_ssd_int8  min =  141.64  max =  148.24  avg =  143.50
      mobilenet_yolo  min =  557.36  max =  581.72  avg =  571.21
  mobilenetv2_yolov3  min =  353.86  max =  369.12  avg =  361.83
         yolov4-tiny  min =  501.15  max =  515.92  avg =  506.44
           nanodet_m  min =  153.47  max =  156.90  avg =  154.92
    yolo-fastest-1.1  min =   82.01  max =   83.51  avg =   82.62
      yolo-fastestv2  min =   65.71  max =   66.68  avg =   66.40
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux 10 (buster)
Release:        10
Codename:       buster
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ uname -r
4.9.170-sun50iw9
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$
```

d) Ubuntu Focal Linux4.9 server version system test results

```
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
          squeezenet  min =   97.04  max =   97.87  avg =   97.42
     squeezenet_int8  min =   74.87  max =   76.53  avg =   75.36
          mobilenet  min =  129.67  max =  134.08  avg =  131.46
     mobilenet_int8  min =   67.85  max =   69.84  avg =   68.45
       mobilenet_v2  min =  118.65  max =  119.99  avg =  119.47
       mobilenet_v3  min =   97.37  max =   98.85  avg =   98.07
         shufflenet  min =   65.06  max =   66.28  avg =   65.45
      shufflenet_v2  min =   60.81  max =   61.40  avg =   61.11
             mnasnet  min =  103.68  max =  107.60  avg =  105.52
    proxylessnasnet  min =  110.98  max =  115.12  avg =  113.02
    efficientnet_b0  min =  157.77  max =  159.13  avg =  158.55
   efficientnetv2_b0  min =  183.36  max =  188.11  avg =  184.91
       regnety_400m  min =  124.32  max =  124.85  avg =  124.60
          blazeface  min =   21.12  max =   22.68  avg =   21.95
          googlenet  min =  276.24  max =  281.58  avg =  278.66
     googlenet_int8  min =  199.85  max =  203.42  avg =  201.28
          resnet18  min =  311.15  max =  331.70  avg =  317.94
     resnet18_int8  min =  185.04  max =  188.46  avg =  186.23
           alexnet  min =  268.10  max =  272.91  avg =  269.84
         vgg16_int8  min =  794.69  max =  817.88  avg =  807.30
          resnet50  min =  657.25  max =  659.90  avg =  658.07
     resnet50_int8  min =  411.41  max =  414.83  avg =  413.04
     squeezenet_ssd  min =  329.91  max =  335.23  avg =  331.41
 squeezenet_ssd_int8  min =  211.84  max =  218.02  avg =  214.15
      mobilenet_ssd  min =  266.72  max =  282.67  avg =  271.74
  mobilenet_ssd_int8  min =  140.36  max =  140.97  avg =  140.64
     mobilenet_yolo  min =  529.34  max =  559.91  avg =  541.46
  mobilenetv2_yolov3  min =  371.21  max =  383.87  avg =  377.91
        yolov4-tiny  min =  501.76  max =  516.57  avg =  508.18
          nanodet_m  min =  155.52  max =  166.02  avg =  158.22
    yolo-fastest-1.1  min =   85.44  max =   89.75  avg =   86.63
      yolo-fastestv2  min =   66.37  max =   67.03  avg =   66.68
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.4 LTS
Release:        20.04
Codename:       focal
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ uname -r
4.9.170-sun50iw9
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$
```

e) Ubuntu Bionic Linux4.9 server version system test results

```
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ ./benchncnn 4 $(nproc) 0 -1
loop_count = 4
num_threads = 4
powersave = 0
gpu_device = -1
cooling_down = 1
          squeezenet  min =  100.80  max =  104.88  avg =  101.89
     squeezenet_int8  min =   73.56  max =   74.35  avg =   74.05
           mobilenet  min =  127.57  max =  141.65  avg =  131.51
      mobilenet_int8  min =   66.00  max =   66.70  avg =   66.32
        mobilenet_v2  min =  119.05  max =  125.64  avg =  121.20
        mobilenet_v3  min =   97.42  max =  100.12  avg =   98.53
          shufflenet  min =   65.89  max =   74.87  avg =   68.74
       shufflenet_v2  min =   60.22  max =   61.39  avg =   60.87
             mnasnet  min =  107.49  max =  111.98  avg =  109.83
     proxylessnasnet  min =  109.60  max =  110.65  avg =  110.16
     efficientnet_b0  min =  158.67  max =  167.94  avg =  161.66
   efficientnetv2_b0  min =  188.55  max =  196.77  avg =  190.78
        regnety_400m  min =  124.03  max =  133.10  avg =  126.77
           blazeface  min =   22.02  max =   22.79  avg =   22.46
           googlenet  min =  273.03  max =  284.48  avg =  277.67
      googlenet_int8  min =  199.92  max =  212.49  avg =  203.93
           resnet18  min =  313.82  max =  325.06  avg =  318.89
      resnet18_int8  min =  174.94  max =  180.87  avg =  177.25
            alexnet  min =  272.26  max =  279.92  avg =  274.40
          vgg16_int8  min =  773.62  max =  817.88  avg =  794.97
           resnet50  min =  641.58  max =  661.81  avg =  650.93
      resnet50_int8  min =  408.98  max =  415.24  avg =  411.01
     squeezenet_ssd  min =  329.31  max =  336.26  avg =  333.11
 squeezenet_ssd_int8  min =  210.64  max =  211.61  avg =  211.16
       mobilenet_ssd  min =  266.34  max =  278.02  avg =  271.36
  mobilenet_ssd_int8  min =  140.69  max =  147.47  avg =  142.61
      mobilenet_yolo  min =  566.12  max =  578.98  avg =  571.36
  mobilenetv2_yolov3  min =  350.73  max =  370.09  avg =  362.37
        yolov4-tiny  min =  499.14  max =  520.08  avg =  509.07
          nanodet_m  min =  154.07  max =  156.35  avg =  155.36
    yolo-fastest-1.1  min =   84.16  max =   84.74  avg =   84.45
      yolo-fastestv2  min =   66.01  max =   67.16  avg =   66.54
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.6 LTS
Release:        18.04
Codename:       bionic
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$ uname -r
4.9.170-sun50iw9
orangepi@orangepizero2:~/ncnn_test_demo_1g/benchncnn_demo$
```

7) NanoDet is an ultra-fast and lightweight mobile Anchor-free object detection model. The test method is as follows

a. The compiled **nanodet** executable file is in the following path. **Note that the execution path of the following command is the upper level directory of the ncnn source code**

```
orangepi@orangepi:~$ ls ncnn/build/examples/nanodet
ncnn/build/examples/nanodet
```

b. First create a new **nanonodet_demo** folder

```
orangepi@orangepi:~$ mkdir nanodet_demo
```

c. Then copy the compiled **nanonodet** executable program to the **nanonodet_demo** folder

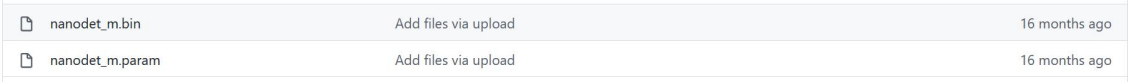
```
orangepi@orangepi:~$ cp ncnn/build/examples/nanodet nanodet_demo/
```

d. Then you need to download the **nanonodet** model file and upload it to the **nanonodet_demo** folder

a) **Nanodet** The model file download address is as follows

```
https://github.com/nihui/ncnn-assets/tree/master/models
```

b) Open the link above, find the two files **nanodet_m.bin** and **nanodet_m.param**, download them, and upload them to the **nanodet_demo** folder of the Linux system of the development board

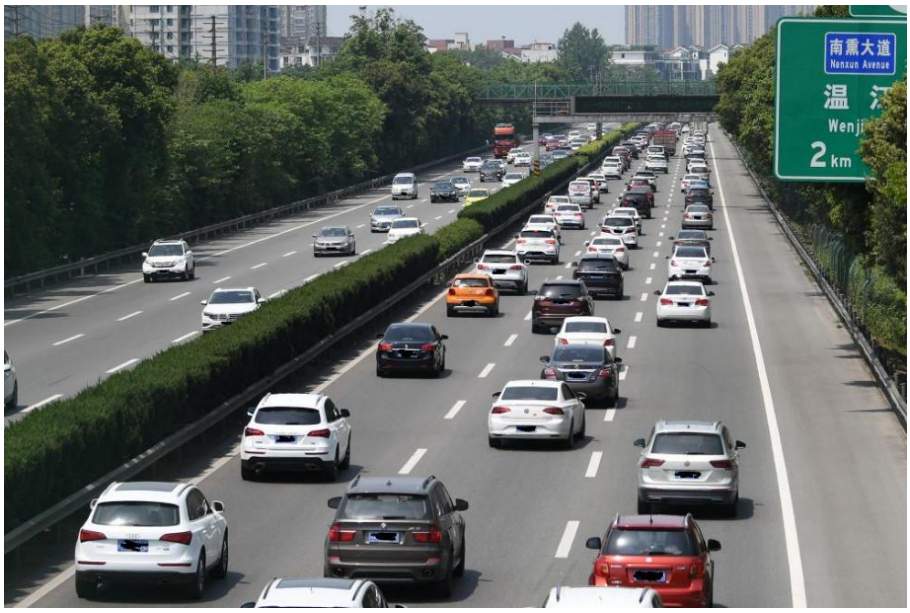


c) At this point, there should be the following three files in the **nanodet_demo** folder

```
orangepi@orangepi:~$ cd nanodet_demo
orangepi@orangepi:~/nanodet_demo$ ls
nanodet  nanodet_m.bin  nanodet_m.param
```

e. Then you need to put the pictures you want to detect in the **nanodet_demo** folder, such as the picture below with many cars (you can use your mobile phone to take a few pictures of traffic or animals, etc.)

```
orangepi@orangepi:~/nanodet_demo$ ls
car.jpg  nanodet  nanodet_m.bin  nanodet_m.param
```



f. Then run the following command to use **nanodet** for target detection, please replace **car.jpg** with the name of your image

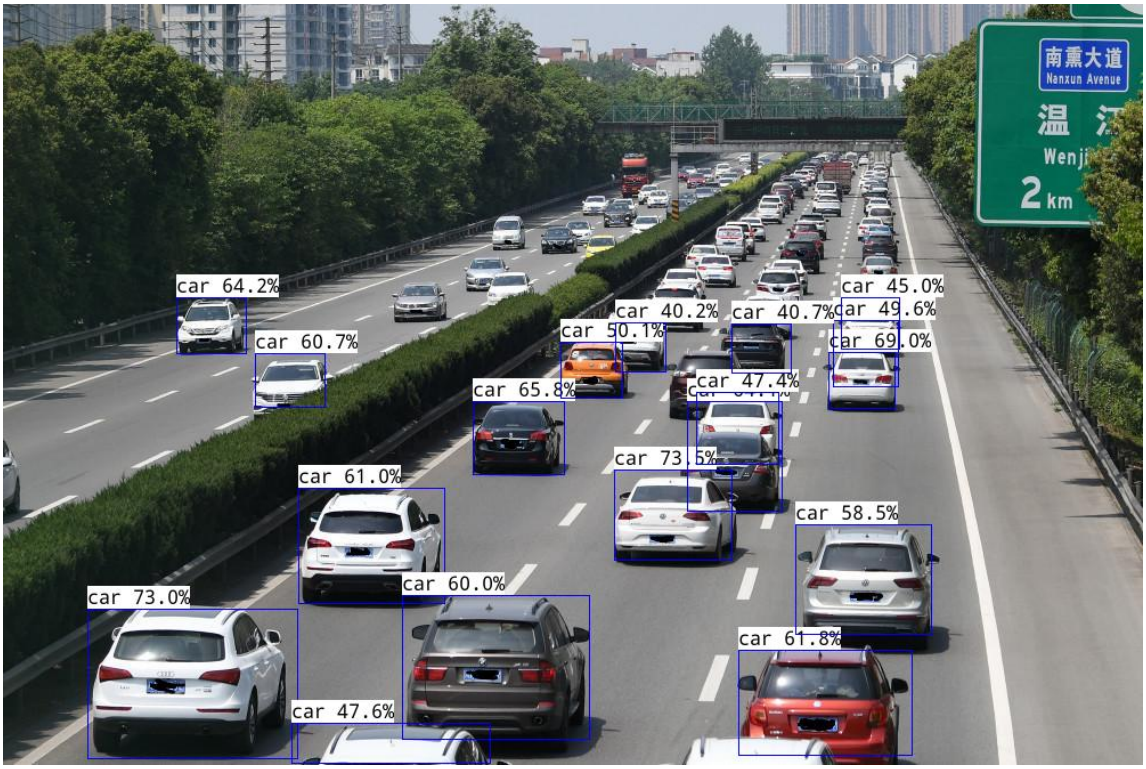
```
orangepi@orangepi:~/nanodet_demo$ ./nanodet car.jpg
2 = 0.73488 at 536.36 408.79 103.68 x 79.63
2 = 0.73003 at 74.47 530.85 184.61 x 131.70
2 = 0.68989 at 724.94 305.76 58.30 x 49.73
2 = 0.65828 at 412.10 348.38 80.33 x 64.65
2 = 0.64167 at 152.09 257.67 61.52 x 49.91
2 = 0.64124 at 600.63 348.06 83.82 x 96.93
2 = 0.61759 at 645.80 566.98 152.59 x 92.63
2 = 0.61004 at 259.78 424.55 128.62 x 101.88
2 = 0.60663 at 221.76 306.18 61.54 x 47.73
2 = 0.60043 at 350.11 518.50 164.37 x 126.59
2 = 0.58546 at 695.82 456.88 119.12 x 96.87
2 = 0.50075 at 489.94 296.66 54.39 x 49.01
2 = 0.49616 at 728.32 277.73 57.70 x 58.59
2 = 0.47553 at 253.21 630.12 174.18 x 35.88
2 = 0.47408 at 608.36 340.09 72.34 x 63.40
2 = 0.44989 at 735.15 257.97 51.55 x 45.79
2 = 0.40665 at 639.82 280.51 52.43 x 43.13
2 = 0.40169 at 537.50 279.67 44.98 x 43.54
imshow save image to image.png
```

```
waitKey stub
```

g. The result of the detection will be saved in a picture named **image.png**

```
orangepi@orangepi:~/nanodet_demo$ ls
car.jpg  image.png  nanodet  nanodet_m.bin  nanodet_m.param
```

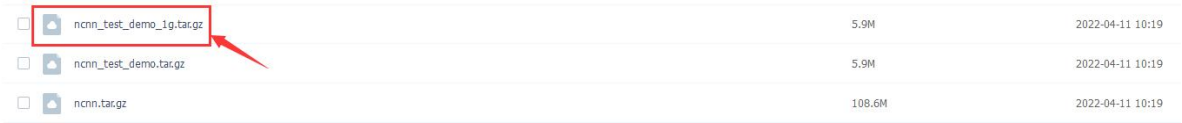
h. If you are using a desktop version of Linux system, you can directly open **image.png** to view it. If you are using a server version of Linux system, you can copy **image.png** to your computer for viewing. The content of **image.png** is shown in the figure below. You can see that the upper left corner of the recognized object will display the type of object and the percentage of reliability



8) In order to facilitate the testing of **benchncnn** and **nanodet**, I compiled an executable file containing only **benchncnn** and **nanodet** and the model files required for the test and packaged it into a **ncnn_test_demo_1g.tar.gz** compressed package and placed it on Google Drive, no need to download Compile the source code of **ncnn**, use this executable program to start the test directly

a. You can download the **ncnn_test_demo_1g.tar.gz** compressed package from the Google Drive link below. Go to the **ncnn** folder and you can see

```
https://drive.google.com/drive/folders/1JAj7xT1ViZKH51BZ-dUqM9bBySQILpZ1?usp=sharing
```



b. downloading the **ncnn_test_demo_1g.tar.gz** compressed package, first upload the **ncnn_test_demo_1g.tar.gz** compressed package to the development board linux system

c. Then use the following command to extract **ncnn_test_demo_1g.tar.gz**

```
orangepi@orangepi:~$ tar ncnn_test_demo_1g.tar.gz
```

d. After decompressing, enter the **ncnn_test_demo_1g** directory and you can see that it contains two subfolders, **benchncnn_demo** and **nanonodet_demo**, which are used to test **benchncnn** and **nanodet** respectively

```
orangepi@orangepi:~$ cd ncnn_test_demo_1g
orangepi@orangepi:~/ncnn_test_demo_1g$ ls
benchncnn_demo  nanodet_demo
```

e. Enter the **benchncnn_demo** folder, and then run the command **./benchncnn 4 $(nproc) 0 -1** to directly test the inference performance of the neural network

```
orangepi@orangepi:~/ncnn_test_demo_1g$ cd benchncnn_demo
orangepi@orangepi:~/ncnn_test_demo_1g/benchncnn_demo$ ./benchncnn 4  \
$(nproc) 0 -1
```

f. Enter the **nanonodet_demo** folder, and then run the **./nanodet car.jpg** command to directly use **nanodet** to detect objects in the **car.jpg** image. You can also put the image you want to detect in the **nanonodet_demo** folder, and then Use **nanodet** to detect

```
orangepi@orangepi:~/ncnn_test_demo_1g$ cd nanodet_demo
orangepi@orangepi:~/ncnn_test_demo_1g/nanodet_demo$ ./nanodet car.jpg
```

Note that the content demonstrated in this section is mainly to prove that ncnn can be compiled and run normally on the Orange Pi development board and system. If there is any problem with the content demonstrated in this section, you can provide technical support (such as ncnn source code download failure, ncnn compilation There are problems, there are problems with benchncnn and nanodet tests), but other things beyond this section cannot provide technical support, please do your own research.

3.38. Installation and testing method of face_recognition face recognition library

Note that the content in this section is tested on the desktop version of Linux system, so please make sure that the system used by the development board is the desktop version system.

In addition, the following installation tests are carried out under the orangepi user, please keep the environment consistent.

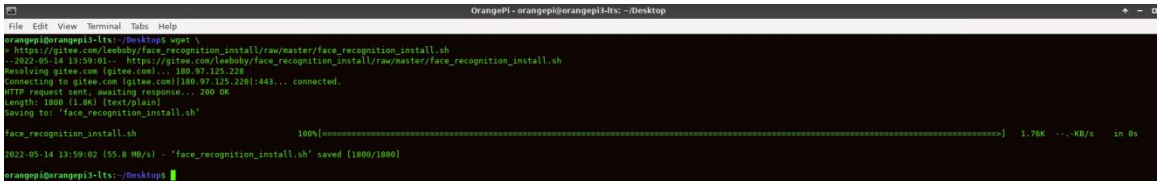
face_recognition / The address of the source code repository is:
https://github.com/ageitgey/face_recognition

face_recognition / The Chinese version of the documentation is:
https://github.com/ageitgey/face_recognition/blob/master/README_Simplified_Chinese.md

3.38.1. Automatic installation of face_recognition using script

1) First open a terminal in the desktop and download **face_recognition_install.sh**

```
orangepi@orangepi:~/Desktop$ wget \
https://gitee.com/leeboby/face_recognition_install/raw/master/face_recognition_install.sh
```



2) Then execute the following command to start installing **face_recognition**

```
orangepi@orangepi:~/Desktop$ bash face_recognition_install.sh
```

3) After face_recognition is installed, it will automatically download the source code of face_recognition, and then automatically run some examples in face_recognition. If you can see the following pictures pop up on the desktop at the end, it means that the face_recognition installation test is successful

3.38.2. Manual installation of face_recognition

1) First create a new **~/.pip** directory, then add the **pip.conf** configuration file, and set the image source of pip to Tsinghua source in it. The commands to be executed are as follows:

```
orangepi@orangepi:~$ mkdir -p ~/.pip
orangepi@orangepi:~$ cat <<EOF > ~/.pip/pip.conf
[global]
timeout = 6000
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
trusted-host = pypi.tuna.tsinghua.edu.cn
EOF
```

2) Then install the dependency package

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y python3-pip libopencv-dev \
python3-opencv imagemagick python3-scipy python3-setuptools python3-wheel \
python3-dev cmake python3-testresources
```

3) Then update pip3

```
orangepi@orangepi:~$ python3 -m pip install -U pip setuptools wheel
```

4) The download address of the dlib whl file is as follows: Before installing **face_recognition,** you first need to install the **dlib** library. Since the dlib library is relatively slow to compile and install on the development board, I saved a compiled dlib whl file on **gitee** and installed it directly after downloading it. The download address of the dlib whl file is as follows:

https://gitee.com/leeboby/python_whl

a. First download the python_whl repository to the Linux system of the development board

```
orangepi@orangepi:~$ git clone --depth=1 https://gitee.com/leeboby/python_whl
```

b. In the python_whl folder, you can see that there are multiple versions of the dlib installation package. The Linux systems corresponding to different versions of dlib are as follows:

Ubuntu18.04	**dlib-19.24.0-cp36-cp36m-linux_aarch64.whl**
Ubuntu20.04	**dlib-19.24.0-cp38-cp38-linux_aarch64.whl**
Ubuntu22.04	**dlib-19.24.0-cp310-cp310-linux_aarch64.whl**
Debian10	**dlib-19.24.0-cp37-cp37m-linux_aarch64.whl**
Debian11	**dlib-19.24.0-cp39-cp39-linux_aarch64.whl**

c. Then you can start to install dlib, the command is as follows

a) Ubuntu18.04

```
orangepi@orangepi:~$ cd python_whl
orangepi@orangepi:~/python_whl$ python3 -m pip install
dlib-19.24.0-cp36-cp36m-linux_aarch64.whl
```

b) Ubuntu20.04

```
orangepi@orangepi:~$ cd python_whl
orangepi@orangepi:~/python_whl$ python3 -m pip install
dlib-19.24.0-cp38-cp38-linux_aarch64.whl
```

c) Ubuntu22.04

```
orangepi@orangepi:~$ cd python_whl
orangepi@orangepi:~/python_whl$ python3 -m pip install
dlib-19.24.0-cp310-cp310-linux_aarch64.whl
```

d) Debian10

```
orangepi@orangepi:~$ cd python_whl
orangepi@orangepi:~/python_whl$ python3 -m pip install
dlib-19.24.0-cp37-cp37m-linux_aarch64.whl
```

e) Debian11

```
orangepi@orangepi:~$ cd python_whl
orangepi@orangepi:~/python_whl$ python3 -m pip install
dlib-19.24.0-cp39-cp39-linux_aarch64.whl
```

d. After installation, if the version number of dlib can be printed normally by using the following command, it means that dlib is installed correctly

```
orangepi@orangepi:~/python_whl$ python3 -c "import dlib; print(dlib.__version__)"
19.24.0
```

5) Then install **face_recognition_models-0.3.0-py2.py3-none-any.whl**

```
orangepi@orangepi:~/python_whl$ python3 -m pip install \
face_recognition_models-0.3.0-py2.py3-none-any.whl
```

6) Then install **face_recognition**

```
orangepi@orangepi:~$ python3 -m pip install face_recognition
```

7) Then you need to **reopen a terminal** to find and run the two commands **face_detection** and **face_recognition**

a. face_recognition command to identify whose face is in a single image or a folder of images

b. face_detection The command is used to locate the position of the face in a single image or a folder of images

```
orangepi@orangepi:~$ which face_detection
/usr/local/bin/face_detection
orangepi@orangepi:~$ which face_recognition
/usr/local/bin/face_recognition
```

Or run the following commands in the terminal, you can find the above two commands without reopening the terminal

```
orangepi@orangepi:~$ export PATH=/home/orangepi/.local/bin:$PATH
```

3.38.3. Test method of face_recognition

Note that the following operations are demonstrated on the desktop, so please connect the HDMI display first, or use NoMachine/VNC to log in to the Linux desktop remotely to test.

1) There are some sample codes in the source code of **face_recognition**, which we can use for testing directly. The download address of the source code of face_recognition is as follows:

a. GitHub official download address

```
orangepi@orangepi:~$ git clone https://github.com/ageitgey/face_recognition.git
```

b. Gitee image download address

```
orangepi@orangepi:~$ git clone https://gitee.com/leeboby/face_recognition.git
```

2) The path of the face_recognition sample code is as follows

```
face_recognition/examples
```

3) The link to the Chinese documentation of face_recognition is as follows, please read it carefully before using face_recognition

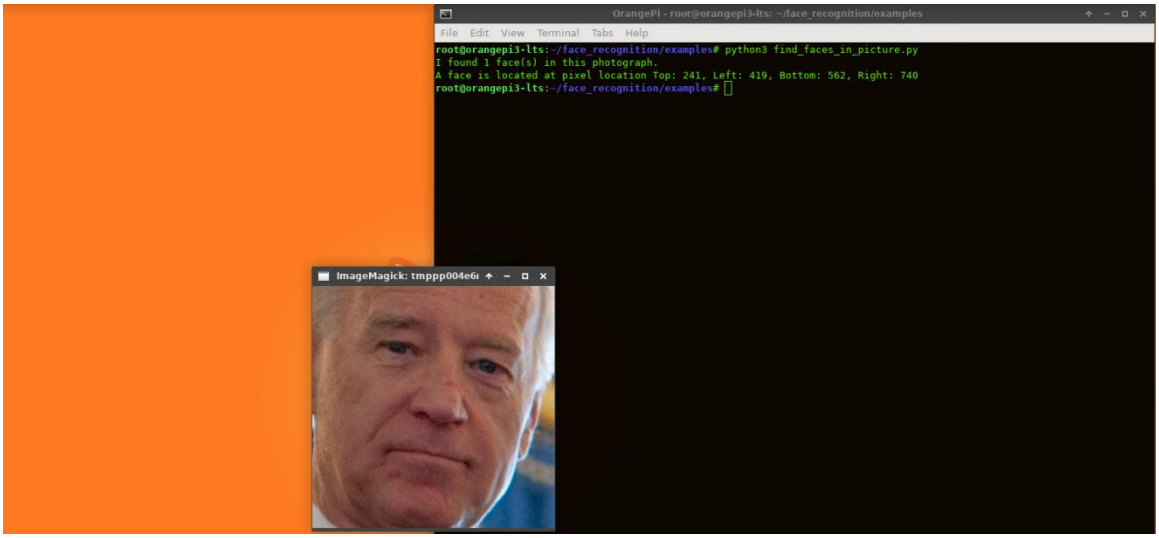
```
https://github.com/ageitgey/face_recognition/blob/master/README_Simplified_Chinese.md
```

4) **find_faces_in_picture.py** is used to locate the position of the face in the picture. The test steps are as follows

a. Open a terminal on the desktop, enter the **face_recognition/examples** directory, and execute the following command

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ python3 find_faces_in_picture.py
I found 1 face(s) in this photograph.
A face is located at pixel location Top: 241, Left: 419, Bottom: 562, Right: 740
```

b. Wait for a while and the following picture will pop up, this is the face located in the test picture

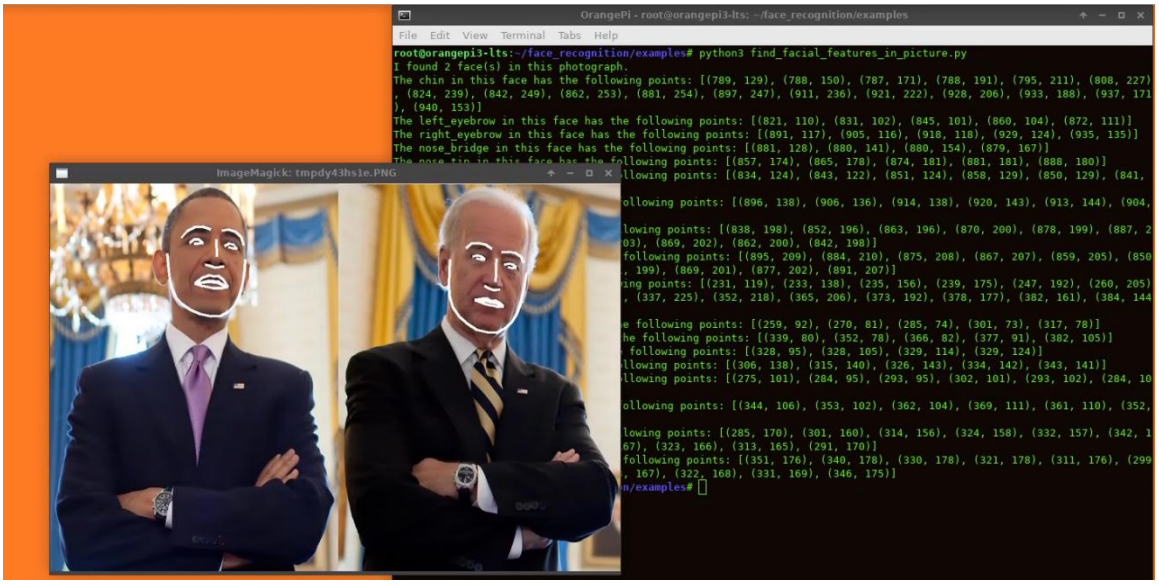


5) **find_facial_features_in_picture.py** is used to identify the key points of the face in a single picture. The test steps are as follows

a. Open a terminal on the desktop, enter the **face_recognition/examples** directory, and execute the following command

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ python3 \
find_facial_features_in_picture.py
```

b. After waiting for a while, the picture below will pop up, and you can see that the outlines of the faces are marked



6) **identify_and_draw_boxes_on_faces.py** is used to identify faces and use box labels.

The test steps are as follows

a. Open a terminal on the desktop, enter the **face_recognition/examples** directory, and execute the following command

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ python3 \
identify_and_draw_boxes_on_faces.py
```

b. After waiting for a while, the following picture will pop up. You can see that the faces in the picture are marked with boxes, and the names of the characters are displayed correctly.

7) **face_distance.py** is used to compare whether two faces belong to the same person at different precisions. First open a terminal, then enter the **face_recognition/examples** directory, and then execute the following command to see the output of the test

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ python3 face_distance.py
The test image has a distance of 0.35 from known image #0
- With a normal cutoff of 0.6, would the test image match the known image? True
- With a very strict cutoff of 0.5, would the test image match the known image? True

The test image has a distance of 0.82 from known image #1
- With a normal cutoff of 0.6, would the test image match the known image? False
- With a very strict cutoff of 0.5, would the test image match the known image?
  False
```

8) **recognize_faces_in_pictures.py** is used to identify who the faces in unknown

pictures are. First open a terminal, then enter the **face_recognition/examples** directory, and then execute the following command, wait for one end to see the test results

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ python3 \
recognize_faces_in_pictures.py
Is the unknown face a picture of Biden? False
Is the unknown face a picture of Obama? True
Is the unknown face a new person that we've never seen before? False
```

9) **facerec_from_webcam_faster.py** is used to identify the face in the USB camera. The test steps are as follows:

a. First, please insert the USB camera into the USB interface of the development board, and then use the **v4l2-ctl** (**note that the l in v4l2 is a lowercase letter l, not the number 1**) command to check the serial number of the device node of the USB camera

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y v4l-utils
orangepi@orangepi:~$ v4l2-ctl --list-devices
cedrus (platform:cedrus):
        /dev/video0

USB2.0 UVC PC Camera: USB2.0 UV (usb-5311000.usb-1):
        /dev/video1
        /dev/video2
```

b. Then open a terminal on the desktop, enter the **face_recognition/examples** directory, and first modify the device serial number of the camera used in **facerec_from_webcam_faster.py**. For example, through the **v4l2-ctl --list-devices** command above, you can see that the USB camera is **/dev/video1**, **then modify 0 in cv2.VideoCapture(0)** to **1**

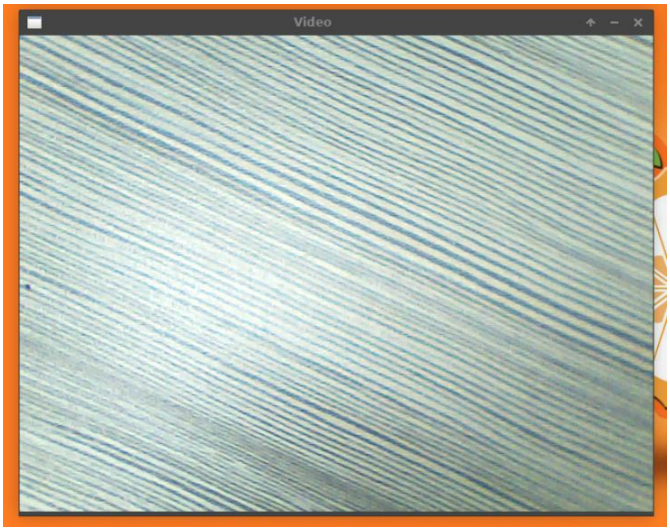
```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ vim \
facerec_from_webcam_faster.py
video_capture = cv2.VideoCapture(1)
```

c. Then execute the following command to run **facerec_from_webcam_faster.py**

```
orangepi@orangepi:~/face_recognition/examples$ python3 \
```

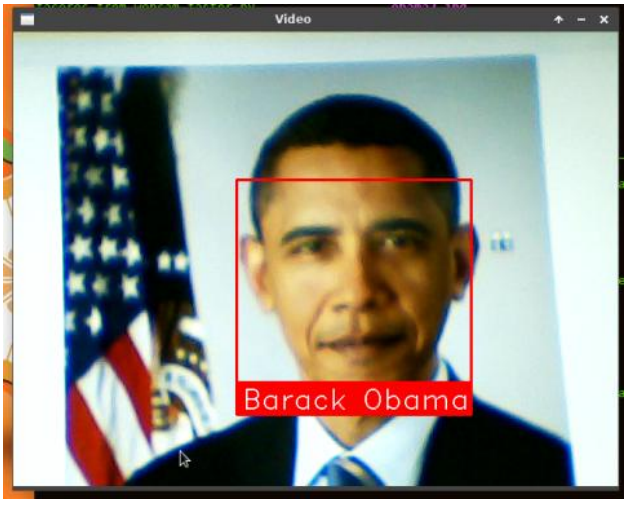
facerec_from_webcam_faster.py

d. Wait for a while and the display screen of the camera will pop up



e. At this point, you can point the camera at yourself. When the camera detects a face, it will use a box to frame the detected face. **Note that when detecting faces, the picture displayed by the camera will be relatively stuck, please do not move too fast**

f. You can also open a picture of Obama, and then use the camera to aim at the opened picture, you can see that not only the face can be marked, but also the name of the detected face can be displayed correctly. **Note that when detecting faces, the picture displayed by the camera will be relatively stuck, please do not move too fast**



10) **web_service_example.py** is a very simple case of using a web service to upload pictures to run face recognition. The back-end server will identify whether the picture is

Obama, and output the recognition results as json key-value pairs. The test steps are as follows :

a. Open a terminal on the desktop, then enter the **face_recognition/examples** directory, and execute the following command **(if it is face_recognition that is automatically installed using a script, then you don't need to install flask)**

```
orangepi@orangepi:~$ python3 -m pip install flask
orangepi@orangepi:~$ cd face_recognition/examples
root@orangepi:~/face_recognition/examples$ python3 web_service_example.py
 * Serving Flask app 'web_service_example' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on all addresses (0.0.0.0)
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://127.0.0.1:5001
 * Running on http://192.168.1.79:5001 (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 500-161-390
```

b. Then run the following command to return the result of image recognition (note that the execution path of the following command is **face_recognition/examples**)

```
orangepi@orangepi:~/face_recognition/examples$ curl -XPOST -F \
"file=@obama2.jpg" http://127.0.0.1:5001
{
  "face_found_in_image": true,
  "is_picture_of_obama": true
}
```

c. We can also copy the picture **face_recognition/examples/obama2.jpg** to other Linux computers, of course, we can also prepare a picture named **obama2.jpg** by ourselves, and then use the following commands to remotely remote from the Linux computer Identify the face through the service running on the development board **(note that the IP address in the command needs to be replaced with the IP address of the development board, and the file name after the file needs to be replaced with the name of the image you want to**

test)

```
test@test:~$ curl -XPOST -F "file=@obama2.jpg" http://192.168.1.79:5001
{
  "face_found_in_image": true,
  "is_picture_of_obama": true
}
```

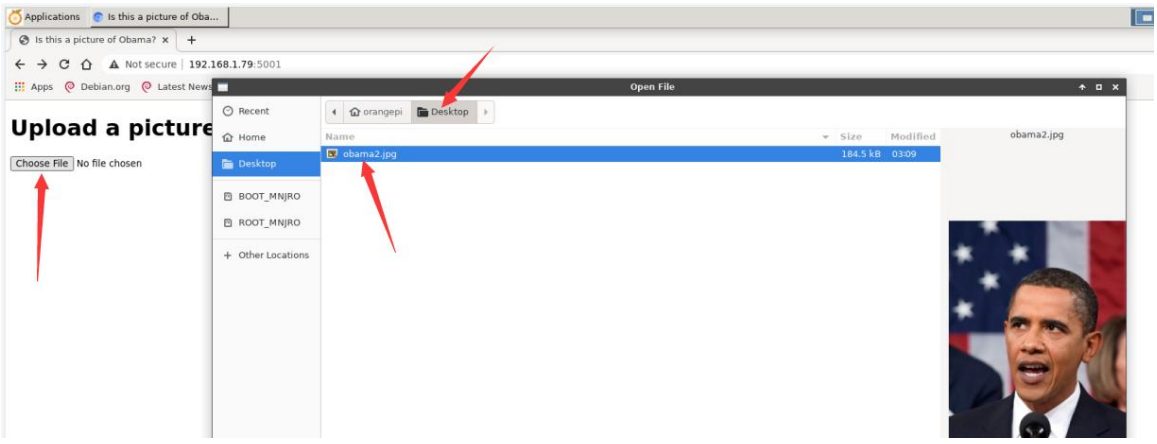
d. The method of using the browser test is as follows:

a) First open the browser, then enter the IP address of **the development board: 5001** in the address bar of the browser, and then you can see the following page

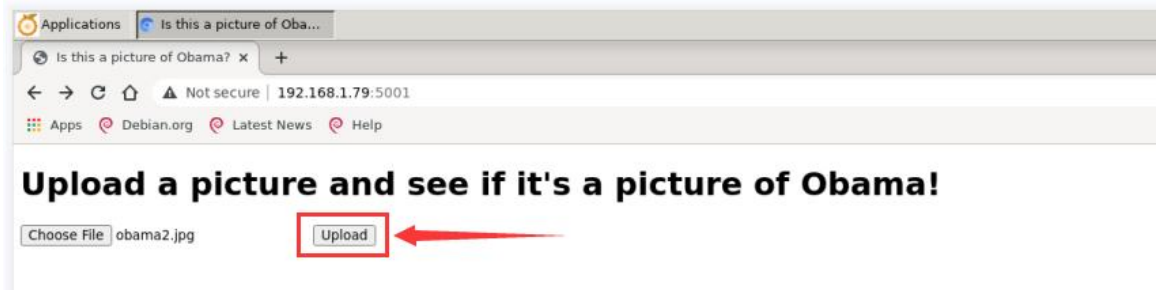
b) Then copy obama2.jpg to the desktop

```
orangepi@orangepi:~/face_recognition/examples$ cp obama2.jpg \
/home/orangepi/Desktop/
```

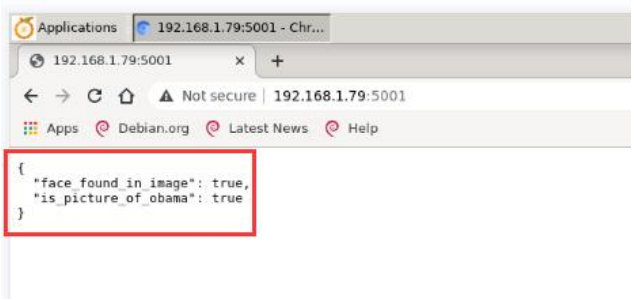
c) Then select the image you just copied in the browser



d) Then click **Upload** to upload the image you just selected for face recognition



e) After a period of time, the test results will be displayed



11) **face_detection** command test example

a. The a.face_detection command-line tool can locate the face position (output pixel coordinates) in a single image or a folder of images. Use **face_detection --help** to view the help information of the face_detection command

```
orangepi@orangepi:~$ face_detection --help
Usage: face_detection [OPTIONS] IMAGE_TO_CHECK

Options:
  --cpus INTEGER    number of CPU cores to use in parallel. -1 means "use all in
                    system"
  --model TEXT      Which face detection model to use. Options are "hog" or
                    "cnn".
  --help            Show this message and exit.
```

b. An example of detecting a single image is as follows:

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ face_detection obama2.jpg
obama2.jpg,302,474,611,164
```

c. An example of using multiple cores to detect multiple images in parallel is as follows:

a) First go to the **face_recognition/examples** folder

b) Then create a new test folder
c) Then copy the jpg images to the test folder
d) Then use all cpus to run **face_detection** in parallel to check the pictures in the test folder, where **--cpus -1** means use the cpu

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ mkdir test
orangepi@orangepi:~/face_recognition/examples$ cp *.jpg test
orangepi@orangepi:~/face_recognition/examples$ face_detection --cpus -1 test
test/obama-240p.jpg,29,261,101,189
test/obama_small.jpg,65,215,169,112
test/obama2.jpg,302,474,611,164
test/two_people.jpg,62,394,211,244
test/two_people.jpg,95,941,244,792
test/obama.jpg,136,624,394,366
test/obama-480p.jpg,65,507,189,383
test/obama-720p.jpg,94,751,273,572
test/obama-1080p.jpg,136,1140,394,882
test/biden.jpg,233,749,542,439
```

12) **face_recognition** command test example
a. **face_recognition** command line tool can recognize whose face is in a single image or a folder of images. Use **face_recognition --help** to view help information for the face_recognition command

```
orangepi@orangepi:~$ face_recognition --help
Usage: face_recognition [OPTIONS] KNOWN_PEOPLE_FOLDER
IMAGE_TO_CHECK

Options:
  --cpus INTEGER                  number of CPU cores to use in parallel (can speed
                                  up processing lots of images). -1 means "use all in
                                  system"
  --tolerance FLOAT               Tolerance for face comparisons. Default is 0.6.
                                  Lower this if you get multiple matches for the same
                                  person.
  --show-distance BOOLEAN         Output face distance. Useful for tweaking tolerance
                                  setting.
```

--help	Show this message and exit.

b. First create a new folder **known_people** with a known name, then copy two pictures to **known_people**, and then copy **obama2.jpg** as **unknown.jpg**, which is the image we want to identify

```
orangepi@orangepi:~$ cd face_recognition/examples
orangepi@orangepi:~/face_recognition/examples$ mkdir known_people
orangepi@orangepi:~/face_recognition/examples$ cp biden.jpg obama.jpg \
known_people
orangepi@orangepi:~/face_recognition/examples$ cp obama2.jpg unkown.jpg
```

c. obama Then you can use the following command to identify the name of the character in the **unknown.jpg** picture, you can see that the unknown.jpg picture is recognized as obama

```
orangepi@orangepi:~/face_recognition/examples$ face_recognition known_people \
unkown.jpg
unkown.jpg,obama
```

d. If we identify an irrelevant picture, unknown_person will be displayed

```
root@orangepi:~/face_recognition/examples$ face_recognition known_people \
alex-lacamoire.png
alex-lacamoire.png,unknown_person
```

e. We can also create a new test folder, and then put multiple pictures in it, and then we can use all the CPUs to recognize all the pictures in parallel

```
orangepi@orangepi:~/face_recognition/examples$ mkdir test
orangepi@orangepi:~/face_recognition/examples$ cp *.jpg *.png test
orangepi@orangepi:~/face_recognition/examples$ face_recognition --cpus -1 \
known_people test
test/obama-240p.jpg,obama
test/alex-lacamoire.png,unknown_person
test/obama_small.jpg,obama
test/unkown.jpg,obama
test/obama2.jpg,obama
test/lin-manuel-miranda.png,unknown_person
test/two_people.jpg,biden
test/two_people.jpg,obama
test/obama-720p.jpg,obama
test/obama.jpg,obama
```

```
test/obama-480p.jpg,obama
test/biden.jpg,biden
test/obama-1080p.jpg,obama
```

3.39. Installation method of Tensorflow

Note that before installing Tensorflow, please make sure that the Linux system used is Debian Buster. The installation method of Tensorflow demonstrated in this section cannot be guaranteed to work normally on other versions of Linux systems.

If you are using the desktop version of Debian Buster, you need to close the desktop before installing Tensorflow, which can free up about 200MB of memory. If the desktop is not closed, the installation will fail due to insufficient memory. For the method of closing the desktop, please refer to the instructions in the section How to disable the desktop in the Linux desktop system. After installation, open the desktop.

If the following error is printed during the installation process, you can ignore it, as long as the installation is not interrupted (this error is caused by insufficient memory):

Write-error on swap-device (253:1:304568)

3.39.1. The method of using script to automatically install Tensorflow

1) First, you need to expand the size of the **/tmp** space. After setting, you need to **restart the linux system** of the development board. The command is as follows:

```
orangepi@orangepi:~$ sudo sed -i 's/nosuid/&,size=2G/' /etc/fstab
orangepi@orangepi:~$ sudo reboot
```

2) After restarting, you can see that the size of the **/tmp** space has changed to 2G

```
orangepi@orangepi:~$ df -h | grep "/tmp"
tmpfs                   2.0G     12K  2.0G     1% /tmp
```

3) Then use the following command to download the tensorflow installation script provided by Orange Pi

```
orangepi@orangepi:~$ wget  \
https://gitee.com/leeboby/tensorflow/raw/master/install_tensorflow.sh
```

4) Then run the **install_tensorflow.sh** script to start installing tensorflow

```
orangepi@orangepi:~$ sudo bash install_tensorflow.sh
```

5) After the tensorflow installation is completed, the version number of tensorflow will be automatically tested and printed. If you can see the following output at the end, it means that the tensorflow installation is successful

```
##### Start Test Tensorflow #####
Tensorflow version is : 2.4.0
###### End Test Tensorflow ######
```

3.39.2. Steps to manually install Tensorflow

1) First, you need to expand the size of the **/tmp** space. After setting, you need to **restart the linux system** of the development board. The command is as follows：

```
orangepi@orangepi:~$ sudo sed -i 's/nosuid/&,size=2G/' /etc/fstab
orangepi@orangepi:~$ sudo reboot
```

2) After restarting, you can see that the size of the **/tmp** space has changed to 2G

```
orangepi@orangepi:~$ df -h | grep "/tmp"
tmpfs                2.0G     12K  2.0G     1% /tmp
```

3) Then use the following command to set the source of pip to Tsinghua source to speed up the download speed of the Python film

```
orangepi@orangepi:~$ mkdir -p ~/.pip
orangepi@orangepi:~$ cat <<EOF > ~/.pip/pip.conf
[global]
timeout = 6000
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
trusted-host = pypi.tuna.tsinghua.edu.cn
EOF
```

4) Then install the dependency film

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install -y python3-pip gfortran    \
libopenblas-dev liblapack-dev libatlas-base-dev libblas-dev    \
libhdf5-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpeg62-turbo-dev    \
```

```
python3-dev pkg-config python3-setuptools python3-wheel
```

5) Then download the **whl** film related to tensorflow

```
orangepi@orangepi:~$ git clone --depth=1 https://gitee.com/leeboby/tensorflow.git
```

6) Then enter the **tensorflow** directory to install the whl package that tensorflow depends on

```
orangepi@orangepi:~$ cd tensorflow
orangepi@orangepi:~/tensorflow$ pip3 install  \
tensorflow/grpcio-1.32.0-cp37-cp37m-linux_aarch64.whl
orangepi@orangepi:~/tensorflow$ pip3 install  \
tensorflow/numpy-1.19.5-cp37-cp37m-linux_aarch64.whl
orangepi@orangepi:~/tensorflow$ pip3 install  \
tensorflow/h5py-2.10.0-cp37-cp37m-linux_aarch64.whl
```

7) Then you can use the following command to install tensorflow

```
orangepi@orangepi:~/tensorflow$ pip3 install  \
tensorflow-2.4.0-cp37-none-linux_aarch64.whl
```

8) After installing tensorflow, you can use the following command to print the version number of tensorflow. If the version number 2.4.0 of tensorflow can be printed out normally, it means that the installation of tensorflow is successful

```
orangepi@orangepi:~/tensorflow$ python3 -c  \
"import tensorflow; print(tensorflow.__version__)"
2.4.0
```

9) References

```
https://github.com/lhelontra/tensorflow-on-arm
https://tf.kmtea.eu/whl/stable.html
https://www.tensorflow.org
https://repo.rock-chips.com/pypi/simple
```

3.40. ROS installation method

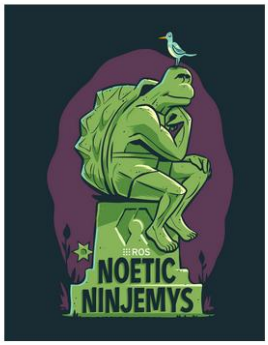
3.40.1. The way to install ROS 1 Noetic

1) The current active version of ROS 1 is as follows, the recommended version is

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 link to the official installation documentation for ROS 1 **Noetic Ninjemys** is as follows:

a. Ubuntu

http://wiki.ros.org/noetic/Installation/Ubuntu

b. Debian

http://wiki.ros.org/noetic/Installation/Debian

3) Ubuntu 20.04 is recommended for Ubuntu Linux in the official installation documentation of ROS **Noetic Ninjemys**, so please make sure that the system used by the development board is **Ubuntu 20.04**. **If you want to use the Debian system, please test**

it yourself, it will not be demonstrated here

http://wiki.ros.org/noetic/Installation

4) First add the software source of ros

```
orangepi@orangepi:~$ sudo sh -c 'echo    \
"deb http://mirrors.ustc.edu.cn/ros/ubuntu $(lsb_release -sc) main"    \
> /etc/apt/sources.list.d/ros-latest.list'
```

5) Then set Keys

```
orangepi@orangepi:~$ sudo apt-key adv    \
--keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key \
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654

Executing: /tmp/apt-key-gpghome.0zbZ9ucMdb/gpg.1.sh --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
gpg: key F42ED6FBAB17C654: public key "Open Robotics <info@osrfoundation.org>"
imported
gpg: Total number processed: 1
gpg:                    imported: 1
```

6) Then update the apt repository cache

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

7) Then install ros

```
orangepi@orangepi:~$ sudo apt install -y ros-noetic-desktop-full
```

8) Before running the commands in ros, you need to set the environment variables first

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
```

9) If you do not want to manually set the environment variables before using the commands in ros, you can add the commands for setting environment variables to **~/.bashrc**, so that the environment variables of ros will be automatically set every time you open a new terminal

```
orangepi@orangepi:~$ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
orangepi@orangepi:~$ source ~/.bashrc
```

10) Now everything needed to run the core ros package has been installed. If you want to create and manage your own ros workspace, you also need to install some other tools. For example, rosinstall is a common command-line tool that lets you download the source tree of some ros packages. Run the following command to install this tool and other dependencies needed to build the ros package

```
orangepi@orangepi:~$ sudo apt install -y python3-rosdep python3-rosinstall   \
python3-rosinstall-generator python3-wstool build-essential
```

11) Before using the ROS tool, you first need to initialize rosdep, and then you can quickly install some system dependencies and some core components in ROS when compiling the source code

Note that running the following command needs to ensure that the development board can access github normally, otherwise an error will be reported due to network problems.

```
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
```

12) The method to verify if ROS is installed correctly

a. First run **roscore**

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ roscore
... logging to
/home/orangepi/.ros/log/132132c4-c873-11ec-9b13-5099013e1a05/roslaunch-orangepi-1
8381.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://orangepi:44425/
ros_comm version 1.15.14
```

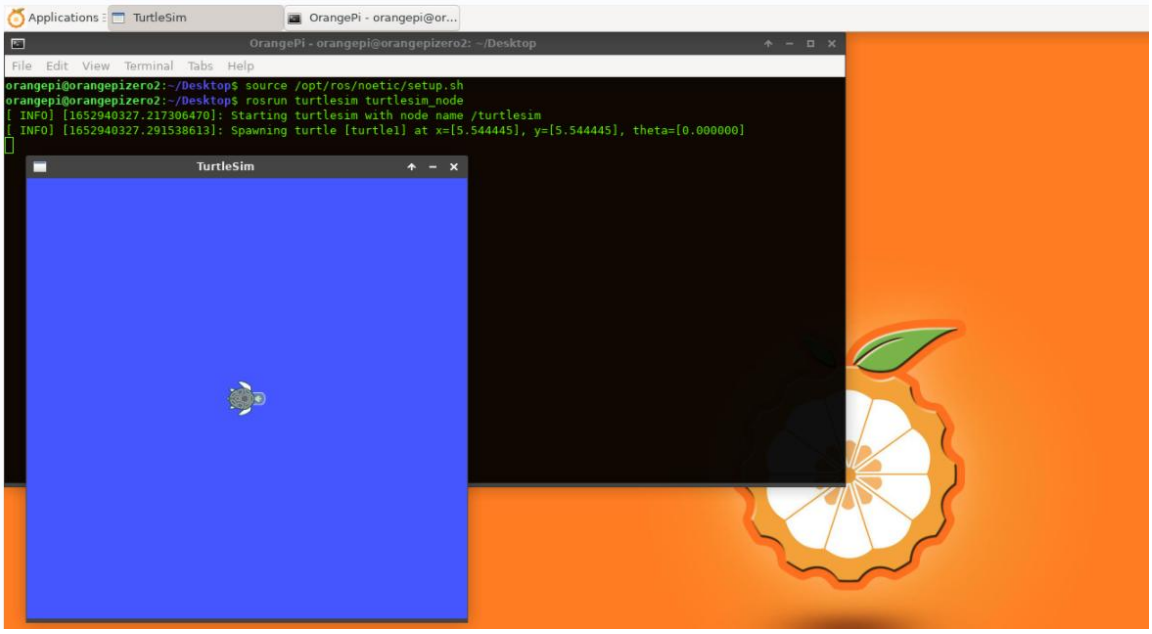
```
SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.15.14

NODES

auto-starting new master
process[master]: started with pid [18389]
ROS_MASTER_URI=http://orangepi:11311/

setting /run_id to 132132c4-c873-11ec-9b13-5099013e1a05
process[rosout-1]: started with pid [18399]
started core service [/rosout]
```

b. Then start a small turtle routine to test whether ROS can be used normally

a) First open a command line terminal window in the desktop

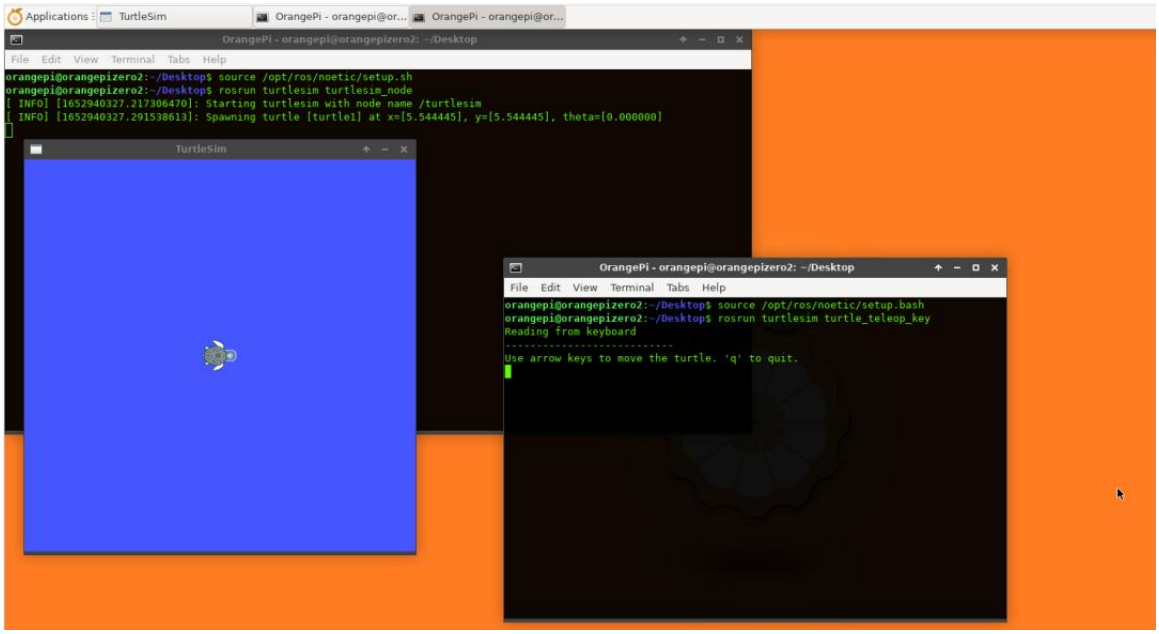
b) Then enter the following command, a small turtle as shown in the figure below will pop up

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ rosrun turtlesim turtlesim_node
```

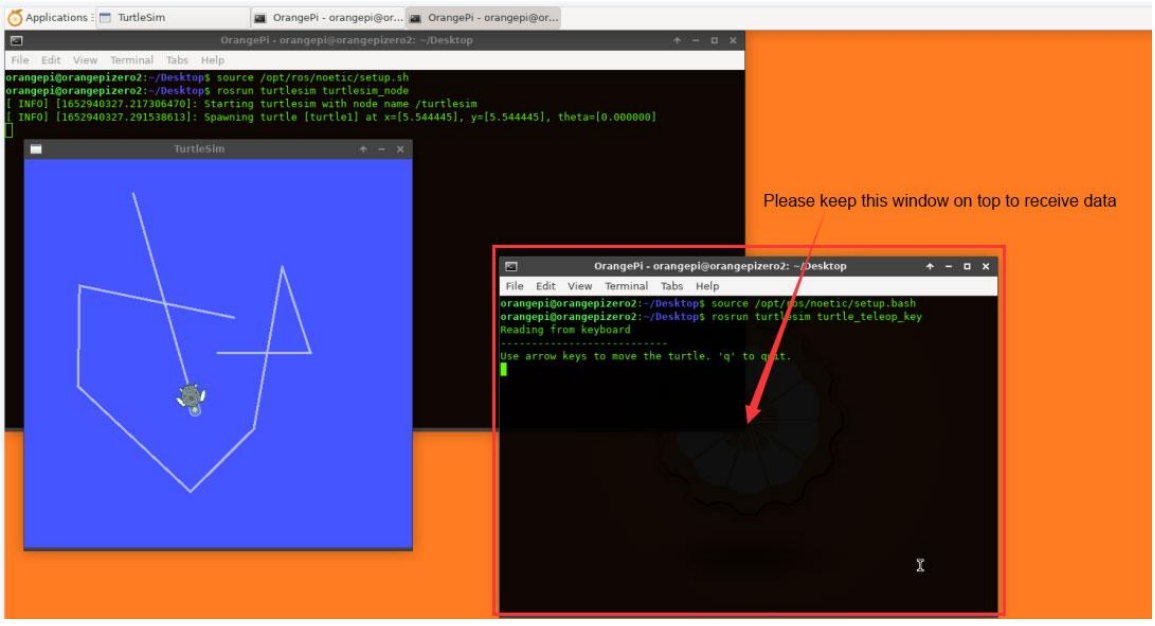


c) Then open a terminal window and enter the following command to run the turtle control program

```
orangepi@orangepi:~$ source /opt/ros/noetic/setup.bash
orangepi@orangepi:~$ rosrun turtlesim turtle_teleop_key
```



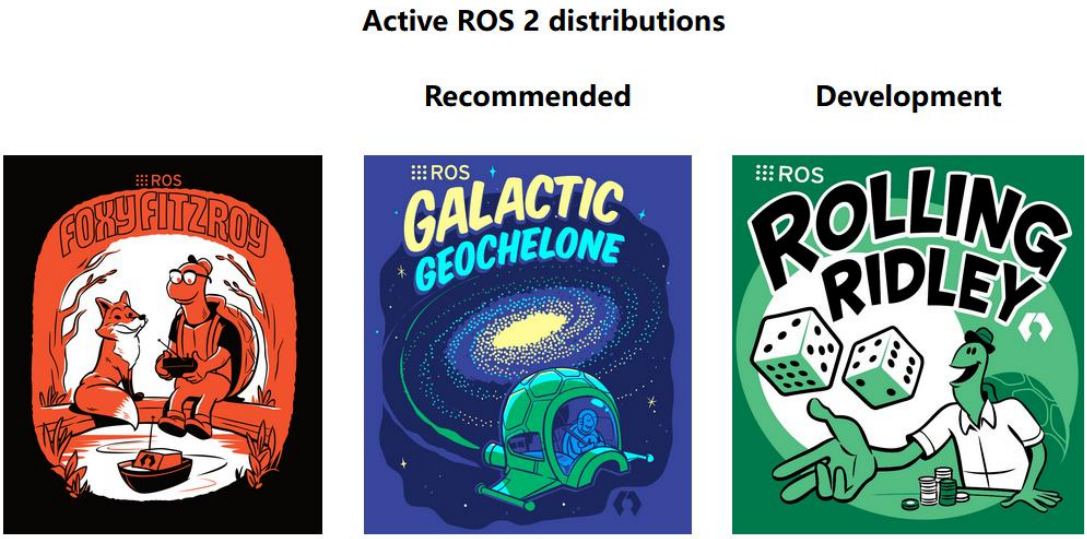
d) Then please keep the terminal window of the turtle control program you just opened at the top. At this time, you can control the small turtle to move up, down, left and right by pressing the direction keys on the keyboard.



3.40.2. The Way to install ROS 2 Galactic

1) The current active version of ROS 2 is as follows, the recommended version is

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 link to the official installation documentation for 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) Ubuntu 20.04 is recommended for Ubuntu Linux in the official installation document of ROS 2 **Galactic Geochelone**, so please make sure that the system used by the development board is **Ubuntu 20.04**. There are several ways to install ROS 2. The

following demonstrates how to install ROS 2 **Galactic Geochelone** through **Debian packages**.

4) First add the key

```
orangepi@orangepi:~$ sudo apt-key adv --keyserver    \
'hkp://keyserver.ubuntu.com:80'    \
--recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654

[sudo] password for orangepi:
Executing: /tmp/apt-key-gpghome.gNBSKx6Ums/gpg.1.sh --keyserver
hkp://keyserver.ubuntu.com:80 --recv-key
C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
gpg: key F42ED6FBAB17C654: public key "Open Robotics <info@osrfoundation.org>"
imported
gpg: Total number processed: 1
gpg:                imported: 1
```

5) Then add the repository source of ROS 2 to the Ubuntu system

```
orangepi@orangepi:~$ echo "deb [arch=$(dpkg --print-architecture)]    \
http://mirrors.ustc.edu.cn/ros2/ubuntu $(source /etc/os-release && echo    \
$UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list    \
> /dev/null
```

6) Then update the apt repository cache

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

7) Then you can install ROS 2 related packages. The following commands will install ROS, RViz, demos, tutorials

```
orangepi@orangepi:~$ sudo apt install -y ros-galactic-desktop
```

8) Before running the **ros2** command, you need to set the environment variable first

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
orangepi@orangepi:~$ ros2 -h
usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...
```

```
ros2 is an extensible command-line tool for ROS 2.

optional arguments:
  -h, --help            show this help message and exit

Commands:
  action      Various action related sub-commands
  bag         Various rosbag related sub-commands
  component   Various component related sub-commands
  daemon      Various daemon related sub-commands
  doctor      Check ROS setup and other potential issues
  interface   Show information about ROS interfaces
  launch      Run a launch file
  lifecycle   Various lifecycle related sub-commands
  multicast   Various multicast related sub-commands
  node        Various node related sub-commands
  param       Various param related sub-commands
  pkg         Various package related sub-commands
  run         Run a package specific executable
  security    Various security related sub-commands
  service     Various service related sub-commands
  topic       Various topic related sub-commands
  wtf         Use `wtf` as alias to `doctor`

  Call `ros2 <command> -h` for more detailed usage.
```

9) You can use the following method to test whether ROS 2 is successfully installed

   a. Open a terminal first, then use the following two commands to run a **C++ talker** that keeps sending Hello World

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
orangepi@orangepi:~$ ros2 run demo_nodes_cpp talker
[INFO] [1649599956.390893735] [talker]: Publishing: 'Hello World: 1'
[INFO] [1649599957.390812753] [talker]: Publishing: 'Hello World: 2'
[INFO] [1649599958.390890143] [talker]: Publishing: 'Hello World: 3'
.....
```

   b. Then open a terminal and use the following two commands to run a **Python**

listener. If you can receive the Hello World sent above, it means that both the C++ and Python APIs of RSO 2 can work normally

```
orangepi@orangepi:~$ source /opt/ros/galactic/setup.bash
orangepi@orangepi:~$ ros2 run demo_nodes_py listener
[INFO] [1649600109.504678962] [listener]: I heard: [Hello World: 154]
[INFO] [1649600110.393777793] [listener]: I heard: [Hello World: 155]
[INFO] [1649600111.393769143] [listener]: I heard: [Hello World: 156]
.....
```

10) For the usage of ROS, please refer to the documentation of ROS 2

http://docs.ros.org/en/galactic/Tutorials.html

11) Run the following command to uninstall ROS 2

```
orangepi@orangepi:~$ sudo apt remove ~nros-galactic-* && sudo apt autoremove
orangepi@orangepi:~$ sudo rm /etc/apt/sources.list.d/ros2.list
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt autoremove
orangepi@orangepi:~$ sudo apt upgrade
```

3.41. Installation method of OpenMediaVault

OpenMediaVault is a Debian-based NAS operating system.

It can be known from the following table: .
Debian10 can only install OpenMediaVault 5.x version;
Debian11 can only install OpenMediaVault 6.x version。

Table 1: openmediavault historical releases

Version	Codename	Base Distro	Status	Date Released
0.2	Ix	Debian 6	EOL	Oct 2011
0.3	Omnious	Debian 6	EOL	Jul 2012
0.4	Fedaykin	Debian 6	EOL	Sep 2012
0.5	Sardoukar	Debian 6	EOL	Aug 2013
1.0	Kralizec	Debian 7	EOL	Sept 2014
2.0	Stoneburner	Debian 7	EOL	Jun 2015
3.0	Erasmus	Debian 8	EOL	Jun 2016
4.0	Arrakis	Debian 9	EOL	Apr 2018
5.0	Usul	Debian 10	Stable	Mar 2020
6.0	Shaitan	Debian 11	In Development	est. Q2/2022

So before installation, please select the version of OpenMediaVault you want to install, and then make sure that the Debian system used by the development board is

the matching system.

In addition, OpenMediaVault officially recommends using the server version of the system, so please do not use the desktop version of the system to install OpenMediaVault.

Can I install openmediavault on top a running Debian system? Yes, but it is recommended that the current running OS not to have a desktop environment installed.

3.41.1. Install OpenMediaVault 5.x on Debian 10

Note that **Debian10** can only install OpenMediaVault 5.x.

1) The official OpenMediaVault documentation is as follows：

a. Documentation for version a.5.x **(stable version of OpenMediaVault)**

https://openmediavault.readthedocs.io/en/5.x/

2) The official documentation for installing OpenMediaVault in Debian10 is as follows:

https://openmediavault.readthedocs.io/en/5.x/installation/on_debian.html

3) First install the keyring of OpenMediaVault, note that the following commands are executed under the **root** user

```
root@orangepi:~# apt-get install -y gnupg
root@orangepi:~# wget -O  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"  \
https://packages.openmediavault.org/public/archive.key
root@orangepi:~# apt-key add  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"
```

4) Then add the package repository of OpenMediaVault, pay attention to switch to the **root** user and enter the following command, the black font part is a complete command, please copy it directly

```
root@orangepi:~# cat <<EOF > /etc/apt/sources.list.d/openmediavault.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul main
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul main
## Uncomment the following line to add software from the proposed repository.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul-proposed main
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul-proposed main
## This software is not part of OpenMediaVault, but is offered by third-party
```

```
## developers as a service to OpenMediaVault users.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public usul partner
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages usul partner
EOF
```

The Tsinghua source is used above. For related instructions, please refer to the following link

https://mirrors.tuna.tsinghua.edu.cn/help/openmediavault/

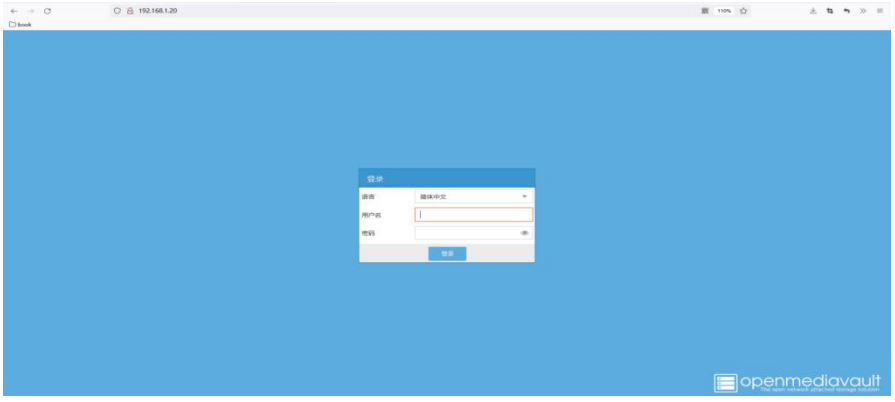
5) Then use the following command to install OpenMediaVault

```
root@orangepi:~# export LANG=C.UTF-8
root@orangepi:~# export DEBIAN_FRONTEND=noninteractive
root@orangepi:~# export APT_LISTCHANGES_FRONTEND=none
root@orangepi:~# apt-get update
root@orangepi:~# apt-get --yes --auto-remove --show-upgraded \
--allow-downgrades --allow-change-held-packages \
--no-install-recommends \
--option DPkg::Options::="--force-confdef" \
--option DPkg::Options::="--force-confold" \
install openmediavault-keyring openmediavault
```

6) Then run the following command. After the operation is completed, enter the IP address of the development board in the browser to open the login page of OpenMediaVault

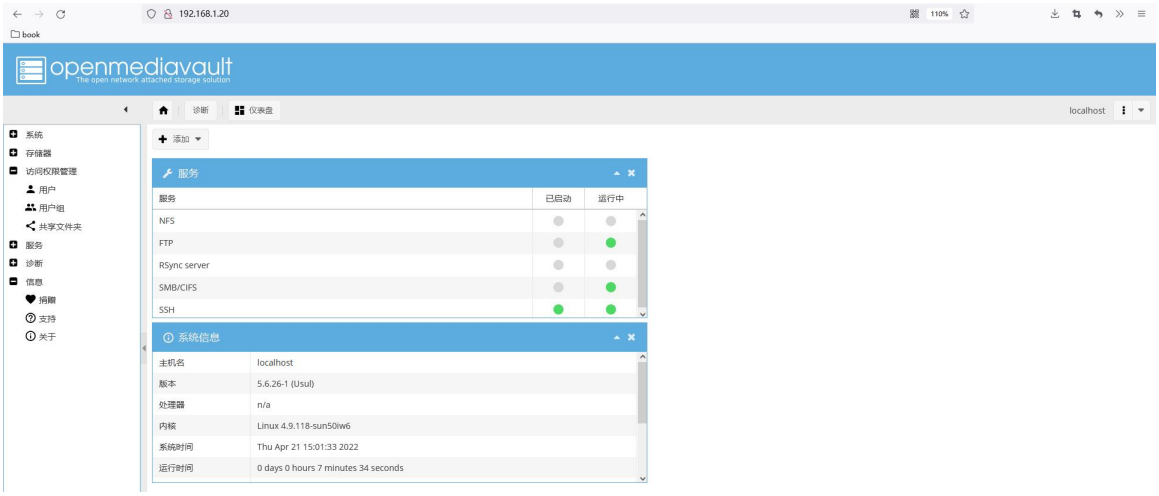
```
root@orangepi:~# omv-confdbadm populate
```

7) The login interface of OpenMediaVault is as follows



8) Then enter the default **username admin** and **password openmediavault**

9) The main interface displayed by OpenMediaVault login is as follows



3.41.2. Debian11 install OpenMediaVault 6.x

Note that **Debian11** can only install OpenMediaVault 6.x.

1) The official documentation of OpenMediaVault is as follows:

https://openmediavault.readthedocs.io/en/latest/

2) The official documentation for installing OpenMediaVault in Debian looks like this:

https://openmediavault.readthedocs.io/en/latest/installation/on_debian.html

3) First install the keyring of OpenMediaVault, note that the following commands are executed under the **root** user

```
root@orangepi:~# apt-get install -y gnupg
root@orangepi:~# wget -O  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"  \
https://packages.openmediavault.org/public/archive.key
root@orangepi:~# apt-key add  \
"/etc/apt/trusted.gpg.d/openmediavault-archive-keyring.asc"
```

4) Then add the package repository of OpenMediaVault, pay attention to switch to the **root** user and enter the following command, the black font part is a complete command, please copy it directly

```
root@orangepi:~# cat <<EOF >> /etc/apt/sources.list.d/openmediavault.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan main
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan main
## Uncomment the following line to add software from the proposed repository.
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan-proposed main
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan-proposed main
## This software is not part of OpenMediaVault, but is offered by third-party
## developers as a service to OpenMediaVault users.
# https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/public shaitan partner
# deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/packages shaitan partner
EOF
```

The Tsinghua source is used above. For related instructions, please refer to the following link

```
https://mirrors.tuna.tsinghua.edu.cn/help/openmediavault/
```

5) Then use the following command to install OpenMediaVault

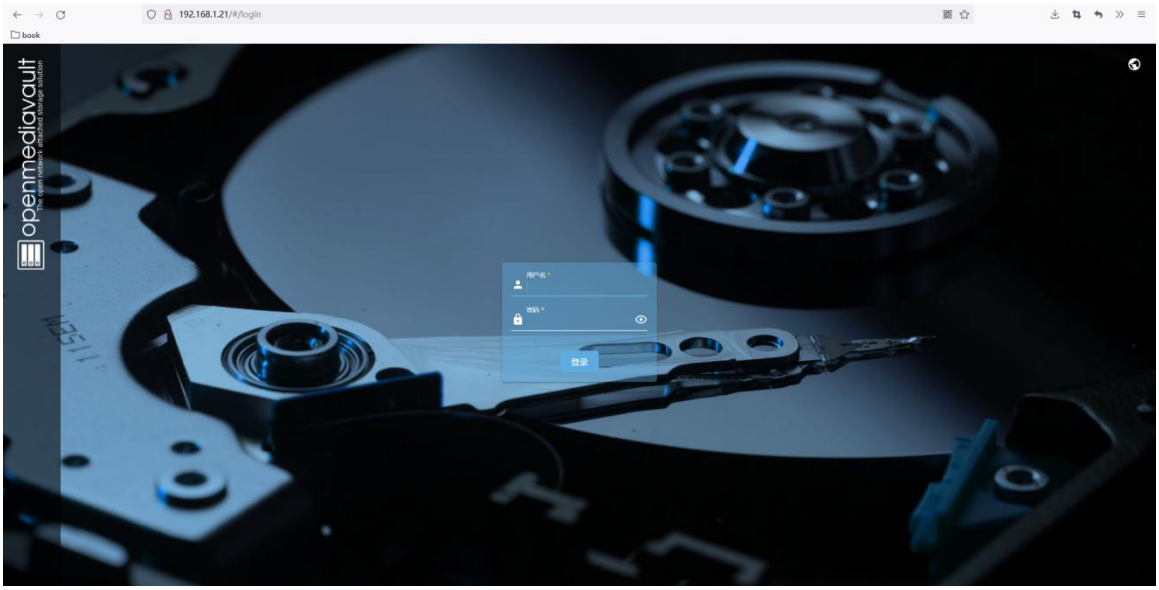
```
root@orangepi:~# export LANG=C.UTF-8
root@orangepi:~# export DEBIAN_FRONTEND=noninteractive
root@orangepi:~# export APT_LISTCHANGES_FRONTEND=none
root@orangepi:~# apt-get update
root@orangepi:~# apt-get --yes --auto-remove --show-upgraded \
--allow-downgrades --allow-change-held-packages \
```

```
--no-install-recommends \
--option DPkg::Options::="--force-confdef" \
--option DPkg::Options::="--force-confold" \
install openmediavault-keyring openmediavault
```

6) Then run the following command. After the operation is complete, enter the IP address of the development board in the browser to open the login page of OpenMediaVault

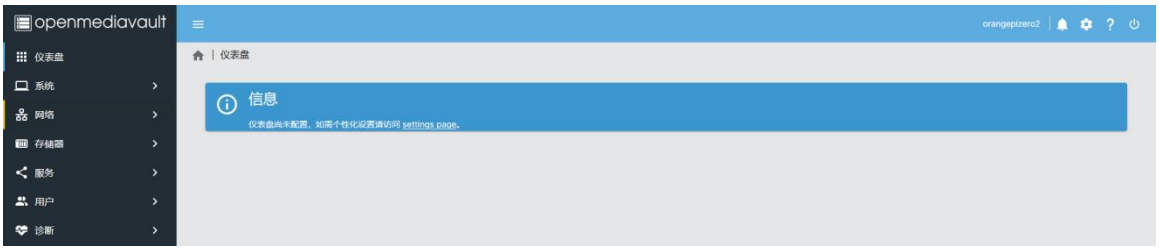
```
root@orangepi:~# omv-confdbadm populate
```

7) The login interface of OpenMediaVault is as follows

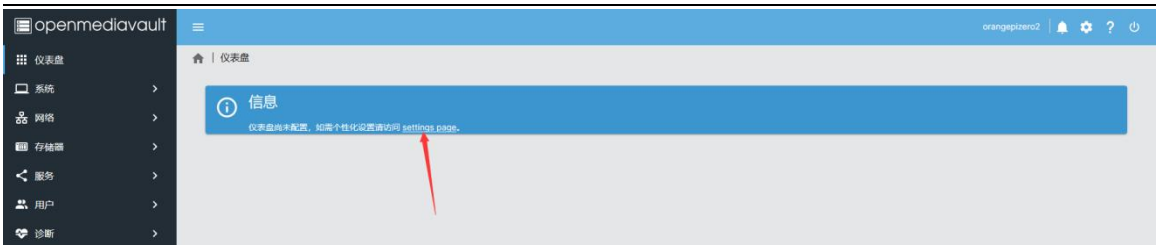


8) Then enter the default username **admin** and password **openmediavault**

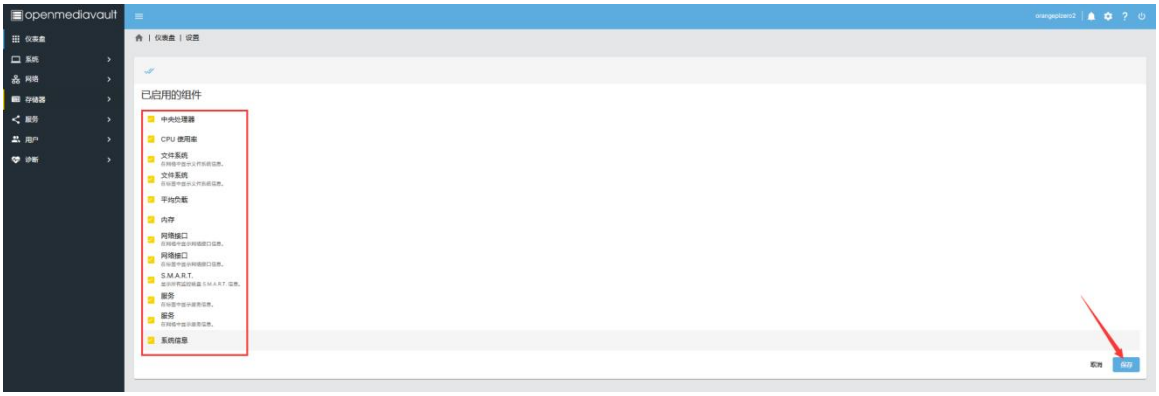
9) The main interface displayed by OpenMediaVault login is as follows



10) Then click on the **setting page**

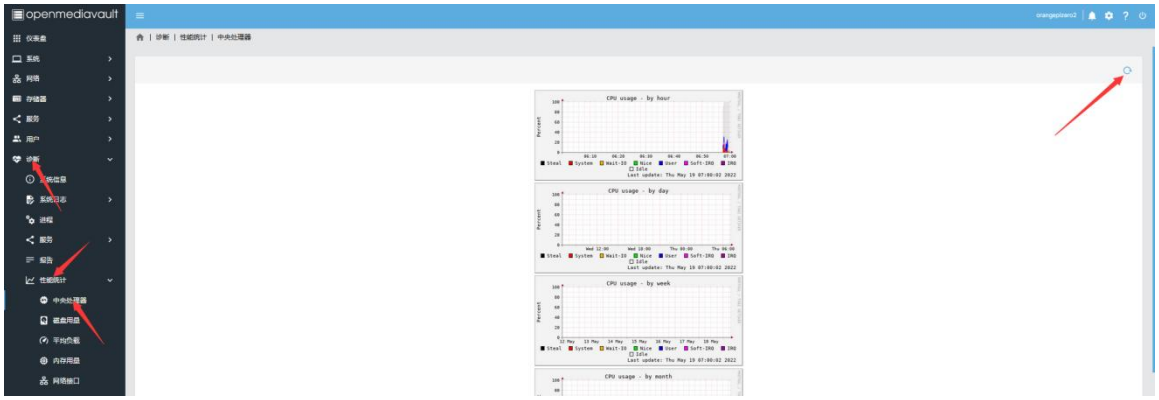


11) Then select all these components, and then click the **save** button to save



12) Then you can see the system information displayed on the dashboard

13) If the interface displayed by the CPU is not normal, you can open the **diagnosis -> performance statistics -> CPU**, and then click the refresh button in the upper right corner to refresh the

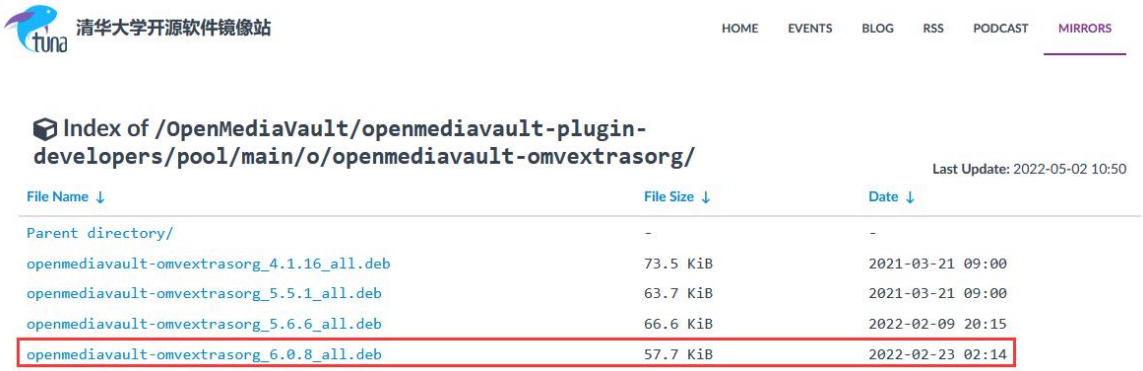


14) The method to install OMV plugin

a. First open the following URL

https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers/pool/main/o/openmediavault-omvextrasorg/

b. Then record the file name of the latest plugin package



c. Then use the following command in the Linux system of the development board to download the plug-in package shown above (**if the following command cannot be downloaded, the name of the plug-in package may have changed, please replace it with the latest name**)

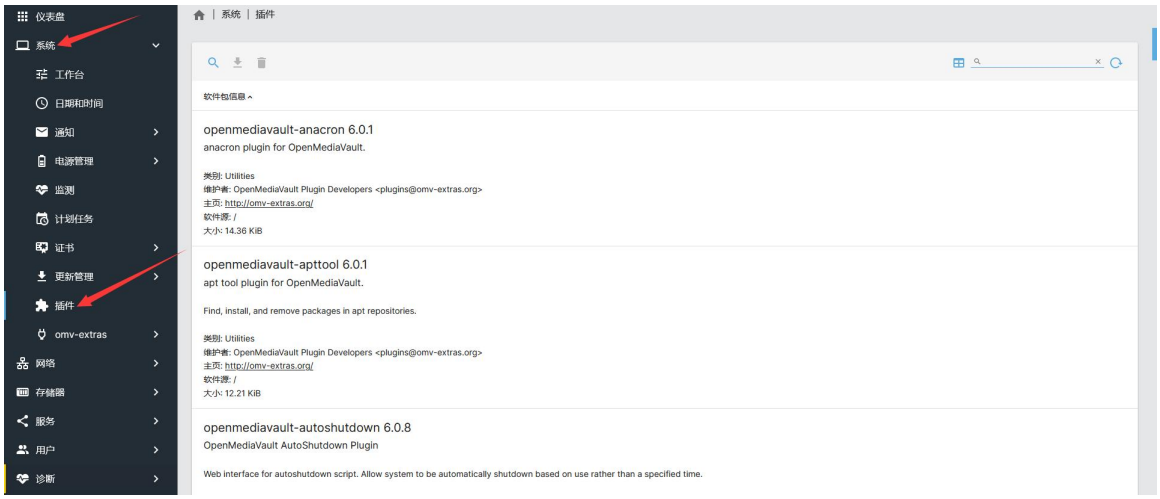
orangepi@orangepi:~$ **wget **
https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers/pool/main/o/openmediavault-omvextrasorg/openmediavault-omvextrasorg_6.0.8_all.deb

d. Then install the deb film just downloaded in the Linux system of the development board

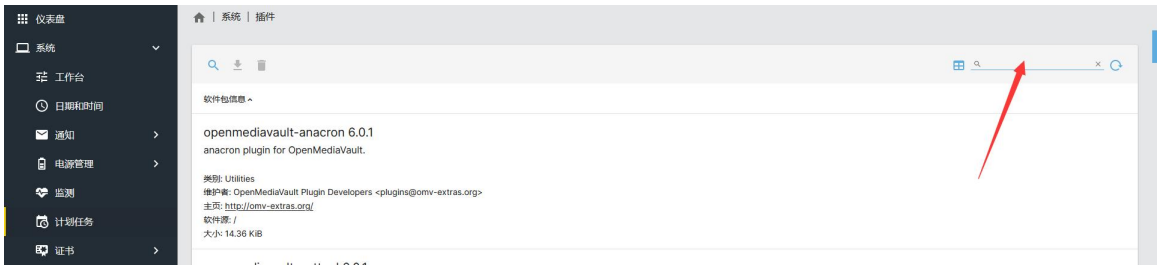
orangepi@orangepi:~$ sudo dpkg -i openmediavault-omvextrasorg_6.0.8_all.deb

```
.....
Summary for orangepizero2
------------
Succeeded: 6 (changed=4)
Failed:    0
------------
Total states run:     6
Total run time: 110.994 s
Processing triggers for openmediavault (6.0.27-1) ...
Updating workbench configuration files ...
Restarting engine daemon ...
```

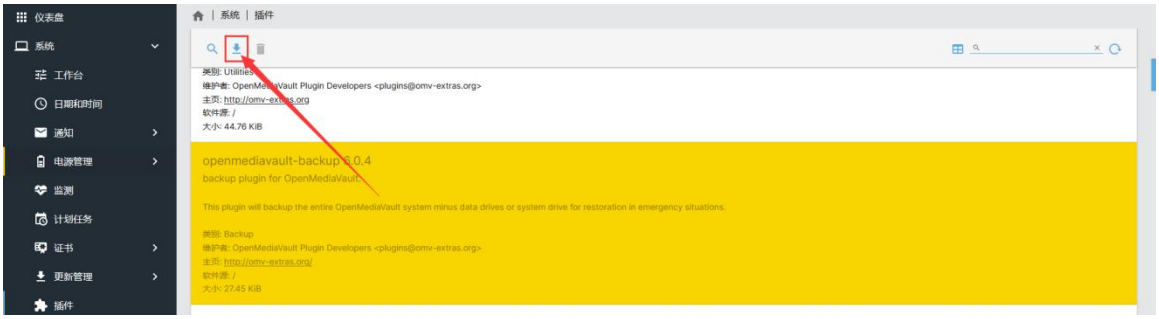
e. Then click **System -> Plugins** to see the following interface



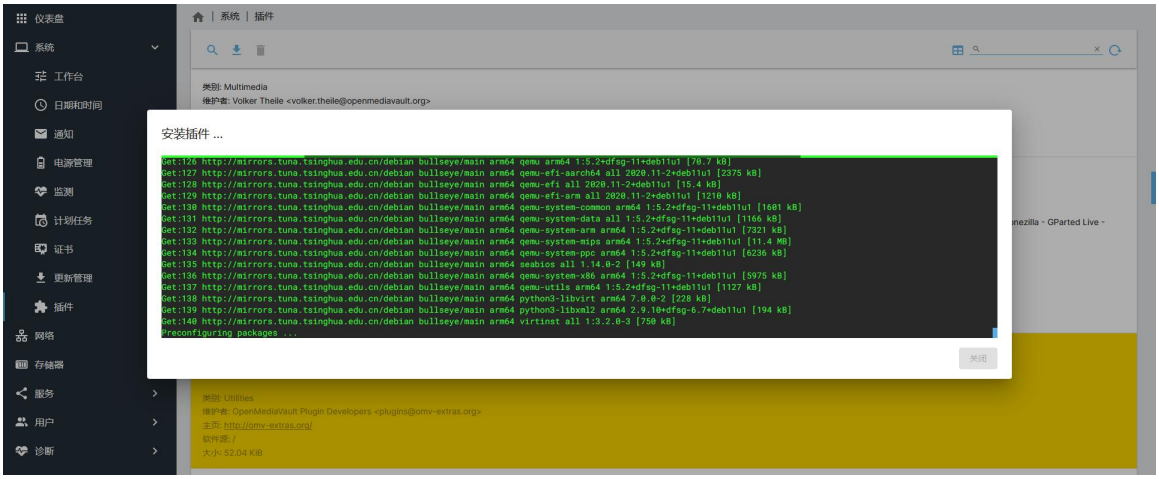
f. Then you can select or search for the plugin you want to install



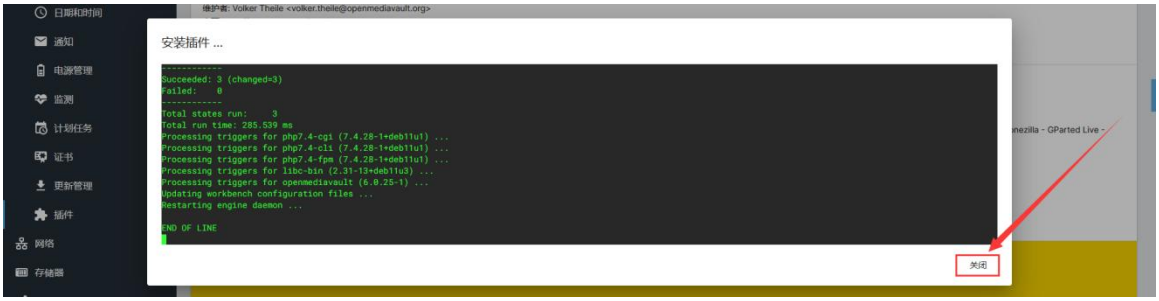
g. Then click the button shown in the figure below to start installing the selected plug-in



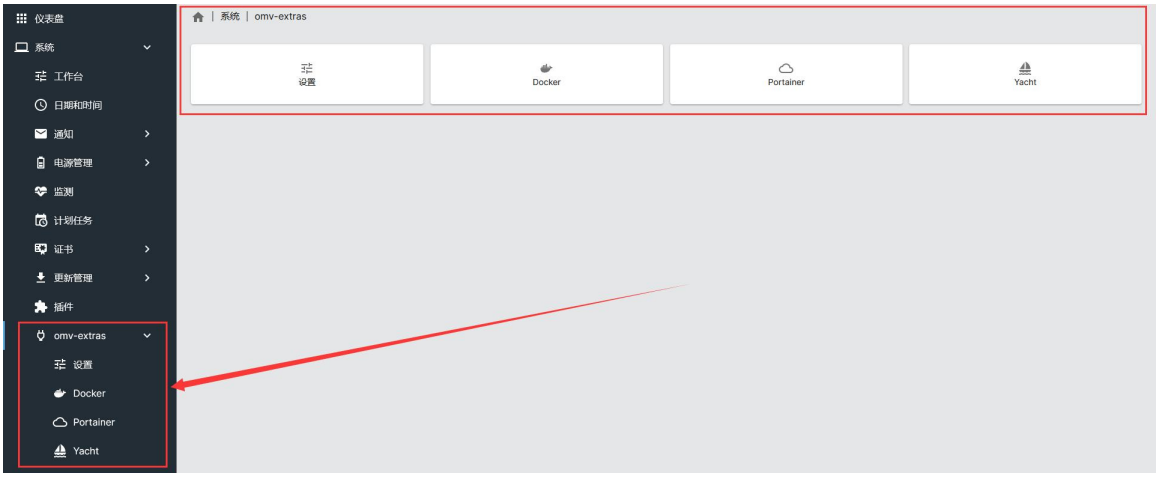
h. The plugin installation process is as follows



i. After the installation is complete, click Close



15) After the plugin package is installed, there will be an **omv-extras** option on the right side of the web interface. **Docker**, **Portainer** and **Yacht** can be installed in omv-extras



16) Before installing docker, please replace the software source of docker. First, use the following command to open **omvextras.list**, and then replace the content in the blue font part. Finally, please remember to use **sudo apt-get update** to update the package index cache of the Linux system

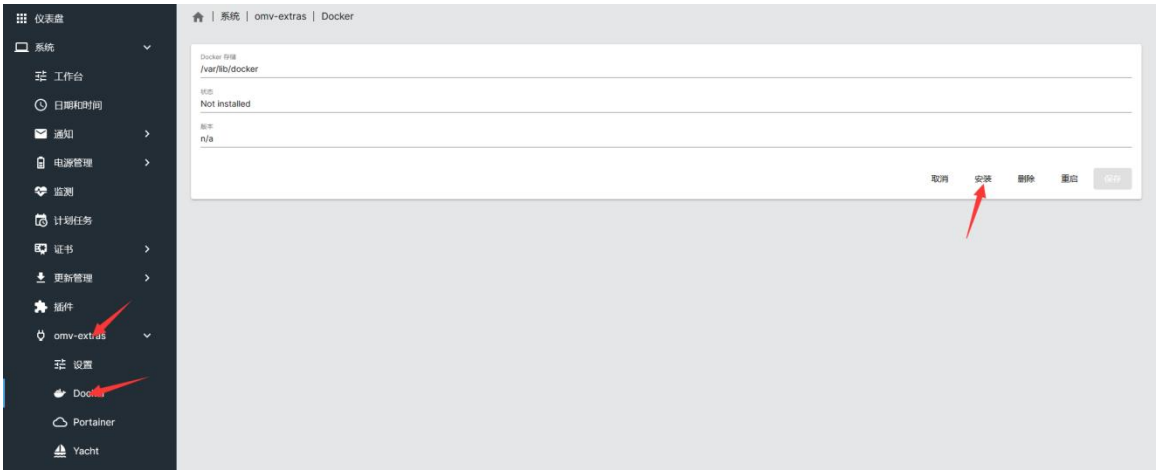
```
orangepi@orangepi:~$ sudo vim /etc/apt/sources.list.d/omvextras.list
deb https://mirrors.tuna.tsinghua.edu.cn/OpenMediaVault/openmediavault-plugin-developers shaitan main
deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/docker-ce/linux/debian bullseye stable
orangepi@orangepi:~$ sudo apt-get update
```

17) The way to install Docker in OMV is as follows

a. First install **apparmor**

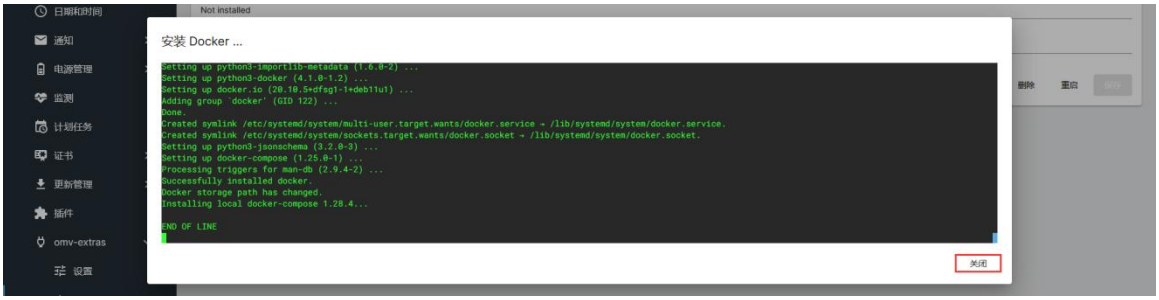
```
orangepi@orangepi:~$ sudo apt install -y apparmor
```

b. Then open the control interface of **Docker**, and then click the **install** button to start installing Docker



c. The display output of the Docker installation process is shown below

d. The display after the Docker installation is completed is as follows, and then click **Close**.

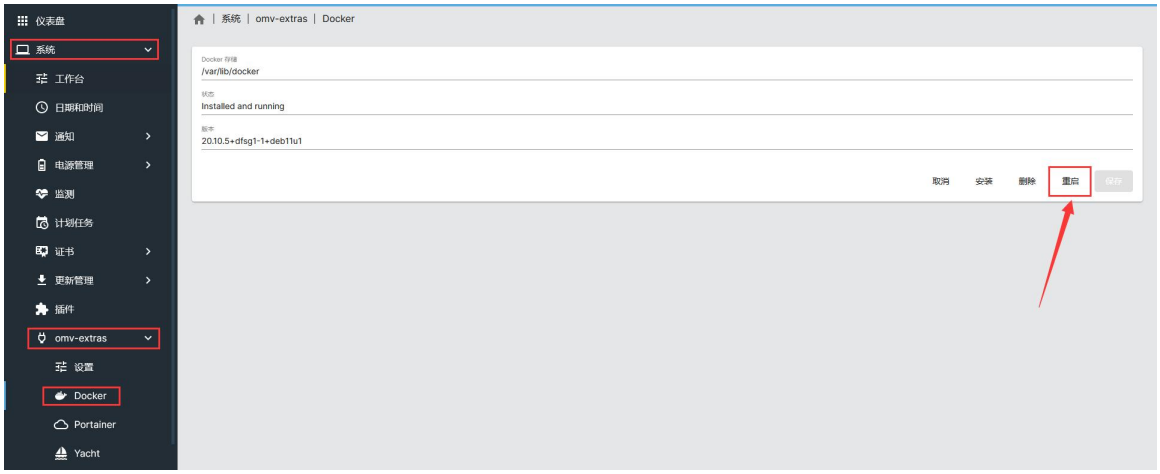


e. Finally, you need to set the address of the Docker warehouse to the domestic address to speed up the download speed of the Docker container. The method looks like this:

a) First open the **/etc/docker/daemon.json** file, and then add the configuration in the red font part below (note that "**data-root": "/var/lib/docker**" should be followed by a ,)

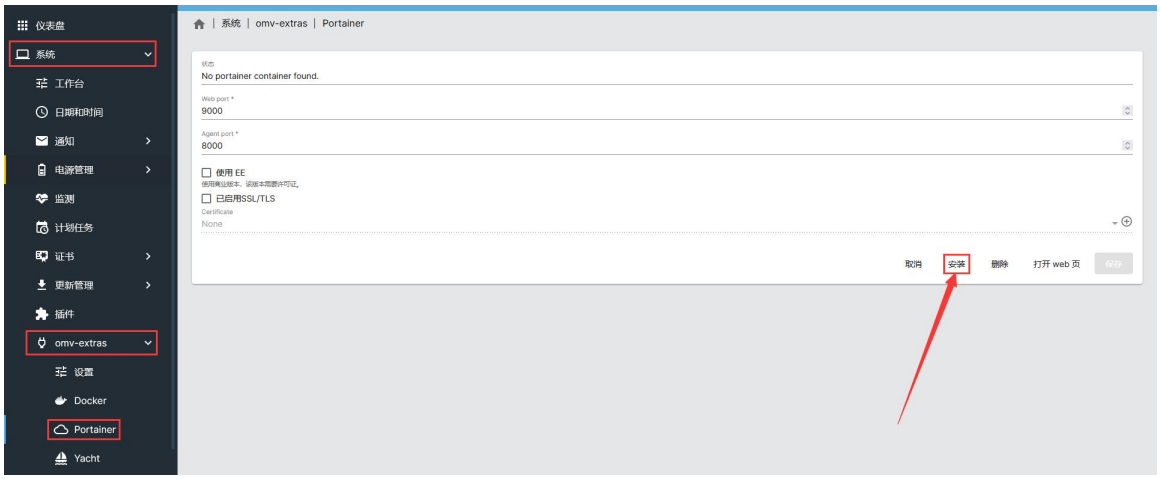
```
orangepi@orangepi:~$ sudo vim /etc/docker/daemon.json
{
  "data-root": "/var/lib/docker",
  "registry-mirrors": ["https://docker.mirrors.ustc.edu.cn"]
}
```

b) Then click the **restart** button on the Docker control interface to restart the Docker service for the configuration to take effect (**if an error is reported, first check whether the above configuration is correct, and then try a few more times, or restart the Debian11 system**)



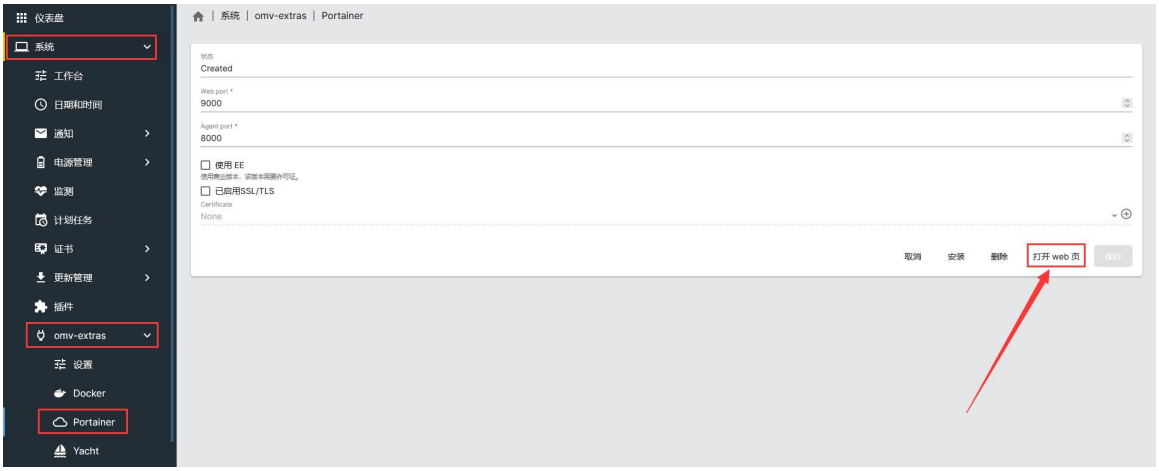
18) **Portainer** is a docker visual management tool. The steps to install Portainer are:：

a. First open Portainer's control interface, and then click the **install** button to start installing Portainer

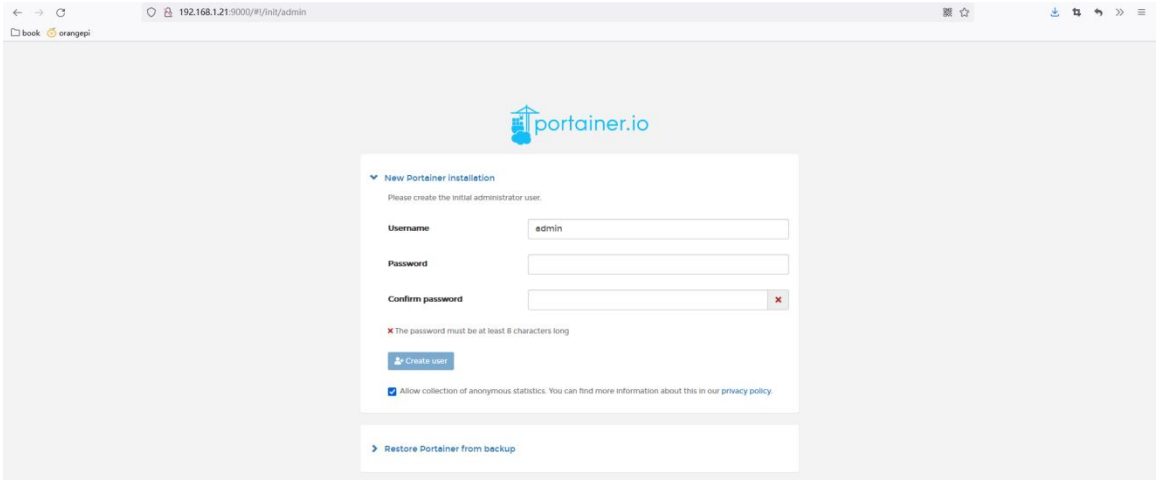


b. After the installation of Portainer is completed, the display is as shown in the figure below, and then click the **close** button to close it.

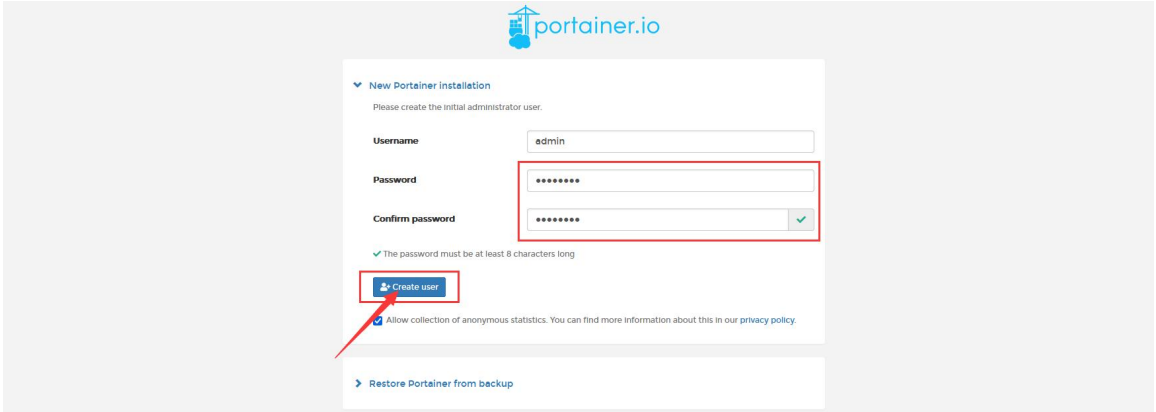
c. Then open Portainer's control interface, and then click to **open the web page** to open Portainer's web control interface



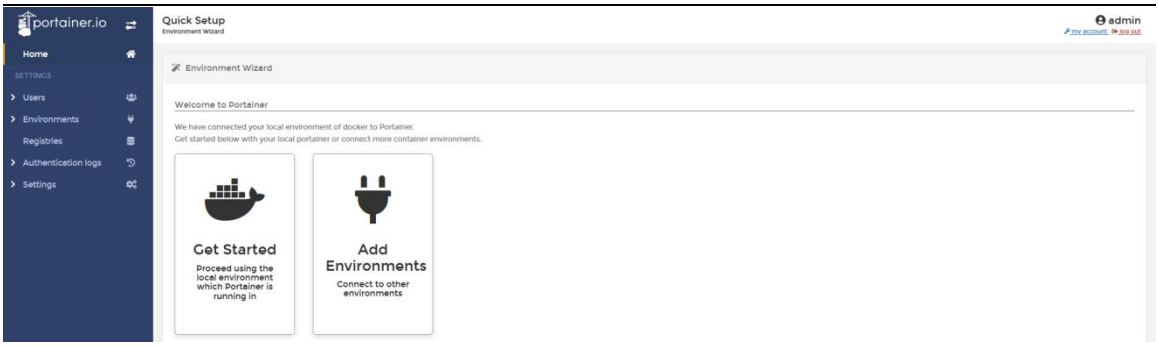
d. The display of d.Portainer's web control interface after opening is as follows



e. Then set the password of Portainer, and then click **Create user** to enter the web control interface of Portainer



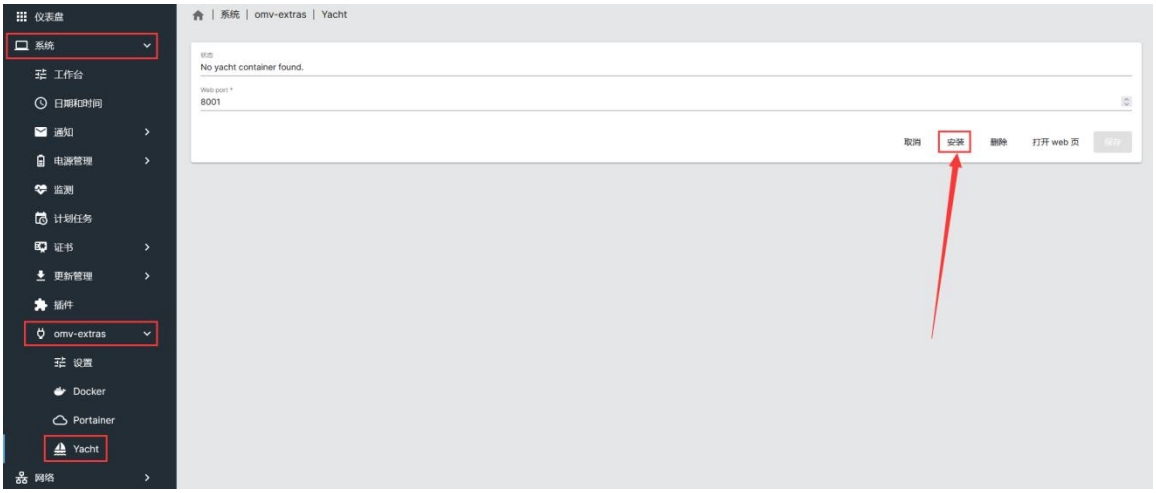
f. The main interface of Portainer after login is shown as below



g. Please study the usage of g.Portainer by yourself

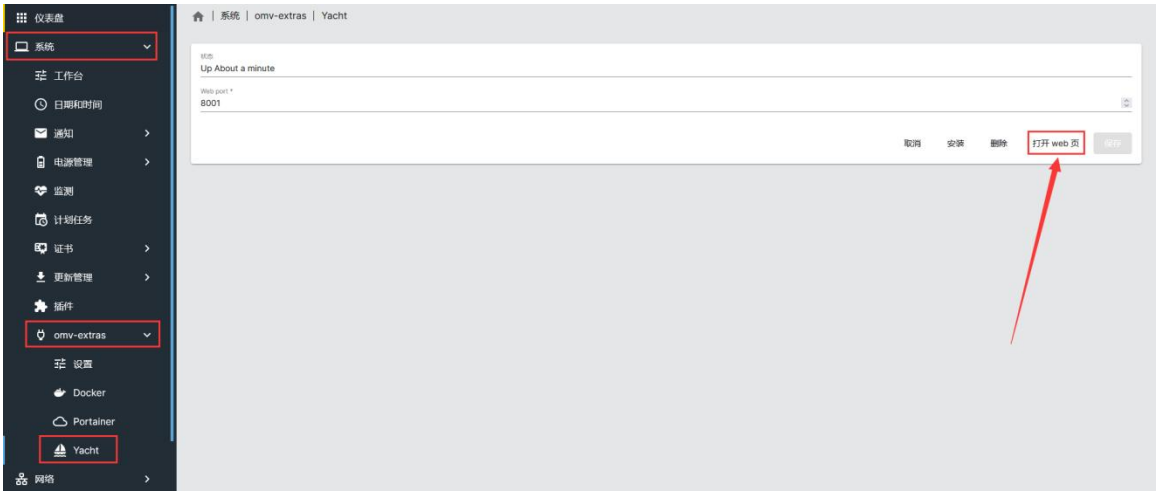
19) **Yacht** and Portainer have similar functions. They are both docker visual management tools. The installation steps are as follows:

a. First open the Yacht control interface in OMV, and then click the **install** button to start installing Yacht

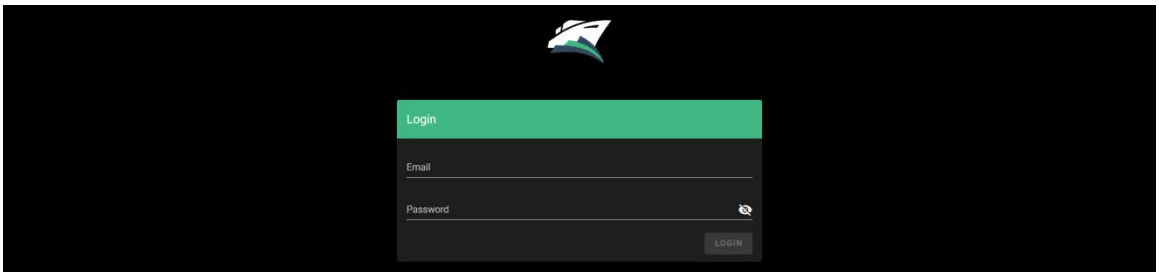


b. The display after Yacht is installed is shown in the figure below, and then click the **close** button to close it

c. Then open Yacht's control interface in OMV, and then click to **open the web page** to open Yacht's web control interface

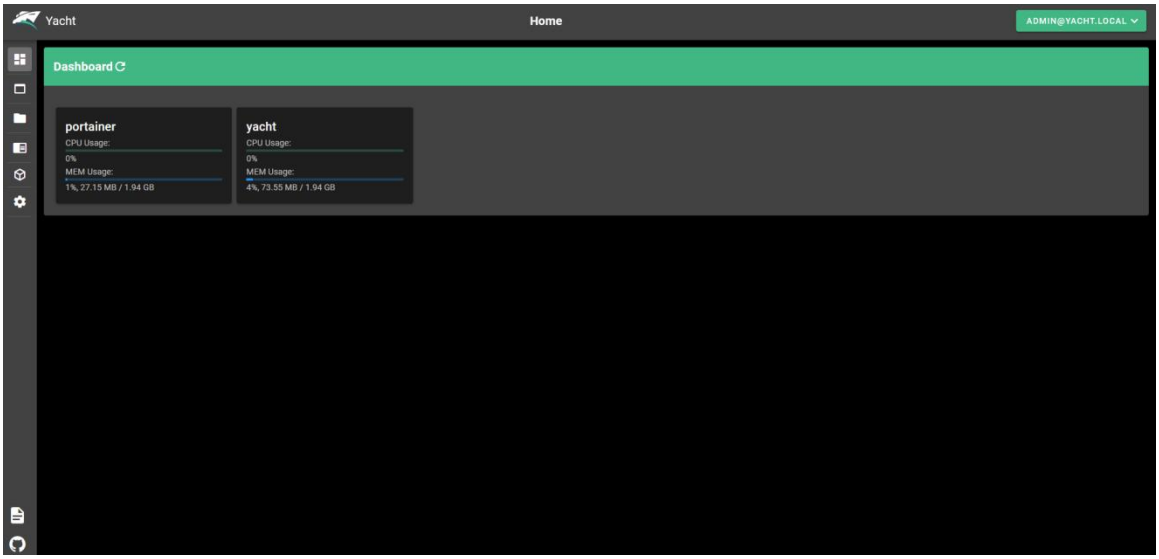


d. The display of Yacht's web control interface after opening is as follows



e. Then **enter** Yacht's default account admin@yacht.local in the Email column, and enter the default password **pass** in the **Password** column, and then click **LOGIN** to enter the Yacht web control interface

f. The main interface after Yacht login is shown as below



g. Please study the usage of g. Yacht by yourself

3.42. Installation method of Pi-hole

Note that this section only provides the installation method of Pi-hole. Please refer to the official documentation of Pi-hole for usage.:

https://pi-hole.net
https://docs.pi-hole.net

Note that Pi-hole does not support Ubuntu22.04 yet, so please do not use this system to install Pi-hole。

1) First download Pi-hole's installation script, then run
   a. Use Orange Pi to provide pi-hole script installation **(recommended)**

Note that the pi-hole installation script provided by Orange Pi does not modify any pi-hole functions, but only solves the problem of download and installation failure caused by inability to access Github normally.

   a) The first method: first download the repository of pi-hole, and then run the installation script

```
orangepi@orangepi:~$ git clone --depth 1 https://gitee.com/leeboby/pi-hole.git   \
Pi-hole
orangepi@orangepi:~$ cd Pi-hole/automated\ install/
orangepi@orangepi:~$ sudo bash basic-install.sh
```

   b) The second method: first download the installation script, then run

```
orangepi@orangepi:~$ wget -O basic-install.sh   \
https://gitee.com/leeboby/pi-hole/raw/master/automated%20install/basic-install.sh
orangepi@orangepi:~$ sudo bash basic-install.sh
```

   b. Install using pi-hole official script (**not recommended if it will not solve network problems**)
      a) The first method: use the following command to download the installation script of pi-hole and run it directly

```
orangepi@orangepi:~$ curl -sSL https://install.pi-hole.net | bash
```

      b) The second method: first download the repository of pi-hole, and then run the installation script

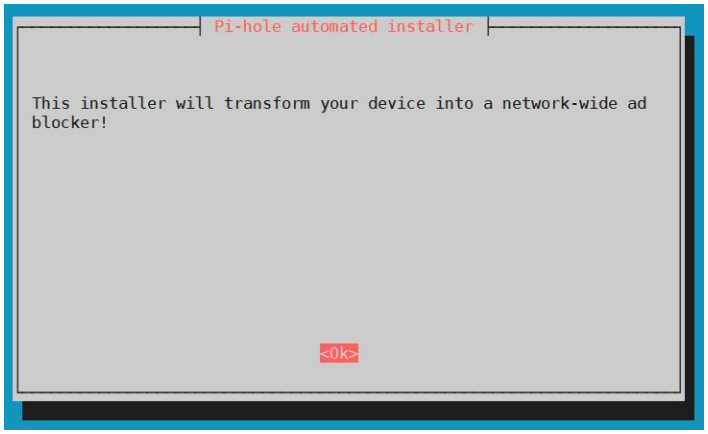
```
orangepi@orangepi:~$ git clone --depth 1 https://github.com/pi-hole/pi-hole.git   \
Pi-hole
```

```
orangepi@orangepi:~$ cd Pi-hole/automated\ install/
orangepi@orangepi:~$ sudo bash basic-install.sh
```

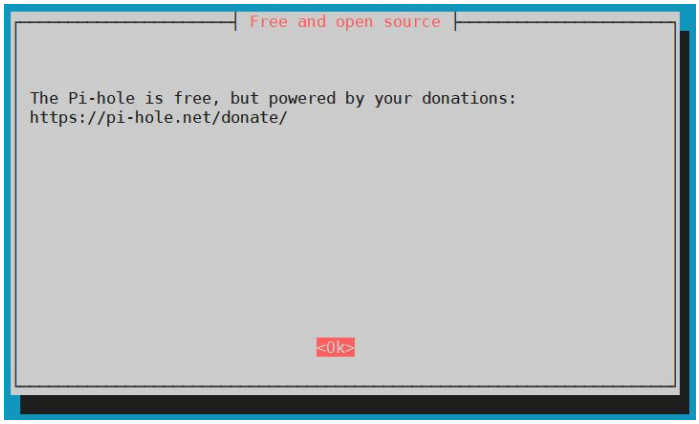
c) The third method: first download the installation script, then run

```
orangepi@orangepi:~$ wget -O basic-install.sh https://install.pi-hole.net
orangepi@orangepi:~$ sudo bash basic-install.sh
```

2) Then press enter

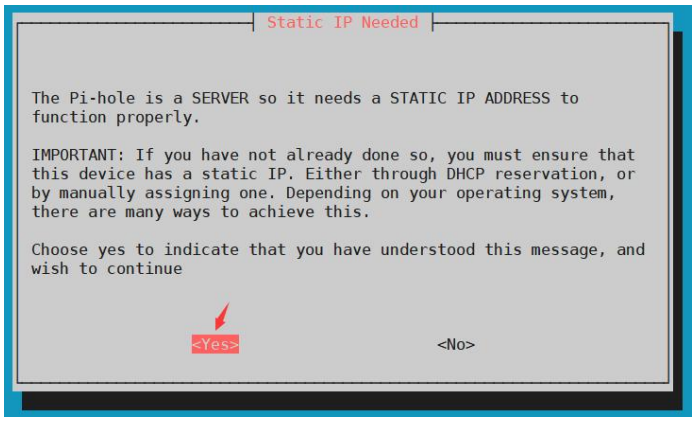


3) Continue to press Enter



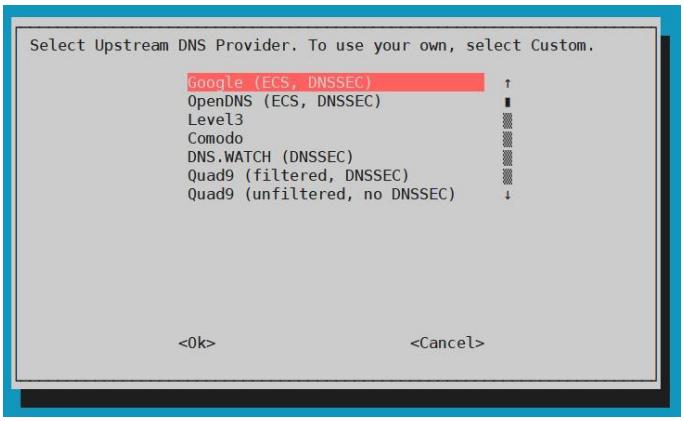
4) Then Pi-hole will prompt you to set a static IP address, please select **Yes** and press Enter

Here, you can not set the static IP address first, and then set it after the installation is completed. For the setting method of the static IP address, please refer to the description in the section on the method of setting the static IP address.

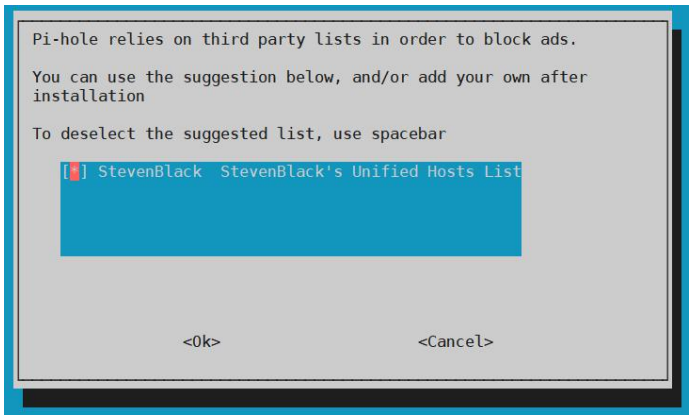


5) Then select the network interface. If you use a network cable, select eth0. If you want to use WIFI, select wlan0. After selecting, press Enter.

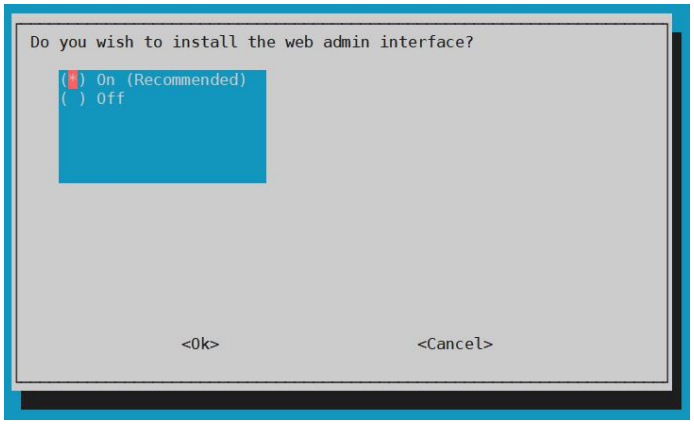
6) Then select the DNS provider, generally select Google.



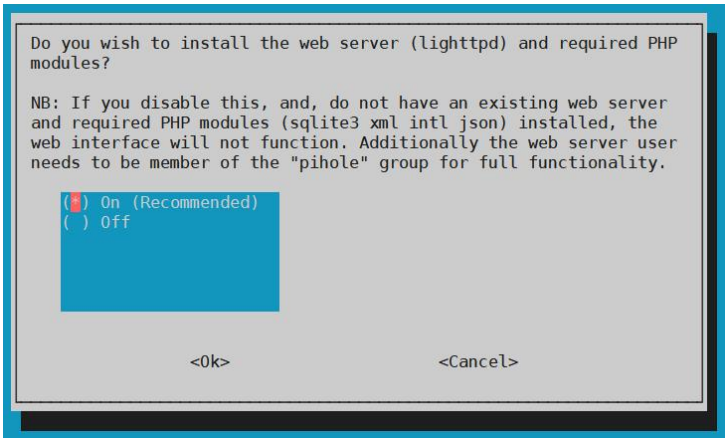
7) Then select the rule list and press Enter



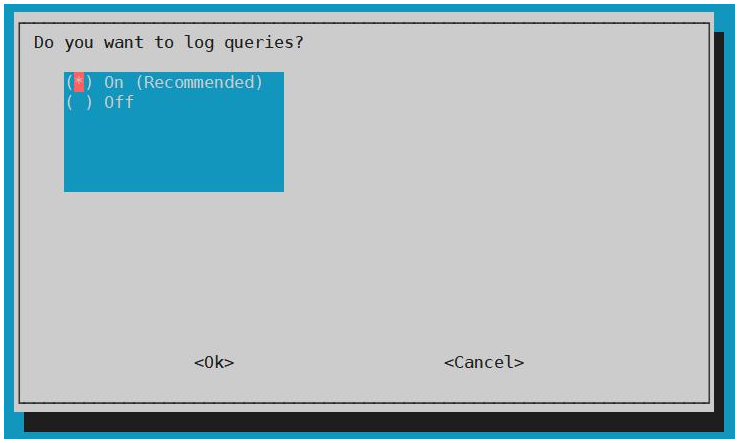
8) Then choose whether to install the web management interface, and press Enter to select the installation



9) Then press Enter to choose to install the web server and the required PHP modules

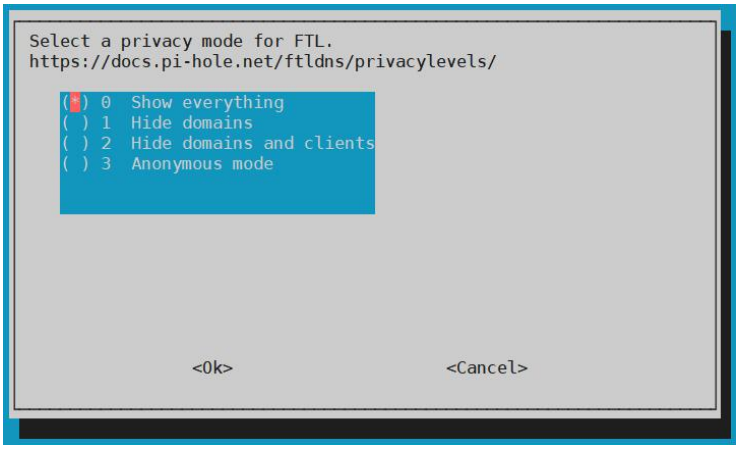


10) Then press Enter to select Open Log

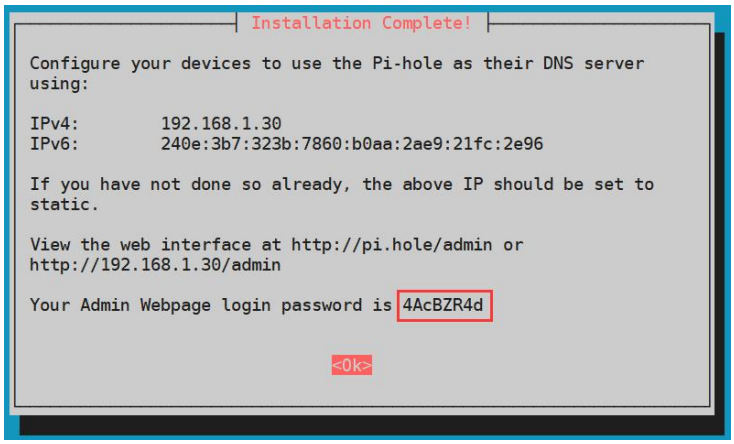


11) Then select the privacy mode, and select the default **0 Show everything** during installation. Anyway, it can be modified after the installation is completed. Please check the following link for the difference between the different options of the privacy mode:

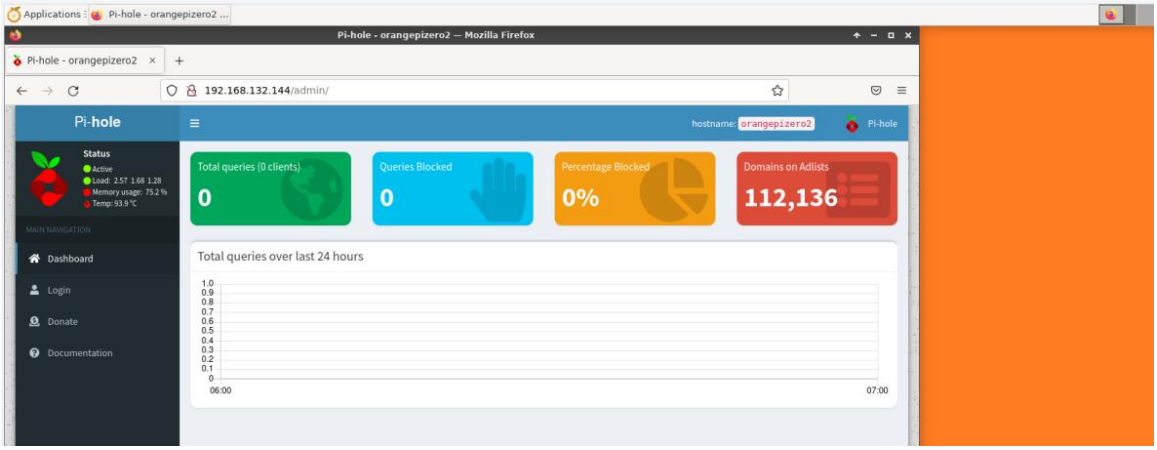
https://docs.pi-hole.net/ftldns/privacylevels/



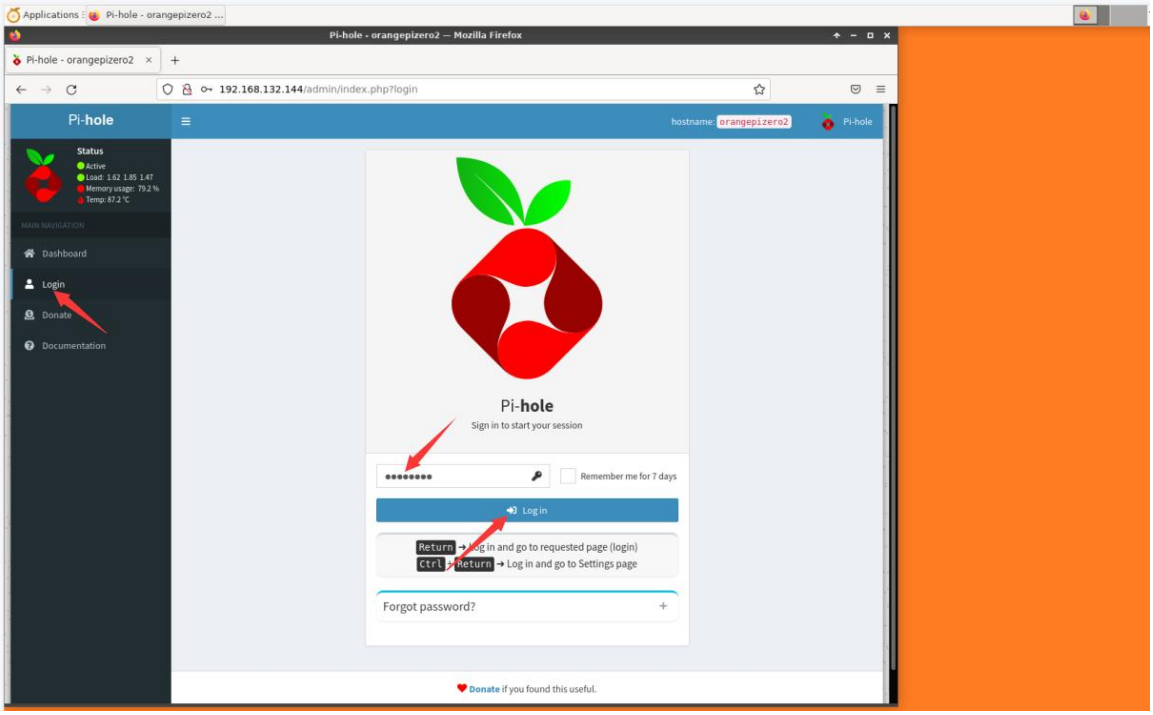
12) Then wait for the pi-hole installation to complete, and finally the following information will be prompted. The key point is to remember the web interface login address and login password



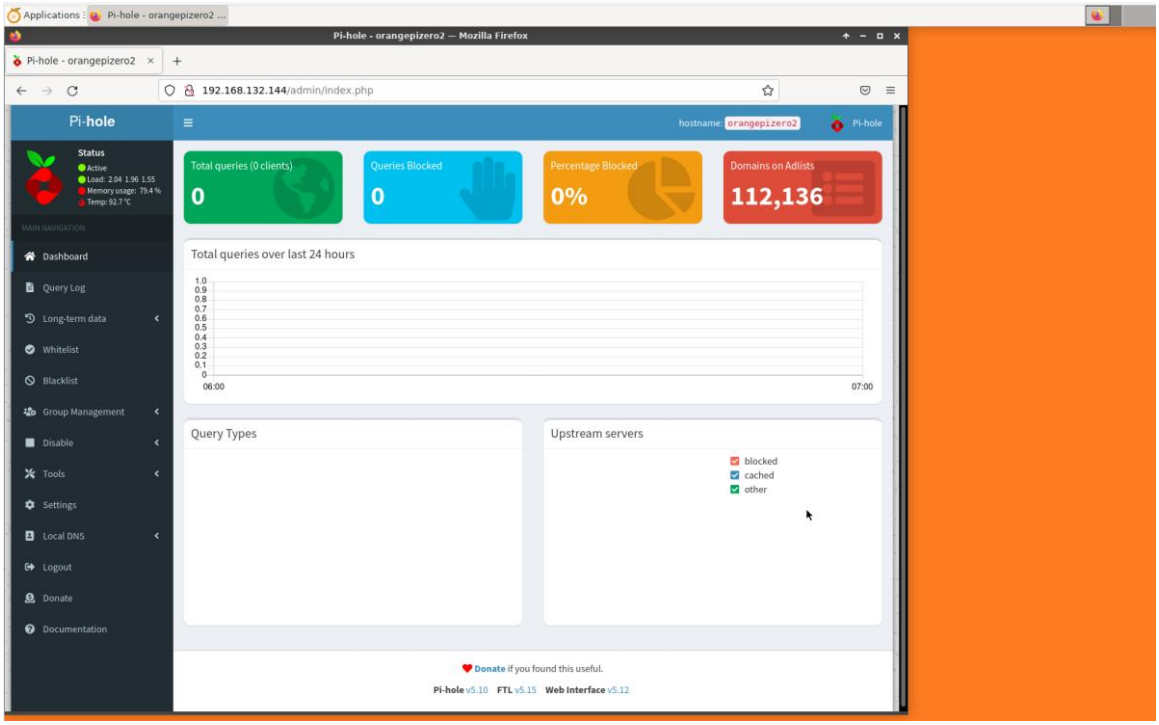
13) Then enter the **IP address /admin** of the development board in the browser to see the web management interface of pi-hole



14) Click **Login** on the left, then enter the password shown above and click **Login** below to log in to pi-hole



15) The interface after logging in is shown as follows, you can see that there are many more options than before logging in



3.43. Introduction to the use of GotoHTTP

1) GotoHTTP can be used to remotely control the development board in the browser. The installation is relatively simple, and it supports devices of various platforms. As long as there is a browser to access the Internet, the development board can be remotely controlled. The link to the GotoHTTP official website documentation is as follows, please read it carefully before using it

https://gotohttp.com/goto/help.12x

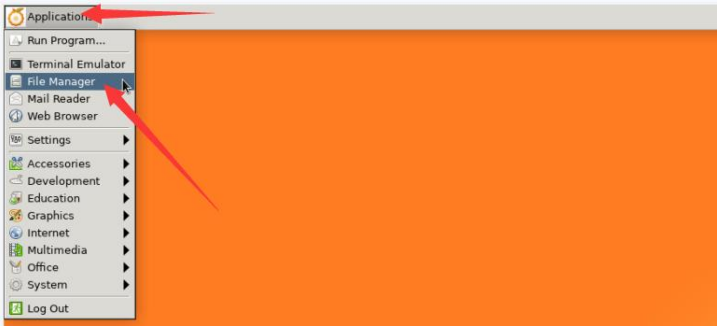
2) GotoHTTP has two versions of compressed packages that provide graphical interface and character interface. If you want to use the graphical interface, first make sure that the Linux system used by the development board is the desktop version system

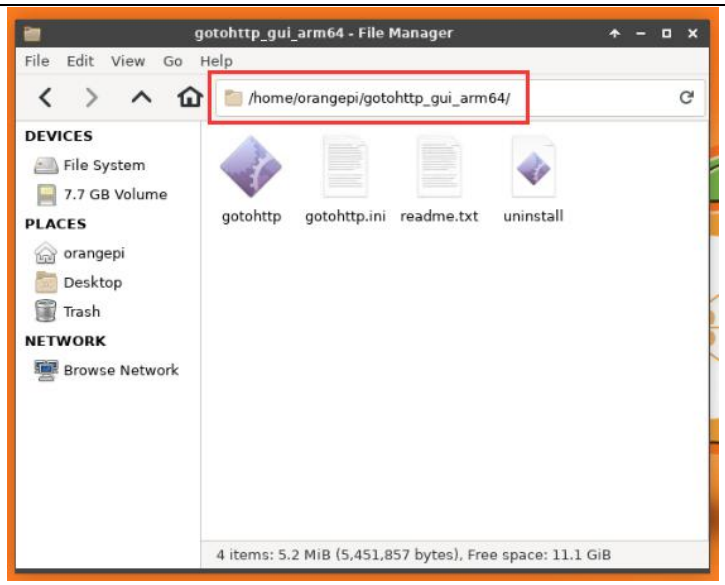
3) The graphical interface version of the GotoHTTP download and run command is as follows

a. First download and unzip **gotohttp_gui_arm64.tar.gz**

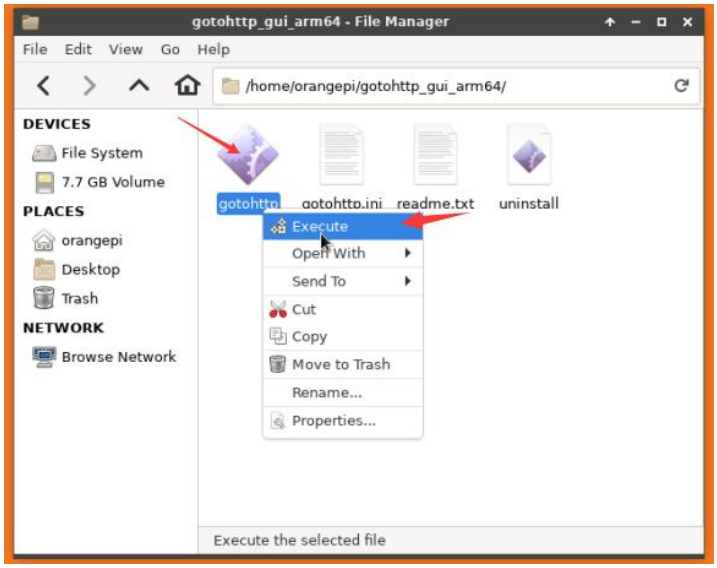
```
orangepi@orangepi:~$ wget http://gotohttp.com/gotohttp_gui_arm64.tar.gz
orangepi@orangepi:~$ tar -xvf gotohttp_gui_arm64.tar.gz
```

b. Then open the folder where **gotohttp_gui_arm64** is located in the desktop





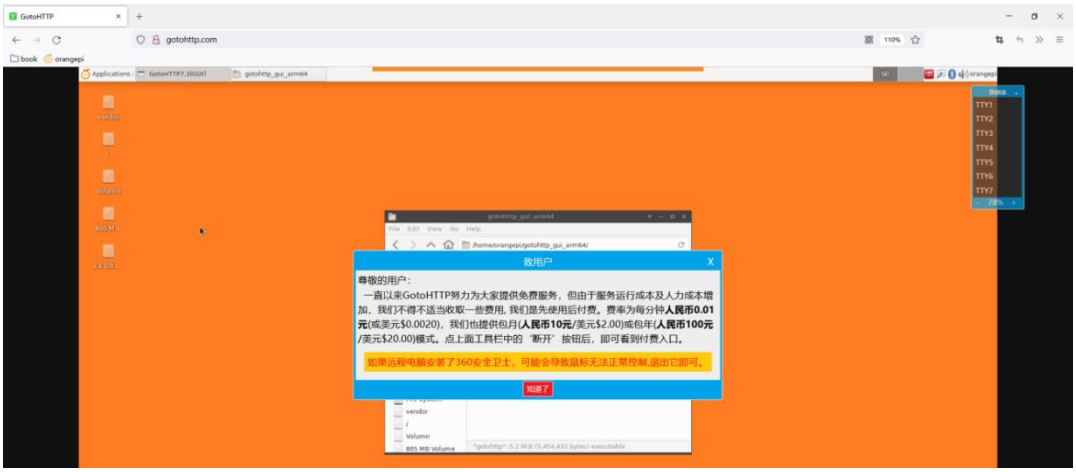
c. Then select the gotohttp executable program, right-click, then select **Execute** to open gotohttp



d. The interface after d.gotohttp is opened is as follows, the key point is to remember **Computer ID** and **Access Code**

e. Then open the browser of the computer or mobile phone, enter **http://gotohttp.com** in the address bar, then enter the **Computer Id** shown above in the **computer ID** column on the right, enter the **Access Code** shown above in the **control code**, and finally Click to start control

f. Then you can see the desktop of the Linux system of the development board in the browser. After gotohttp is opened, a payment prompt will pop up. **After use, you need to pay**



4) The command to download and run the GotoHTTP version of the character interface version is as follows

a. First download and decompress **gotohttp_cli_arm64.tar.gz**

```
orangepi@orangepi:~$ wget http://gotohttp.com/gotohttp_cli_arm64.tar.gz
orangepi@orangepi:~$ tar -xvf gotohttp_cli_arm64.tar.gz
```

b. Then use the following command to run GotoHTTP

```
orangepi@orangepi:~$ sudo ./gotohttp_cli_arm64/gotohttp_cli
```

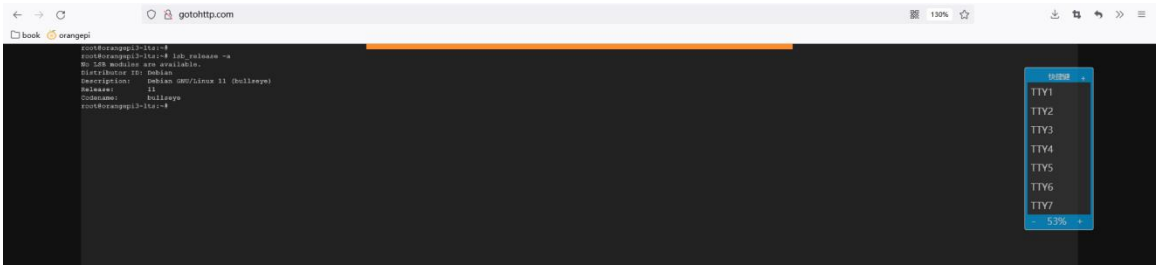
c. If you can see the following output after running GotoHTTP, it means the startup is normal. In addition, you can see that there are output **Computer Id** and **Access Code** below, please remember them, which will be used later

```
Starting GotoHTTP...
orangepi@orangepi:~$ Registering to server.....
Connecting to server.....
Connected.(secure connection)

File Transfer: ON,    Try Framebuffer: OFF

This computer is ready for controlling remotely.
To control it, please open the URL http://gotohttp.com in web browser, then input Computer Id: 137133946 and Access Code: 6103
```

d. Then open the browser of the computer or mobile phone, enter **http://gotohttp.com** in the address bar, then enter the **Computer Id** shown above in the **computer ID** column, enter the **Access Code** shown above in the **control code**, and finally click start control

e. Then you can see the command line input interface of the development board

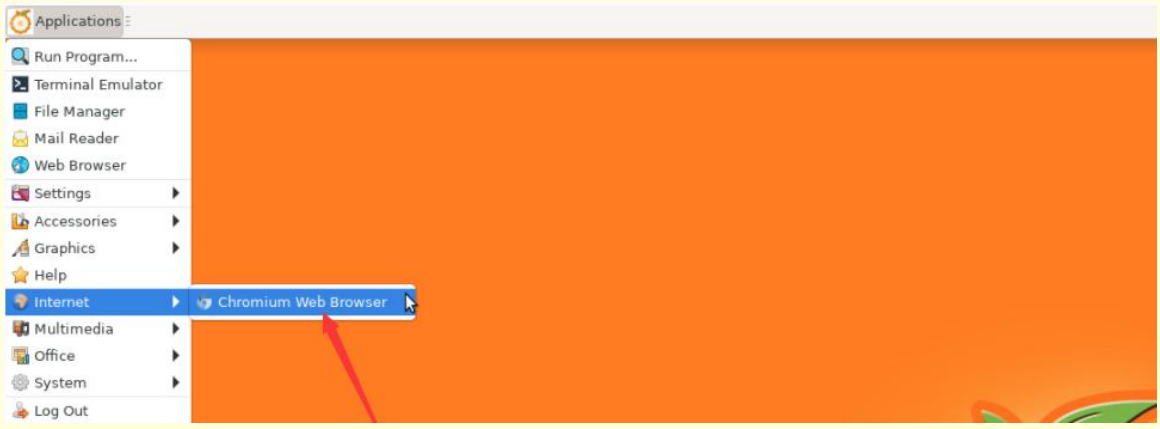
Linux system in the browser



3.44. Ubuntu22.04 method of installing browser

3.44.1. Ubuntu22.04 Chromium browser installation method

If you are using the latest version of the Ubuntu 22.04 image, the deb version of the Chromium browser is pre-installed. Before installation, please check whether the Chromium browser already exists in the following location. If so, you don't need to install it again, you can use it directly.



The software repository of Ubuntu 22.04 only supports the snap version of the Chromium browser, but it cannot be opened due to the GPU problem after installation, so please do not install the Chromium browser through the snap method.

1) Although the snap version of the Chromium browser that comes with Ubuntu 22.04 cannot be used, we can install the deb version of the Chromium browser through the following method：

a. The storage address of the Chromium browser installation package of the a.deb version is as follows

https://launchpad.net/~saiarcot895/+archive/ubuntu/chromium-beta

```
https://ppa.launchpadcontent.net/saiarcot895/chromium-beta/ubuntu
```

b. First add the PPA source of the Chromium browser to the system source

```
orangepi@orangepi:~$ sudo add-apt-repository ppa:saiarcot895/chromium-beta
```

c. When you see the following prompt, please press Enter to confirm

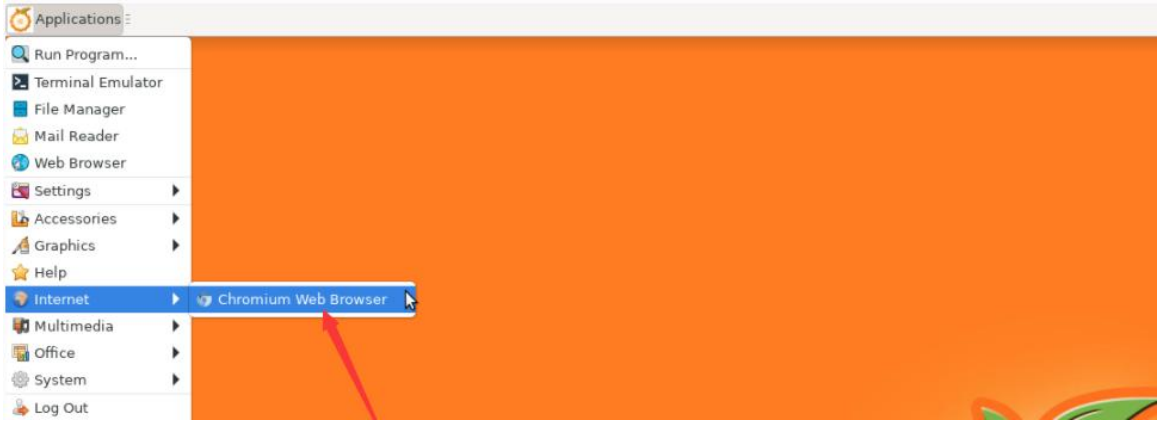
```
=== Packaging ===

Packaging for this PPA is at https://github.com/saiarcot895/chromium-ubuntu-build. A separate branch is created for each upstream branch number.
More info: https://launchpad.net/~saiarcot895/+archive/ubuntu/chromium-beta
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
```

d. Then use the following command to install the deb version of the Chromium browser

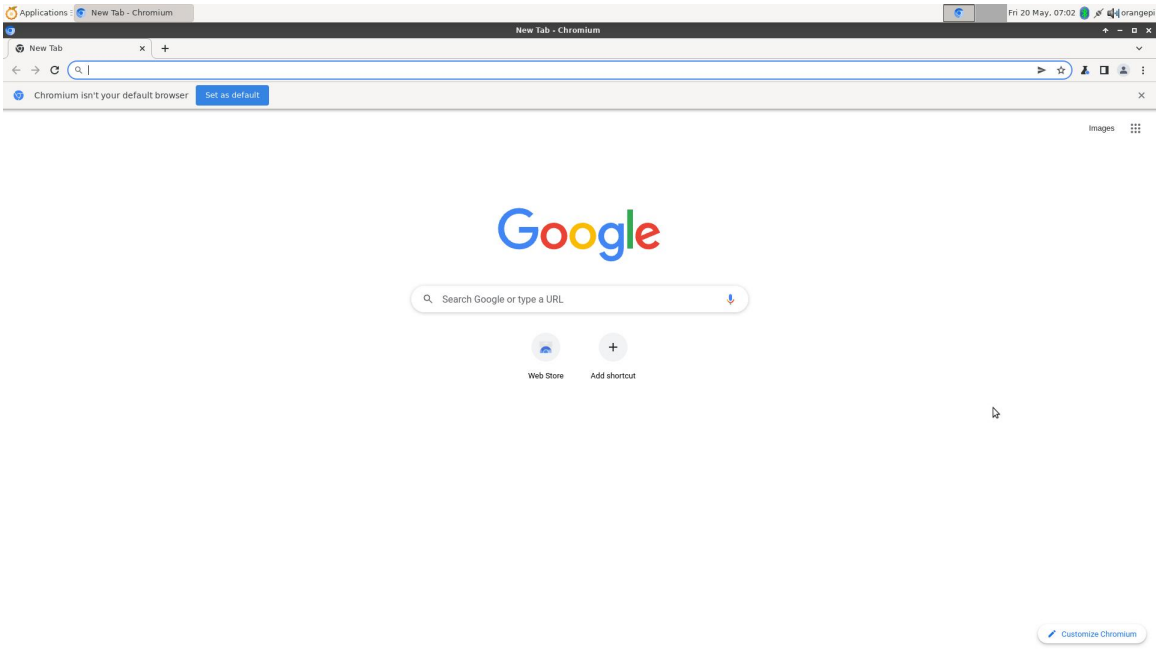
```
orangepi@orangepi:~$ sudo apt install -y chromium-browser
```

It should be noted that installing the Chromium browser in this way in China is generally very slow, and it is necessary to ensure that the network environment is better.

e. After installation, you can see the shortcut of Chromium browser in the application



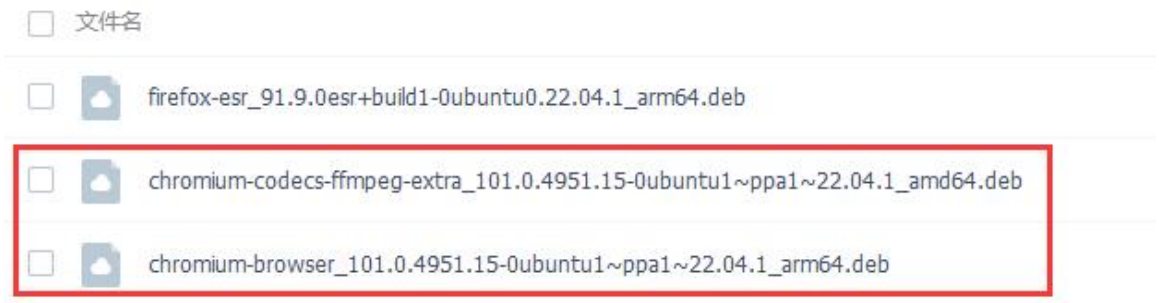
f. The display after opening the Chromium browser is as follows



2) The display after opening the Chromium browser is as follows

a. The deb package required by the Chromium browser can be downloaded from the link below

https://drive.google.com/drive/folders/1c7o6fBjTZYoervmc7oTtGemL5oww6tz0?usp=sharing



b. After downloading, upload it to the Ubuntu system of the development board

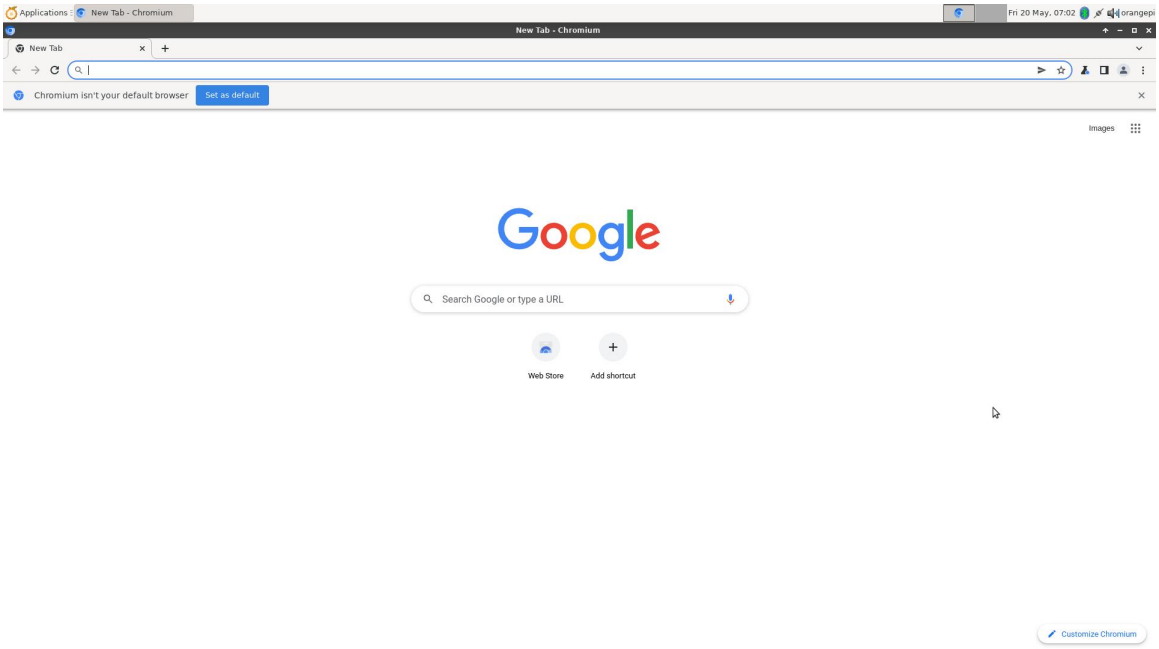
c. Then use the following command to install the deb package of the Chromium browser

```
orangepi@orangepi:~$ sudo dpkg -i  \
chromium-codecs-ffmpeg-extra_101.0.4951.15-0ubuntu1~ppa1~22.04.1_arm64.deb
orangepi@orangepi:~$ sudo dpkg -i  \
chromium-browser_101.0.4951.15-0ubuntu1~ppa1~22.04.1_arm64.deb
```

d. After installation, you can see the shortcut of the Chromium browser in the

application

e. The display after opening the Chromium browser is as follows



3.44.2. Installation method of Ubuntu22.04 Firefox browser

1) Ubuntu22.04 only supports the snap version of the Firefox browser by default. The installation command is:

```
orangepi@orangepi:~$ sudo apt install firefox
```

2) The firefox-esr version browser can be installed by the following methods:

a. First add firefox-esr version PPA source

```
orangepi@orangepi:~$ sudo add-apt-repository ppa:mozillateam/ppa
PPA publishes dbgsym, you may need to include 'main/debug' component
Repository: 'deb https://ppa.launchpadcontent.net/mozillateam/ppa/ubuntu/ jammy main'
Description:
```

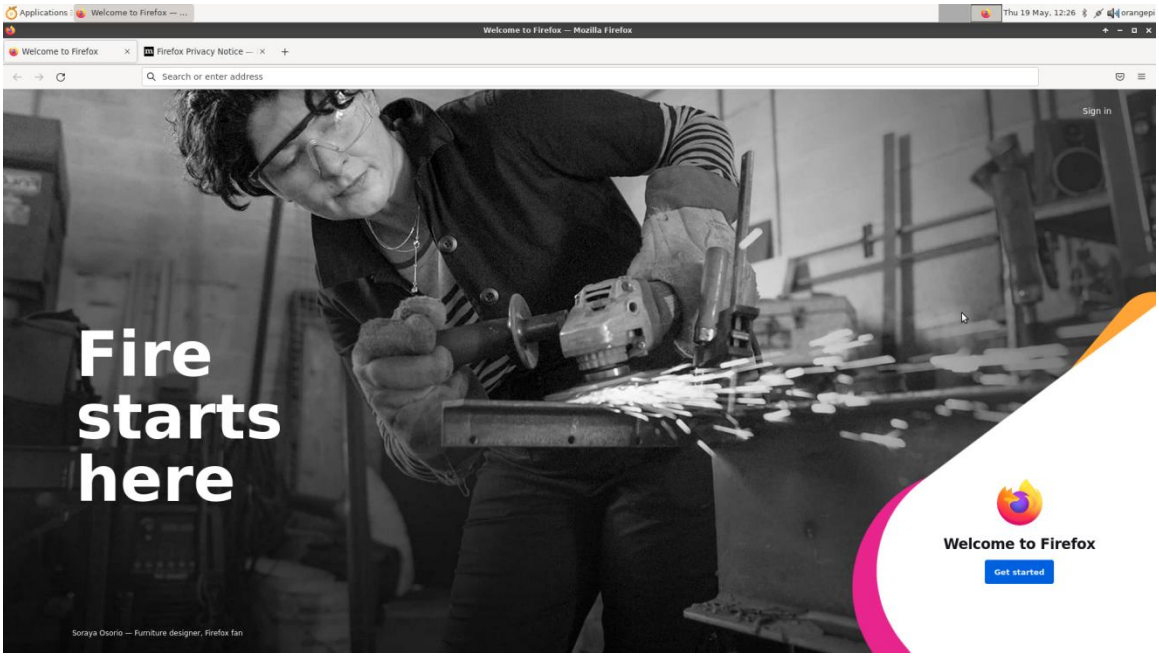
```
Mozilla Team's Firefox 91 ESR and Thunderbird 91 or 78 stable builds
More info: https://launchpad.net/~mozillateam/+archive/ubuntu/ppa
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.      <--- Please enter here to confirm
```

b. Then install firefox-esr using the command below

```
orangepi@orangepi:~$ sudo apt install firefox-esr
```

c. After installation, you can see the shortcut of firefox-esr browser in the application

d. The display after opening the firefox-esr browser is as follows



3.45. GPU Test Instructions

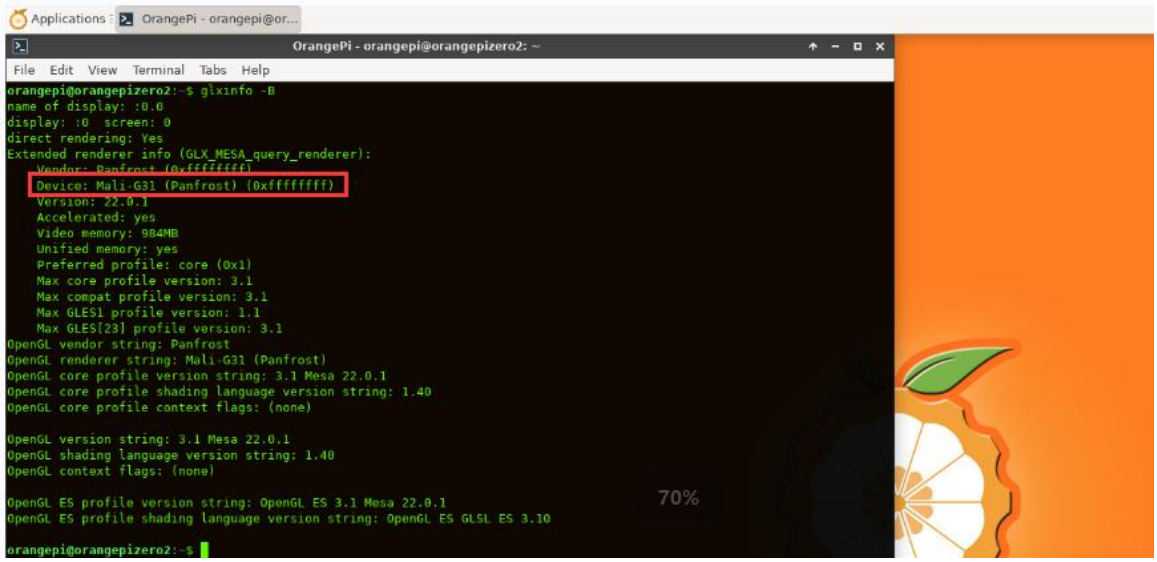
3.45.1. Ubuntu22.04 Linux5.16 system GPU test instructions

Note that the following tests are all tested on the desktop version of the system, so please make sure that the system used by the development board is the desktop version of the system.

1) Please connect the HDMI display first, all the following commands are operated in the desktop displayed by HDMI, please do not use ssh remote login or use serial port to log in to the Linux system

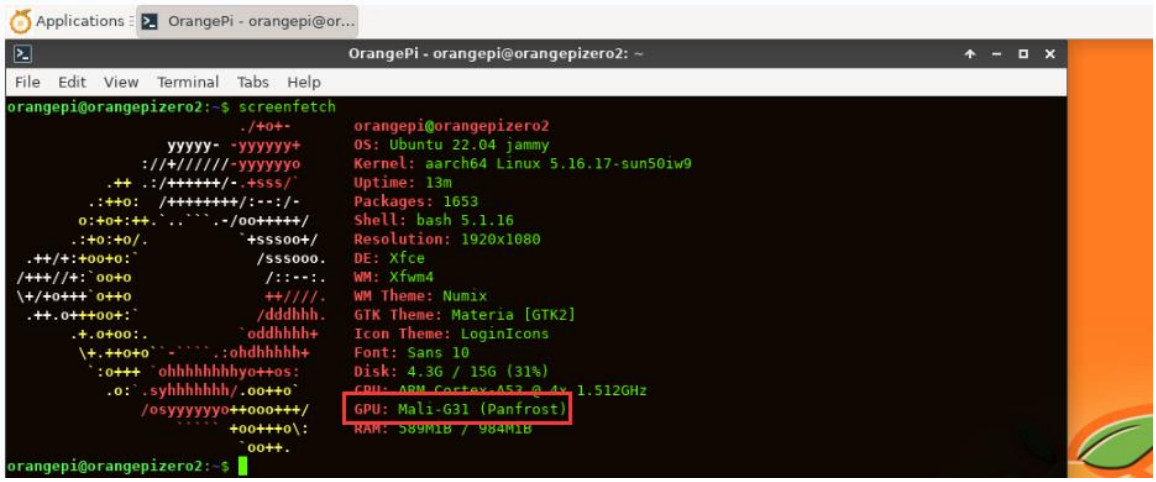
2) After entering the desktop, first open a terminal, and then use the **glxinfo -B** command to see that the GPU driver used is **Mali-G31 (Panfrost)** instead of **llvmpipe**

```
orangepi@orangepi:~$ glxinfo -B
```



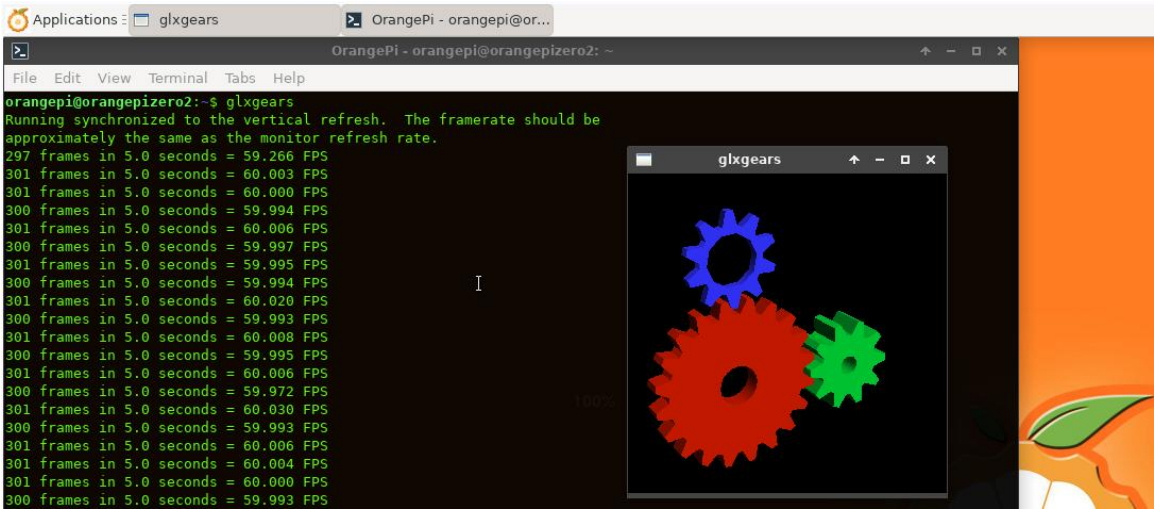
3) Using the **screenfetch** command, you can also see that the GPU driver is using **Mali-G31 (Panfrost)**

```
orangepi@orangepi:~$ sudo apt install -y screenfetch
orangepi@orangepi:~$ screenfetch
```



4) **The Glx-Gears** test is shown below

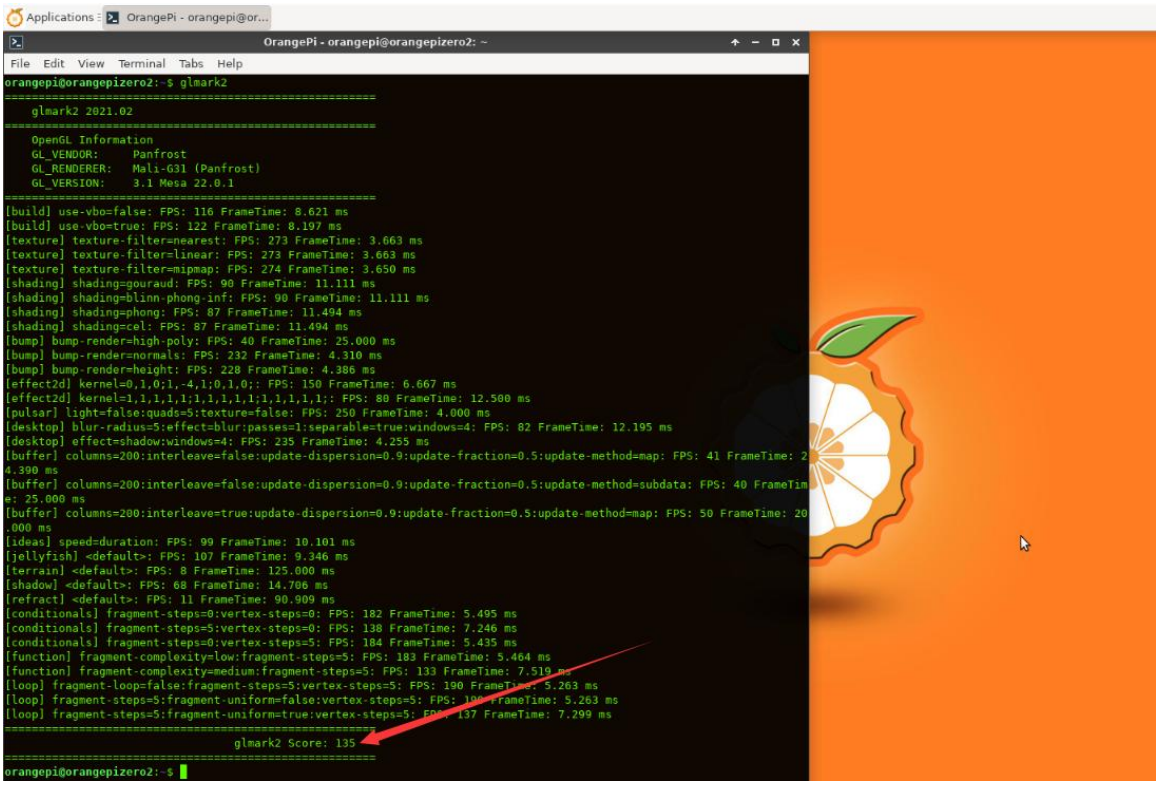
```
orangepi@orangepi:~$ glxgears
```



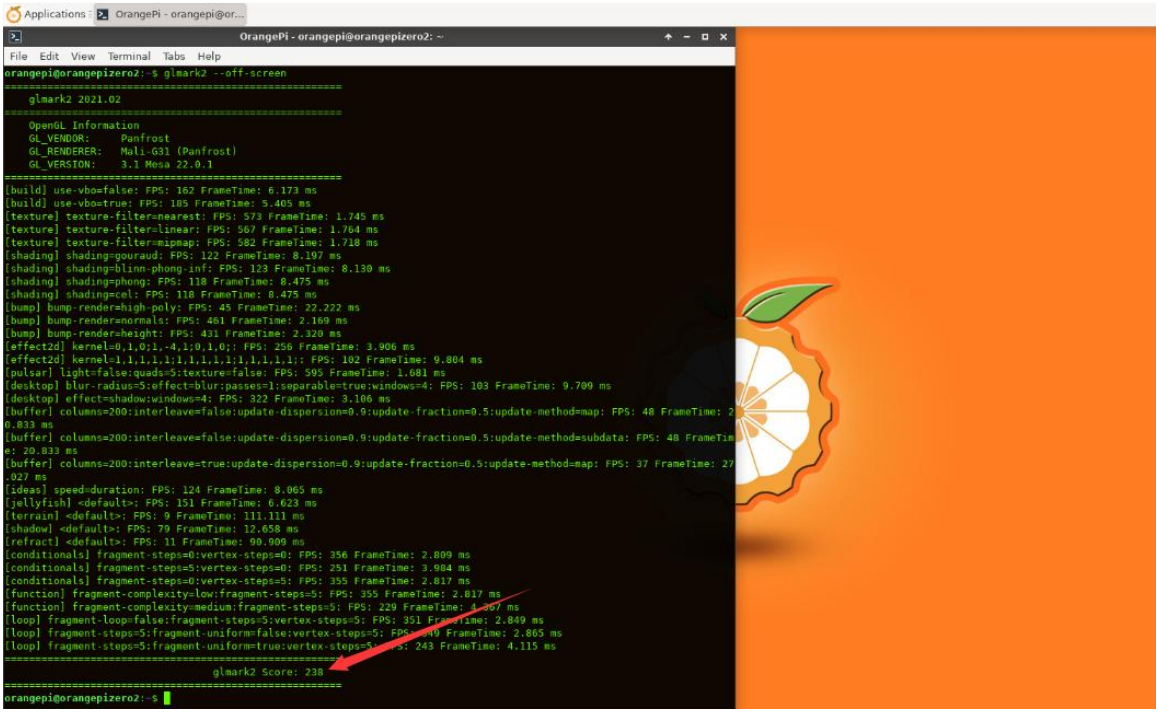
5) **glmark2-es2** is a benchmark tool for OpenGL (ES) 2.0. Using glmark2 can test the performance of GPU OpenGL ES 2.0

```
orangepi@orangepi:~$ sudo apt install glmark2-es2
orangepi@orangepi:~$ glmark2-es2
orangepi@orangepi:~$ glmark2-es2 --off-screen
```

a. The test scores for the a.**glmark2-es2** command are as follows

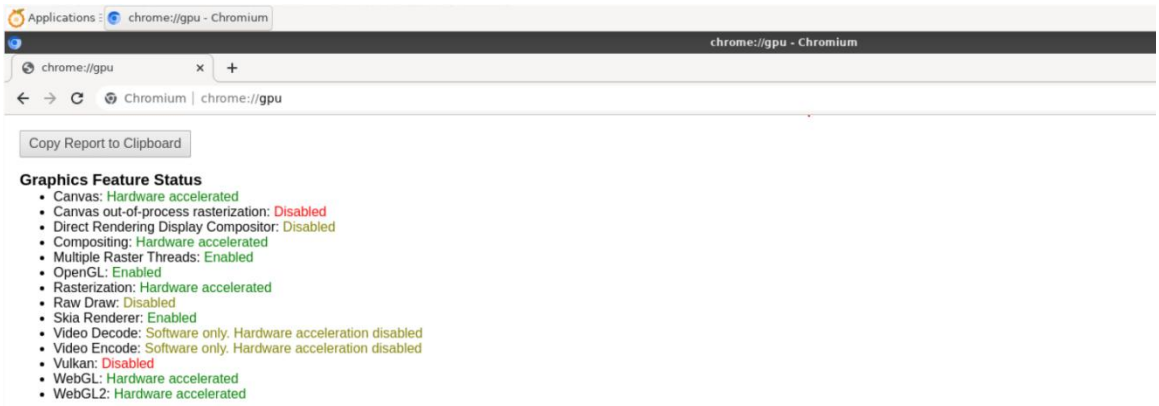


b. The test scores for the b.**glmark2-es2 --off-screen** command are as follows



6) Then you can open the Chromium browser

7) Enter **chrome://gpu** in the address bar to check the GPU support



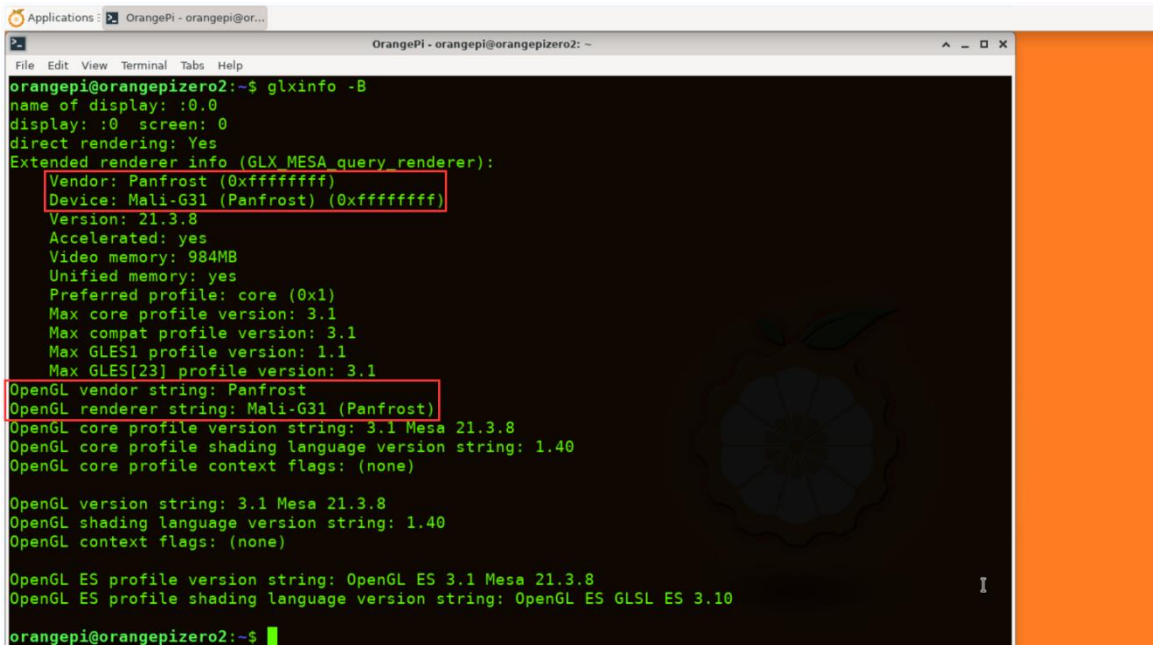
3.45.2. Debian12 Linux5.16 system GPU test instructions

Note that the following tests are all tested on the desktop version of the system, so please make sure that the system used by the development board is the desktop version of the system.

1) Please connect the HDMI display first, all the following commands are operated in the desktop displayed by HDMI, please do not use ssh remote login or use serial port to log in to the Linux system

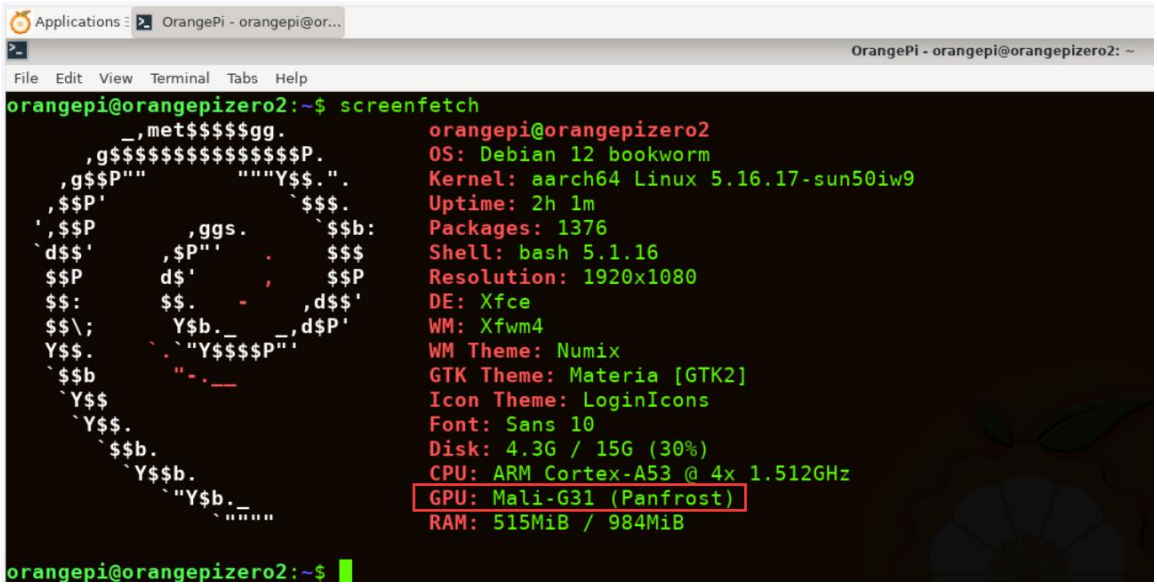
2) After entering the desktop, first open a terminal, and then use the **glxinfo -B** command to see that the GPU driver used is **Mali-G31 (Panfrost)** instead of **llvmpipe**

```
orangepi@orangepi:~$ glxinfo -B
```



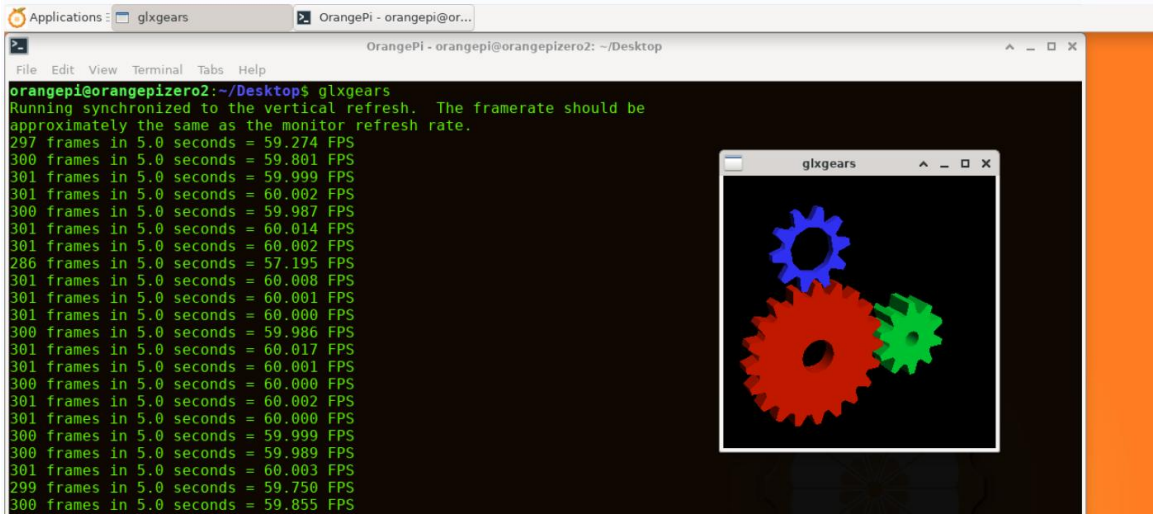
3) Using the **screenfetch** command, you can also see that the GPU driver is using **Mali-G31 (Panfrost)**

```
orangepi@orangepi:~$ sudo apt install -y screenfetch
orangepi@orangepi:~$ screenfetch
```



4) The **Glx-Gears** test is shown below

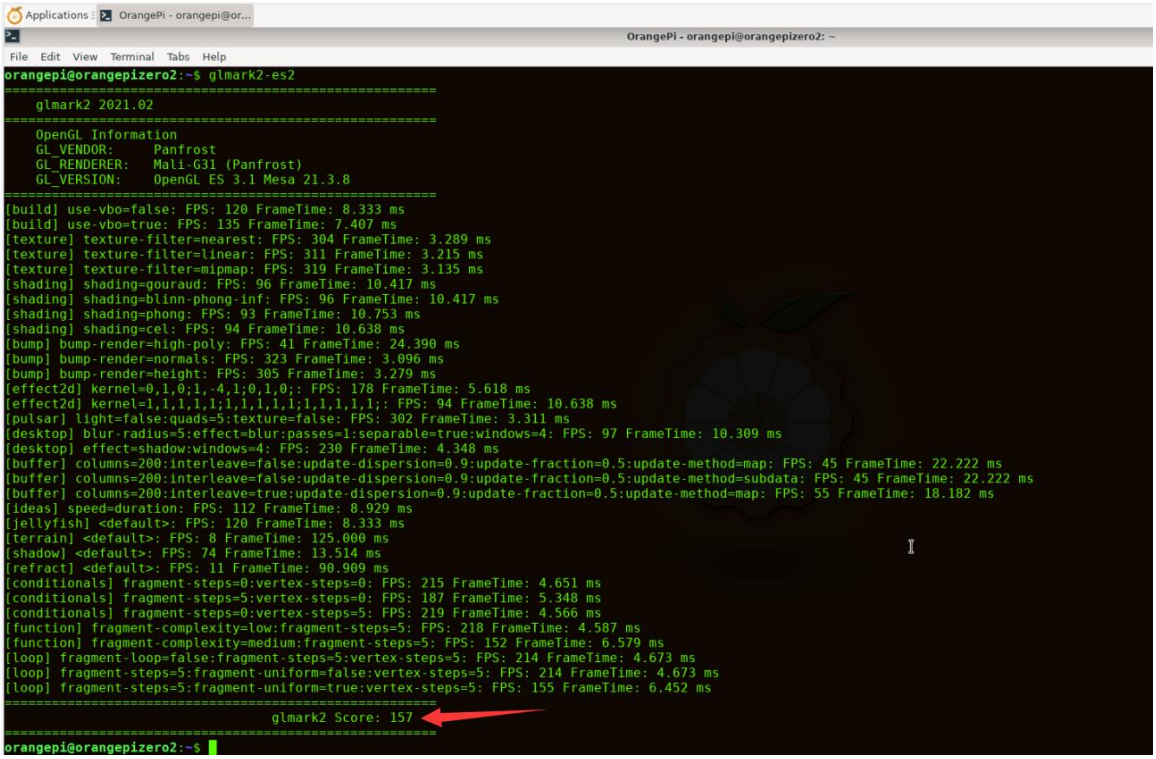
```
orangepi@orangepi:~$ glxgears
```



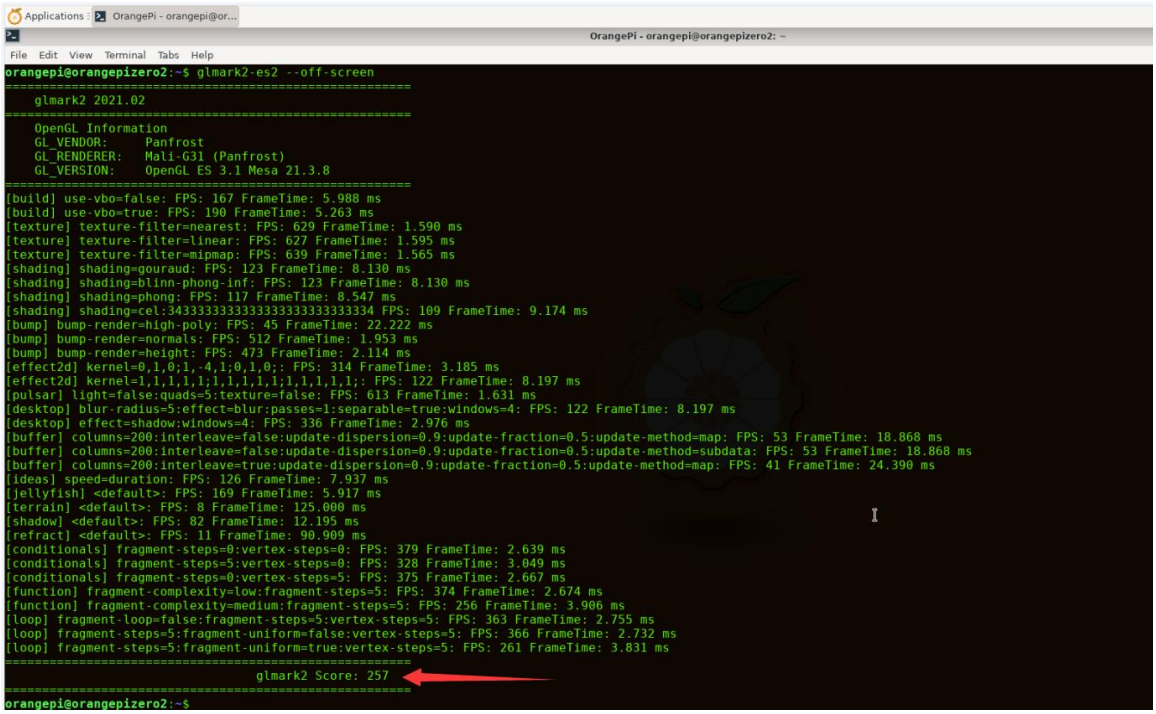
5) **glmark2-es2** is a benchmark tool for OpenGL (ES) 2.0. Using glmark2 can test the performance of GPU OpenGL ES 2.0

```
orangepi@orangepi:~$ sudo apt install glmark2-es2-x11
orangepi@orangepi:~$ glmark2-es2
orangepi@orangepi:~$ glmark2-es2 --off-screen
```

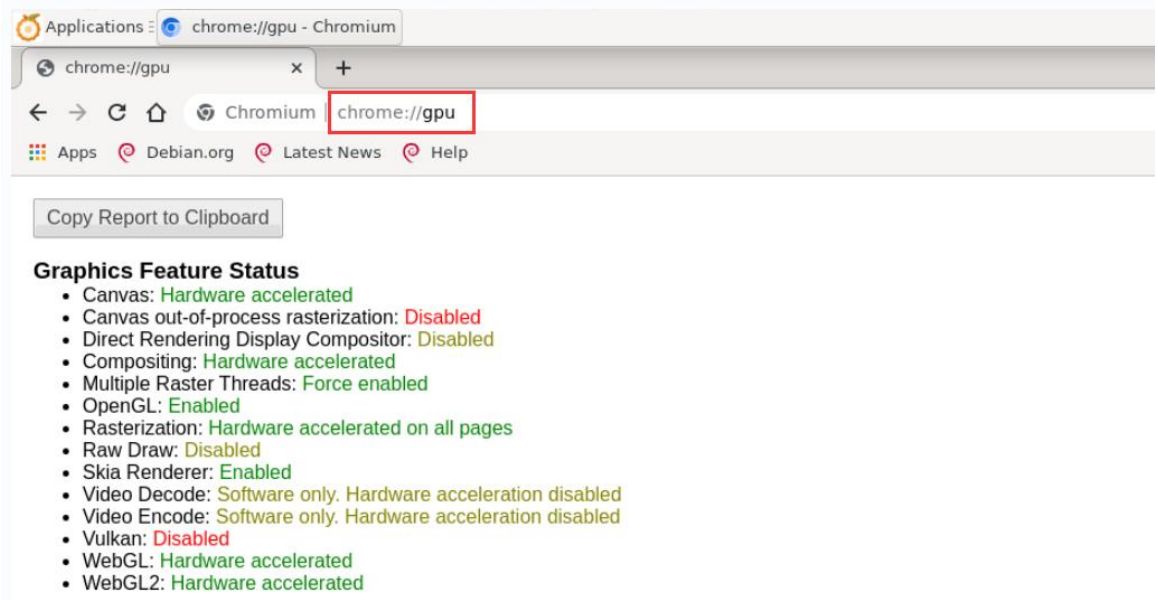
a. The test scores for the a.**glmark2-es2** command are as follows



b. The test scores for the **glmark2-es2 --off-screen** command are as follows



6) In addition, you can open the Chromium browser, and then enter **chrome://gpu** in the address bar to view the GPU support



3.46. Partial programming language test supported by Linux system

3.46.1. Debian Buster System

1) Debian Buster is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of a.gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 8.3.0-6) 8.3.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
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 Buster is installed with Python2 and Python3 by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python
Python 2.7.16 (default, Oct 10 2019, 22:02:15)
[GCC 8.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
orangepi@orangepi:~$ python3
Python 3.7.3 (default, Jan 22 2021, 20:04:44)
```

```
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in the Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python hello_world.py
Hello World!
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Buster does not install Java compilation tools and runtime environment by default

a. You can install openjdk with the following command, the default version in Debian Buster is openjdk-11

```
orangepi@orangepi:~$ sudo apt install -y openjdk-11-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 11.0.13 2021-10-19
OpenJDK Runtime Environment (build 11.0.13+8-post-Debian-1deb10u1)
OpenJDK 64-Bit Server VM (build 11.0.13+8-post-Debian-1deb10u1, mixed mode)
```

c. Write the Java version of **hello_world.java**

```
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!
```

3.46.2. Debian Bullseye System

4) Debian Bullseye is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of a.gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 10.2.1-6) 10.2.1 20210110
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
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) 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 the Python language

```
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) 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 version of Java

```
orangepi@orangepi:~$ java --version
```

c. Write the Java version of **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!
```

3.46.3. Ubuntu Bionic system

1) Ubuntu Bionic is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of a.gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu/Linaro 7.5.0-3ubuntu1~18.04) 7.5.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
```

```
PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
root@orangepi:~# gcc -o hello_world hello_world.c
root@orangepi:~# ./hello_world
Hello World!
```

2) Ubuntu Bionic has Python3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.6.9 (default, Oct  8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in the Python language

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Ubuntu Bionic does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk, the default version in Debian Buster is openjdk-17

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 17.0.2 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-118.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-118.04, mixed mode, sharing)
```

c. Write the Java version of **hello_world.java**

```
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!
```

3.46.4. Ubuntu Focal system

1) Ubuntu Focal is installed with the gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of a.gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");
```

```
        return 0;
}
```

c. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Ubuntu Focal has Python3 installed by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.8.10 (default, Sep 28 2021, 16:10:42)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in the Python language

```
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 install openjdk-17 using the following command

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 17.0.2 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-120.04, mixed mode, sharing)
```

c. Write the Java version of **hello_world.java**

```
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!
```

3.46.5. Ubuntu Jammy system

4) Ubuntu Jammy is installed with gcc compilation toolchain by default, which can directly compile C language programs in the Linux system of the development board

a. The version of gcc is shown below

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu 11.2.0-19ubuntu1) 11.2.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
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 Python3 installed by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.10.4 (main, Apr  2 2022, 09:04:19) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

b. Write the **hello_world.py** program in the Python language

```
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 use the following command to install openjdk-18

```
orangepi@orangepi:~$ sudo apt install -y openjdk-18-jdk
```

b. After installation, you can check the version of Java

```
orangepi@orangepi:~$ java --version
openjdk 17.0.2 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-120.04, mixed mode, sharing)
```

c. Write the Java version of **hello_world.java**

```
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!
```

3.47. The method to shut down and restart the development board

1) During the operation of the Linux system, if the power is directly unplugged, some data may be lost in the file system. It is recommended to use the **poweroff** command to shut down the Linux system of the development board before powering off, and then unplug the power.

```
orangepi@orangepi:~$ sudo poweroff
```

Note that after shutting down the development board, you need to unplug the power supply to turn it on.

2) Use the **reboot** command to restart the Linux system on the development board

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

4. Instructions for use of Android TV system

4.1. Supported Android Versions

Android version	kernel version
Android 10.0 TV version	**linux4.9**

4.2. Android 10 TV function adaptation

Function	State

HDMI video	OK
HDMI audio	OK
USB2.0 x 3	OK
TF card boot	OK
Network card	OK
IR	OK
WIFI	OK
WIFI hotsport	OK
Bluetooth	OK
BLE Bluetooth	OK
Headphone Audio	OK
TV-OUT	OK
USB Camera	OK
LED Light	OK
Temperature Sensor	OK
Hardware watchdog	OK
Mali GPU	OK
Video codec	OK

4.3. On-board LED light display description

	green light	red light
u-boot startup phase	Off	On
The kernel boots into the system	On	Off
GPIO port	PC13	PC12

4.4. The method to return to the previous interface on Android

1) We generally use the mouse and keyboard to control the Android system of the development board. When entering some interfaces and need to return to the previous interface or desktop, you can only return to the **right mouse button**, and the keyboard cannot be returned.

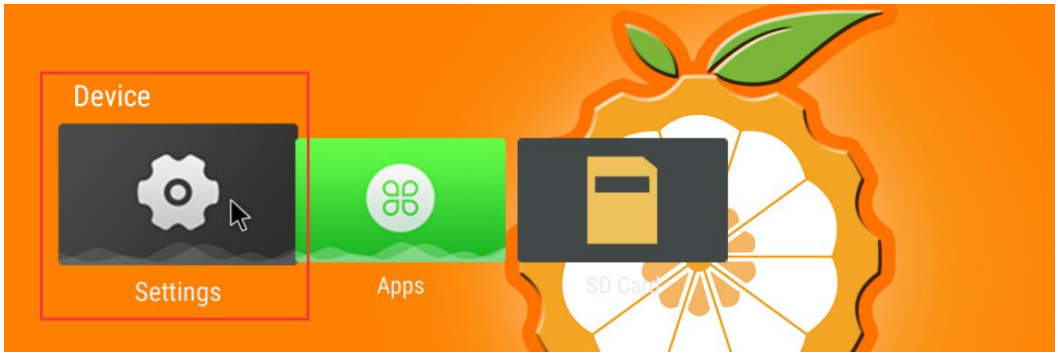
2) If you have purchased an infrared remote control (other remote controls are not available) and an expansion board that are matched with the development board, after inserting the expansion board into the development board, you can return to the previous

menu through the return key in the remote control. The position of the return key is as shown in the figure below.

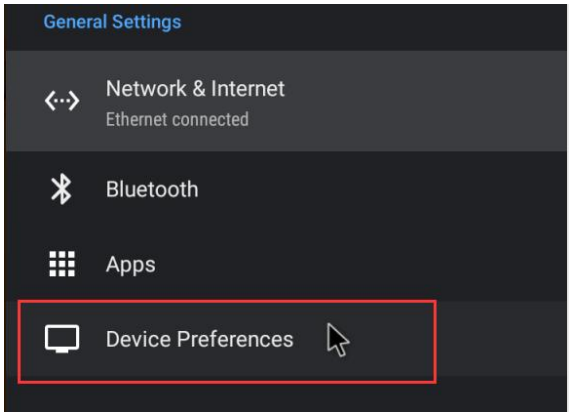
4.5. The method to use ADB

4.5.1. Enable USB debugging option

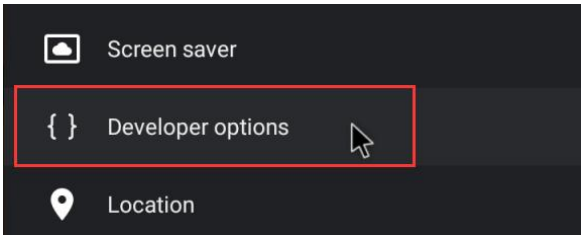
1) Select **Settings**



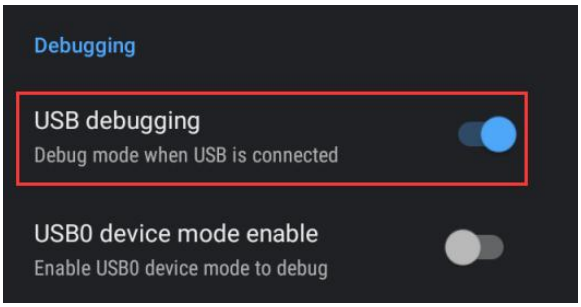
2) Then select **Device Preferences**



3) Then select **Developer options**



4) Finally find **USB debugging** and make sure it is turned on



4.5.2. Using network connection adb debugging

Using network adb does not require the USB Typc C interface data cable to connect the computer and the development board, but communicates through the network, so first make sure that the wired or wireless network of the development board has been connected, and then obtain the IP address of the development board, and then to use.

1) Make sure the **USB debugging** option is turned on

2) Make sure that the **service.adb.tcp.port** of the Android system is set to the port

number 5555

```
cupid-p2:/ # getprop | grep "adb.tcp"
[service.adb.tcp.port]: [5555]
```

3) If **service.adb.tcp.port** is not set, you can use the following command to set the port number of network adb

```
cupid-p2:/ # setprop service.adb.tcp.port 5555
cupid-p2:/ # stop adbd
cupid-p2:/ # start adbd
```

4) Install adb tool on Ubuntu PC

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

5) Then connect 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
```

6) Then you can log in to the android system through adb shell on the Ubuntu PC

```
test@test:~$ adb shell
cupid-p2:/ #
```

4.5.3. Use the data cable to connect adb debugging

1) First make sure the **USB debugging option** is turned on

2) Prepare a USB Typc C interface data cable, insert one end of the USB interface into the USB interface of the computer, and insert one end of the USB Type C interface into

the power interface of the development board. In this case, the development board is powered by the computer's USB interface, so please make sure that the computer's USB interface can provide the most power to drive the development board

3) Install adb tool on Ubuntu PC

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

4) Check to identify the ADB device

```
test@test:~$ adb devices
List of devices attached
8c00141167058911ccd     device
```

5) Then you can log in to the android system through adb shell on the Ubuntu PC

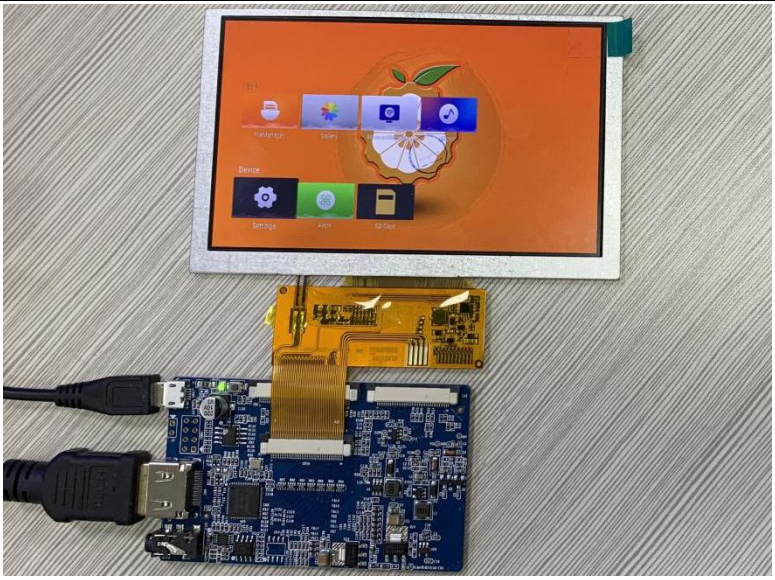
```
test@test:~$ adb shell
cupid-p2:/ #
```

4.6. Orange Pi 5-inch TFT LCD screen test

1) First, prepare the 5-inch TFT LCD screen of the orange pi. The wiring method of the screen cable and the adapter board is as shown in the figure below. Please do not reverse it. In addition, please ensure that the screen cable and the cable socket are in place. , if the contact is not in place, the screen output will have problems

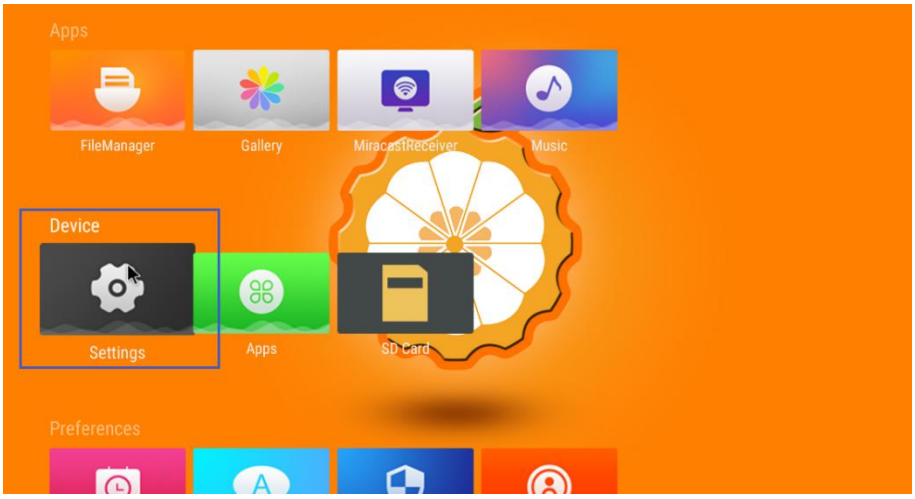
2) Then use the Micro HDMI cable to connect the Micro HDMI port of the development board to the HDMI port of the adapter board. The screen needs to be powered separately. Please insert a 5V/2A power supply into the Micro USB port of the adapter board.

3) After the development board is started, the screen can see the desktop of the Android system

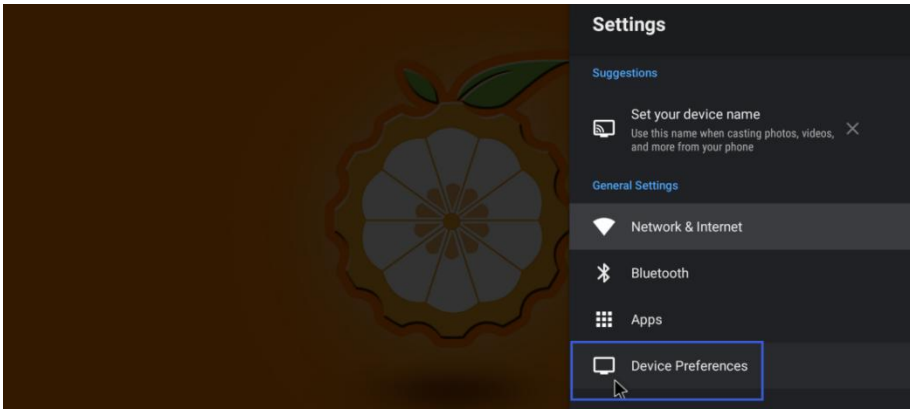


4) When the Android system is started for the first time, it is normal for the screen display to be jittery or blurry, because the default HDMI resolution of the Android system is 1080p, but the TFT LCD screen does not support this resolution. After the first boot into the Android system Please modify the HDMI output resolution to **480p** first, then the screen display will be very stable

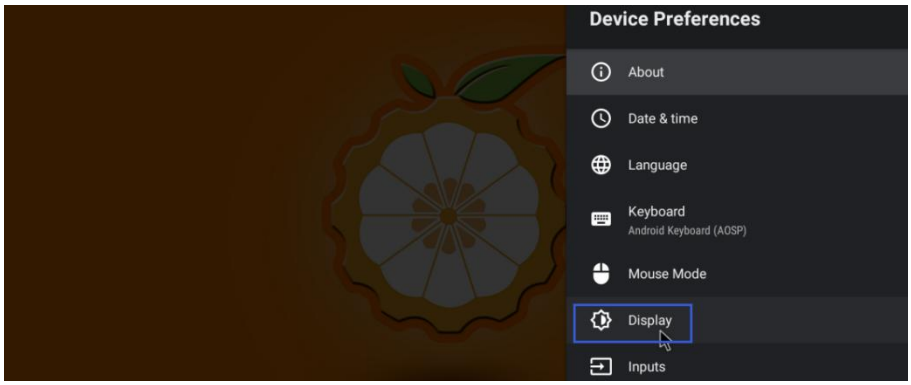
a. First select **Setting**



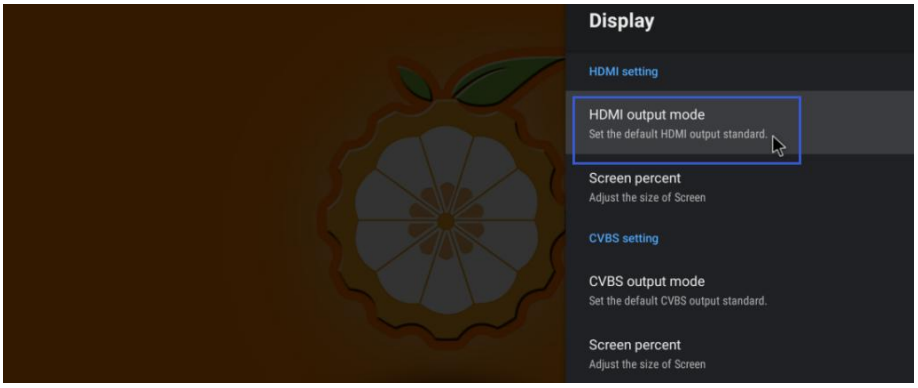
b. Then select **Device Preferences**



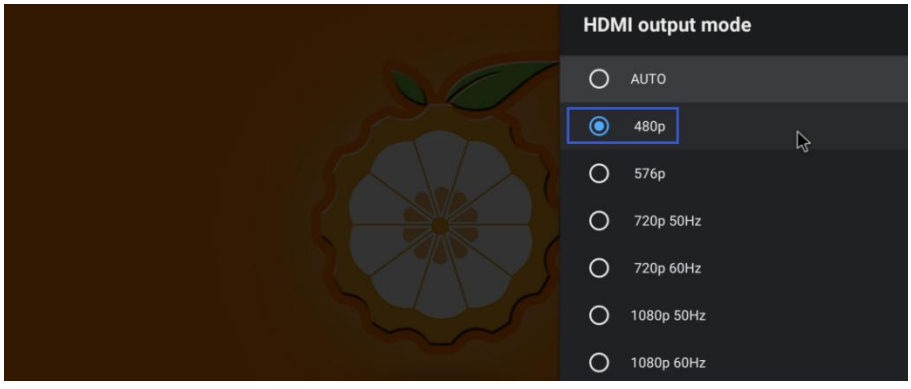
c. Then select **Display**



d. Then select **HDMI output mode**



e. Finally, select 480p to set the HDMI resolution to 480p

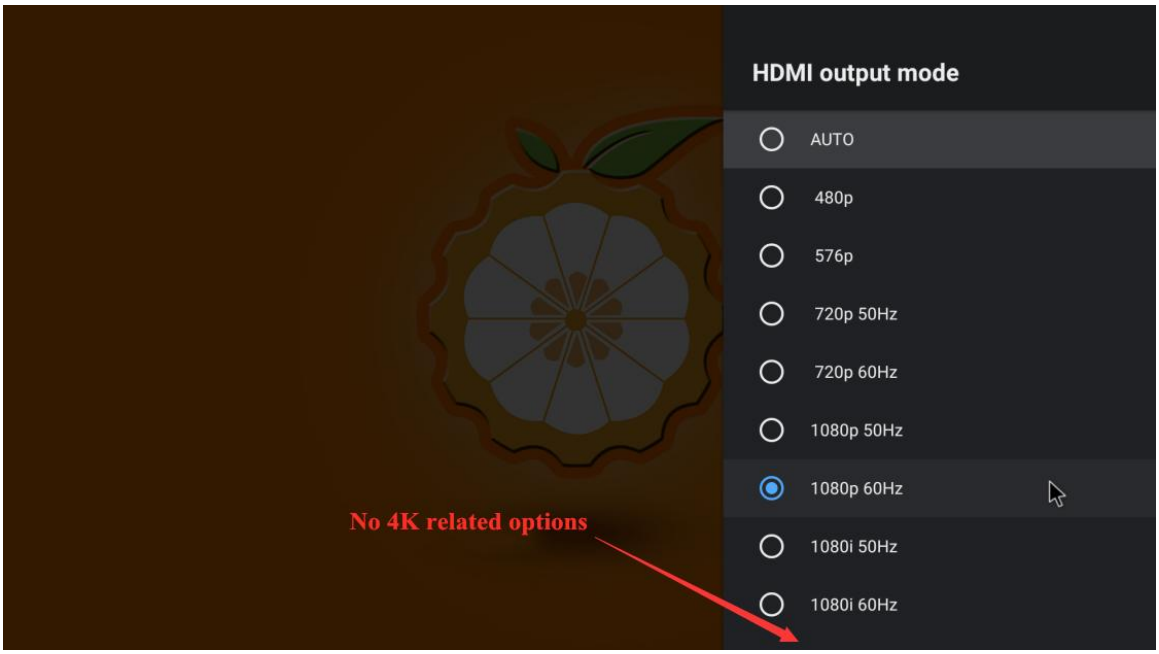


f. At this time, the display output of the LCD screen will be very stable.

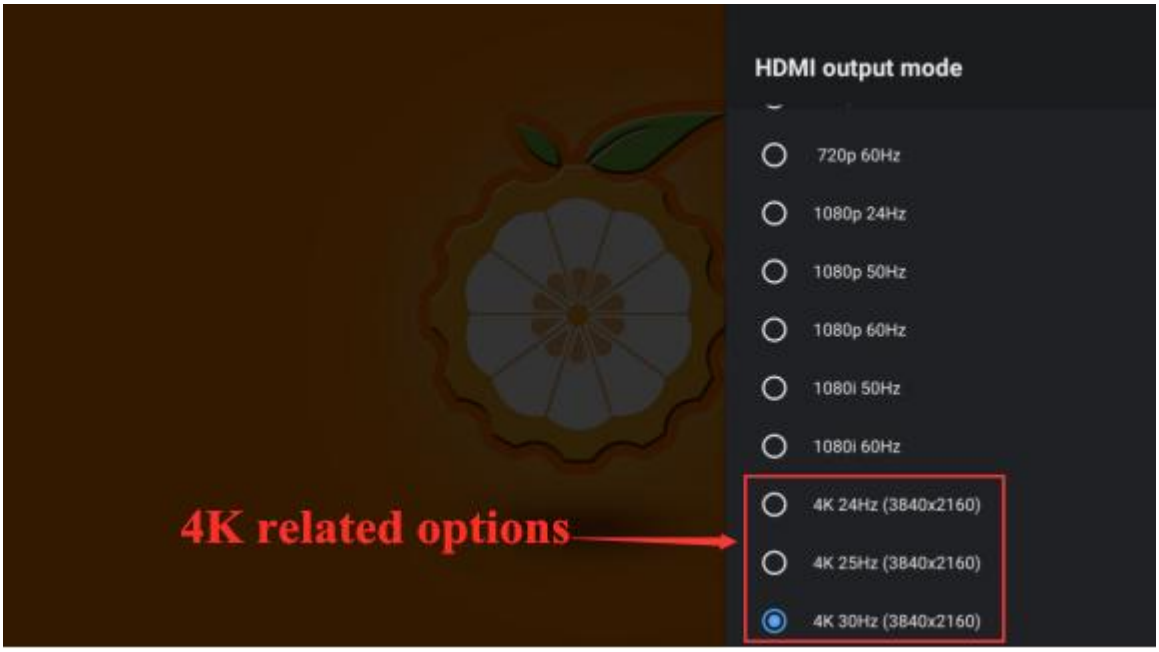
After turning off the power of the development board, please remember to turn off the power of the screen adapter board at the same time, and turn it on before starting the development board next time.

4.7. HDMI 4K Display Instructions

1) If the Micro HDMI of the development board is connected to a TV or monitor that does not support 4K, the 4K related options cannot be seen when viewing the resolution supported by HDMI in the settings



2) Only when the Micro HDMI of the development board is connected to a TV or monitor that supports 4K, the 4K-related options can be seen in the resolutions supported by HDMI



4.8. HDMI to VGA display test

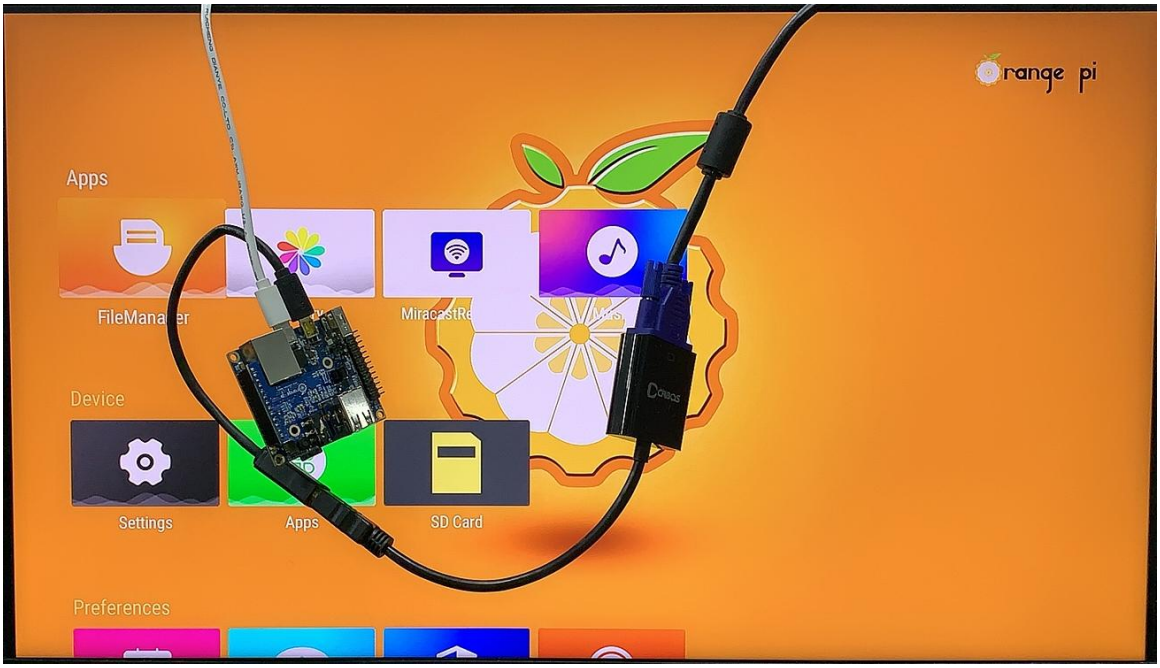
3) First you need to prepare the following accessories

a. HDMI to VGA converter

b. A VGA cable and a Micro HDMI male to HDMI female adapter cable

c. A monitor or TV that supports VGA interface

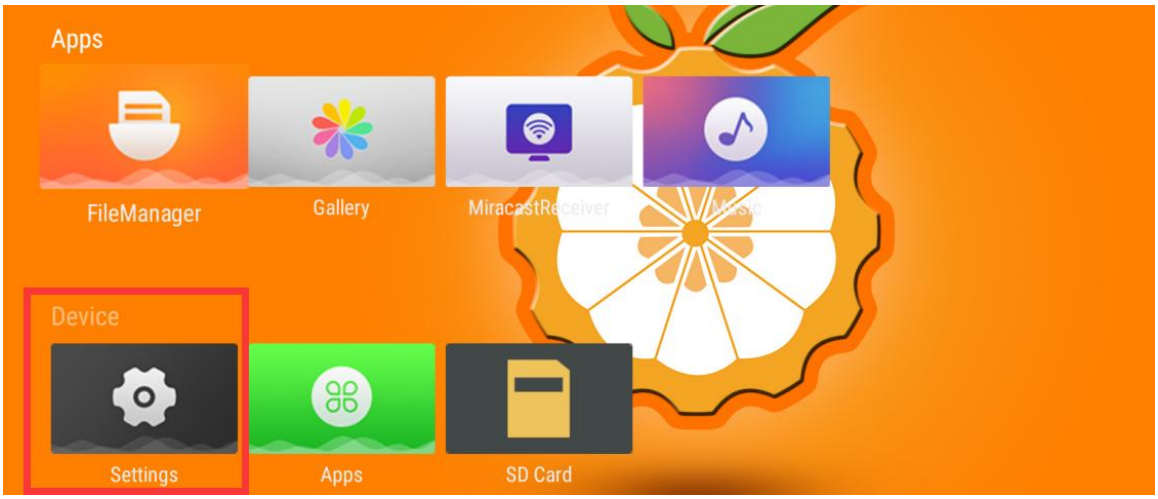
4) HDMI to VGA display test as shown below



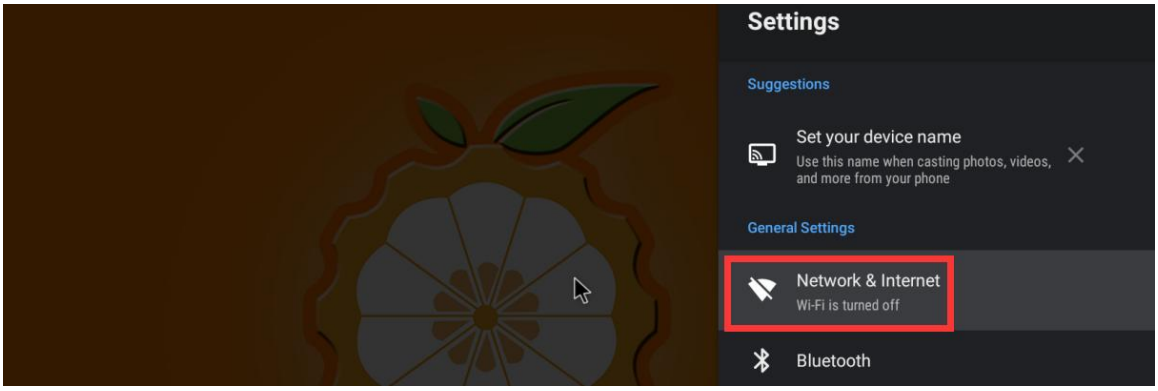
When using HDMI to VGA display, the development board and the Android system of the development board do not need to do any settings, as long as the Micro HDMI interface of the development board can display normally. So if there is a problem with the test, please check the HDMI to VGA converter, VGA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

4.9. Wi-Fi connection method

1) First select **Settings**



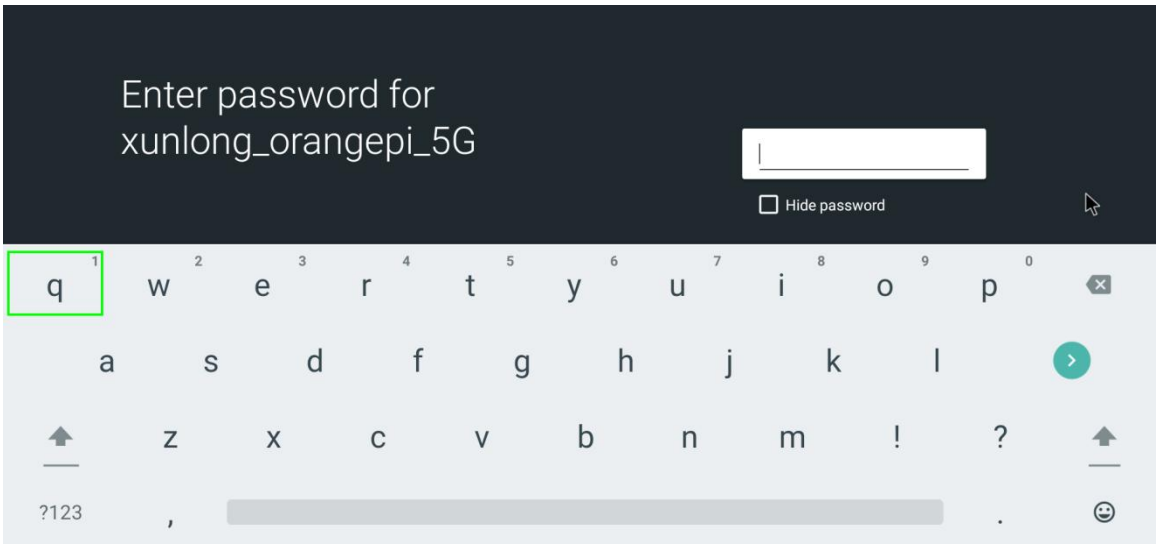
2) Then select **Network & Internet**



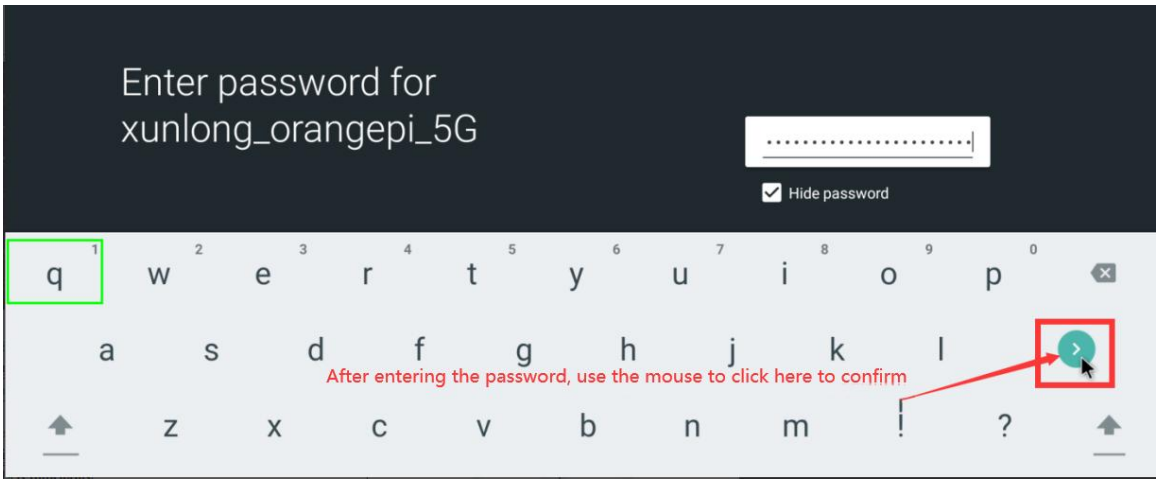
3) Then turn on WI-FI

4) After turning on WI-FI, you can see the searched signal under **Available networks**

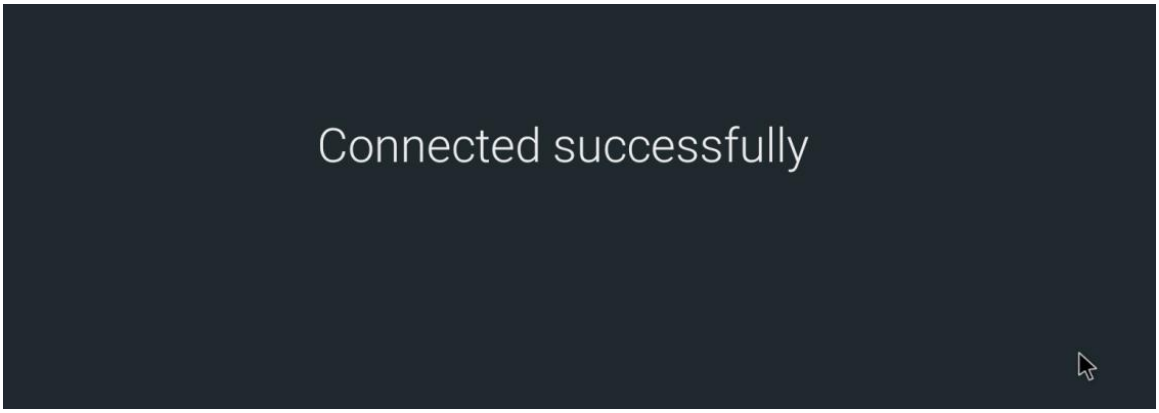
5) After selecting the WI-FI you want to connect to, the password input interface shown in the figure below will pop up



6) Then use the keyboard to enter the password corresponding to WI-FI, and then use the **mouse** to click the Enter button in the virtual keyboard to start connecting to WI-FI



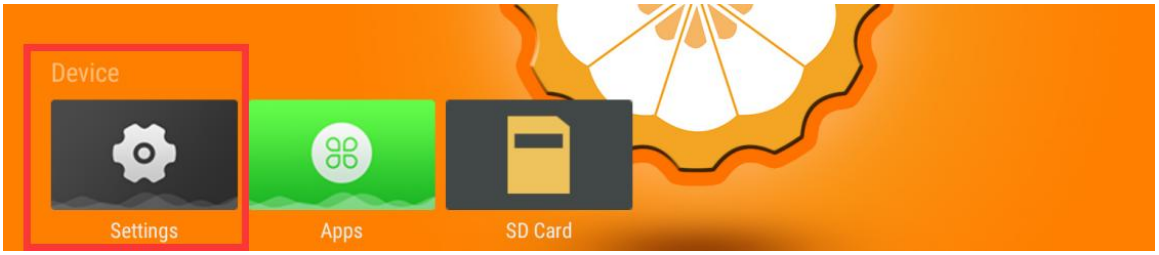
7) The display after the WI-FI connection is successful is shown in the figure below



4.10. The method to use WI-FI hotspot

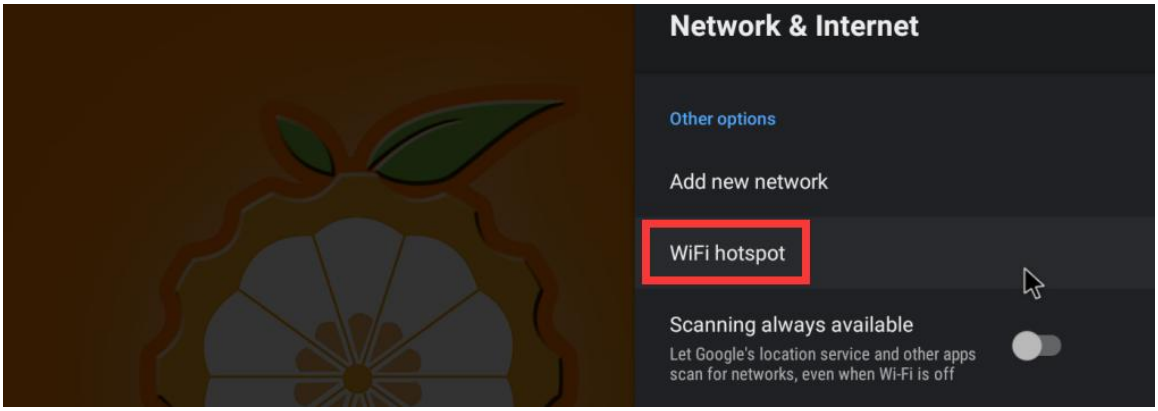
1) First, please make sure that the Ethernet port is connected to the network cable and can access the Internet normally

2) Then select **Settings**

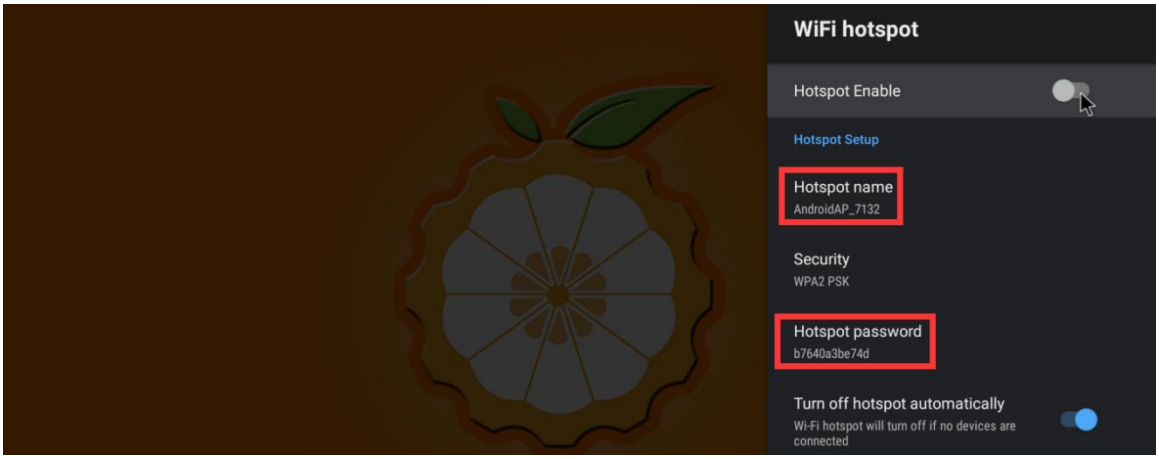


3) Then select **Network & Internet**

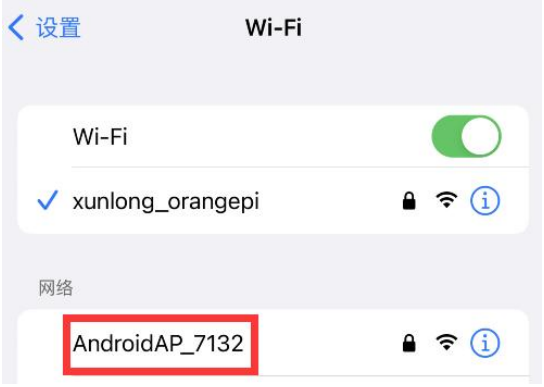
4) Then select **WIFI hotspot**



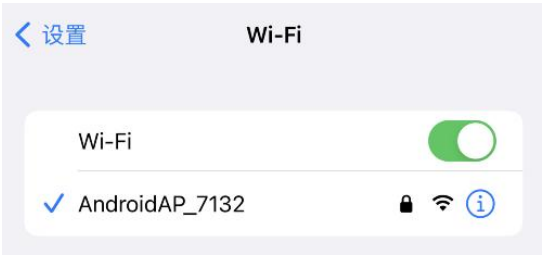
5) Then open **Hotspot Enable**, you can also see the name and password of the generated hotspot in the figure below, remember them and use them when connecting to the hotspot (**if you need to modify the name and password of the hotspot, you need to close Hotspot Enable first, then modify it**)



6) **Hotspot password** At this point, you can take out your mobile phone. If everything is normal, you can find the WIFI hotspot with the same name **(AndroidAP_7132 here)** displayed under the **Hotspot name** in the picture above in the WI-FI list searched by the mobile phone. Then you can click on **AndroidAP_7132** to connect to the hotspot. The password can be seen under the Hotspot password in the picture above.



7) After the connection is successful, it will display as shown in the figure below (different mobile phone interfaces will be different, the specific interface is subject to your mobile phone display). At this point, you can open a web page on your mobile phone to see if you can access the Internet. If the web page can be opened normally, it means that the **WI-FI Hotspot** of the development board can be used normally.



4.11. Bluetooth connection method

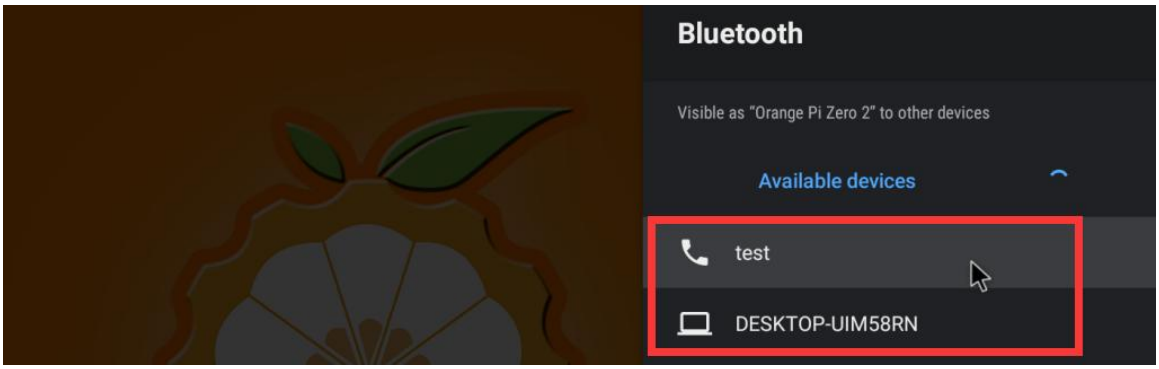
1) First select **Settings**

2) Then select **Bluetooth**

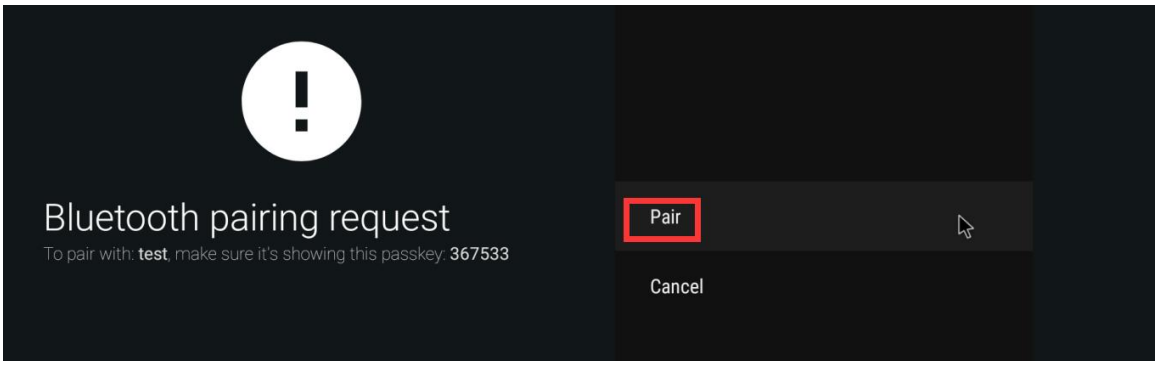
3) Then turn on **Bluetooth Enable**

4) Then click **Pair new device** to start scanning the surrounding bluetooth devices

5) The searched Bluetooth devices will be displayed under **Available devices**

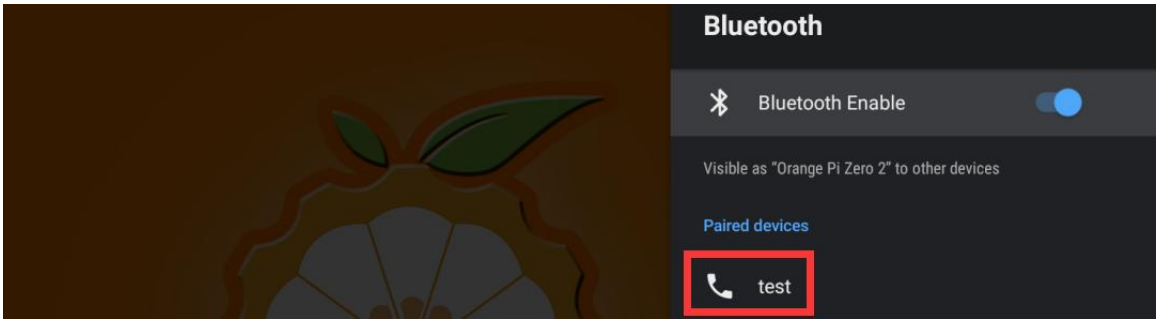


6) Then click on the Bluetooth device you want to connect to start pairing. When the following interface pops up, please use the mouse to select the **Pair** option



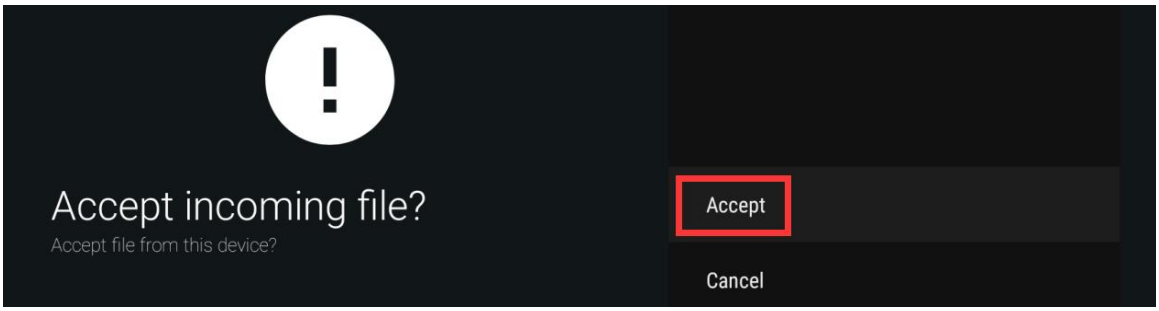
7) The configuration process of the development board and the **Android phone's** Bluetooth is tested here. At this time, the following confirmation interface will pop up on the mobile phone. After clicking the pairing button on the mobile phone, the pairing process will start.

8) After the pairing is completed, open the **paired devices** and you can see the paired Bluetooth devices

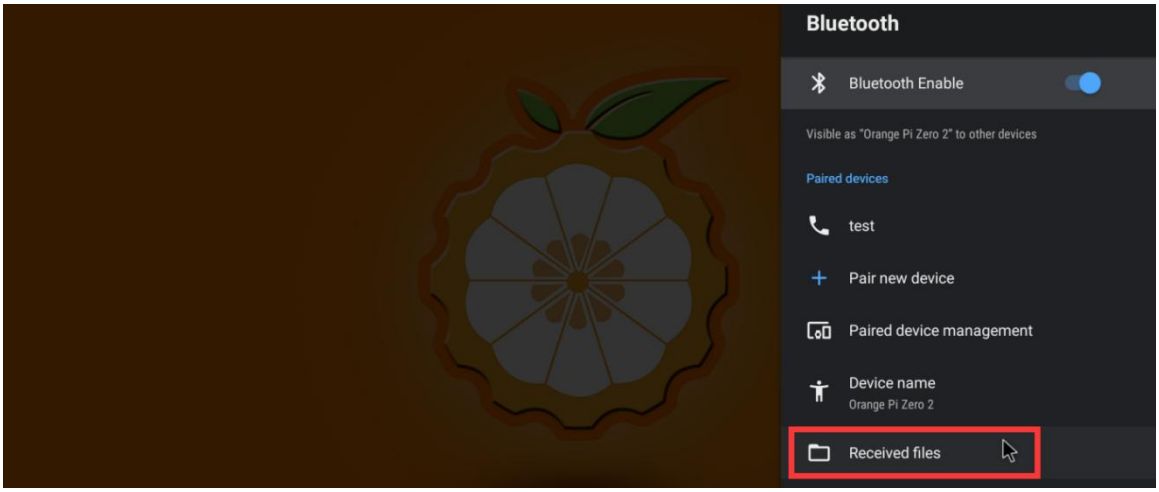


9) At this time, you can use the mobile phone Bluetooth to send a picture to the

development board. After sending, you can see the following confirmation interface in the Android system of the development board, and then click **Accept** to start receiving the picture sent by the mobile phone.

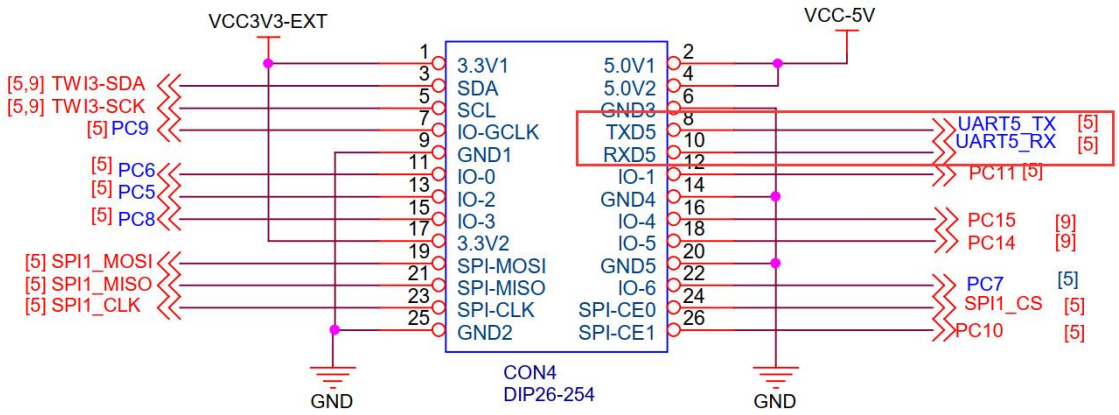


10) The pictures received by Bluetooth of the Android system of the development board can be viewed in **Received files**



4.12. Test method of serial port in 26pin interface

1) From the schematic diagram of the 26pin interface, the uart available for Orange Pi Zero 2 is uart5



2) After entering the Android system, please confirm whether there is a uart5 device node under **/dev**

d. The command viewed using adb is as follows

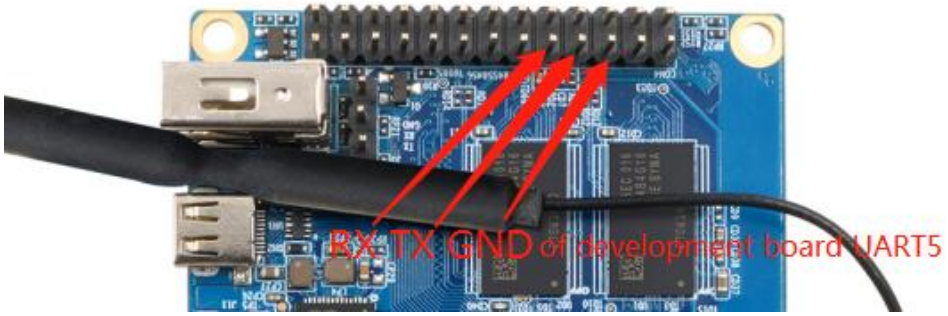
```
test@test:~$ adb connect 192.168.1.82
connected to 192.168.1.82:5555
test@test:~$ adb shell ls /dev/ttyS5
/dev/ttyS5
```

e. The command to view using the debug serial port is as follows

```
cupid-p2:/ # ls /dev/ttyS5
/dev/ttyS5
```

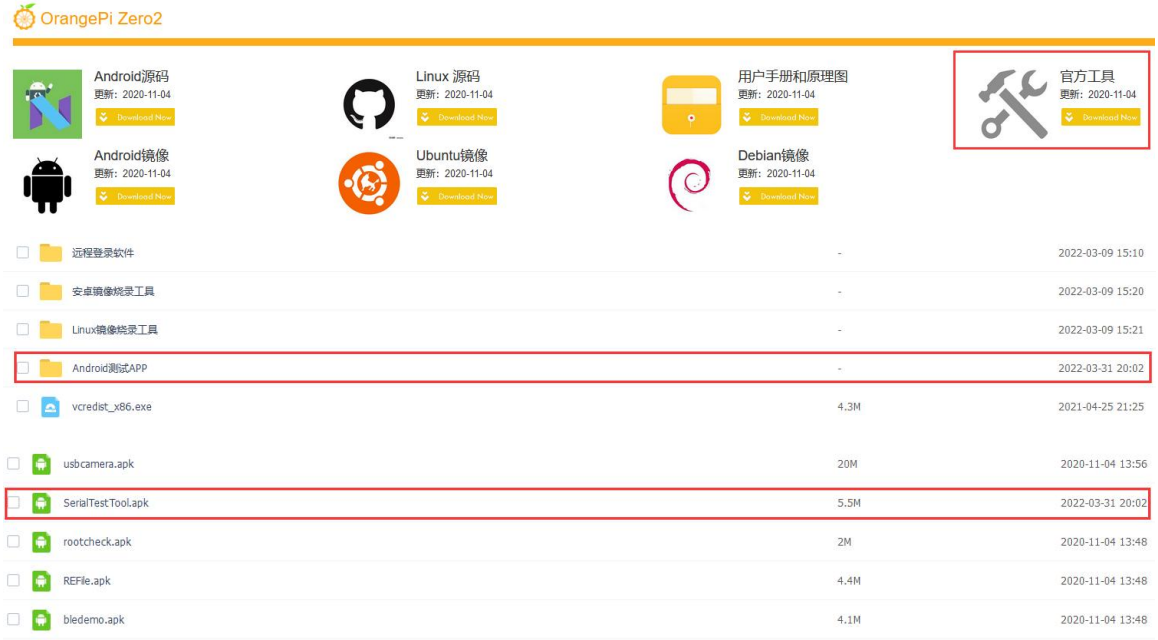
3) Then start to test the uart5 interface, first use a 2.54mm DuPont cable to short-circuit the rx and tx of the uart5 interface to be tested

	uart5
Tx Pin	Corresponds to pin 8
Rx Pin	Corresponds to pin 10



4) Then download the serial port debugging assistant **APP - SerialTestTool.apk**, this

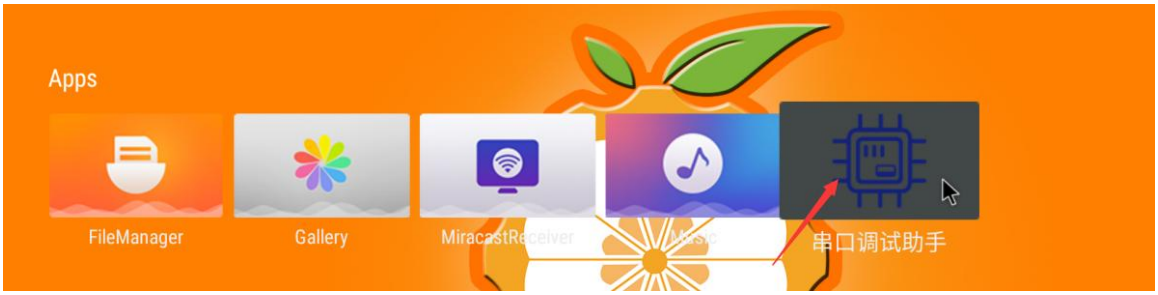
APP can be downloaded in the official tool of Orange Pi Zero 2



5) Then install **SerialTestTool.apk**. The command to install using adb is as follows. If you don't use adb, you can use U disk to copy and install

```
test@test:~$ adb install SerialTestTool.apk
```

6) Then open the serial port debugging assistant APP, the installed position of the serial port debugging assistant is shown in the figure below



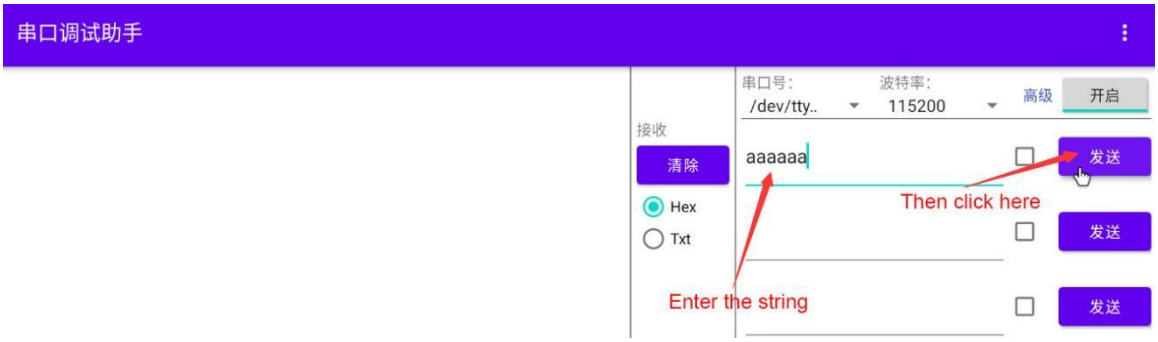
7) Then set the serial port debugging assistant

a. Select the serial port number **/dev/ttyS5**

b. b. Baud rate selection **115200**

c. Finally, remember to click the **close button** in the upper right corner to open the serial port

8) Then you can send a string to test the loopback function of the serial port to see if the serial port can receive the sent string



9) As shown in the figure below, if the serial port debugging assistant can receive the sent string, it means that the serial port can be used normally

4.13. The method to use the USB camera

1) Insert a USB camera into the USB interface of the development board, and then confirm that the kernel module related to the USB camera has been loaded normally

```
console:/ # lsmod
Module                  Size  Used by
```

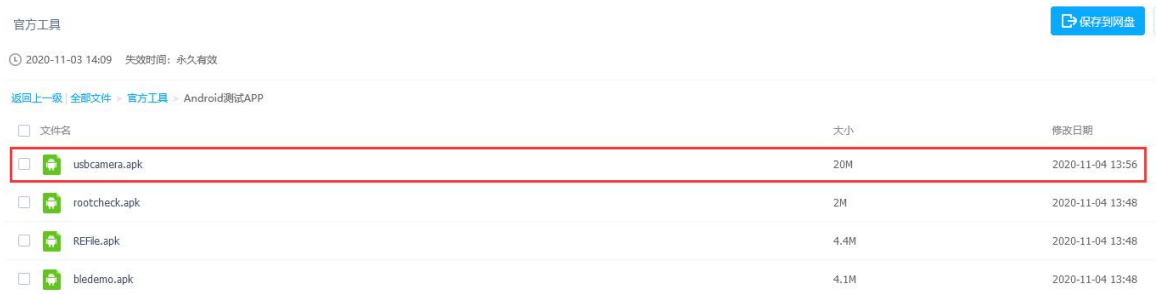
```
sprdwl_ng              405504  0
sprdbt_tty              36864  2
uwe5622_bsp_sdio       274432  2 sprdwl_ng,sprdbt_tty
uvcvideo               102400  0
videobuf2_v4l2          28672  1 uvcvideo
videobuf2_vmalloc       16384  1 uvcvideo
videobuf2_memops        16384  1 videobuf2_vmalloc
videobuf2_core          49152  2 uvcvideo,videobuf2_v4l2
mali_kbase             532480  7
```

2) If the USB camera is recognized normally, the corresponding video device node will be generated under **/dev**

```
console:/ # ls /dev/video0
/dev/video0
console:/ # ls -l /sys/class/video4linux/ -lh
total 0
lrwxrwxrwx 1 root root 0 2020-11-02 20:46:01.187678078 +0800 video0 -> ../../devices/platform/soc/5200000.ehci1-controller/usb1/1-1/1-1:1.0/video4linux/video0
console:/ #
```

3) Then make sure the adb connection between the Ubuntu PC and the development board is normal

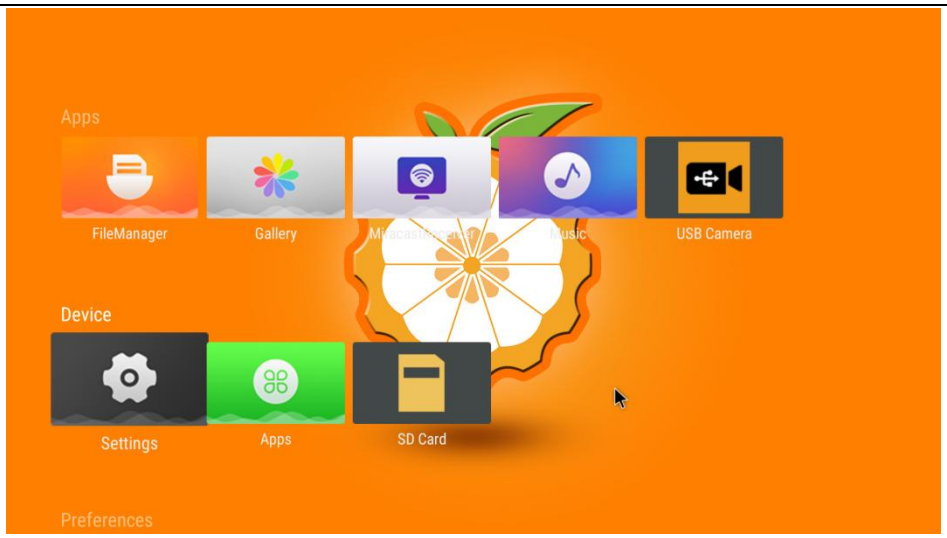
4) Download the USB camera test APP in the **official tool** on the Orange Pi Zero 2 data download page



5) Then use the adb command to install the USB camera test APP to the Android system, of course, you can also use the U disk copy method to install

```
test@test:~$ adb install usbcamera.apk
```

6) After installation, you can see the startup icon of the USB camera on the Android desktop

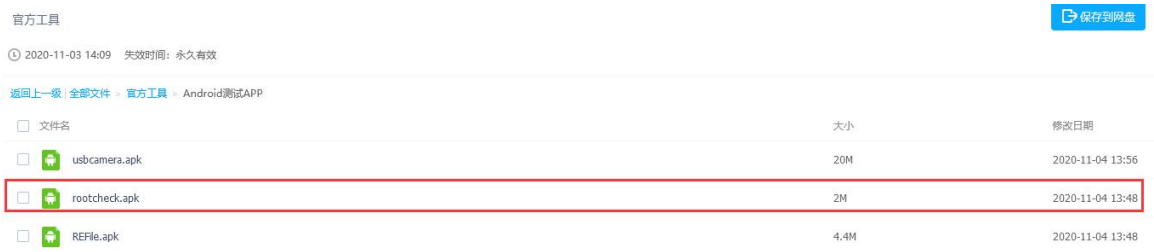


7) Then double-click to open the USB camera APP to see the output video of the USB camera

4.14. Android system ROOT description

The Android 10.0 system released by Orange Pi has been ROOT, you can use the following method to test

1) Download **rootcheck.apk** in the official tool on the Orange Pi Zero 2 data download page



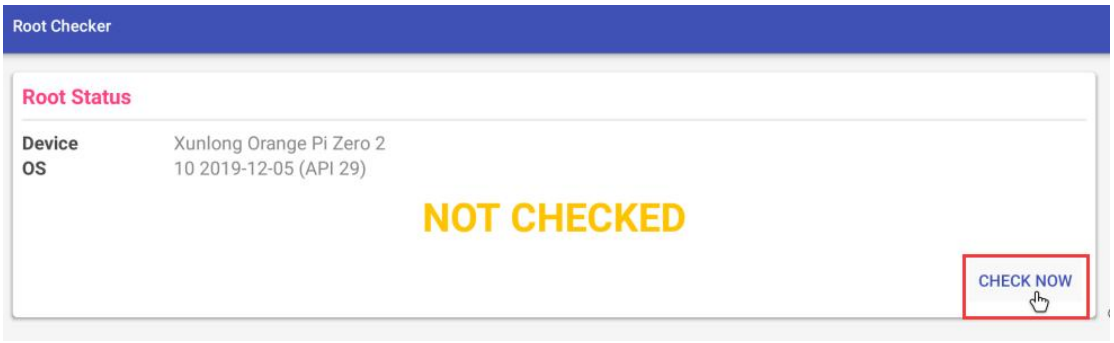
2) Then make sure the adb connection between the Ubuntu PC and the development board is normal

3) Then use the adb command to install rootcheck.apk into the Android system, of course, you can also use the U disk copy method to install

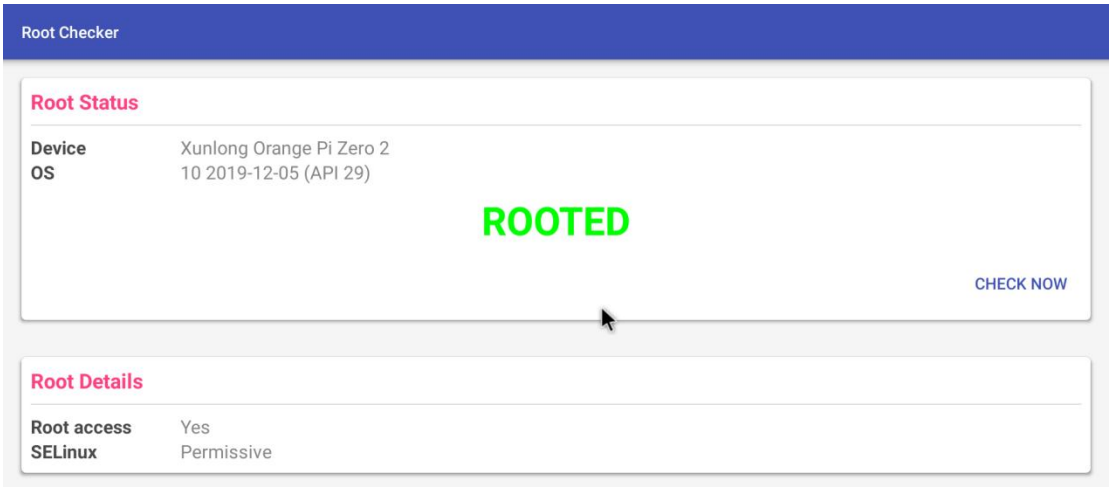
```
test@test:~$ adb install rootcheck.apk
```

4) After installation, you can see the startup icon of the ROOT test tool on the Android desktop

5) The display interface after opening the **ROOT test tool** for the first time is shown in the figure below



6) Then you can click "**Check Now**" to start the check of the ROOT status of the Android system. After the check, the display is as follows, and you can see that the Android system has obtained the ROOT permission.



4.15. Some Android APP installation instructions

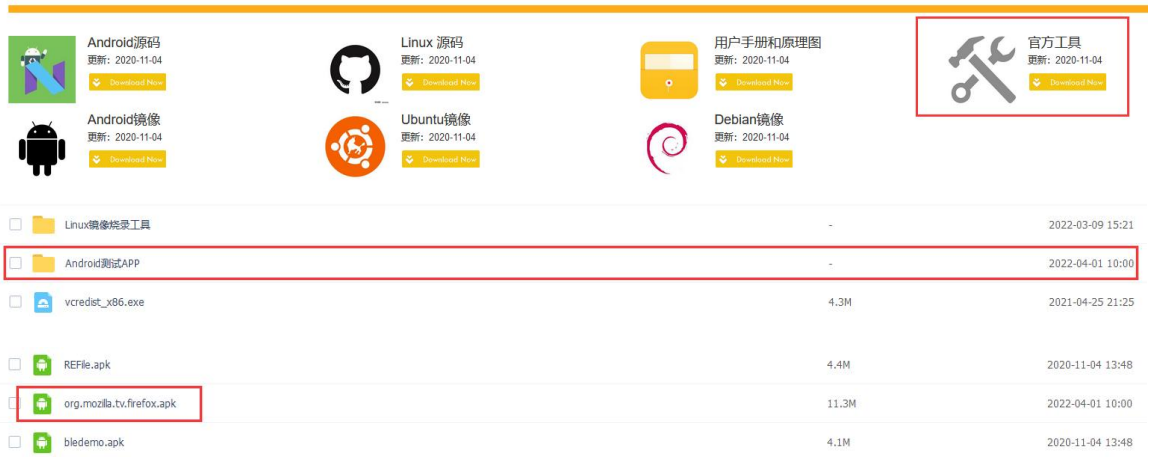
The Allwinner H616 chip is mainly used in TV boxes, so the Android provided by Allwinner is also the TV version of the Android system. If you need to install an APP, you can first install an application market such as Dangbei, which is specially created for smart TVs, and then install the required APP through Dangbei.

In addition, it should be noted that many mobile APPs used on mobile phones cannot be installed normally in the Android TV system. Although some can be installed normally, there are many problems in use. Before installing the software, it is best to check whether the manufacturer of the software provides an installation package for the TV version. For example, Tencent Video or iQiyi both provide an installation package for the TV version of the TV. Compared with the mobile APP, the TV version is much smaller in size, consumes much less resources, and is smoother to use.

4.15.1. Browser Installation Instructions

1) The first thing to note is that the Android system provided by Allwinner is the TV version of the Android system, so the APP installed on the mobile terminal of Firefox or Chrome will be very stuck. If you need to install a browser, please select the TV version of the installation package, but there are very few TV version browsers on the market. At present, the TV version of the browser previously developed by Firefox can be found better, although it has stopped updating (open A warning message will be displayed at the top after the APP), but there is no problem in using it and it is relatively smooth.

2) Firefox TV version APP can be downloaded from the **official tool**



3) The source code location of the Firefox TV version browser is as follows

https://github.com/mozilla-mobile/firefox-tv/releases

4.15.2. Tencent Video Installation Instructions

1) If you need to install Tencent Video to watch TV series and movies, please install the TV version of Tencent Video TV. The Android version of the mobile terminal cannot be used normally. The download address of the TV version of Tencent Video TV is as follows

http://v.qq.com/download.html

4.15.3. Youku Video Installation Instructions

1) If you need to install Youku to watch TV series and movies, please install the Kumiao TV version of the TV terminal provided by Youku. The Android version of the mobile terminal cannot be used normally. The download address of the Kumiao TV version is as follows

https://youku.com/product/index?spm=a2ha1.14919748_WEBHOME_GRAY.uerCenter.5!5~5~5~A

4.15.4. iQIYI Video Installation Instructions

2) If you need to install iQIYI to watch TV series and movies, please use iQIYI's Kiwi TV version, the download address is as follows

http://app.iqiyi.com/tv/player/

4.15.5. Lebo screencasting Installation Instructions

1) The download address is as follows

https://www.lebo.cn/Download.jsp

2) In the Android TV system of the development board, you need to install the TV version of LeBo.

3) Then install LeBo screencasting in the mobile phone, you can cast the screen of the mobile phone to the HDMI display connected to the development board

5. Linux SDK - Instructions for using the old version of orangepi-build

5.1. Compilation system requirements

1) The Linux SDK, namely **orangepi-build**, only supports running on a computer with **Ubuntu18.04** installed, so before downloading **orangepi-build**, please make sure that the Ubuntu version installed on your computer is Ubuntu18.04. The command to check the Ubuntu version installed on the computer is as follows. If the Release field does not display **18.04**, it means that the current Ubuntu version does not meet the requirements.

Please change the system before performing the following operations.

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.5 LTS
Release:        18.04
Codename:       bionic
test@test:~$
```

2) If the computer is installed with Windows system and there is no computer with Ubuntu18.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu18.04 virtual machine in the Windows system. But please note, do not compile orangepi-build on the WSL virtual machine, because orangepi-build has not been tested in the WSL virtual machine, so it cannot be guaranteed that orangepi-build can be used normally in WSL, and please do not use the Linux system of the **development board**. using orangepi-build in

3) The download address of the installation image of the Ubuntu18.04 **amd64** version is as follows

https://repo.huaweicloud.com/ubuntu-releases/18.04.6/ubuntu-18.04.6-desktop-amd64.iso

4) After installing Ubuntu18.04 in the computer or virtual machine, please set the software source of Ubuntu18.04 to Tsinghua source first, otherwise it is easy to make mistakes due to network reasons when installing the software later

a. For the method of replacing Tsinghua source, please refer to the description of this webpage.

https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/

b. Note that the Ubuntu version needs to be switched to 18.04

c. The content of the **/etc/apt/sources.list** file that needs to be replaced is

```
test@test:~$ sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak
test@test:~$ sudo vim /etc/apt/sources.list
# The source image is commented by default to improve the speed of apt update, you can uncomment it yourself if necessary
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse

# Pre-release software sources, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted
```

```
universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
```

d. After the replacement, you need to update the package information and ensure that no errors are reported

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

e. In addition, since the source code such as the kernel and U-boot are stored on GitHub, it is very important to ensure that the computer can download the code from GitHub normally when compiling the image.

5.2. Get the source code of linux sdk

5.2.1. Download orangepi-build from github

1) The linux sdk actually refers to the code of orangepi-build. Orangepi-build is modified based on the armbian build compilation system. Using orangepi-build, multiple versions of linux images can be compiled. First download the code of orangepi-build, currently H616 series development boards already support legacy branch and current branch

```
test@test:~$ sudo apt update
test@test:~$ sudo apt -y install git
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git
```

Downloading the code of orangepi-build through the git clone command does not require entering the username and password of the github account (the same is true for downloading other codes in this manual). If the Ubuntu PC prompts the user who needs to enter the github account after entering the git clone command The name and password are usually wrongly entered in the address of the orangepi-build repository behind git clone. Please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account here.

2) The legacy branch generally uses the BSP version of u-boot and linux kernel code provided by Allwinner, the current branch generally uses the u-boot and kernel code close to the Linux mainline version, and the u-boot and linux kernel currently used by H616 series development boards As follows

Branch	u-boot version	Linux kernel version
legacy	u-boot 2018.05	linux4.9

current	u-boot 2021.07	linux5.13

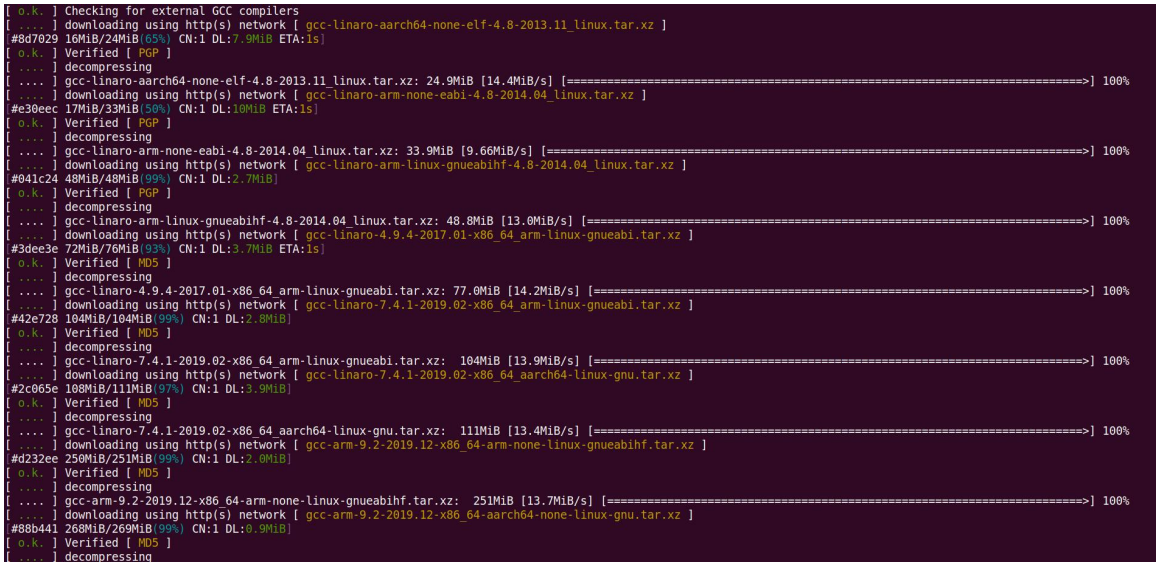
3) Orangepi-build will contain the following files and folders after downloading
   a. **build.sh**: Compile startup script
   b. **external**: Contains configuration files, specific scripts and source code of some programs needed to compile the image, etc.
   c. **LICENSE**: GPL 2 license file
   d. **README.md**: orangepi-build documentation
   e. **scripts**: Generic script for compiling linux images

```
test@test:~/orangepi-build$ ls
build.sh  external  LICENSE  README.md  scripts
```

If you download the code of orangepi-build from github, you may find that orangepi-build does not contain the source code of u-boot and linux kernel after downloading, and the cross-compilation toolchain is not required to compile u-boot and linux kernel. , which is normal, because these things are stored in other separate github repositories or some servers (the addresses will be detailed below). orangepi-build will specify the addresses of u-boot, linux kernel and cross-compilation toolchain in scripts and configuration files. When orangepi-build is running, when it finds that these things are not available locally, it will automatically download from the corresponding place.

5.2.2. Download the cross-compilation toolchain

1) When the **build.sh** script in orangepi-build is run for the first time, the cross-compilation **toolchain** will be automatically downloaded and stored in the **toolchains** folder under orangepi-build. After that, the build in orangepi-build will be run every time. sh script, it will also check whether the cross-compilation **toolchain** in **toolchains** exists. If it does not exist or there is an update, the download will be restarted. If it exists, it will be used directly without repeated download.



2) The default image website of the cross-compilation toolchain in China is the open source software image site of Tsinghua University. If you need to download the compressed package of the cross-compilation toolchain separately, you can open the following website, and then select the desired version to download.

https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/

3) After the **toolchains** is downloaded, it will contain multiple versions of the cross-compilation toolchain

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

4) The cross-compilation toolchain used to compile the H616 Linux kernel source code is

a. linux4.9

gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu

b. linux5.13

gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu

5) The cross-compilation toolchain used to compile the H616 u-boot source code is

a. u-boot 2018.05

gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi

b. u-boot 2021.07

gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu

5.2.3. Orangepi-build complete directory structure description

1) After the orangepi-build repository is downloaded, it does not contain the source code of the linux kernel, u-boot and the cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository

a. The git repository where the Linux kernel source code is stored is as follows

a) linux4.9

https://github.com/orangepi-xunlong/linux-orangepi/tree/**orange-pi-4.9-sun50iw9**

b) linux5.13

https://github.com/orangepi-xunlong/linux-orangepi/tree/**orange-pi-5.13-sunxi64**

b. The git repository where the source code of u-boot is stored is as follows

a) u-boot 2018.05

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/**v2018.05-sun50iw9**

b) u-boot 2021.07

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/**v2021.07-sunxi**

If you are not familiar with orangepi-build and do not know the detailed process of compiling the linux kernel and u-boot, please do not download and use the above linux kernel and u-boot source code for compilation operation, because the compilation script and configuration file of orangepi-build Some adjustments and optimizations will be made to u-boot and linux. If you do not use orangepi-build to compile u-boot and linux, you may encounter problems of compilation failure or failure to start.

2) When you run the build.sh script in orangepi-build for the first time, the cross-compilation toolchain, u-boot and linux kernel source code will be automatically downloaded. After successfully compiling the linux image once, the files you can see in

orangepi-build and folder has

a. **build.sh**: Compile startup script
b. **external**: Contains configuration files, scripts for specific functions and source code of some programs needed to compile the image. The rootfs compressed package cached in the process of compiling the image is also stored in external
c. **kernel**: Store the source code of the linux kernel, the folder named **orange-pi-4.9-sun50iw9** stores the kernel source code of the legacy branch of the H616 series development board, and the folder named **orange-pi-5.13-sunxi64** stores the The kernel source code of the current branch of the H616 development board (if only the linux image of the legacy branch is compiled, then only the kernel source code of the legacy branch can be seen; if only the linux image of the current branch is compiled, then only the kernel of the current branch can be seen source code), please do not manually modify the name of the folder of the kernel source code. If modified, the compilation system will re-download the kernel source code when it is running.
d. **LICENSE**: GPL 2 license file
e. **README.md**: orangepi-build documentation
f. **output**:Store the compiled u-boot, linux and other deb packages, compilation logs, and compiled images and other files
g. **scripts**: Generic script for compiling linux images
h. **toolchains**: Store the cross-compilation toolchain
i. **u-boot**: Store the source code of u-boot, the folder named **v2018.05-sun50iw9** stores the u-boot source code of the legacy branch of the H616 series development board, and the folder named **v2021.07-sunxi** stores the H616 The u-boot source code of the current branch of the development board (if only the linux image of the legacy branch is compiled, then only the u-boot source code of the legacy branch can be seen; if only the linux image of the current branch is compiled, then only the u-boot source code of the legacy branch can be seen. The u-boot source code of the current branch), please do not manually modify the name of the folder of the u-boot source code. If it is modified, the u-boot source code will be downloaded again when the compilation system is running.
j. **userpatches**: Store the configuration files needed to compile the script

```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts
toolchains  u-boot  userpatches
```

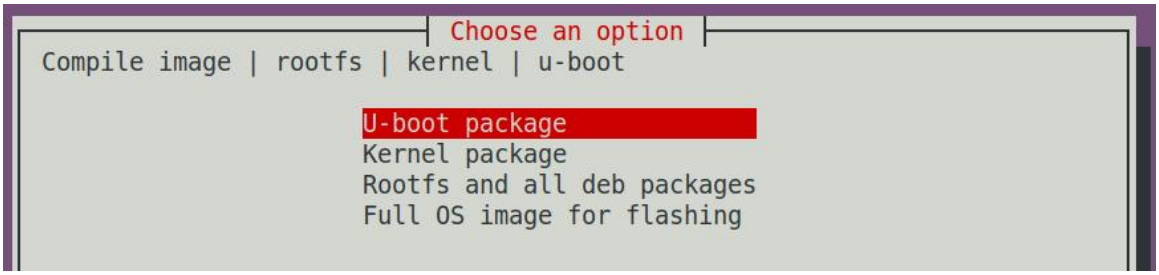
5.3. Compile u-boot

First of all, it should be noted that there is no source code of boot0 in the source code of u-boot 2018.05 of the legacy branch. This part of the source code is not open, and only the bin file of boot0 is provided. The u-boot 2021.07 of the current branch does not have this problem, and all the codes can be obtained

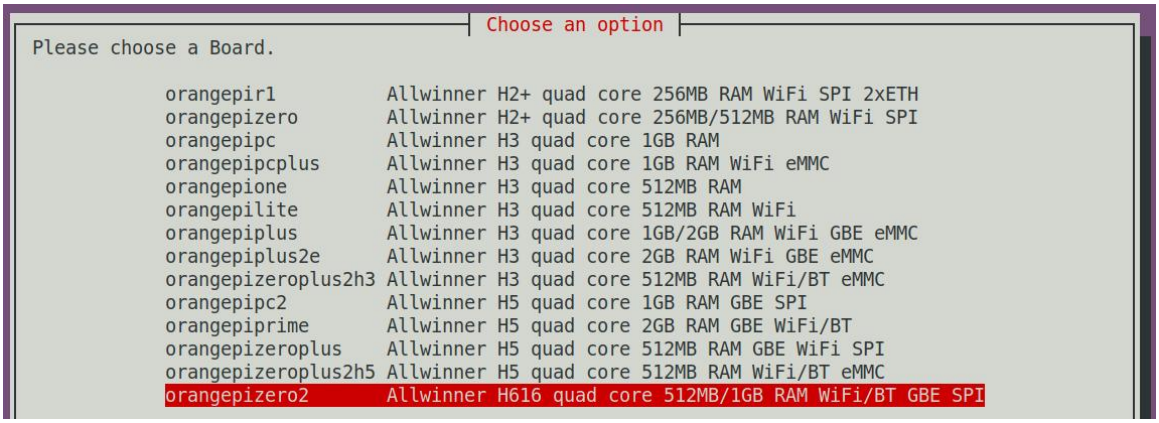
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **U-boot package** and press Enter



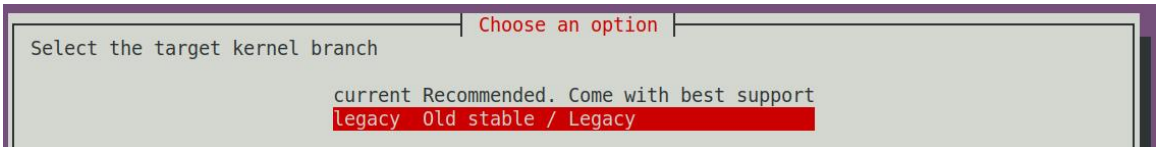
3) Then select the model of the development board



4) Then select branch

a. current will compile u-boot v2021.07

b. legacy will compile u-boot v2018.05



5) Then it will start compiling u-boot, some of the information prompted when compiling is explained as follows (take the legacy branch as an example)

a. u-boot source code version

```
[ o.k. ] Compiling u-boot [ v2018.05 ]
```

b. The version of the cross-compile toolchain

```
[ o.k. ] Compiler version [ arm-linux-gnueabi-gcc 7.4.1 ]
```

c. The path to the generated u-boot deb package

```
[ o.k. ] Target directory [ orangepi-build/output/debs/u-boot ]
```

d. The package name of the u-boot deb package generated by compilation

```
[ o.k. ] File name [ linux-u-boot-legacy-orangepizero2_2.2.0_arm64.deb ]
```

e. Compilation time used

```
[ o.k. ] Runtime [ 1 min ]
```

f. Repeat the command to compile u-boot, use the following command to start compiling u-boot directly without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2
BRANCH=legacy BUILD_OPT=u-boot KERNEL_CONFIGURE=no ]
```

6) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-legacy-orangepizero2_2.2.0_arm64.deb
```

7) The files contained in the generated u-boot deb package are as follows

a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \
linux-u-boot-legacy-orangepizero2_2.2.0_arm64.deb .     (Note that there is a "." at
                                                          the end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-legacy-orangepizero2_2.2.0_armhf.deb    usr
```

b. The decompressed file is as follows

```
test@test:~orangepi-build/output/debs/u-boot$ tree usr/
usr/
└── lib
    ├── linux-u-boot-legacy-orangepizero2_2.2.0_arm64
    │   ├── boot0_sdcard.fex       //binary for boot0
```

```
    │   └── boot_package.fex       //u-boot binaries
    └── u-boot
        ├── LICENSE
        ├── orangepi_zero2_defconfig          //The configuration file used to
                                              compile the u-boot source code
        ├── orangepizero2-u-boot.dts          //dts file used by u-boot source code
        └── platform_install.sh               //u-boot burning script

3 directories, 6 files
```

8) The orangepi-bulid compilation system will first synchronize the u-boot source code with the u-boot source code of the github server when compiling the u-boot source code, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code (**You need to compile u-boot once before closing this function, otherwise you will be prompted that the source code of u-boot cannot be found**), otherwise the modifications will be restored. The method is as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

9) When debugging the u-boot code, you can use the following method to update the u-boot in the linux image for testing

a. Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-legacy-orangepizero2_2.2.0_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepizero2-legacy
```

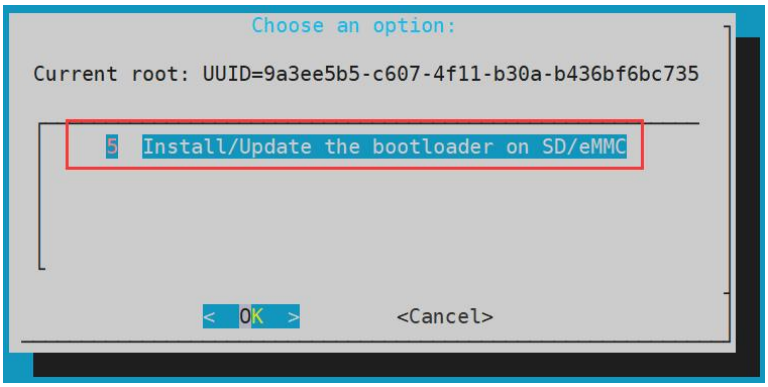
c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-legacy-orangepizero2_2.2.0_arm64.deb
```

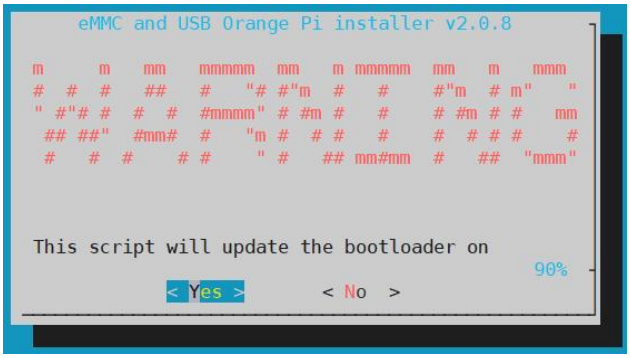
d. Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

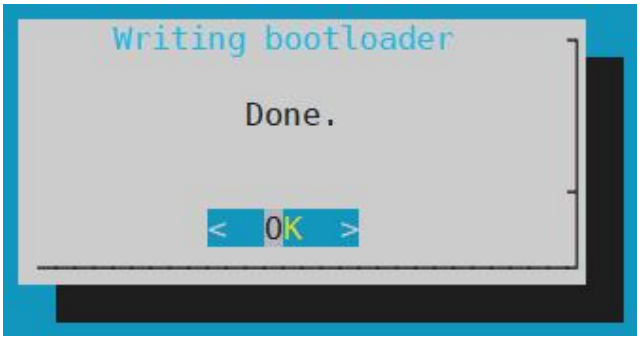
e. Then select **5 Install/Update the bootloader on SD/eMMC**



f. After pressing the Enter key, a Warring will pop up first



g. Press the Enter key again to start updating u-boot. After the update, the following information will be displayed



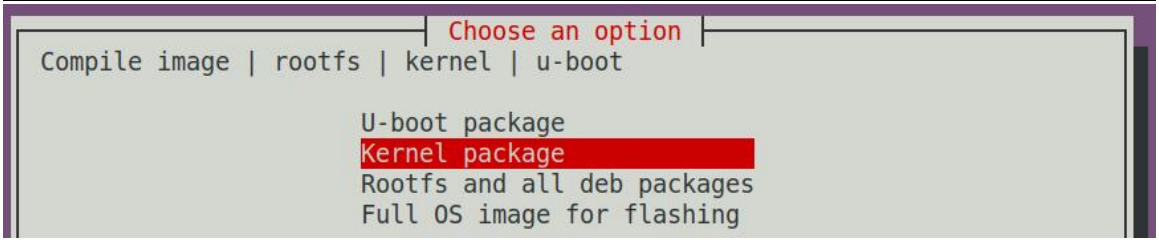
h. Then you can restart the development board to test whether the modification of u-boot takes effect

5.4. Compile the linux kernel

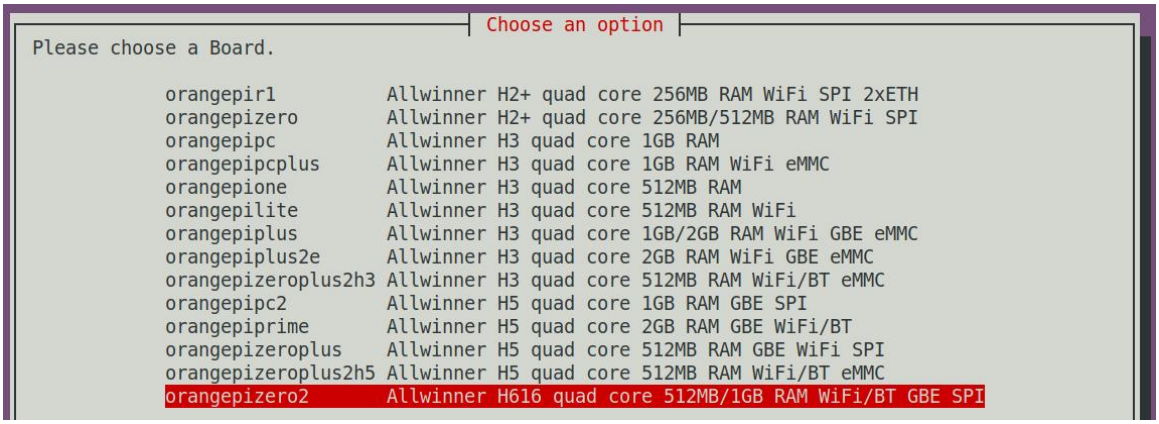
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Kernel package**, then press Enter



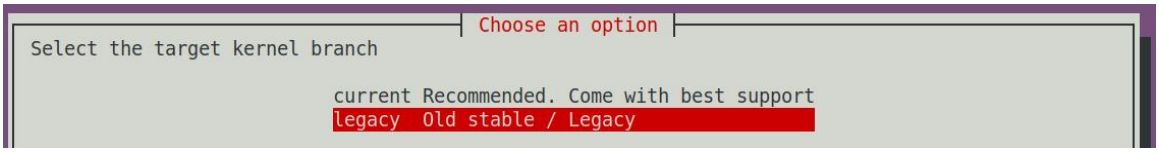
3) Then select the model of the development board



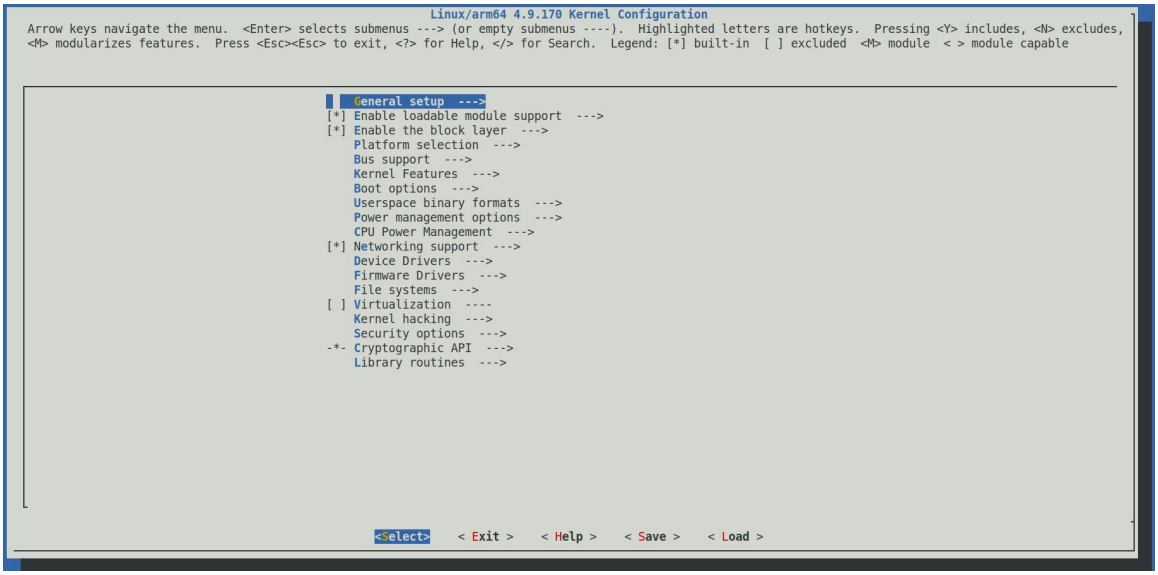
4) Then select branch

a. current will compile linux 5.13

b. legacy will compile linux4.9



5) Then the kernel configuration interface opened by **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration. If you do not need to modify the kernel configuration, you can simply exit. After exiting, the kernel source code will be compiled.



a. If you do not need to modify the configuration options of the kernel, when running the build.sh script, pass in **KERNEL_CONFIGURE=no** to temporarily shield the configuration interface of the pop-up kernel

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

b. You can also set **KERNEL_CONFIGURE=no** in the orangepi-build/userpatches/config-default.conf configuration file to permanently disable this feature

c. If the following error is displayed when compiling the kernel, this is because the terminal interface of the Ubuntu PC is too small, so the interface of make menuconfig cannot be displayed. Please adjust the terminal of the Ubuntu PC to the maximum, and then re-run the build.sh script

```
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:560: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
[ error ] ERROR in function compile_kernel [ compilation.sh:376 ]
[ error ] Error kernel menuconfig failed
[ o.k. ] Process terminated
```

6) Part of the information prompted when compiling the kernel source code is explained as follows

a. The version of the linux kernel source code

```
[ o.k. ] Compiling legacy kernel [ 4.9.170 ]
```

b. The version of the cross-compilation toolchain used

```
[ o.k. ] Compiler version [ aarch64-none-linux-gnu-gcc 9.2.1 ]
```

c. The configuration file used by the kernel by default and the path where it is stored

```
[ o.k. ] Using kernel config file [ config/kernel/linux-sun50iw9-legacy.config ]
```

d. If **KERNEL_CONFIGURE=yes**, the final configuration file .config used by the kernel will be copied to **output/config**. If the kernel configuration is not modified, the final configuration file is the same as the default configuration file

```
[ o.k. ] Exporting new kernel config [ output/config/linux-sun50iw9-legacy.config ]
```

e. The path to the generated kernel-related deb package

```
[ o.k. ] Target directory [ output/debs/ ]
```

f. The package name of the kernel image deb package generated by compilation

```
[ o.k. ] File name [ linux-image-legacy-sun50iw9_2.2.0_arm64.deb ]
```

g. Compilation time used

```
[ o.k. ] Runtime [ 5 min ]
```

h. Finally, the compilation command to repeat the compilation of the last selected kernel will be displayed. Use the following command to directly start compiling the kernel source code without selecting through the graphical interface.

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2
BRANCH=legacy BUILD_OPT=kernel KERNEL_CONFIGURE=yes ]
```

7) View the compiled and generated kernel-related deb packages

a. **linux-dtb-legacy-sun50iw9_2.2.0_arm64.deb** Not used yet, don't worry about it

b. **linux-headers-legacy-sun50iw9_2.2.0_arm64.deb** Include kernel header files

c. **linux-image-legacy-sun50iw9_2.2.0_arm64.deb** Contains kernel images and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-legacy-sun50iw9_2.2.0_arm64.deb
output/debs/linux-headers-legacy-sun50iw9_2.2.0_arm64.deb
output/debs/linux-image-legacy-sun50iw9_2.2.0_arm64.deb
```

8) The files contained in the generated linux-image deb package are as follows

a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-legacy-sun50iw9_2.2.0_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-legacy-sun50iw9_2.2.0_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-legacy-sun50iw9_2.2.0_arm64.deb  usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi_build/output/debs/test$ tree -L 2
.
├── boot
│   ├── config-4.9.170-sun50iw9                 //The configuration file used to compile
                                                  the kernel source code
│   ├── System.map-4.9.170-sun50iw9
│   └── vmlinuz-4.9.170-sun50iw9          //The configuration file used to compile the
                                          kernel source code
├── etc
│   └── kernel
├── lib
│   └── modules                                 //Compile the generated kernel module
├── linux-image-legacy-sun50iw9_2.2.0_arm64.deb
└── usr
    ├── lib
    └── share

8 directories, 4 files
```

9) When the orangepi-bulid compilation system compiles the linux kernel source code, it first synchronizes the linux kernel source code with the linux kernel source code of the github server, so if you want to modify the linux kernel source code, you first need to turn off the update function of the source code (**you need to compile the linux kernel once This function can only be turned off after the source code, otherwise it will prompt that the source code of the linux kernel cannot be found**), otherwise the modification

will be restored, the method is as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

10) If the kernel is modified, you can use the following methods to update the kernel and kernel modules of the development board linux system

a. Upload the compiled deb package of the linux kernel to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-legacy-sun50iw9_2.2.0_arm64.deb root@192.168.1.207:/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-sun50iw9
```

c. Install the deb package of the new linux kernel just uploaded

```
root@orangepi:~# dpkg -i linux-image-legacy-sun50iw9_2.2.0_arm64.deb
```

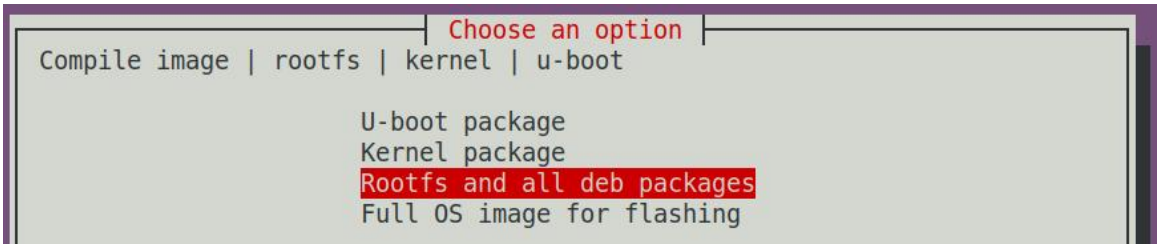
d. Then restart the development board, and then check whether the kernel-related modifications have taken effect

5.5. Compile rootfs

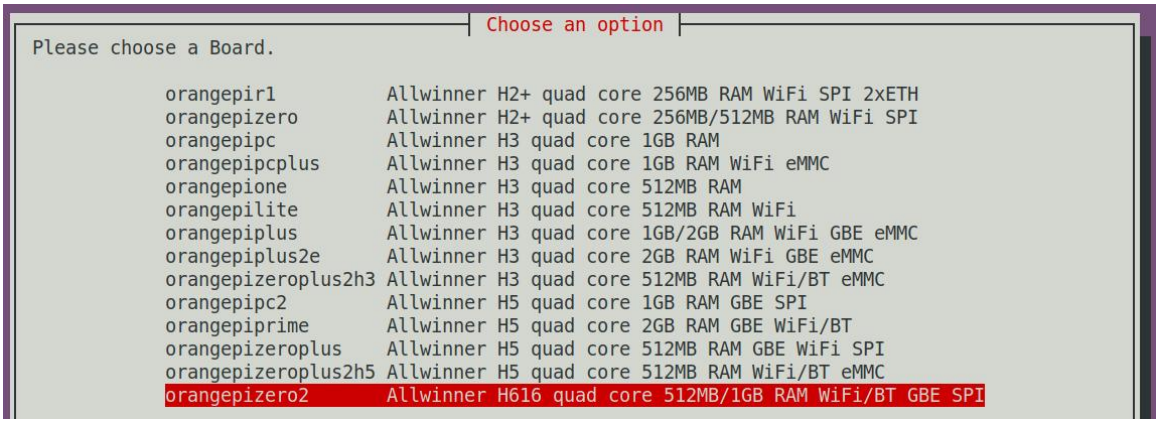
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

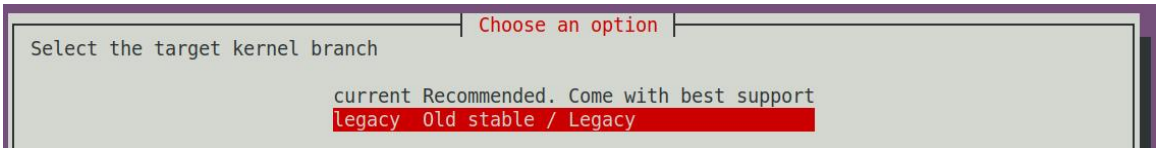
2) Select **Rootfs and all deb packages**, then press Enter



3) Then select the model of the development board



4) Then select branch type

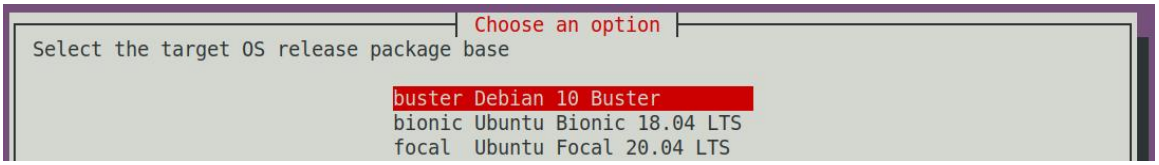


5) Then select the type of rootfs

a. **buster** means Debian 10

b. **bionic** means Ubuntu 18.04

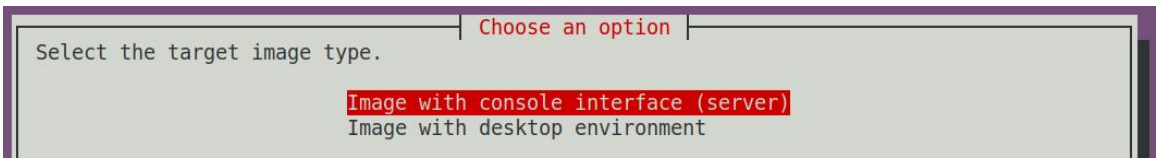
c. **focal** means Ubuntu 20.04



6) Then select the type of image

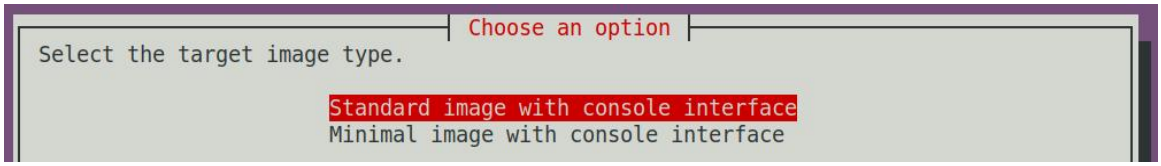
a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size

b. **Image with desktop environment** Indicates a image with a desktop, which is relatively large in size



7) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal

version will be much less than the Standard version.



8) After selecting the type of image, the rootfs will be compiled, and some of the information prompted during compilation are as follows

a. type of rootfs

[o.k.] local not found [Creating new rootfs cache for **bionic**]

b. The storage path of the rootfs compressed package generated by compilation

[o.k.] Target directory [**external/cache/rootfs**]

c. The name of the rootfs compressed package generated by compilation

[o.k.] File name [**bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4**]

d. Compilation time

[o.k.] Runtime [**13 min**]

e. Repeat the command to compile rootfs, use the following command to start compiling rootfs directly without selecting through the graphical interface

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepizero2 BRANCH=legacy BUILD_OPT=rootfs RELEASE=bionic BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes**]

9) View the rootfs compressed package generated by compilation

a. bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4is the compressed package of rootfs, and the meaning of each field of the name is

a) **bionic** indicates the type of linux distribution for rootfs

b) **cli** indicates that rootfs is the type of the server version, and if it is dekstop, it indicates the type of the desktop version

c) **arm64** represents the architecture type of rootfs

d) **153618961f14c28107ca023429aa0eb9** is the MD5 hash value generated by the package names of all software packages installed by rootfs. As long as the list of packages installed by rootfs is not modified, this value will not change, and the compilation script will judge by this MD5 hash value. Do I need to recompile rootfs

b. bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list lists the

package names of all film installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4
bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list
```

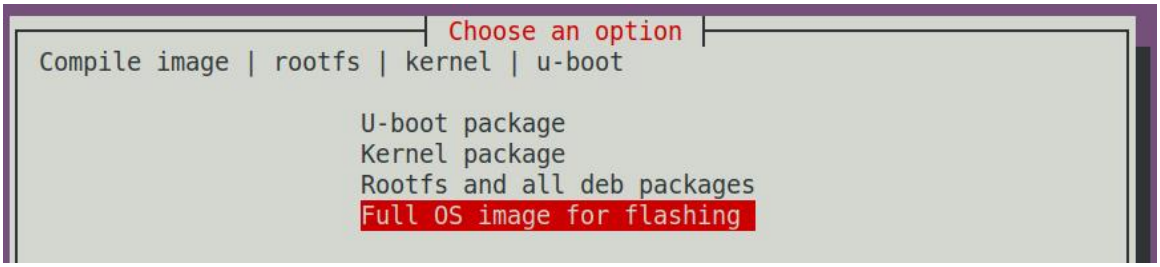
10) If the required rootfs already exists under **external/cache/rootfs**, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it has There is a rootfs available for cache, if there is one, use it directly, which can save a lot of download and compilation time

5.6. Compile the linux image

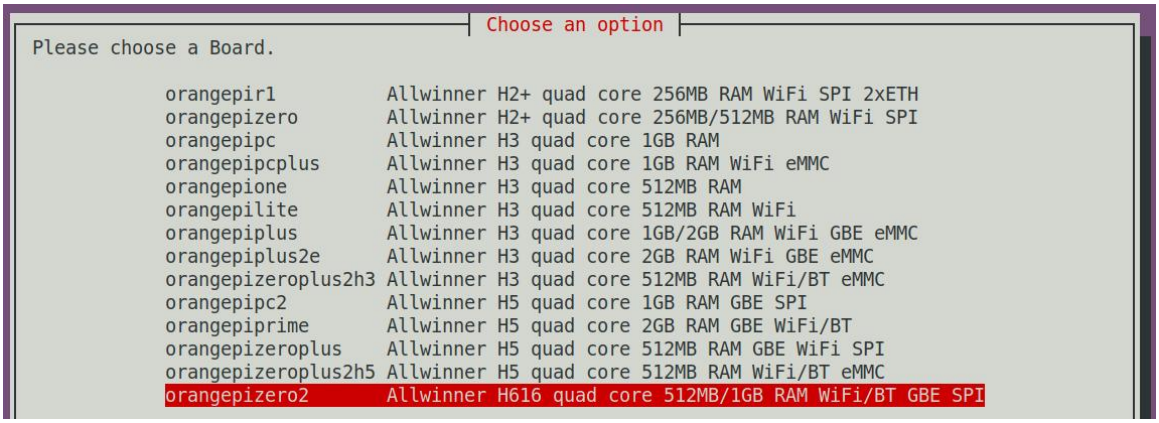
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Full OS image for flashing** and press Enter



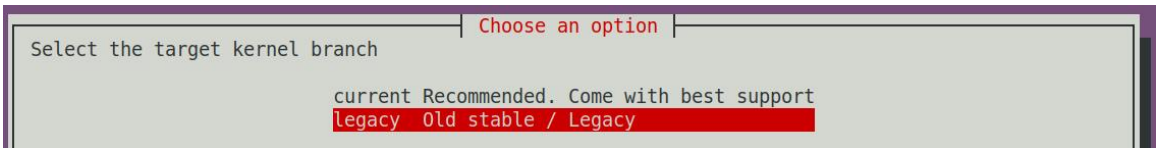
3) Then select the model of the development board



4) Then select branch

a. current will compile linux 5.13

b. legacy will compile linux4.9

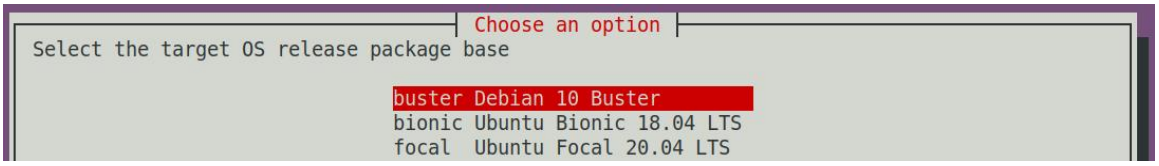


5) Then select the type of rootfs

a. **Buster** means Debian 10

b. **bionic** means Ubuntu 18.04

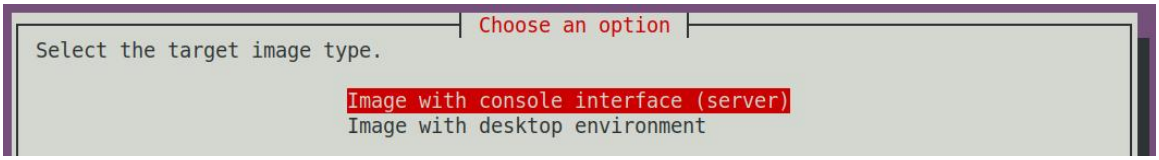
c. **focal** means Ubuntu 20.04



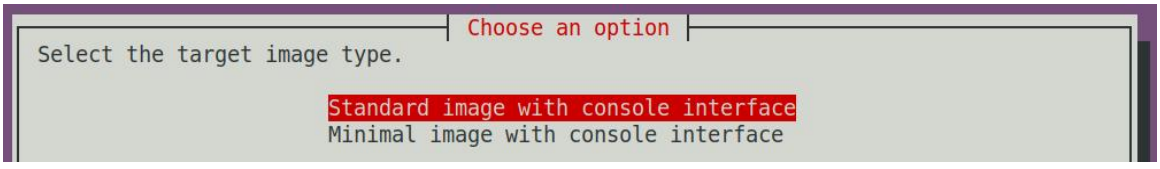
6) Then select the type of image

a. **Image with console interface (server)** Indicates the image of the server version, which is relatively small in size

b. **Image with desktop environment** Indicates a image with a desktop, which is relatively large in size



7) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.



8) After selecting the type of image, the linux image will be compiled. The general process of compilation is as follows

a. Initialize the compilation environment of the Ubuntu PC and install the software packages required for the compilation process

b. Download the source code of u-boot and linux kernel (if cached, only update the code)

c. Compile u-boot source code and generate u-boot deb package

d. Compile the linux source code to generate linux-related deb packages

e. Make a deb package of linux firmware

f. Make the deb package of the orangepi-config tool

g. Make board-level supported deb packages

h. If you are compiling the desktop version of the image, you will also create a desktop-related deb package

i. Check whether the rootfs has been cached, if there is no cache, then recreate the rootfs, if it has been cached, directly decompress and use

j. Install the deb package generated earlier into rootfs

k. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configuration, etc.

l. Then make an image file and format the partition, the default type is ext4

m. Copy the configured rootfs to the mirrored partition

n. then update initramfs

o. Finally, write the bin file of u-boot into the image through the dd command

9) After compiling the image, the following information will be prompted

a. The storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/orangepizero2_2.2.0_ubuntu_bionic_server_linux4.9.170/orangepizero2_2.2.0_ubuntu_bionic_server_linux4.9.170.img ]
```

b. Compilation time used

```
[ o.k. ] Runtime [ 19 min ]
```

c. Repeat the command to compile the image, use the following command to start compiling the image directly without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2 BRANCH=legacy BUILD_OPT=image RELEASE=bionic BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

6. Linux SDK - Instructions for using the new version of orangepi-build

1. What is the difference between the compilation host of the new version and the old version?

The old version of orangepi-build was run on an x64 computer or virtual machine with Ubuntu 18.04.

The new version of orangepi-build is run on an x64 computer or virtual machine of Ubuntu21.04.

If you want to use the new version of orangepi-build described in this chapter to compile Linux images, you first need to prepare a computer or virtual machine with Ubuntu 21.04 installed.

2. What is the difference between the source code storage address of the new version and the old version?

The old version of the orangepi-build source code is stored in the main branch of the orangepi-build repository:
https://github.com/orangepi-xunlong/orangepi-build/tree/main

The source code of the new version of orangepi-build is stored in the next branch of the orangepi-build repository:
https://github.com/orangepi-xunlong/orangepi-build/tree/next

3. What is the difference between the new version and the old version supporting the kernel?

The old version of orangepi-build mainly supports Linux4.9 and Linux5.13 (5.13 is no longer updated).

The new version of orangepi-build currently supports Linux 5.16.

4. What is the difference between the types of Linux distributions supported by the new version and the old version?

Older versions of orangepi-build mainly support Debian10、 Ubuntu18.04、Ubuntu20.04。

The new version of orangepi-build mainly supports Debian11/12、Ubuntu20.04、Ubuntu22.04。

Ubuntu 22.04 will be released after the test is stable.

5. What is the difference between the image names generated by the new version and the old version?

The version of the image compiled by the old version of orangepi-build is named 2.xx.

The image version generated by the new version of orangepi-build is 3.xx. Desktop images also add fields for desktop types, such as xfce.

6.1. Compilation system requirements

1) The Linux SDK, orangepi-build, only supports running on computers with **Ubuntu 21.04** installed, so before downloading orangepi-build, please make sure that the Ubuntu version installed on your computer is Ubuntu 21.04. The command to check the Ubuntu version installed on the computer is as follows. If the Release field shows other than **21.04**, it means that the current Ubuntu version does not meet the requirements. Please change the system before performing the following operations.

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 21.04
Release:        21.04
Codename:       hirsute
test@test:~$
```

2) If the computer is installed with Windows system and there is no computer with Ubuntu 21.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu 21.04 virtual machine in the Windows system. But please note, do not compile orangepi-build on the WSL virtual machine, because orangepi-build has not been tested

in the WSL virtual machine, so it cannot be guaranteed that orangepi-build can be used normally in WSL, and please do not use the Linux system of the development board. using orangepi-build in

3) The installation image download address of Ubuntu 21.04 **amd64** version is:

https://repo.huaweicloud.com/ubuntu-releases/21.04/ubuntu-21.04-desktop-amd64.iso

4) After installing Ubuntu 21.04 in the computer or virtual machine, please set the software source of Ubuntu 21.04 to Tsinghua source, otherwise it is easy to make mistakes due to network reasons when installing the software later

a. For the method of replacing Tsinghua source, please refer to the description of this webpage.

https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/

b. Note that the Ubuntu version needs to be switched to 21.04

c. The content of the **/etc/apt/sources.list** file that needs to be replaced is

test@test:~$ **sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak**
test@test:~$ **sudo vim /etc/apt/sources.list**
The source image is commented by default to improve the speed of apt update, you can uncomment it yourself if necessary
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe multiverse
deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute main restricted universe multiverse

```
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-security main restricted universe multiverse

# Pre-release software sources, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ hirsute-proposed main restricted universe multiverse
```

d. After the replacement, you need to update the package information and ensure that no errors are reported

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

e. In addition, since the source code such as the kernel and U-boot are stored on GitHub, it is very important to ensure that the computer can download the code from GitHub normally when compiling the image.

6.2. Get the source code of linux sdk

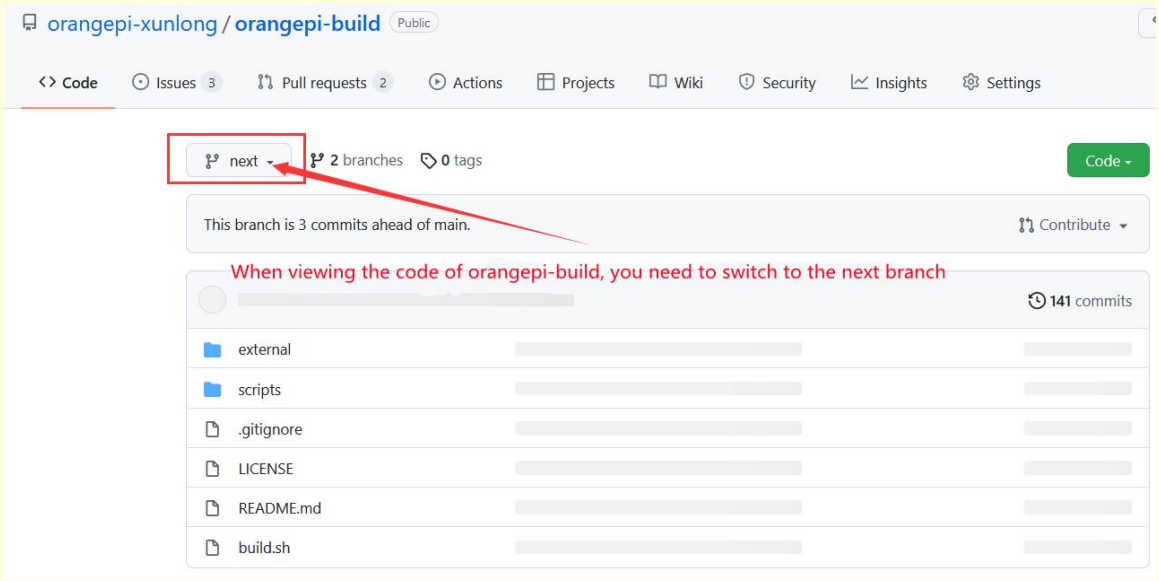
6.2.1. Download orangepi-build from github

1) The linux sdk actually refers to the code of orangepi-build. Orangepi-build is modified based on the armbian build compilation system. Using orangepi-build, multiple versions of linux images can be compiled. First download the code of orangepi-build. Currently, the H616 series development boards in the Linux SDK already support the legacy branch and the current branch.

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

```
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git -b next
```

Note that the instructions in this section are for the source code of the next branch of orangepi-build. The above git clone command needs to specify the branch of the source code of orangepi-build as next.



Downloading the code of orangepi-build through the git clone command does not require entering the username and password of the github account (the same is true for downloading other codes in this manual). If the Ubuntu PC prompts the user who needs to enter the github account after entering the git clone command The name and password are usually wrongly entered in the address of the orangepi-build repository behind git clone. Please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account here.

2) The u-boot and linux kernel versions currently used by the H616 series development boards are as follows

branch	u-boot version	linux kernel version
current	u-boot 2021.10	linux5.16

3) Orangepi-build will contain the following files and folders after downloading

f. **build.sh**: Compile startup script
g. **external**: Contains configuration files, specific scripts and source code of some programs needed to compile the image, etc.
h. **LICENSE**: GPL 2 license file
i. **README.md**: orangepi-build documentation
j. **scripts**: Generic script for compiling linux images

```
test@test:~/orangepi-build$ ls
build.sh  external  LICENSE  README.md  scripts
```

If the code of orangepi-build is downloaded from github, after downloading, you may find that orangepi-build does not contain the source code of u-boot and linux kernel, nor does it require cross-compilation tools to compile u-boot and linux kernel chain, this is normal, because these things are stored in other separate github repositories or some servers (the addresses will be detailed below). orangepi-build will specify the addresses of u-boot, linux kernel and cross-compilation toolchain in scripts and configuration files. When orangepi-build is running, when it finds that these things are not available locally, it will automatically download from the corresponding place.

6.2.2. Download the cross-compilation toolchain

1) When orangepi-build runs for the first time, it will automatically download the cross-compilation toolchain and put it in the toolchains folder. After running the build.sh script of orangepi-build, it will check whether the cross-compilation toolchain in toolchains exists. , if it does not exist, it will restart the download, if it exists, it will be used directly, and the download will not be repeated

```
[ o.k. ] Checking for external GCC compilers
[ .... ] downloading using http(s) network [ gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz ]
#8d7029 16MiB/24MiB(65%) CN:1 DL:7.9MiB ETA:1s]
[ o.k. ] Verified [ PGP ]
[ .... ] decompressing
[ .... ] gcc-linaro-aarch64-none-elf-4.8-2013.11_linux.tar.xz: 24.9MiB [14.4MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-linaro-arm-none-eabi-4.8-2014.04_linux.tar.xz ]
#e30eec 17MiB/33MiB(50%) CN:1 DL:10MiB ETA:1s]
[ o.k. ] Verified [ PGP ]
[ .... ] decompressing
[ .... ] gcc-linaro-arm-none-eabi-4.8-2014.04_linux.tar.xz: 33.9MiB [9.66MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz ]
#041c24 48MiB/48MiB(99%) CN:1 DL:2.7MiB]
[ o.k. ] Verified [ PGP ]
[ .... ] decompressing
[ .... ] gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux.tar.xz: 48.8MiB [13.0MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz ]
#3dee3e 72MiB/76MiB(93%) CN:1 DL:3.7MiB ETA:1s]
[ o.k. ] Verified [ MD5 ]
[ .... ] decompressing
[ .... ] gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi.tar.xz: 77.0MiB [14.2MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz ]
#42e728 104MiB/104MiB(99%) CN:1 DL:2.8MiB]
[ o.k. ] Verified [ MD5 ]
[ .... ] decompressing
[ .... ] gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi.tar.xz:  104MiB [13.9MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz ]
#2c065e 108MiB/111MiB(97%) CN:1 DL:3.9MiB]
[ o.k. ] Verified [ MD5 ]
[ .... ] decompressing
[ .... ] gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu.tar.xz:  111MiB [13.4MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz ]
#d232ee 250MiB/251MiB(99%) CN:1 DL:2.0MiB]
[ o.k. ] Verified [ MD5 ]
[ .... ] decompressing
[ .... ] gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf.tar.xz:  251MiB [13.7MiB/s] [==========================================================================>] 100%
[ .... ] downloading using http(s) network [ gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu.tar.xz ]
#88b441 268MiB/269MiB(99%) CN:1 DL:0.9MiB]
[ o.k. ] Verified [ MD5 ]
[ .... ] decompressing
```

2) The image website of the cross-compilation tool chain in China is the open source software image site of Tsinghua University

https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/

3) **Toolchains** After downloading, it will contain multiple versions of the cross-compilation toolchain

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

4) The cross-compilation toolchain used to compile the H616 Linux kernel source code is

a. linux5.16

```
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
```

5) The cross-compilation toolchain used to compile the H616 u-boot source code is

a. v2021.10

gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu

6.2.3. Orangepi-build complete directory structure description

1) After the orangepi-build repository is downloaded, it does not contain the source code of the linux kernel, u-boot and the cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository

a. The git repository where the linux kernel source code is stored is as follows, pay attention to switch the branch of the linux-orangepi repository to

a) Linux5.16

https://github.com/orangepi-xunlong/linux-orangepi/tree/**orange-pi-5.16-sunxi64**

b. The git repository where the u-boot source code is stored is as follows. Note that the branch of the u-boot-orangepi repository is switched to

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/**v2021.10-sunxi**

If you are not familiar with orangepi-build and do not know the detailed process of compiling the linux kernel and u-boot, please do not download and use the above linux kernel and u-boot source code for compilation operation, because the compilation script and configuration file of orangepi-build Some adjustments and optimizations will be made to u-boot and linux. If you do not use orangepi-build to compile u-boot and linux, you may encounter problems of compilation failure or failure to start.

2) When orangepi-build runs for the first time, it will download the cross-compilation toolchain, u-boot and linux kernel source code. After successfully compiling a linux image, the files and folders that can be seen in orangepi-build are:

a. **build.sh**: Compile startup script

b. **external**: Contains configuration files, scripts for specific functions and source code of some programs needed to compile the image. The rootfs compressed package cached during the process of compiling the image is also stored in external

c. **kernel**: Store the source code of the linux kernel. The folder named **orange-pi-5.16-sunxi64** stores the kernel source code of the current branch of the H616 series development board. Please do not manually modify the name of the folder of the kernel source code. If modified, compile the system The kernel

source code will be re-downloaded at runtime

d. **LICENSE**: GPL 2 license file
e. **README.md**: orangepi-build documentation
f. **output**: Store the compiled u-boot, linux and other deb packages, compilation logs, and compiled images and other files
g. **scripts**: Generic script for compiling linux images
h. **toolchains**: Store the cross-compilation toolchain
i. **u-boot**: Store the source code of u-boot. The folder named **v2021.10-sunxi** stores the u-boot source code of the H616 series development board. Please do not manually modify the name of the folder of the u-boot source code. If modified, compile The u-boot source code will be re-downloaded when the system is running
j. **userpatches**: Store the configuration files needed to compile the script

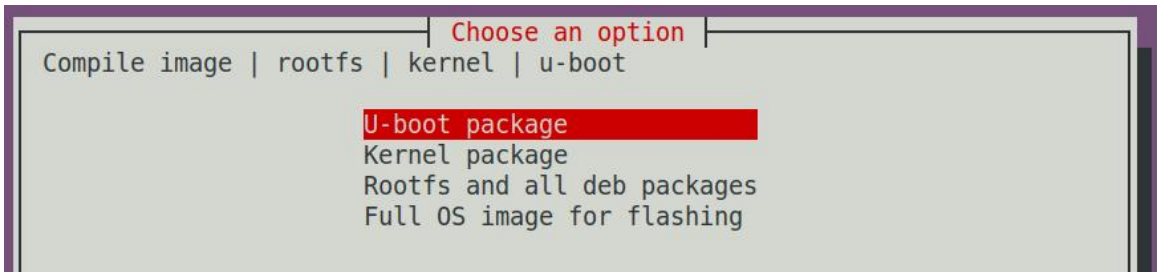
```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts
toolchains  u-boot  userpatches
```

6.3. Compile u-boot

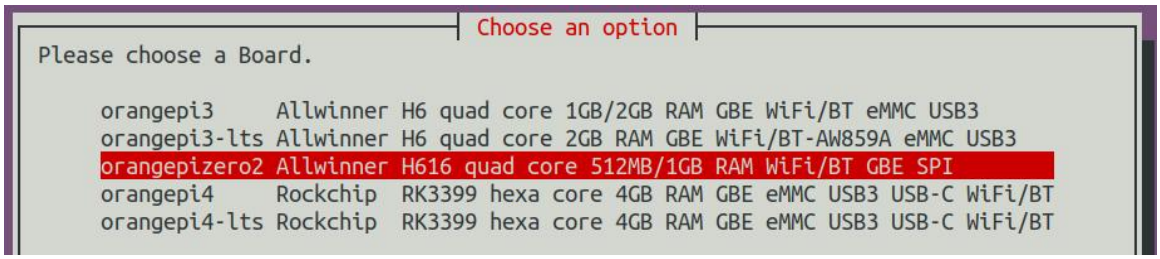
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **U-boot package** and press Enter



3) Then select the model of the development board



4) Then it will start to compile u-boot, and some of the information prompted during compilation are as follows

a. u-boot source code version

[o.k.] Compiling u-boot [**v2021.10**]

b. The version of the cross-compile toolchain

[o.k.] Compiler version [**aarch64-none-linux-gnu-gcc 9.2.1**]

c. The path to the generated u-boot deb package

[o.k.] Target directory [**orangepi-build/output/debs/u-boot**]

d. The package name of the u-boot deb package generated by compilation

[o.k.] File name [**linux-u-boot-current-orangepizero2_3.0.0_arm64.deb**]

e. Compilation time used

[o.k.] Runtime [**1 min**]

f. Repeat the command to compile u-boot, use the following command to start compiling u-boot directly without selecting through the graphical interface

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepizero2 BRANCH=current BUILD_OPT=u-boot KERNEL_CONFIGURE=yes**]

5) View the compiled u-boot deb package

test@test:~/orangepi-build$ **ls output/debs/u-boot/**
linux-u-boot-current-orangepizero2_3.0.0_arm64.deb

6) The files contained in the generated u-boot deb package are as follows

a. Use the following command to decompress the deb package

test@test:~/orangepi-build$ **cd output/debs/u-boot**
test@test:~/orangepi_build/output/debs/u-boot$ $ **dpkg -x** \
linux-u-boot-current-orangepizero2_3.0.0_arm64.deb . (Note that there is a "." at the end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ **ls**

```
linux-u-boot-current-orangepizero2_3.0.0_arm64.deb    usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-current-orangepizero2_3.0.0_arm64
    │   └── u-boot-sunxi-with-spl.bin
    └── u-boot
        ├── LICENSE
        ├── orangepi_zero2_defconfig
        └── platform_install.sh
```

7) The orangepi-bulid compilation system will first synchronize the u-boot source code with the u-boot source code of the github server when compiling the u-boot source code, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code (**You need to compile u-boot once before closing this function, otherwise it will prompt that the source code of u-boot cannot be found. If it is a source code compressed package downloaded from Google Drive, there is no such problem, because the source code of u-boot have been cached**), otherwise the modifications will be restored, as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

8) When debugging the u-boot code, you can use the following method to update the u-boot in the linux image for testing

a. Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-current-orangepizero2_3.0.0_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepizero2-current
```

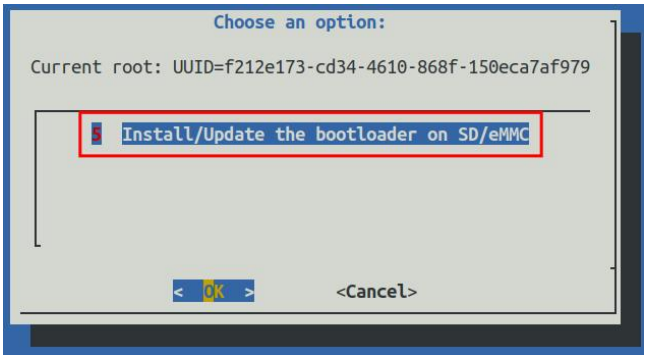
c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-current-orangepizero2_3.0.0_arm64.deb
```

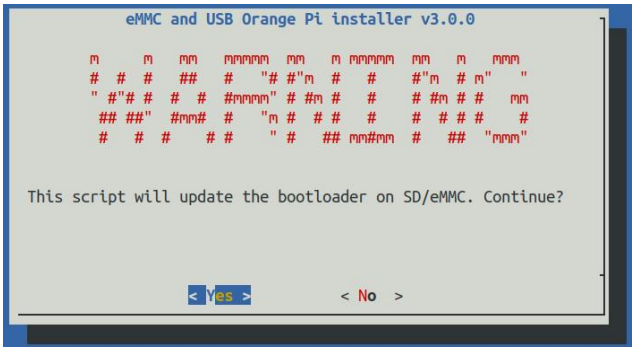
d. Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

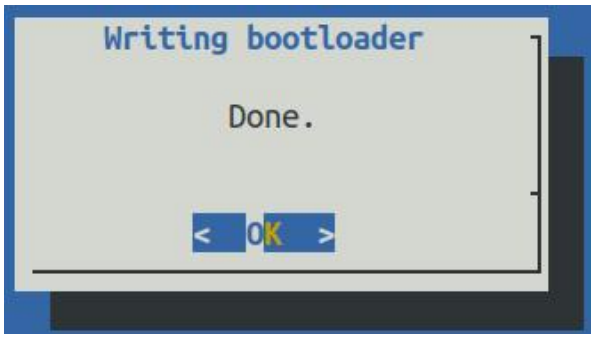
e. then select **5 Install/Update the bootloader on SD/eMMC**



f. f. After pressing the Enter key, a Warring will pop up first



g. Press the Enter key again to start updating u-boot. After the update, the following information will be displayed



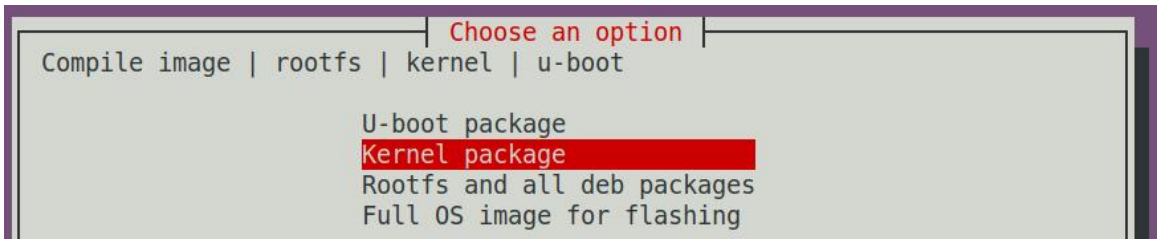
h. Then you can restart the development board to test whether the modification of u-boot takes effect

6.4. Compile the linux kernel

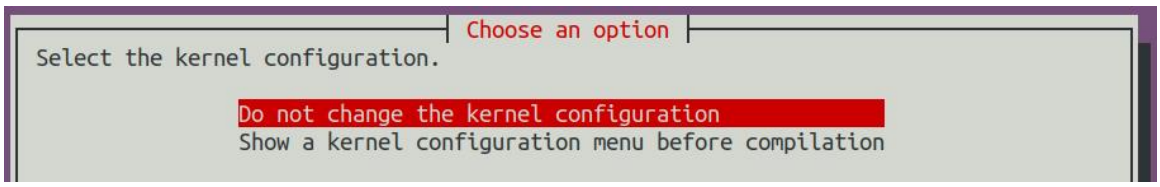
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

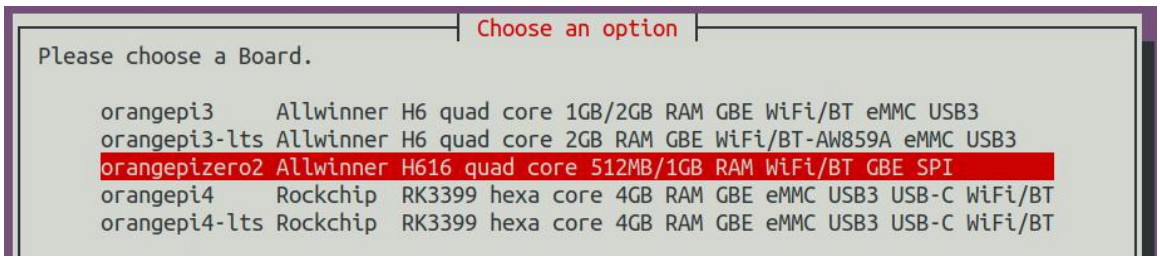
2) Select **Kernel package**, then press Enter



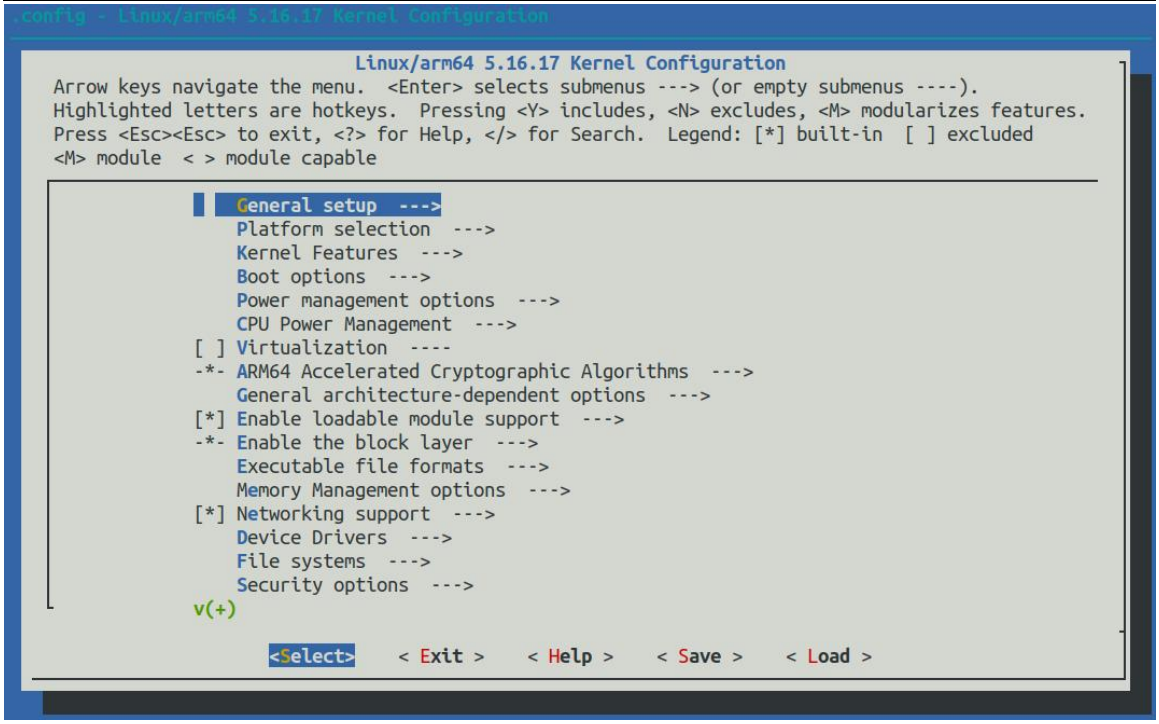
3) Then you will be prompted whether you need to display the kernel configuration interface. If you do not need to modify the kernel configuration, you can choose the first one. If you need to modify the kernel configuration, choose the second one.



4) Then select the model of the development board



5) If you choose to display the kernel configuration menu (the second option) in the second step, the kernel configuration interface opened by **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration, save and exit after modification. , it will start compiling the kernel source code after exiting



a. If you do not need to modify the configuration options of the kernel, when running the build.sh script, pass in **KERNEL_CONFIGURE=no** to temporarily shield the configuration interface of the pop-up kernel

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

b. You can also set **KERNEL_CONFIGURE=no** in the orangepi-build/userpatches/config-default.conf configuration file to permanently disable this feature

c. If the following error is displayed when compiling the kernel, this is because the terminal interface of the Ubuntu PC is too small, so the interface of make menuconfig cannot be displayed. Please adjust the terminal of the Ubuntu PC to the maximum, and then re-run the build.sh script

```
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:560: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
[ error ] ERROR in function compile_kernel [ compilation.sh:376 ]
[ error ] Error kernel menuconfig failed
[ o.k. ] Process terminated
```

6) Part of the information prompted when compiling the kernel source code is explained as follows

a. The version of the linux kernel source code

```
[ o.k. ] Compiling current kernel [ 5.16.17 ]
```

b. The version of the cross-compilation toolchain used

```
[ o.k. ] Compiler version [ aarch64-none-linux-gnu-gcc 9.2.1 ]
```

c. The configuration file used by the kernel by default and the path where it is stored

```
[ o.k. ] Using kernel config file [ config/linux-5.16-sun50iw9-current.config ]
```

d. The path to the generated kernel-related deb package

```
[ o.k. ] Target directory [ output/debs/ ]
```

e. Package name of the kernel image deb package generated by compilation

```
[ o.k. ] File name [ linux-image-current-sun50iw9_3.0.0_arm64.deb ]
```

f. Compilation time used

```
[ o.k. ] Runtime [ 5 min ]
```

g. Finally, the compilation command to repeat the compilation of the last selected kernel will be displayed. Use the following command to directly start compiling the kernel source code without selecting through the graphical interface.

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2 BRANCH=current BUILD_OPT=kernel KERNEL_CONFIGURE=yes ]
```

7) View the compiled and generated kernel-related deb packages

a. **linux-dtb-current-sun50iw9_3.0.0_arm64.deb** contains dtb files used by the kernel

b. **linux-headers-current-sun50iw9_3.0.0_arm64.deb** include kernel header files

c. **linux-image-current-sun50iw9_3.0.0_arm64.deb** contains kernel images and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-current-sun50iw9_3.0.0_arm64.deb
output/debs/linux-image-current-sun50iw9_3.0.0_arm64.deb
output/debs/linux-headers-current-sun50iw9_3.0.0_arm64.deb
```

8) The files contained in the generated linux-image deb package are as follows

a. Use the following command to decompress the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-current-sun50iw9_3.0.0_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-current-sun50iw9_3.0.0_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-current-sun50iw9_3.0.0_arm64.deb  usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/test$ tree -L 2
.
├── boot
│   ├── config-5.16.17-sun50iw9
│   ├── System.map-5.16.17-sun50iw9
│   └── vmlinuz-5.16.17-sun50iw9
├── etc
│   └── kernel
├── lib
│   └── modules
├── linux-image-current-sun50iw9_3.0.0_arm64.deb
└── usr
    ├── lib
    └── share

8 directories, 4 files
```

9) The orangepi-Bulid compiler will first synchronize the Linux kernel source code with github server Linux kernel source code, so if you want to modify the Linux kernel source code, First of all need to be closed source update feature (**need complete compiled a Linux kernel source code to close the function, otherwise it will prompt can not find the Linux kernel source code, if from Google Drive to download the source code package, there is no this problem, because Linux is the source of all cached**), otherwise the changes will be restored, The method is as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

10) If the kernel has been modified, the following methods 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 of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-current-sun50iw9_3.0.0_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the deb package of the installed linux kernel

```
root@orangepi:~# apt purge -y linux-image-current-sun50iw9
```

c. Install the deb package of the new linux kernel just uploaded

```
root@orangepi:~# dpkg -i linux-image-current-sun50iw9_3.0.0_arm64.deb
```

d. Then restart the development board, and then check whether the kernel-related modifications have taken effect

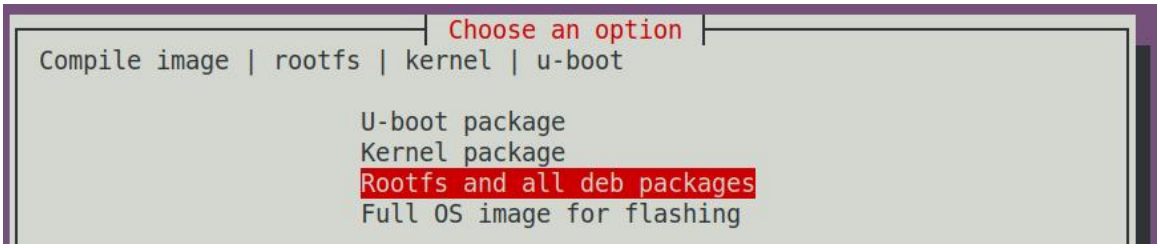
```
root@orangepi:~# reboot
```

6.5. Compile rootfs

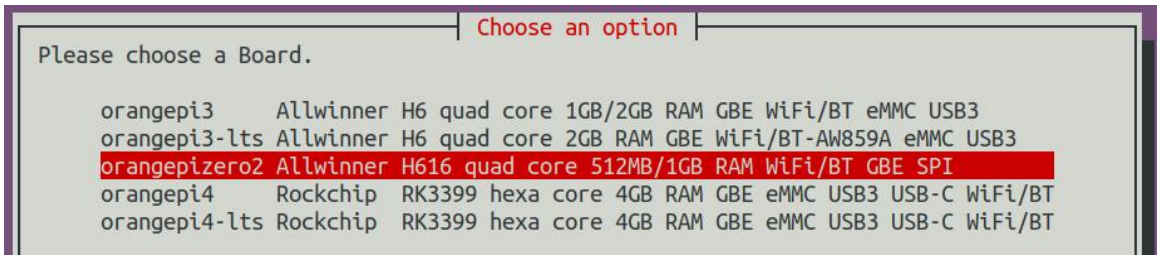
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

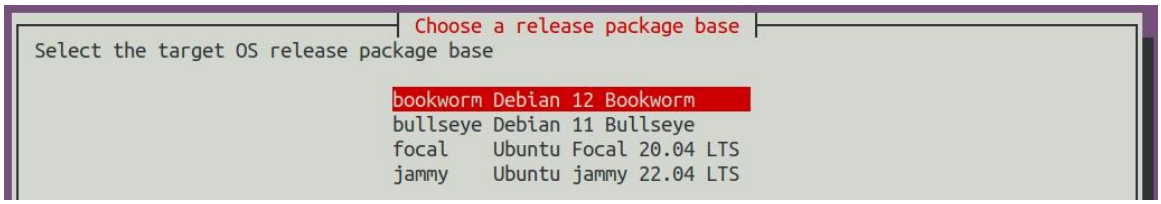
2) Select **Rootfs and all deb packages**, then press Enter



3) Then select the model of the development board



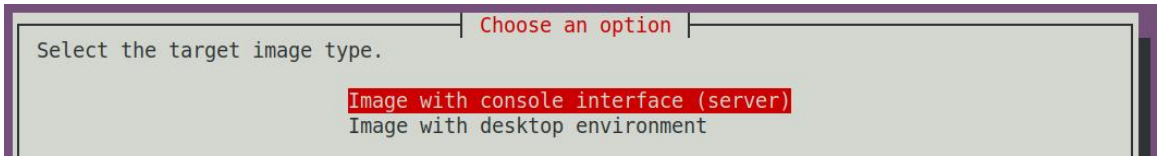
4) Then select the type of rootfs



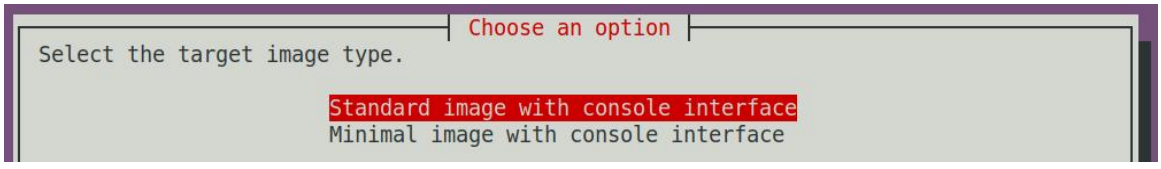
5) Then select the type of image

a. **Image with console interface (server)** indicates the image of the server version, which is relatively small in size

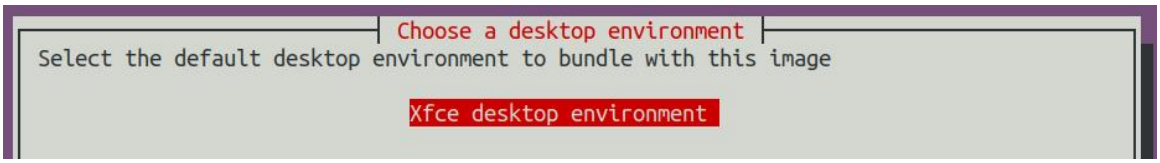
b. **Image with desktop environment** indicates a image with a desktop, which is relatively large in size

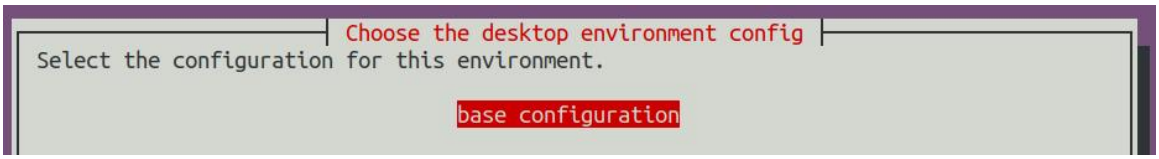


6) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.

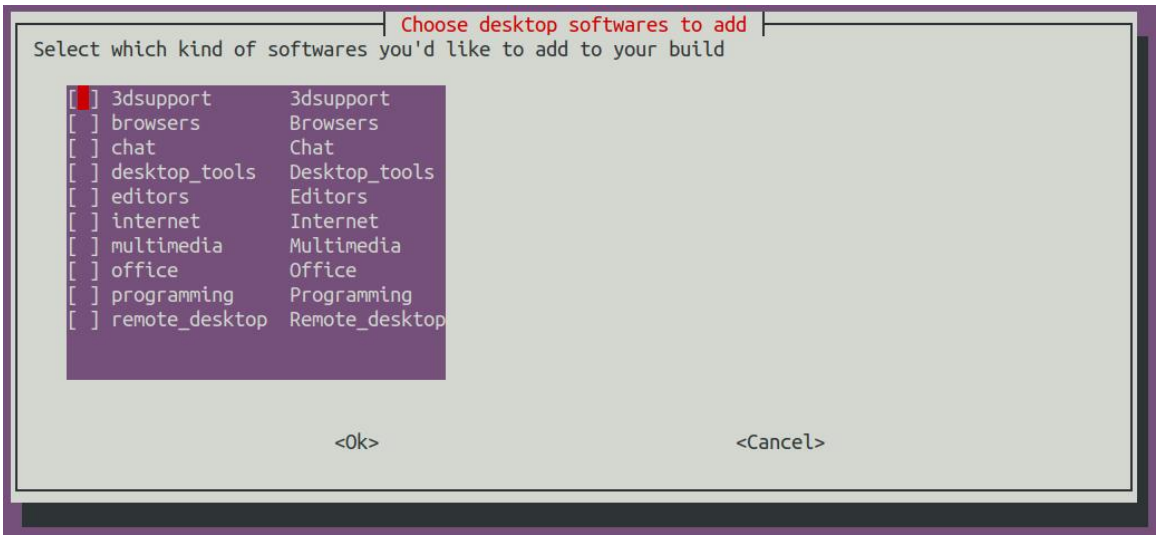


7) If you are compiling the desktop version of the image, you also need to select the type of desktop environment, but only XFCE is currently supported, so just press Enter.





You can then select additional packages that need to be installed. For example, if you need to install a browser, you can choose **browsers**.



Which packages are included in each selection can be seen in the code of orangepi-build. You can also modify these configuration files to add the packages you want to install:

a. First enter the **external/config/desktop** directory to see the desktop configuration folders of different linux distributions. Note that not all the Orange Pi development boards that can be seen in the code are supported and tested.

```
test@test:~/orangepi-build$ cd external/config/desktop
test@test:~/orangepi-build/external/config/desktop$ ls
bionic  bookworm  bullseye  buster  focal  jammy  README.md  sid
```

b. Then select the type of distribution you want to view or modify, and enter the corresponding directory, such as **bullseys**

```
test@test:~/orangepi-build/external/config/desktop$ cd bullseye
```

```
test@test:~/orangepi-build/external/config/desktop/bullseye$ ls
appgroups  environments
```

c. Then enter the **appgroups** directory to see all app groups

```
test@test:~/orangepi-build/external/config/desktop/bullseye$ cd appgroups
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ ls
3dsupport  browsers  chat  desktop_tools  editors  internet  multimedia
office  programming  remote_desktop
```

d. Open the packages file under different groups to view the software contained in the group

```
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat  \
programming/packages
build-essential
clang
meld
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat  \
office/packages
libreoffice
```

8) Then it will start to compile rootfs, and some of the information prompted during compilation are as follows

a. type of rootfs

```
[ o.k. ] local not found [ Creating new rootfs cache for bullseye ]
```

b. The storage path of the rootfs compressed package generated by compilation

```
[ o.k. ] Target directory [ external/cache/rootfs ]
```

c. The name of the rootfs compressed package generated by compilation

```
[ o.k. ] File name
 [ bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4 ]
```

d. Compilation time

```
[ o.k. ] Runtime [ 13 min ]
```

e. Repeat the command to compile rootfs, use the following command to start compiling rootfs directly without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2
BRANCH=current BUILD_OPT=rootfs RELEASE=bullseye
BUILD_MINIMAL=no  BUILD_DESKTOP=no
KERNEL_CONFIGURE=yes  ]
```

9) View the rootfs compressed package generated by compilation

a. bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4 is the compressed package of rootfs, and the meaning of each field of the name is
   a) **bullseye** indicates the type of linux distribution of rootfs
   b) **xfce** indicates that the rootfs is of the desktop version, and if it is **cli**, it indicates the server version
   c) **arm64** represents the architecture type of rootfs
   d) **25250ec7002de9e81a41de169f1f89721** is the MD5 hash value generated by the package names of all packages installed by rootfs. As long as the list of packages installed by rootfs is not modified, this value will not change, and the compilation script will use this MD5 hash value to Determine if you need to recompile rootfs

b. bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list lists the package names of all packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.current
bullseye-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list
```

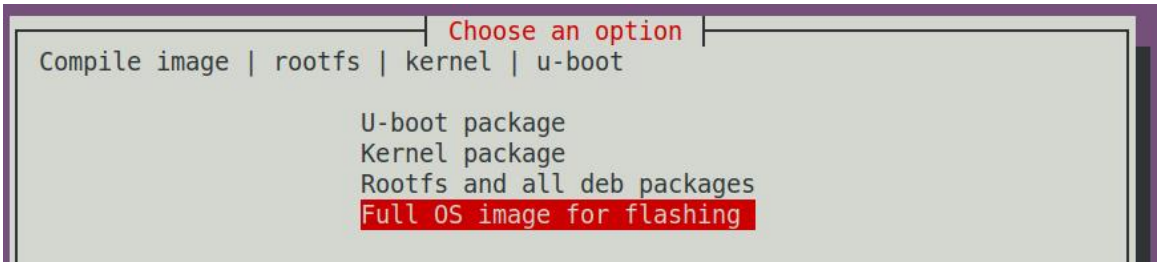
10) If the required rootfs already exists under **external/cache/rootfs**, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it has There is a rootfs available for cache, if there is one, use it directly, which can save a lot of download and compilation time

6.6. Compile the linux image

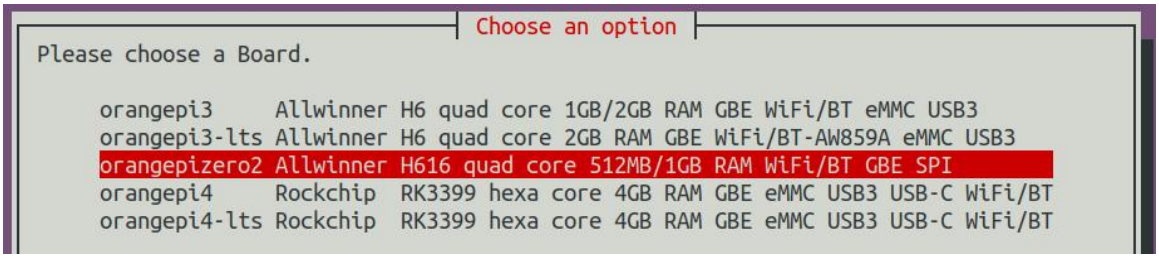
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

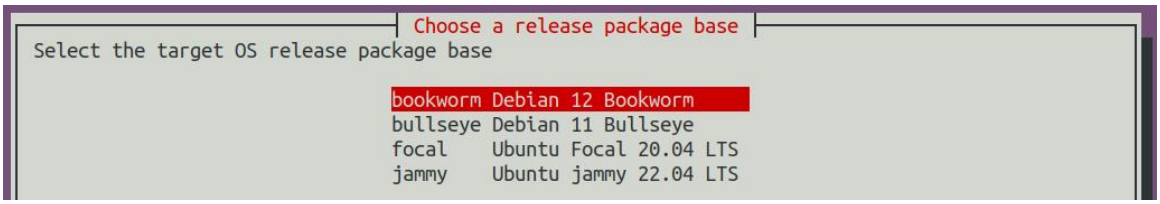
2) Select **Full OS image for flashing** and press Enter



3) Then select the model of the development board



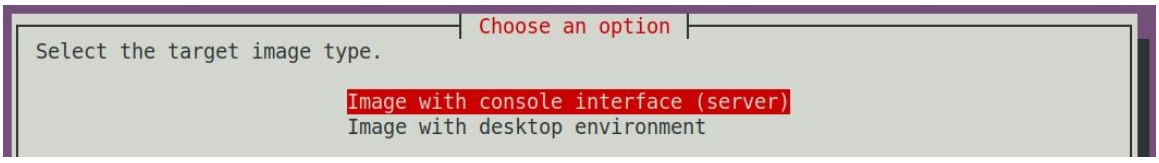
4) Then select the type of rootfs



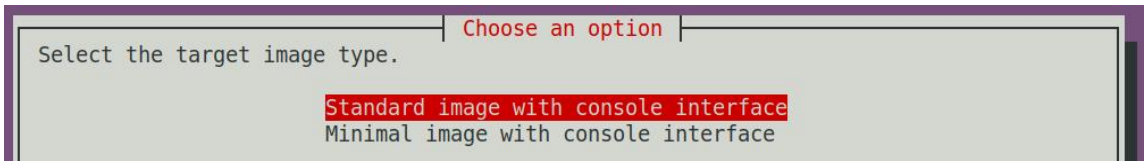
5) Then select the type of image

a. **Image with console interface (server)** represents the image of the server version, which is relatively small in size

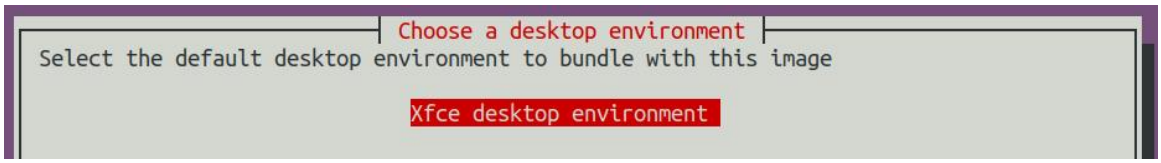
b. **Image with desktop environment** means an image with a desktop, which is relatively large in size

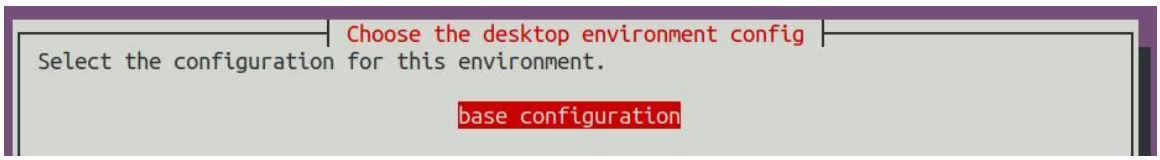


6) If you are compiling the image of the server version, you can also choose to compile the Standard version or the Minimal version. The software pre-installed in the Minimal version will be much less than the Standard version.

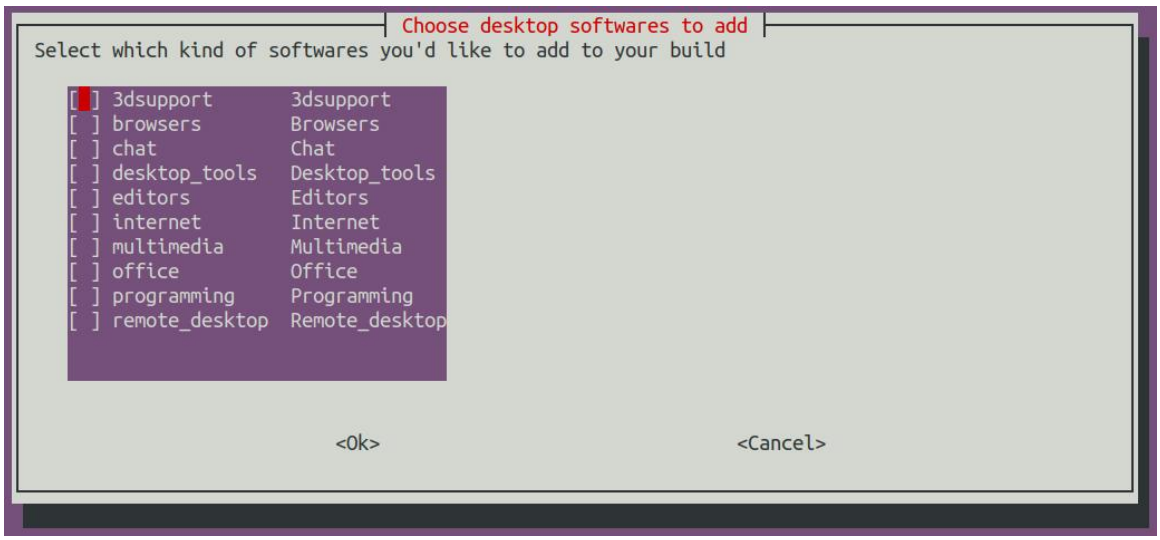


7) If you are compiling the desktop version of the image, you also need to select the type of desktop environment, but only XFCE is currently supported, so just press Enter.





You can then select additional packages that need to be installed. For example, if you need to install a browser, you can choose **browsers**.



Which packages are included in each selection can be seen in the code of orangepi-build. You can also modify these configuration files to add the packages you want to install:

a. First enter the **external/config/desktop** directory to see the desktop configuration folders of different linux distributions. Note that not all the Orange

Pi development boards that can be seen in the code are supported and tested.

```
test@test:~/orangepi-build$ cd external/config/desktop
test@test:~/orangepi-build/external/config/desktop$ ls
bionic  bookworm  bullseye  buster  focal  jammy  README.md  sid
```

b. Then select the type of distribution you want to view or modify, and enter the corresponding directory, such as **bullseys**

```
test@test:~/orangepi-build/external/config/desktop$ cd bullseye
test@test:~/orangepi-build/external/config/desktop/bullseye$ ls
appgroups  environments
```

c. Then enter the **appgroups** directory to see all app groups

```
test@test:~/orangepi-build/external/config/desktop/bullseye$ cd appgroups
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ ls
3dsupport  browsers  chat  desktop_tools  editors  internet  multimedia
office  programming  remote_desktop
```

d. Open the packages file under different groups to view the software contained in the group

```
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat  \
programming/packages
build-essential
clang
meld
test@test:~/orangepi-build/external/config/desktop/bullseye/appgroups$ cat  \
office/packages
libreoffice
```

8) Then it will start to compile the linux image. The general process of compilation is as follows

a. Initialize the compilation environment of the Ubuntu PC and install the software packages required for the compilation process

b. Download the source code of u-boot and linux kernel (if cached, only update the code)

c. Compile u-boot source code and generate u-boot deb package

d. Compile the linux source code to generate linux-related deb packages

e. Make a deb package of linux firmware

f. Make the deb package of the orangepi-config tool

g. Make board-level supported deb packages

h. If you are compiling the desktop version of the image, you will also create a desktop-related deb package

i. Check whether the rootfs has been cached, if there is no cache, then recreate the rootfs, if it has been cached, directly decompress and use

j. Install the deb package generated earlier into rootfs

k. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configuration, etc.

l. Then make an image file and format the partition, the default type is ext4

m. Copy the configured rootfs to the mirrored partition

n. then update initramfs

o. Finally, write the bin file of u-boot into the image through the dd command

9) After compiling the image, the following information will be prompted

a. The storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/orangepizero2_3.0.0_debian_bullseye_linux5.10.43_xfce_desktop/orangepizero2_3.0.0_debian_bullseye_linux5.10.43_xfce_desktop.img ]
```

b. Compilation time used

```
[ o.k. ] Runtime [ 19 min ]
```

c. Repeat the command to compile the image, use the following command to start compiling the image directly without selecting through the graphical interface

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepizero2 BRANCH=current BUILD_OPT=image RELEASE=bullseye BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

7. Instructions for using Android SDK

The compilation of the Android SDK is carried out on a PC with Ubuntu 14.04 installed. Other versions of Ubuntu systems may have some differences. The image download address of the Ubuntu14.04 amd64 version is as follows: https://repo.huaweicloud.com/ubuntu-releases/14.04/ubuntu-14.04.6-desktop-amd64.iso

The Android SDK is the original SDK released by Allwinner. If you want to use the Android image compiled by the Android SDK on the Orange Pi development board, you need to adapt it to the Orange Pi development board to ensure that all functions can be used normally.

7.1. Download the source code of Android SDK

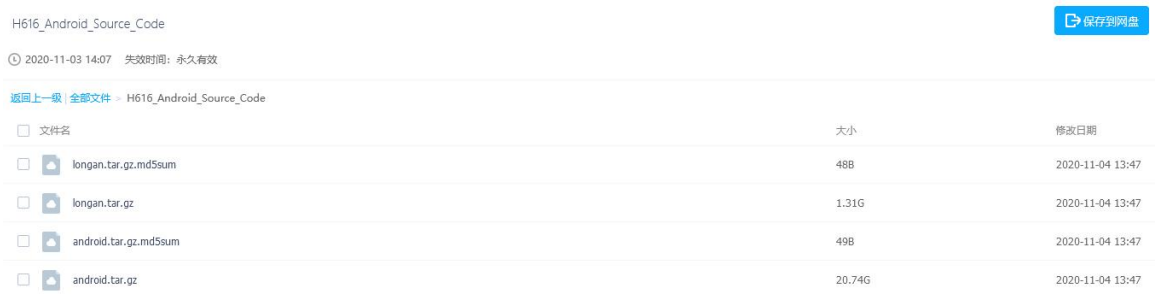
1) The Android source code folder of H616 contains the following 4 files

a. **android.tar.gz**：android related source code

b. **android.tar.gz.md5sum**：MD5 checksum file of android.tar.gz

c. **longan.tar.gz**：Contains u-boot, linux kernel and other source code (excluding the source code of boot0)

d. **longan.tar.gz.md5sum**：MD5 checksum file of d.longan.tar.gz



2) After downloading the Android source code, please check whether the MD5 checksum is correct. If it is not correct, please download the source code again.

```
test@test:~$ md5sum -c android.tar.gz.md5sum
android.tar.gz: Sure
test@test:~$ md5sum -c longan.tar.gz.md5sum
longan.tar.gz: Sure
```

3) Then unzip the Android source code

a. android: Store android related source code

b. longan: Store the source code of the linux kernel and u-boot (excluding the source code of boot0), and other configuration files

```
test@test:~$ tar -zxf android.tar.gz
test@test:~$ tar -zxf longan.tar.gz
test@test:~$ ls
android  longan
```

7.2. Build Android Compilation Environment

1) Use Ubuntu 14.04 to compile the Android 10 source code, you need to ensure that Ubuntu 14.04 uses the linux 4.4 kernel, otherwise an error will be reported when compiling. When installing the Ubuntu14.04 system, if you use the latest version of the Ubuntu14.04 image file **ubuntu-14.04.6-desktop-amd64.iso**, then the kernel defaults to linux 4.4 after the system is installed. If your current Ubuntu14.04 system If it is not installed using the latest image file, then first check whether the kernel version is linux 4.4, if not, please upgrade the kernel to linux 4.4, or use **ubuntu-14.04.6-desktop-amd64.iso** to reinstall the Ubuntu system

```
test@test:~$ uname -a
Linux ubuntu 4.4.0-142-generic #168~14.04.1-Ubuntu SMP Sat Jan 19 11:26:28 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

2) Install JDK

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

3) Configure JAVA environment variables

a. First determine the installation path of java, generally

```
test@test:~$ ls /usr/lib/jvm/java-8-openjdk-amd64
ASSEMBLY_EXCEPTION  bin  docs  include  jre  lib  man  src.zip
THIRD_PARTY_README
```

b. Then use the following command to export java environment variables

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

4) Install platform support software

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

test@test:~$ sudo apt-get install u-boot-tools
```

7.3. Compile the android image

7.3.1. Compile the kernel

1) First configure the compilation environment

```
test@test:~$ cd longan
test@test:~/longan$ ./build.sh config

Welcome to mkscript setup progress
All available platform:
    0. android
    1. linux
Choice [android]: 0
All available ic:
    0. h313
    1. h616
    2. h700
Choice [h616]: 1
All available board:
    0. fpga
    1. ft
    2. p1
    3. p2
    4. perf1
```

```
    5. perf1_axp152
    6. perf2
    7. perf3
    8. qa
Choice [p2]: 3
INFO: kernel defconfig: generate longan/kernel/linux-4.9/.config by
longan/kernel/linux-4.9/arch/arm64/configs/sun50iw9p1smp_h616_android_defconfig
*** Default configuration is based on 'sun50iw9p1smp_h616_android_defconfig'
#
# configuration written to .config
#
```

2) Then start compiling

```
test@test:~/longan$ ./build.sh
```

3) The output after compilation is as follows

```
sun50iw9p1 compile Kernel successful


INFO: build kernel OK.
INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: ----------------------------------------
INFO: build lichee OK.
INFO: ----------------------------------------
```

7.3.2. Compile Android source code

1) The command to compile Android is as follows

```
test@test:~$ cd android
test@test:~/android$ source build/envsetup.sh
test@test:~/android$ lunch cupid_p2-eng
test@test:~/android$ extract-bsp
test@test:~/android$ make -j8
```

2) After the compilation is completed, the following information will be printed

```
##### build completed successfully (01:51 (mm:ss)) ####
```

3) Then use the **pack** command to package and generate an Android image

```
test@test:~/android$ pack
......
----------image is at----------

longan/out/h616_android10_p2_uart0.img

pack finish
use pack4dist for release
```

4) The path where the generated Android image is stored is

```
longan/out/h616_android10_p2_uart0.img
```