Orange Pi 4 Pro
User Manual

Catalogue

1. Basic features of Orange Pi 4 Pro

1. 1. What is Orange Pi 4 Pro

The Orange Pi 4 Pro uses the Allwinner A733 octa-core processor, integrating an octa-core high-performance CPU, a dual-core Arm Cortex-A76 + a hexa-core Arm Cortex-A55 architecture, a 12nm process, and a maximum main frequency of 2.0GHz. It supports a 3T NPU to meet the needs of edge intelligent AI acceleration applications. It supports up to 16GB of LPDDR5 and supports up to 8K@24fps H.265/VP9/AVS2 video format decoding and up to 4K@30fps H.265/H.264 video format encoding. It also has a rich interface, including Gigabit Ethernet, PCIe3.0, USB3.0, MIPI-CSI, MIPI-DSI, 40Pin expansion interface, and other common functional interfaces. It supports operating systems such as Ubuntu, Debian, and Android 13.

Orange Pi 4 Pro provides a solid hardware foundation for the scenario-based implementation of generative AI and artificial intelligence algorithms. It can be widely used in smart industrial control, smart commercial displays, retail payments, smart education, commercial robots, in-vehicle terminals, visual driver assistance, edge computing, and smart power distribution terminals.

1. 2. Purpose of Orange Pi 4 Pro

We can use it to achieve：

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

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

1. 3. Who is the Orange Pi 4 Pro designed for?

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

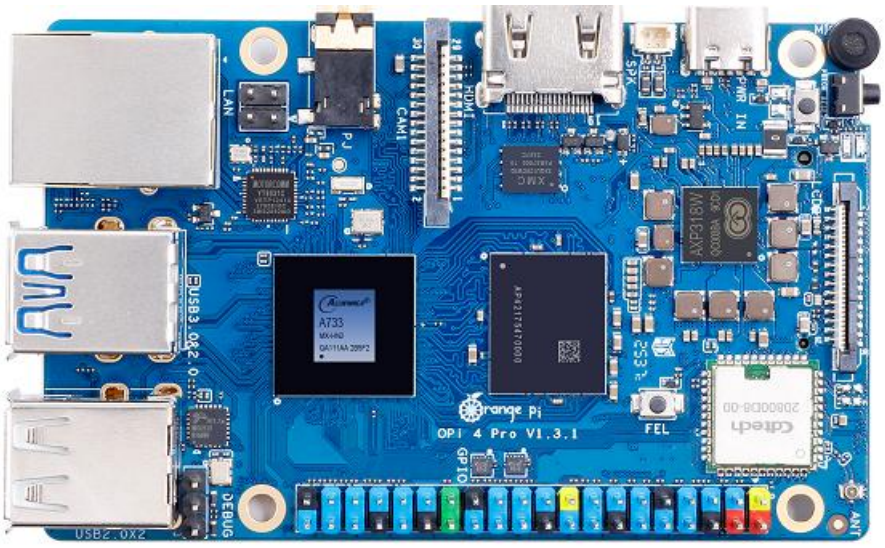
1. 4. Orange Pi 4 Pro Hardware Features

Hardware Features	
Processor	A733, 2 Cortex-A76 cores + 6 Cortex-A55 cores, maximum frequency 2.0GHz RISC-V E902 coprocessor（200MHz） GPU：Imagination BXM-4-64 NPU：3TOPS
Memory	Highest support 16GB LPDDR5
Storage	eMMC module Optional：16GB/32GB/64GB/128GB SPI Flash：128Mb（Default）、256Mb Optional M.2 M-KEY Socket：PCIe3.0 NVMe SSD uSD card slot：supports up to 128GB uSD card
Wi-Fi + Bluetooth	Wi-Fi + Bluetooth two-in-one module Wi-Fi6 + BT 5.4，BLE
Ethernet	Gigabit Ethernet (onboard PHY chip: YT8531CA), support PoE power supply
Display	1x HDMI TX 2.0 interface up to 4K@60fps 1x 4-lane MIPI-DSI
Camera	1x 2-lane MIPI-CSI camera interface 1x 4-lane MIPI-CSI camera interface
USB	1xUSB Type-A 3.0 3xUSB Type-A 2.0 HOST
Audio	3.5mm headphone jack (supports audio input/output), single

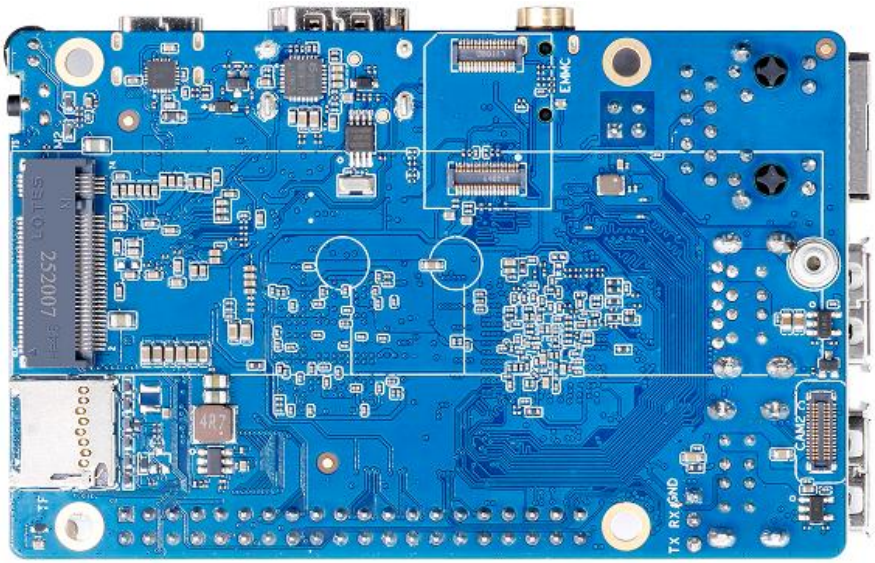
	speaker, single microphone
Button	1 * BOOT，1 * RESET，1 * PWR ON
40Pin	40Pin function expansion interface, supports the following interface types： GPIO、UART、I2C、SPI、PWM
DEBUG	3Pin debug serial port
Power Supply	Type-C 5V 3A DCIN
Supported OS	Ubuntu、Debian、Android13 etc
Appearance Specifications	
PCB	89mm*56mm
Weight	58g

1.5. Orange Pi 4 Pro top and bottom views

Top-level view：

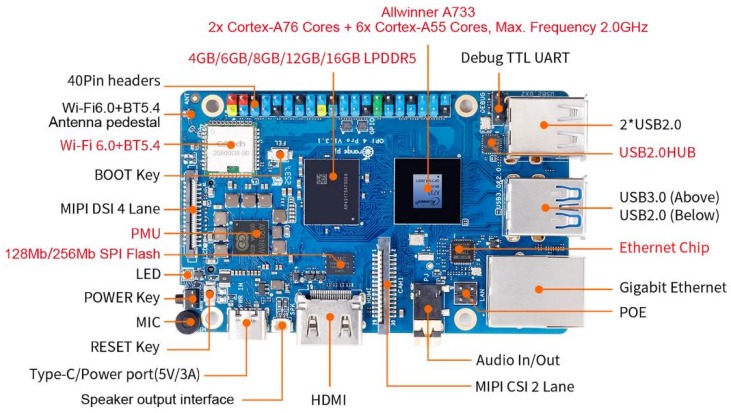


Bottom-level view：

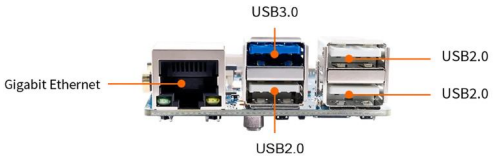


1. 6. Orange Pi 4 Pro interface details

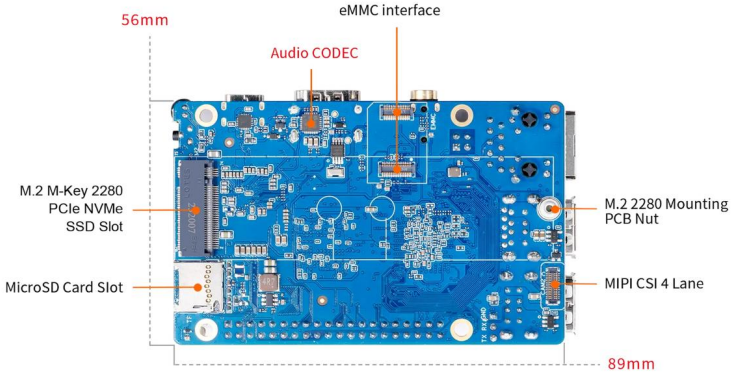
Product display



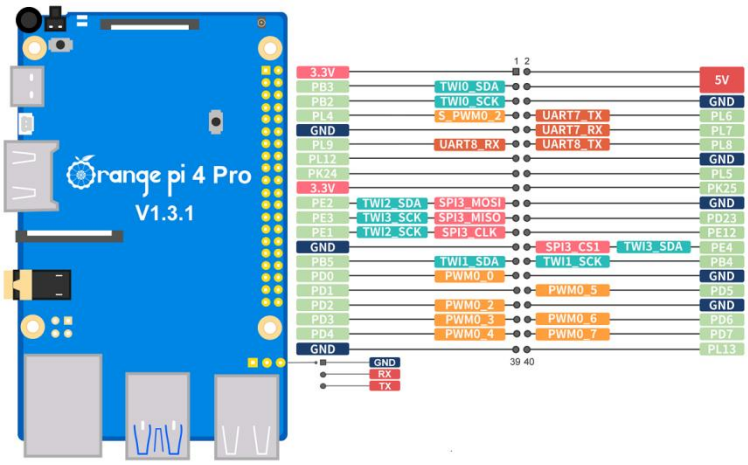
Top View



Side View



Bottom View

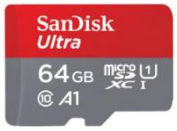


The diameters of the four positioning holes are 3.0mm

2. Introduction to the use of development board

2. 1. Prepare the necessary accessories

1) TF card, minimum 8GB capacity, class 10 or higher high-speed SanDisk card



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

3) HDMI interface display

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

5) 10.1-inch MIPI screen, used to display the system interface of the development board (this screen is common to both the adapter board and OPi5Plus/OPi5B/OPi5/OPi5Pro/OPi5Max/OPi4A/OPi4Pro)

6) Power adapter: Orange Pi 4 Pro recommends using a 5V/3A Type-C power adapter

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

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

8) USB camera

9) 100M or 1000M Ethernet cable to connect the development board to the Internet

10) USB2.0 male-to-male data cable, used for adb debugging, burning images to eMMC, etc.

11) USB to TTL module and DuPont line are needed to connect the development board and the computer when using the serial port debugging function.

Note that the TTL level used by the development board is 3.3v. In addition to the USB to TTL module shown above, other similar 3.3v USB to TTL modules are generally also acceptable.

12) X64 computer with Ubuntu and Windows operating systems installed

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

2.2. Download the development board image and related information

1) The download link for the Chinese version is

http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-4-Pro.html

2) The download link for the English version is

http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-4-Pro.html

3) The information mainly includes

a. **Linux source code**：Save on Github
b. **Android image**：Save on Baidu Cloud and Google Drive
c. **Ubuntu image**：Save on Baidu Cloud and Google Drive
d. **Debian image**：Save on Baidu Cloud and Google Drive
e. **User manual and schematics**：Chip-related data sheets will also be placed here

f. **Official Tools：** Mainly includes the software needed during the use of the development board

2. 3. Method for burning Linux image to TF card based on Windows PC

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

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

1) First, prepare a TF card with a capacity of 8GB or larger. The transmission speed of the TF card must be **class 10** or above. It is recommended to use a TF card from a brand such as SanDisk.

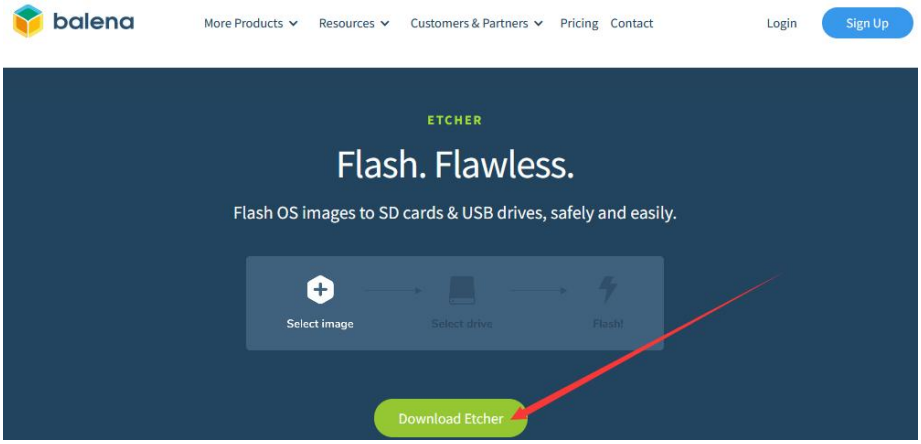
2) Then use the card reader to insert the TF card into the computer

3) Download the compressed Linux operating system image file you want to burn from the **Orange Pi data download page**, and then use decompression software to decompress it. Among the decompressed files, the file ending with "**.img**" is the operating system image file, which is generally over 1GB in size.

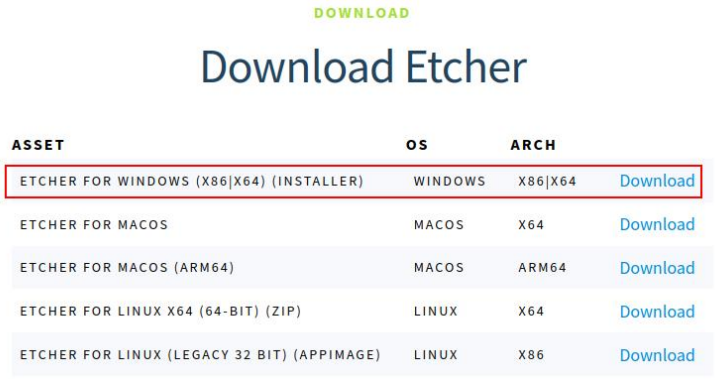
4) Then download the Linux image burning software - **balenaEtcher**, the download address is

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

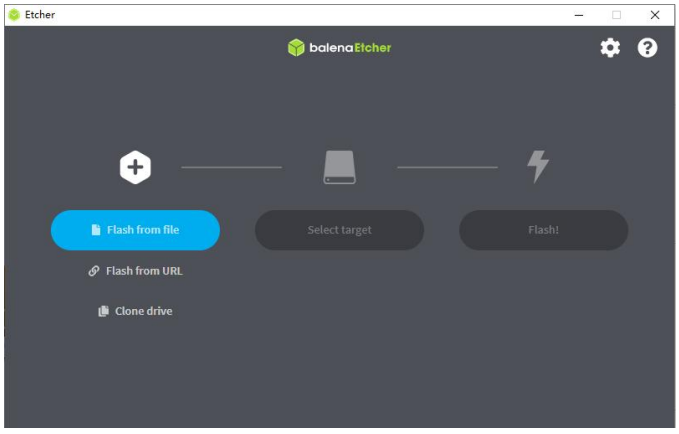
5) After entering the balenaEtcher download page, click the green download button to jump to the software download selection interface



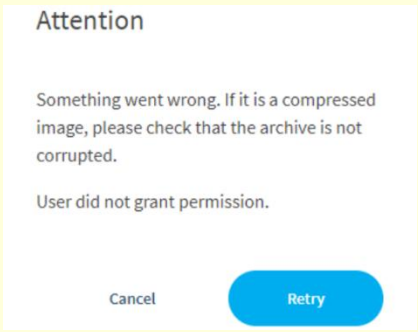
6) Then download the installation package of balenaEtcher Windows version.



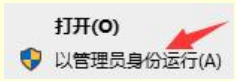
7) Then install balenaEtcher and open it. The balenaEtcher interface after opening is as shown below



If the following error is prompted when opening balenaEtcher:



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



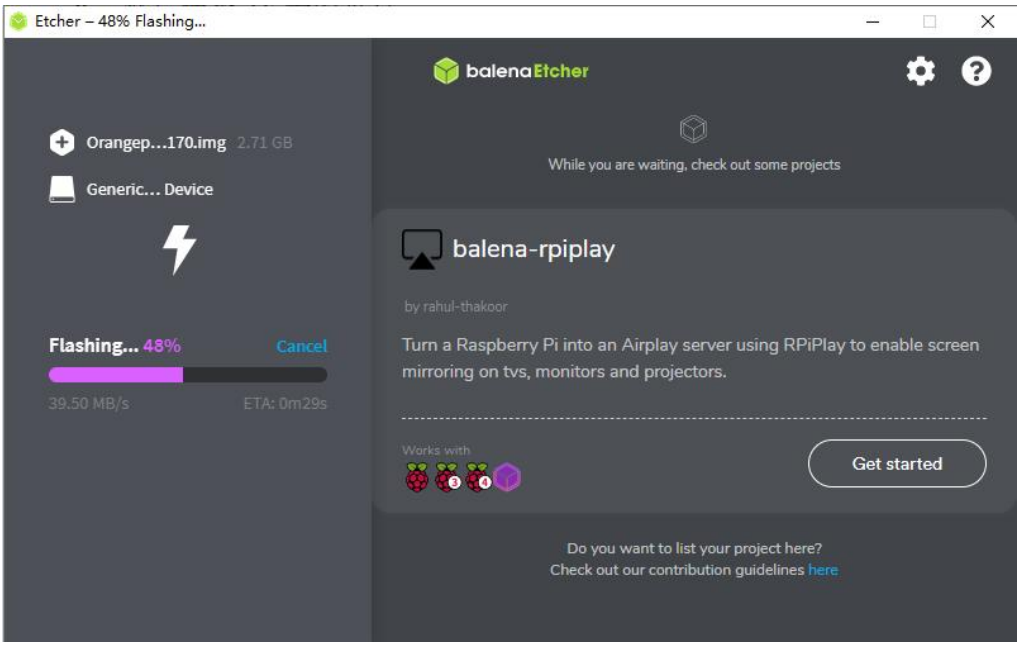
8) The specific steps for burning the Linux image using balenaEtcher are as follows

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

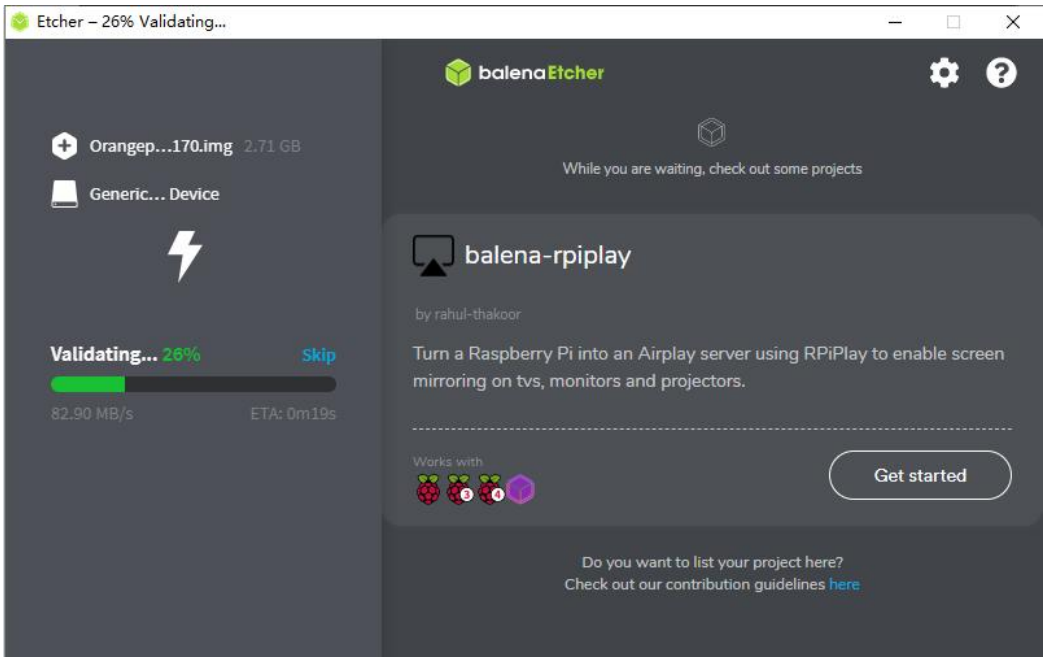
b. Then select the drive letter of the TF card

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

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

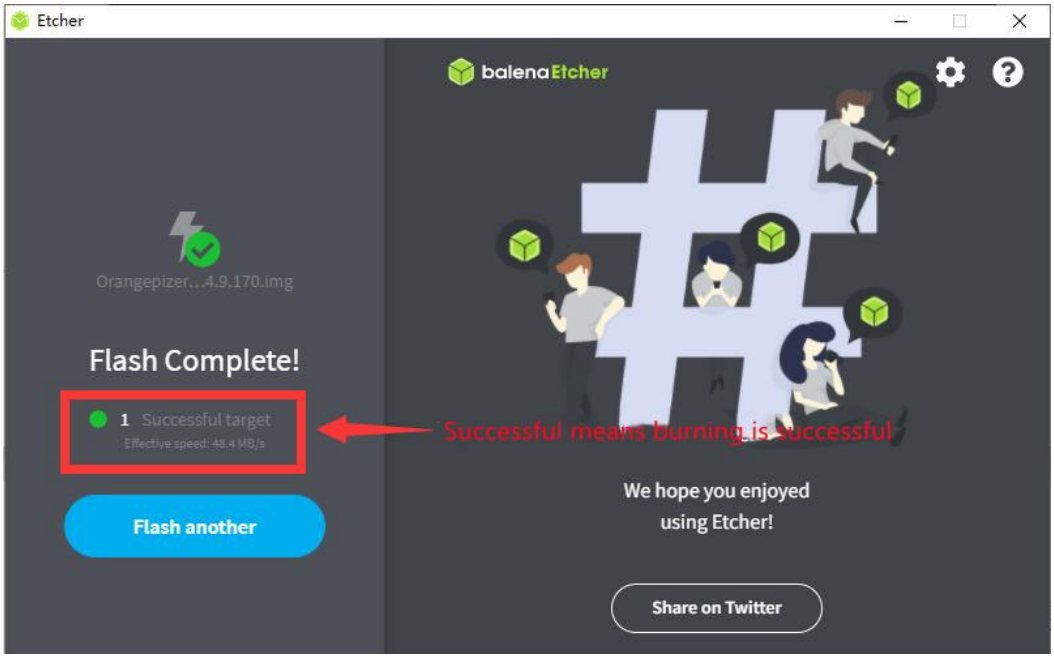


10) After the Linux image is burned, balenaEtcher will verify the image burned to the TF card by default to ensure that there are no problems with the burning process. As shown in the figure below, a green progress bar indicates that the image has been burned and balenaEtcher is verifying the burned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



11) After the burning is completed successfully, the display interface of balenaEtcher is as shown below. If the green indicator icon is displayed, it means that the image is burned

successfully. At this time, you can exit balenaEtcher, then pull out the TF card and insert it into the TF card slot of the development board for use.



2. 3. 2. How to burn Linux image using Win32Diskimager

1) First, prepare a TF card with a capacity of 8GB or larger. The transmission speed of the TF card must be **class 10** 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

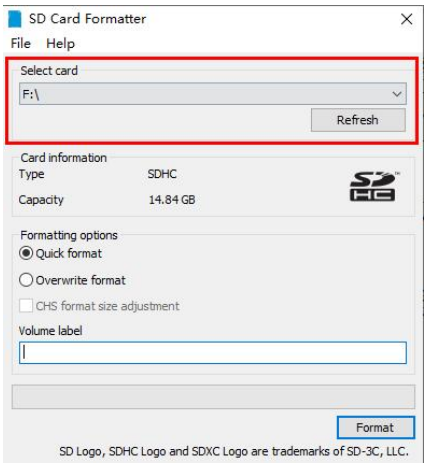
3) Then format the TF card

a. You can use **SD Card Formatter** to format the TF card. The download address is:

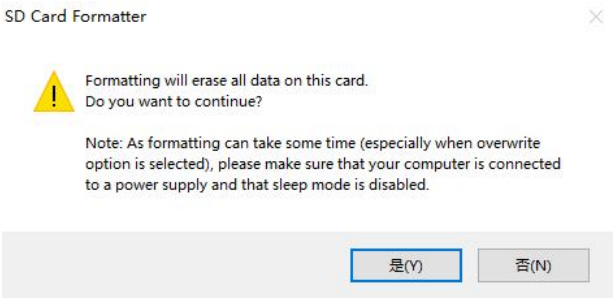
https://www.sdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip

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

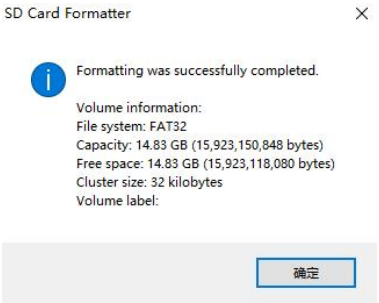
c. If only a TF card is inserted into the computer, the drive letter of the TF card will be displayed in the "**Select card**" column. If multiple USB storage devices are inserted into the computer, you can select the drive letter corresponding to the TF card through the drop-down box.



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



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



4) Download the compressed Linux operating system image file you want to burn from the Orange Pi data download page, and then use decompression software to decompress it. Among the decompressed files, the file ending with "**.img**" is the operating system image file, which is generally over 1GB in size.

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

a. The download page for Win32Diskimager is

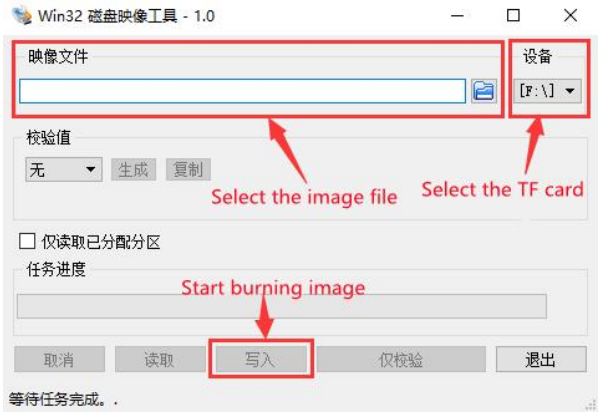
http://sourceforge.net/projects/win32diskimager/files/Archive/

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

a) First select the path of the image file

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

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



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

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

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

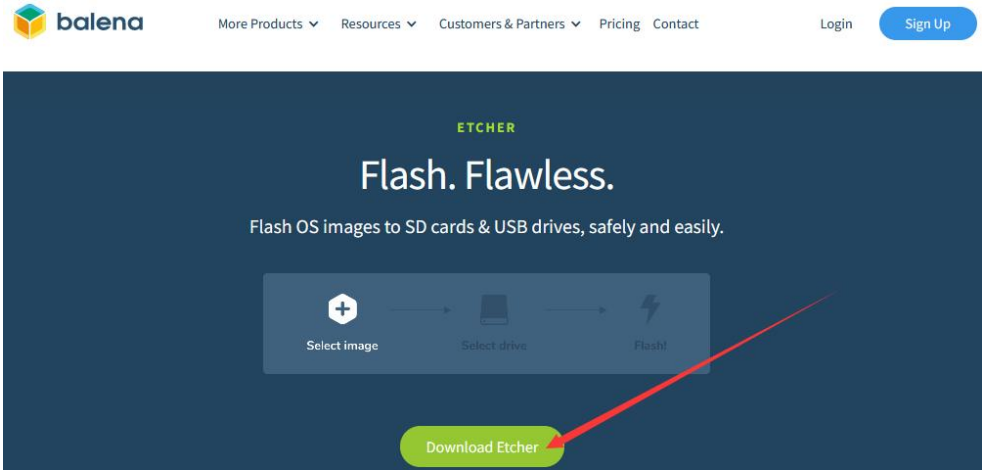
1) First, prepare a TF card with a capacity of 8GB or larger. The transmission speed of the TF card must be **class 10** or above. It is recommended to use a TF card from a brand such as SanDisk.

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

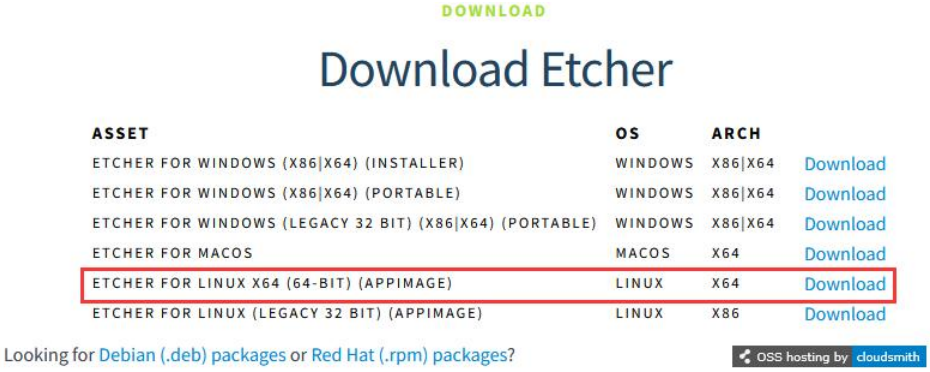
3) Download the balenaEtcher software from the download address:

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

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



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



6) Download the Linux operating system image file compressed package you want to burn from the **Orange Pi data download page**, and then use decompression software to decompress it. After decompression, the file ending with "**.img**" is the operating system image file, which is generally larger than 1GB in size. The decompression command for the compressed package ending with 7z is as follows：

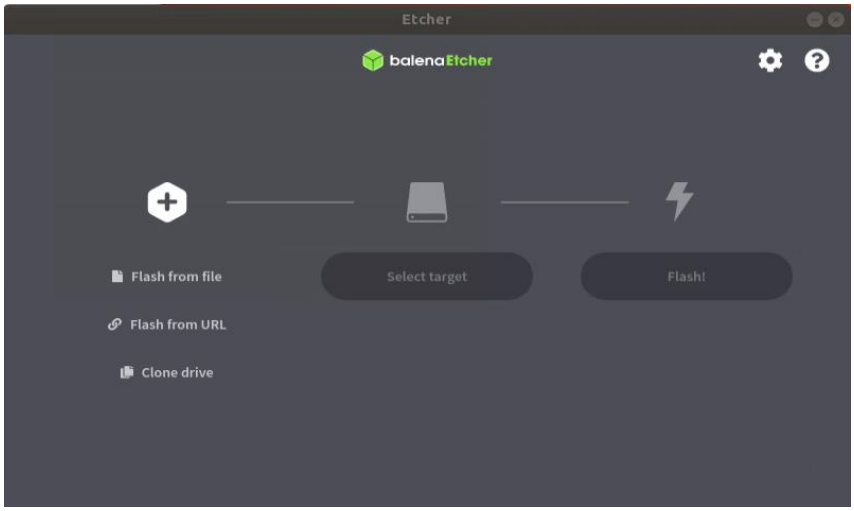
```
test@test:~$ 7z x Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.7z
test@test:~$ ls Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.*
```

```
Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.7z
Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.sha      #Checksum File
Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.img      #Image file
```

7) After decompressing the image, you can use the **sha256sum -c *.sha** command to calculate the checksum. If it is successful, it means the downloaded image is correct and you can burn it to the TF card with confidence. If it prompts that the **checksum does not match**, it means there is a problem with the downloaded image. Please try to download it again.

```
test@test:~$ sha256sum -c *.sha
Orangepi4pro_1.0.0_ubuntu_jammy_desktop_linux5.15.147.img: success
```

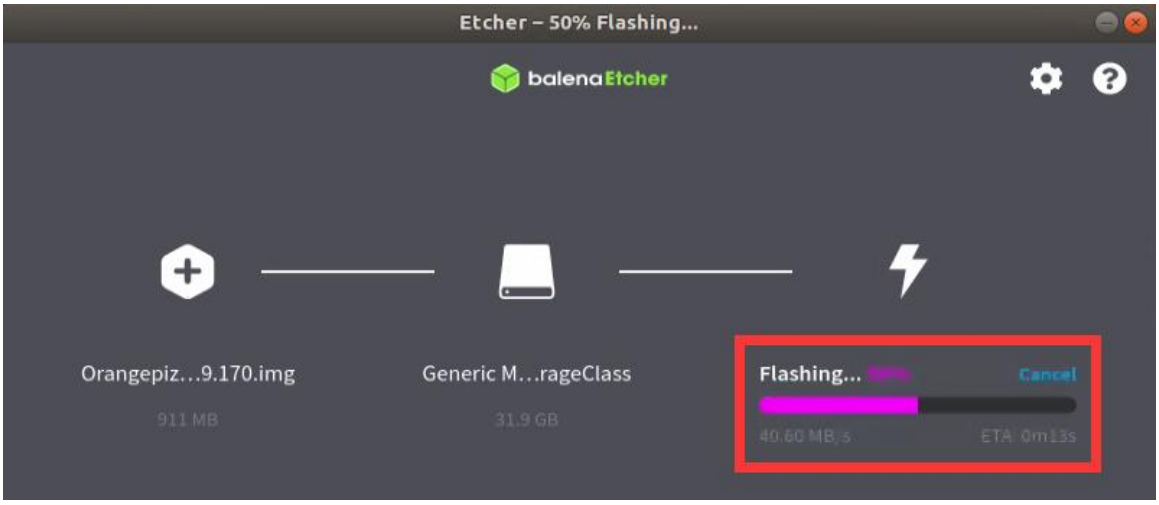
8) Then double-click **balenaEtcher-1.14.3-x64.AppImage** in the graphical interface of the Ubuntu PC to open balenaEtcher (**no installation required**). The interface of balenaEtcher after opening is as shown in the figure below



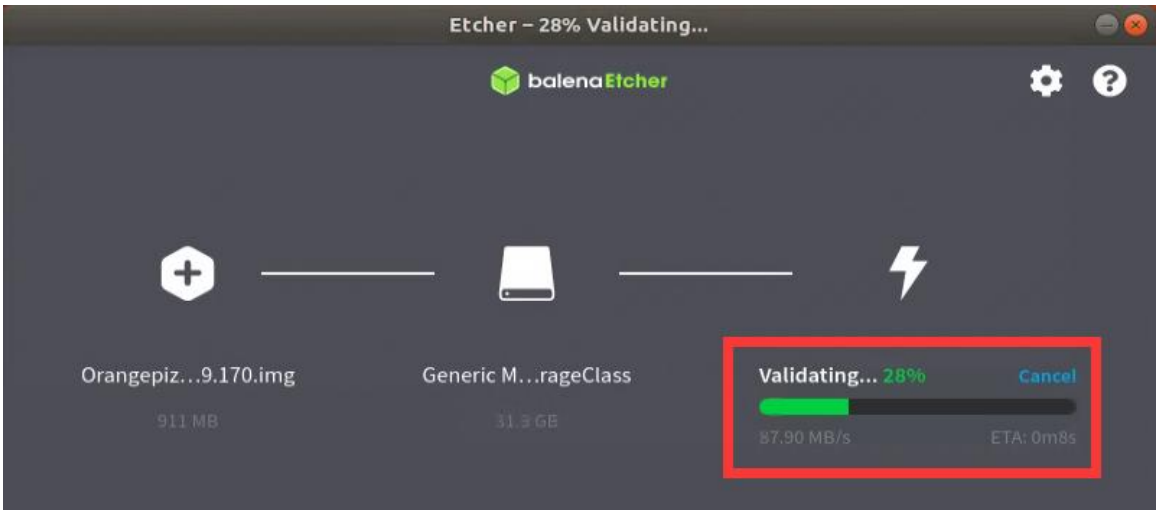
9) The specific steps for burning the Linux image using balenaEtcher are as follows

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

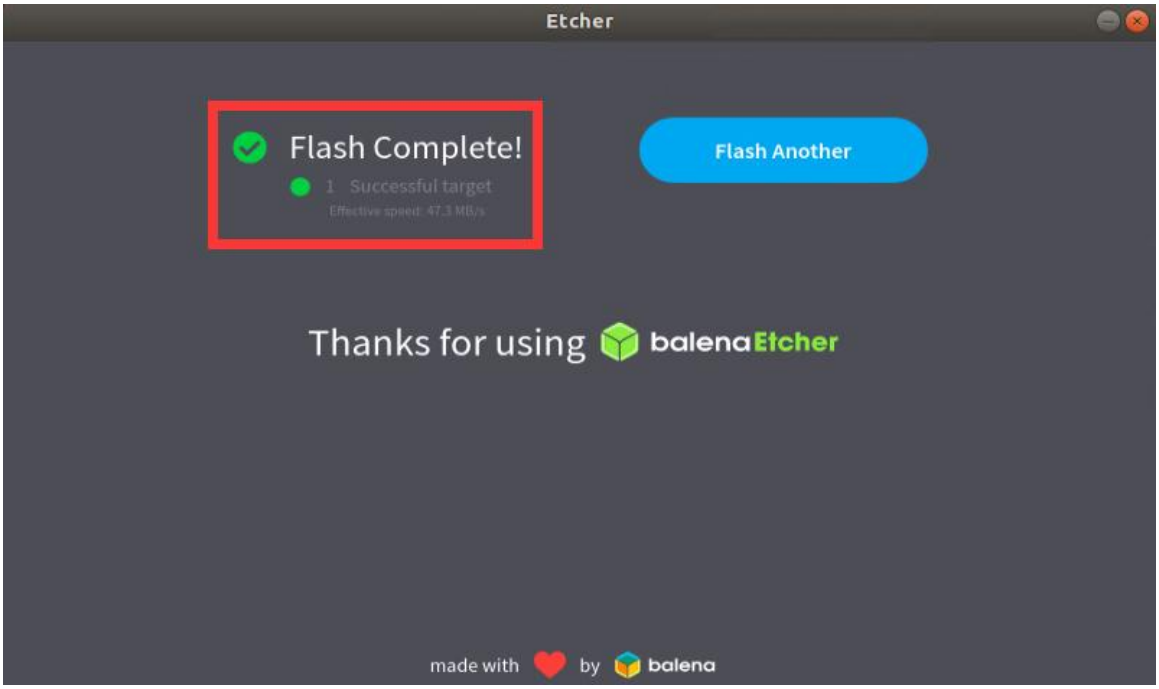
10) The interface displayed during the process of burning the Linux image in balenaEtcher is shown in the figure below. In addition, the progress bar is purple, indicating that the Linux image is being burned to the TF card.



11) After the Linux image is burned, balenaEtcher will verify the image burned to the TF card by default to ensure that there are no problems with 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 burning is completed successfully, the display interface of balenaEtcher is as shown below. If the green indicator icon is displayed, it means that the image is burned successfully. At this time, you can exit balenaEtcher, then remove the TF card and insert it into the TF card slot of the development board for use.



2. 5. How to burn Linux image to eMMC

See the method of burning Linux image to EMMC

2. 6. Method of burning Linux image to SPIFlash+NVMe SSD

See the method of burning Linux image to SPIFlash+NVMe SSD

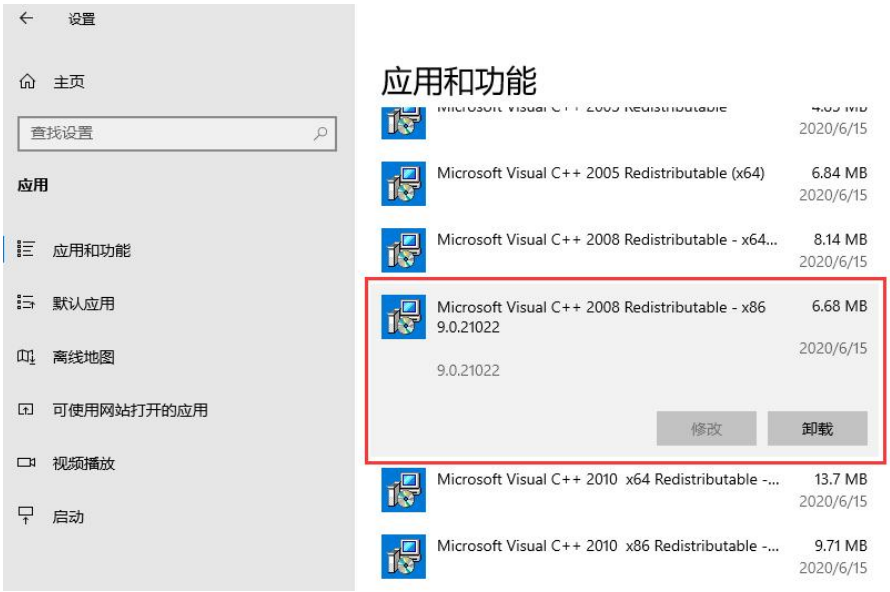
2. 7. How to burn Android image to TF card

The Android image of the development board can only be burned to a TF card using the PhoenixCard software on the Windows platform. The PhoenixCard software version must be PhonixCard-4.2.8.

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

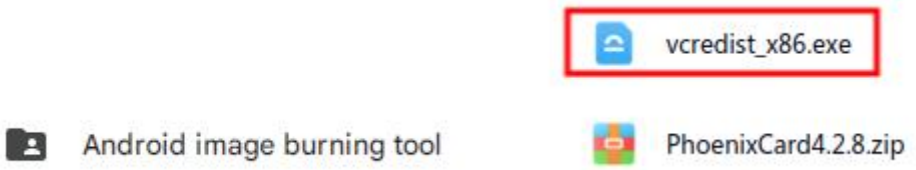
In addition, PhoenixCard does not have versions for Linux and Mac platforms, so it is impossible to burn Android images to TF cards on Linux and Mac platforms.

1) First, make sure that **Microsoft Visual C++ 2008 Redistrbutable - x86** is installed on your Windows system.



2) If **Microsoft Visual C++ 2008 Redistrbutable - x86** is not installed, the following error message will appear when formatting a TF card or burning an Android image using **PhoenixCard**:

3) The **Microsoft Visual C++ 2008 Redistrbutable - x86** installation package can be downloaded from the Orange Pi 4 Pro **official tools** or from the **Microsoft official website**.



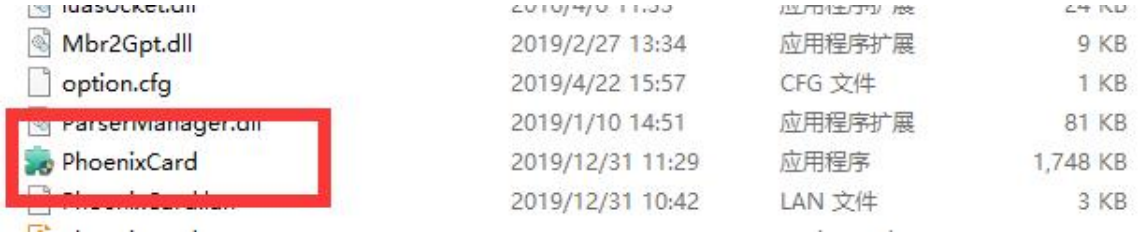
4) Then prepare a TF card with a capacity of 8GB or larger. The transmission speed of the TF card must be **class 10** or above. It is recommended to use a TF card from a brand such as SanDisk.

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

6) Download the Android image and PhoenixCard flashing tool from **Orange Pi's download page**. Please make sure the version of the PhoenixCard tool is **PhoenixCard-4.2.8. Do not use a version lower than 4.2.8 of the PhoenixCard software to flash the Android image**. Flashing an Android image with a version lower than this may cause problems.

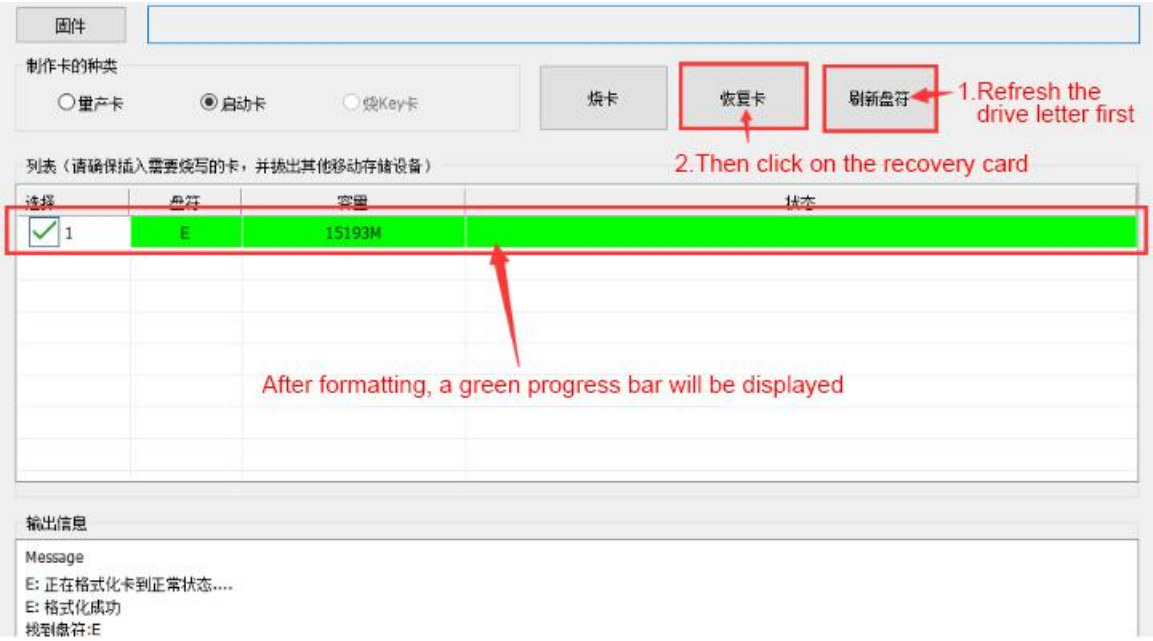
7) Use decompression software to decompress the downloaded Android image file. The file ending with "**.img**" is the Android image file, which is over 1GB in size. If you don't know how to decompress the Android image file, you can install **360 compression software** to decompress the image.

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



9) After opening PhoenixCard, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that the drive letter displayed is the same as the drive letter of the TF card you want to burn**. If it is not displayed, you can try to unplug the TF card or click the "**Refresh Drive Letter**" button in PhoenixCard.

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



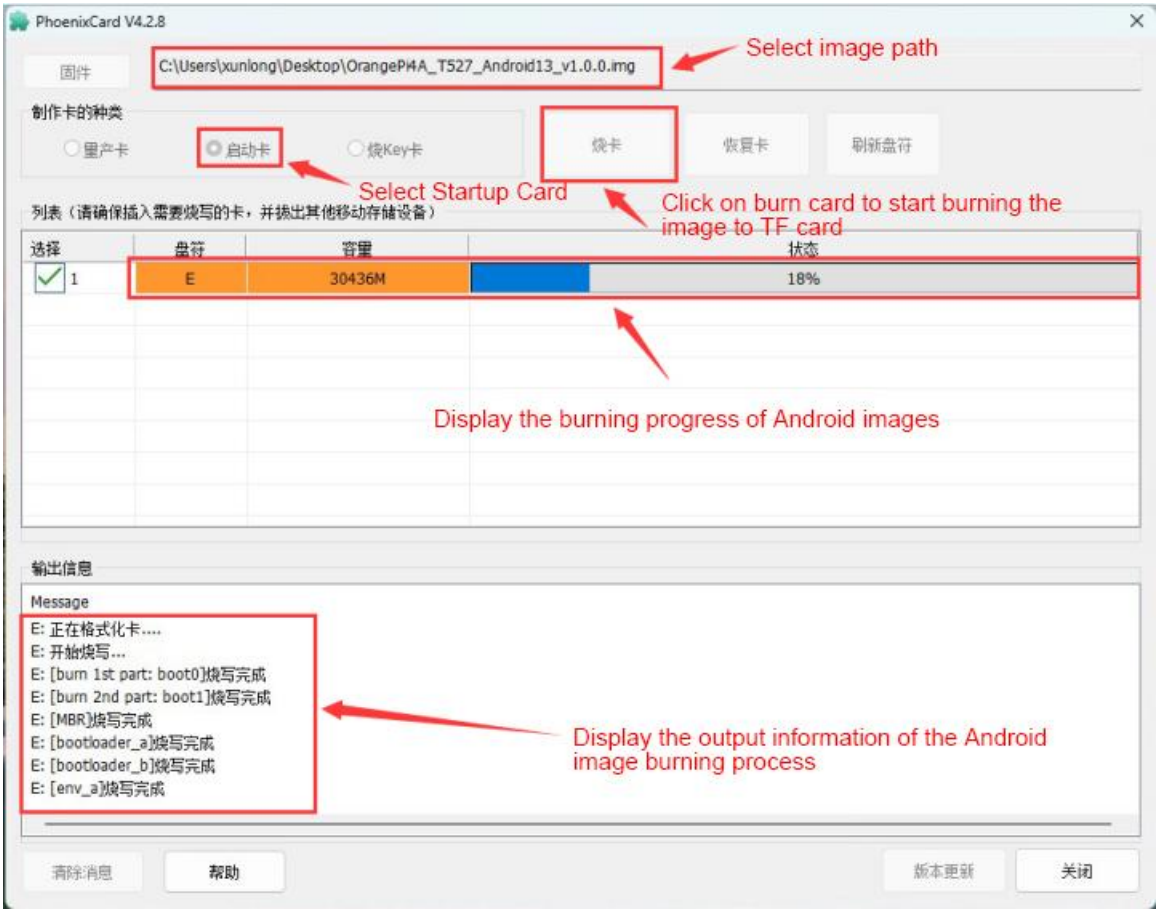
If there is a problem with formatting, please try to unplug and plug in the TF card and test again. If the problem still exists after plugging and plugging in the TF card, you can restart the Windows computer or try another computer and try again.

11) Then start writing the Android image to the TF card

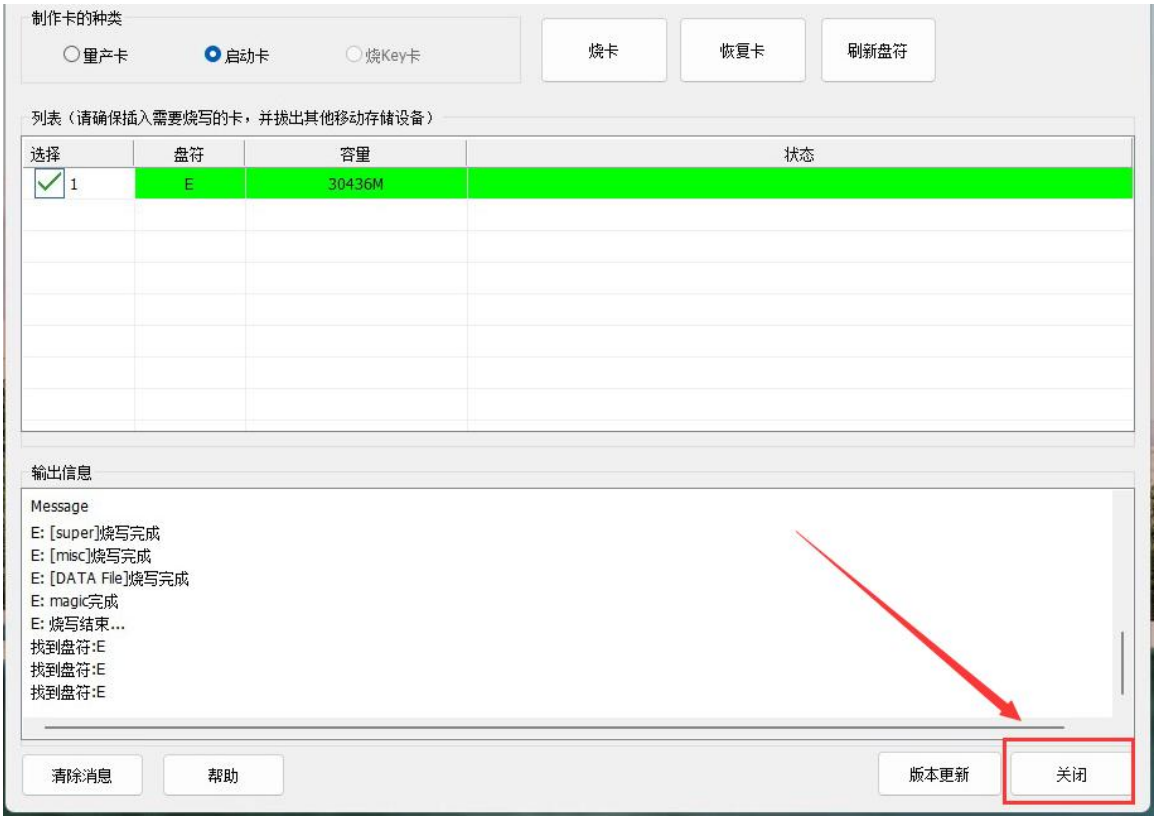
a. First, select the path of the Android image in the "**Firmware**" column

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

c. Then click the "Burn Card" button to start burning



12) After burning, the PhoenixCard display is as shown below. Click the "**Close**" button to exit PhoenixCard. Then you can remove the TF card from the computer and insert it into the development board to start.



After burning the Android system, only one 128 MB partition will be visible on the TF card in Windows, as shown in the figure below. (Some computers may pop up more than 20 disk partitions, but only the 128 MB partition will be visible.) Please note that this is normal and do not assume that the TF card is damaged. This is because the Android system has more than 20 partitions, but most of them cannot be properly recognized in the Windows system. At this time, please safely remove the TF card and insert it into the development board to boot.

After the Android system is started, use the following command to view the twenty or so partitions in the TF card：

```
console:/ # ls /dev/block/mmcblk0*
/dev/block/mmcblk0      /dev/block/mmcblk0p18  /dev/block/mmcblk0p27
/dev/block/mmcblk0p1    /dev/block/mmcblk0p19  /dev/block/mmcblk0p28
/dev/block/mmcblk0p10   /dev/block/mmcblk0p2   /dev/block/mmcblk0p3
/dev/block/mmcblk0p11   /dev/block/mmcblk0p20  /dev/block/mmcblk0p4
/dev/block/mmcblk0p12   /dev/block/mmcblk0p21  /dev/block/mmcblk0p5
/dev/block/mmcblk0p13   /dev/block/mmcblk0p22  /dev/block/mmcblk0p6
/dev/block/mmcblk0p14   /dev/block/mmcblk0p23  /dev/block/mmcblk0p7
/dev/block/mmcblk0p15   /dev/block/mmcblk0p24  /dev/block/mmcblk0p8
/dev/block/mmcblk0p16   /dev/block/mmcblk0p25  /dev/block/mmcblk0p9
/dev/block/mmcblk0p17   /dev/block/mmcblk0p26
```

Using the df -h command, you can see that after burning the Android system, there is about 24 GB of space left on the 32GB TF card (not all of the twenty or so partitions will be mounted in the Android system, so just focus on the visible partitions).

```
console:/ # df -h
Filesystem                Size Used Avail Use% Mounted on
tmpfs                     963M 1.2M  961M   1% /dev
tmpfs                     963M    0  963M   0% /mnt
/dev/block/dm-0           803M 803M     0 100% /
/dev/block/dm-4           232K  36K  196K  16% /system_dlkm
/dev/block/dm-1            88M  88M     0 100% /vendor
/dev/block/dm-3            11M  11M     0 100% /vendor_dlkm
/dev/block/dm-2           106M 106M     0 100% /product
tmpfs                     963M 8.0K  963M   1% /apex
tmpfs                     963M 488K  962M   1% /linkerconfig
/dev/block/mmcblk0p21      10M 120K   10M   2% /metadata
/dev/block/mmcblk0p22      80M  54M   26M  67% /treadahead
/dev/block/mmcblk0p26      16M    0   16M   0% /oem
/dev/block/mmcblk0p27      64M 4.0K   64M   1% /Reserve0
/dev/block/dm-5            26G 1.8G   24G   8% /data
tmpfs                     963M    0  963M   0% /data_mirror
/dev/fuse                  26G 1.8G   24G   8% /mnt/user/0/emulated
/dev/block/vold/public:179,1 128M  18M  109M  15% /mnt/media_rw/extsd
/dev/fuse                 128M  18M  109M  15% /mnt/user/0/extsd
```

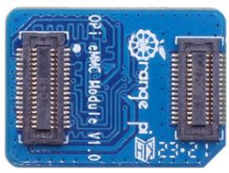
2.8. How to burn Android image to eMMC

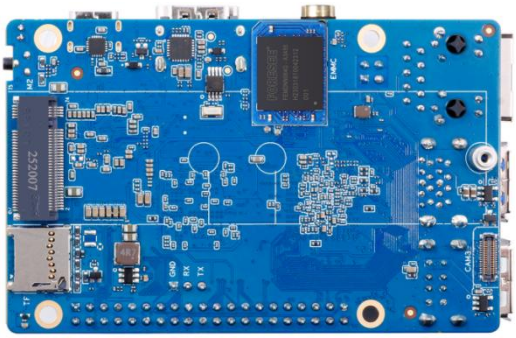
The Android image of the development board can only be burned to the eMMC using the PhoenixCard software on the Windows platform. The PhoenixCard software version must be PhonixCard-4.2.8.

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

In addition, PhoenixCard software does not have versions for Linux and Mac platforms, so it is impossible to burn Android images to eMMC on Linux and Mac platforms.

1) The development board has a reserved expansion interface for the eMMC module. Before burning the system to the eMMC, you first need to purchase an eMMC module that matches the eMMC interface of the development board. Then install the eMMC module on the development board. The eMMC module and the method of inserting it into the development board are shown below：

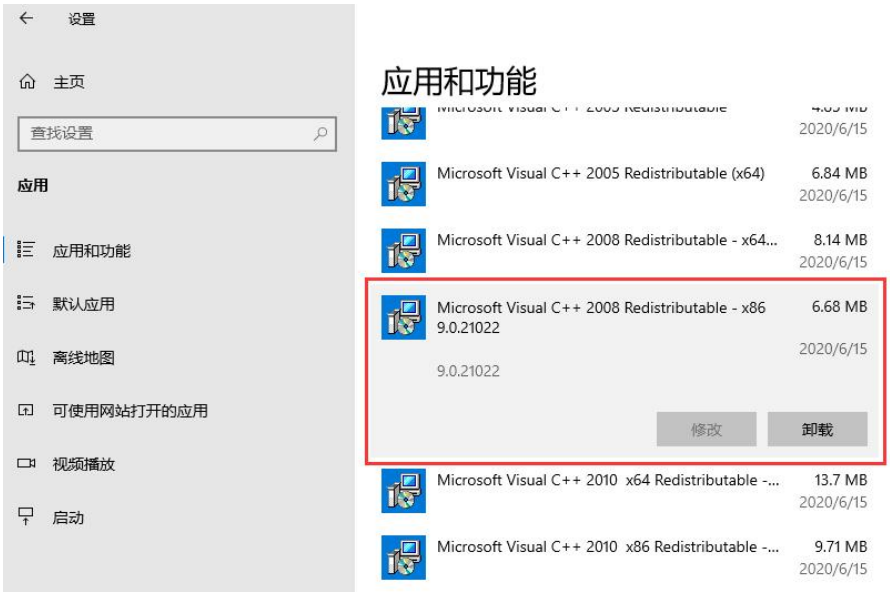




2) First of all, please note that this method requires the use of a TF card to complete, mainly divided into the following two steps

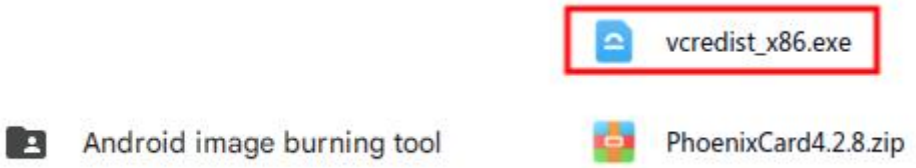
a. First use PhoenixCard to burn the Android firmware into the TF card in the form of a mass production card
b. Then use the TF card to burn the Android firmware to the eMMC

3) Please make sure that **Microsoft Visual C++ 2008 Redistrbutable - x86** is installed on your Windows system.



4) If **Microsoft Visual C++ 2008 Redistrbutable - x86** is not installed ,When using **PhoenixCard** to format a TF card or burn an Android image, the following error message will be displayed:

5) The installation package for **Microsoft Visual C++ 2008 Redistrbutable - x86** can be downloaded from the Orange Pi 4 Pro **official tools** or from the **Microsoft official website**.



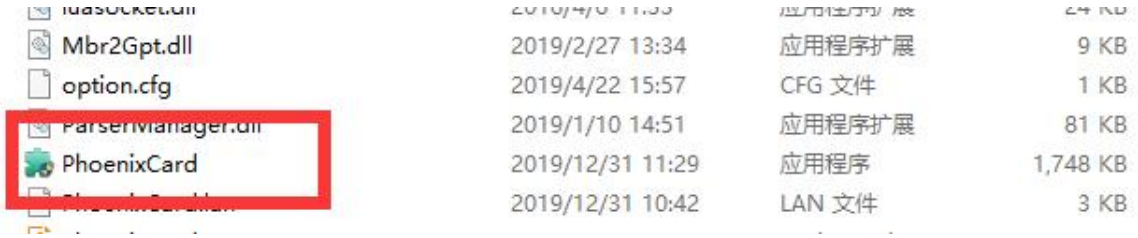
6) Then prepare a TF card with a capacity of 8GB or larger. The transmission speed of the TF card must be **class 10** or above. It is recommended to use a TF card from a brand such as SanDisk.

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

8) Download the Android image and PhoenixCard flashing tool from **Orange Pi's data download page**. Please make sure the version of the PhoenixCard tool is **PhoenixCard-4.2.8. Do not use a version lower than 4.2.8 of the PhoenixCard software to flash the Android image**. Flashing an Android image with a version lower than this may cause problems.

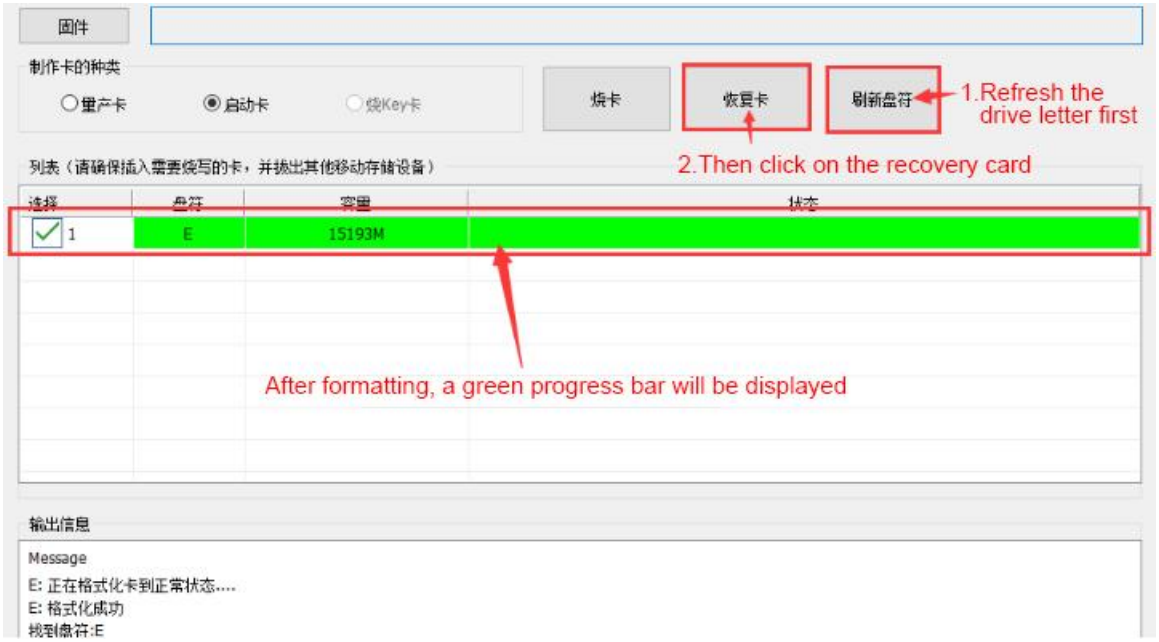
9) Use decompression software to decompress the downloaded Android image file. The file ending with "**.img**" is the Android image file, which is over 1GB in size. If you don't know how to decompress the Android image file, you can install **360 compression software** to decompress the image.

10) Then use the decompression software to decompress **PhonixCard4.2.8.zip**. This software does not need to be installed. Find PhoenixCard in the decompressed folder and open it.



11) After opening PhoenixCard, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that the drive letter displayed is the same as the drive letter of the TF card you want to burn**. If it is not displayed, you can try to unplug the TF card or click the "**Refresh Drive Letter**" button in PhoenixCard.

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



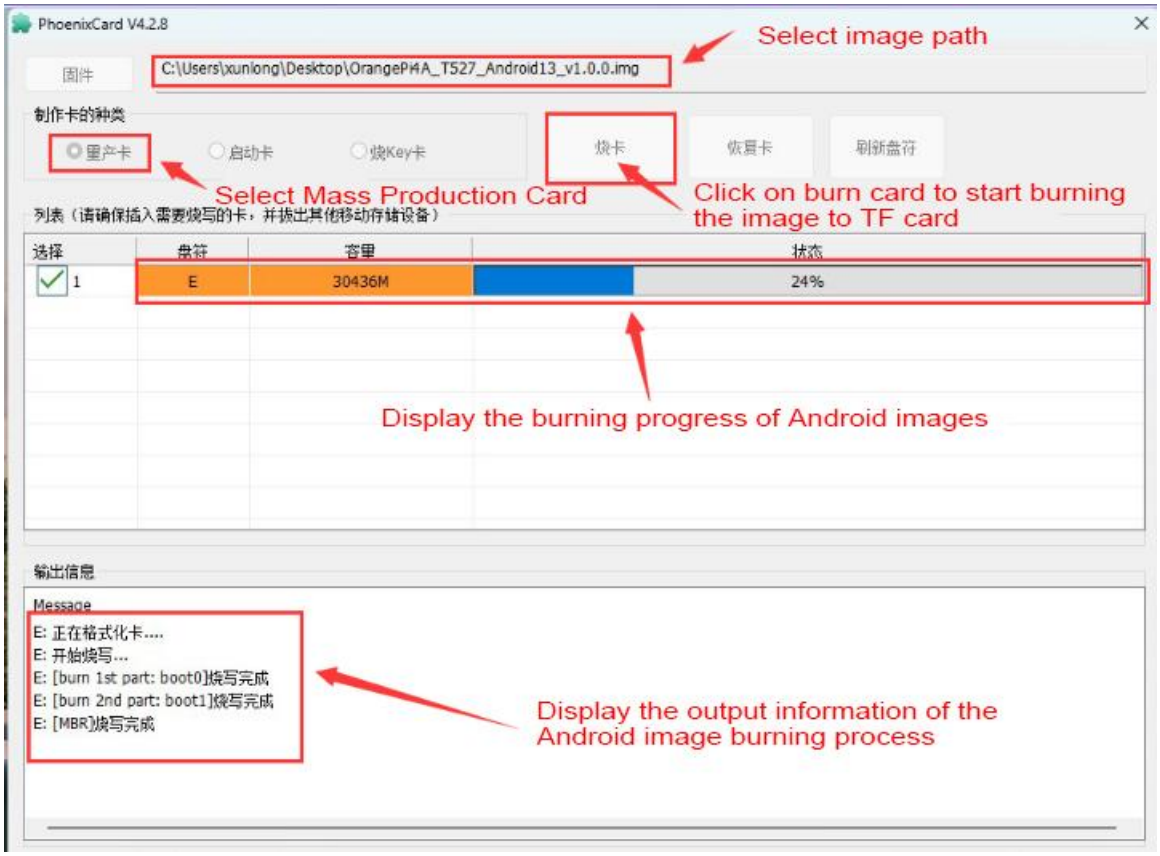
If there is a problem with formatting, please try to unplug and plug in the TF card and test again. If the problem still exists after plugging and plugging in the TF card, you can restart the Windows computer or try another computer and try again.

13) Then start writing the Android image to the TF card

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

b. Select "**Mass Production Card**" in "**Card Type**"

c. Then click the "Burn Card" button to start burning



14) After burning, the PhoenixCard display will be as shown below. Click the "**Close**" button to exit PhoenixCard.

15) Then insert the TF card into the development board. After powering on the development board, it will automatically burn the Android firmware in the TF card to the eMMC of the development board. After the burning is completed, it will automatically shut down and the LED light on the development board will go out.

16) At this point, you can remove the TF card and then power on again, and the Android system in the eMMC will start.

2. 9. Start the Orange Pi Development Board

1) Insert the TF card with the burned image into the TF card slot of the Orange Pi development board.

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

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

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

5) Connect a **high-quality** power adapter with a 5V/3A (5V/4A is also acceptable) USB Type C port.

Remember not to plug in a power adapter with a voltage output greater than 5V, as this will burn out the development board.

Many unstable issues during system startup are often caused by power supply problems, so a reliable power adapter is essential. If you experience repeated restarts during startup, replace the power supply or Type-C cable and try again.

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

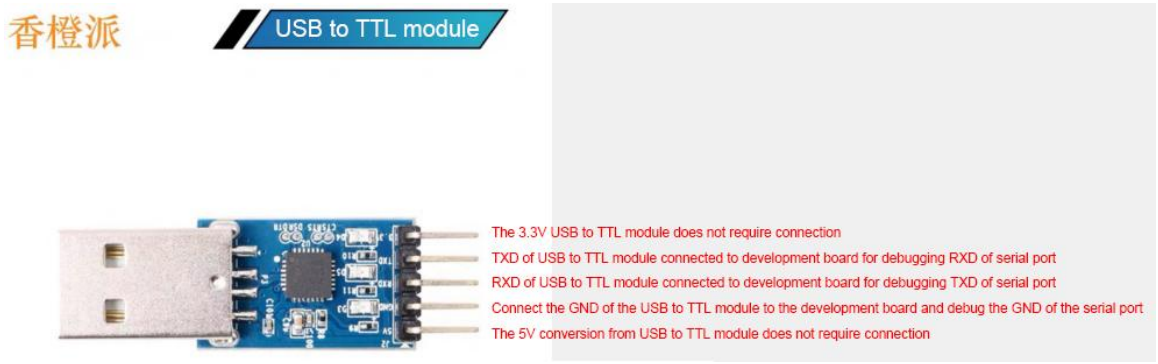
7) If you want to view the system output information through the debug serial port, please use a serial cable to connect the development board to the computer. For serial port

connection methods, please refer to the section "**How to use the debug serial port**"

2. 10. How to use the debug serial port

2. 10. 1. Debug serial port connection instructions

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

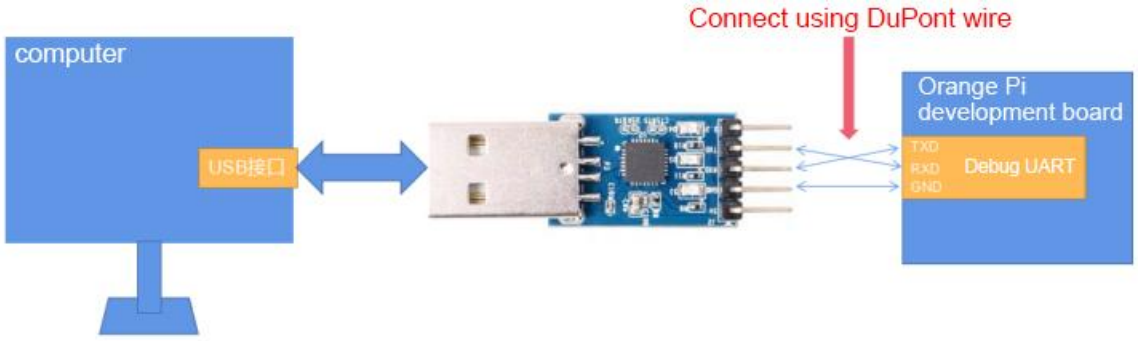


2) The corresponding relationship between the debug serial port GND, TX and RX pins of the development board is shown in the figure below.

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 via a DuPont line.

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

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



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

The TX and RX of the serial port need to be cross-connected. If you don't want to carefully distinguish the order of TX and RX, you can connect the TX and RX of the serial port randomly first. If there is no output when testing the serial port, then swap the order of TX and RX. In this way, there will always be one order that is correct.

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

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

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

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

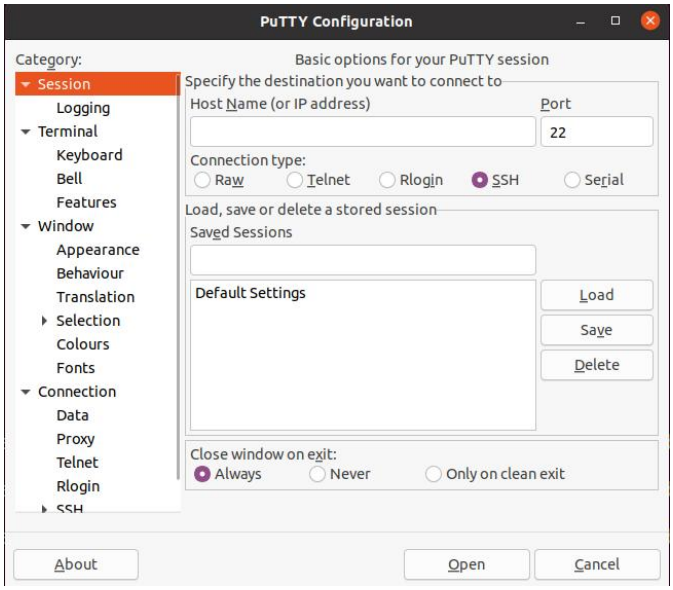
2) Then use the following command to install putty on the Ubuntu PC

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

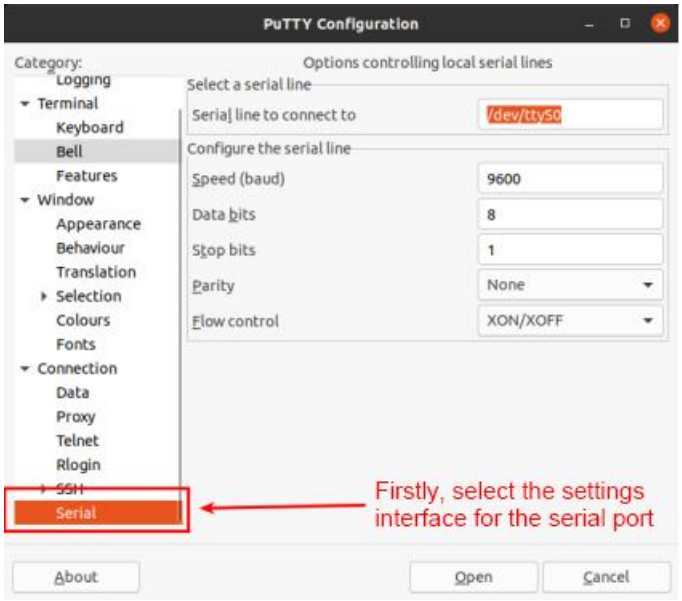
3) Then run putty, **remember to add sudo permissions**

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

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



5) First select the serial port setting interface

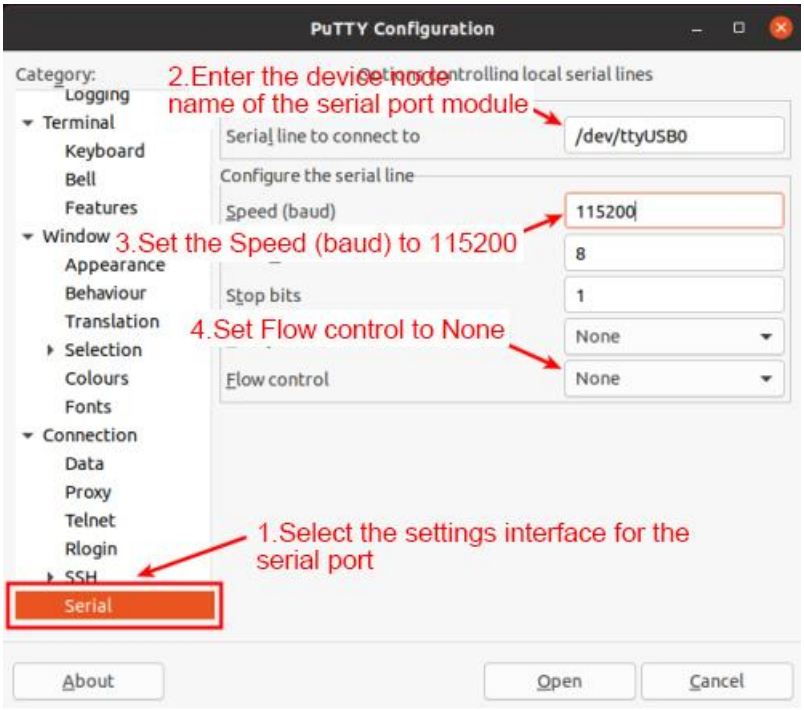


6) Then set the serial port parameters

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

b. Set **Speed(baud)** to 115200 (the baud rate of the serial port)

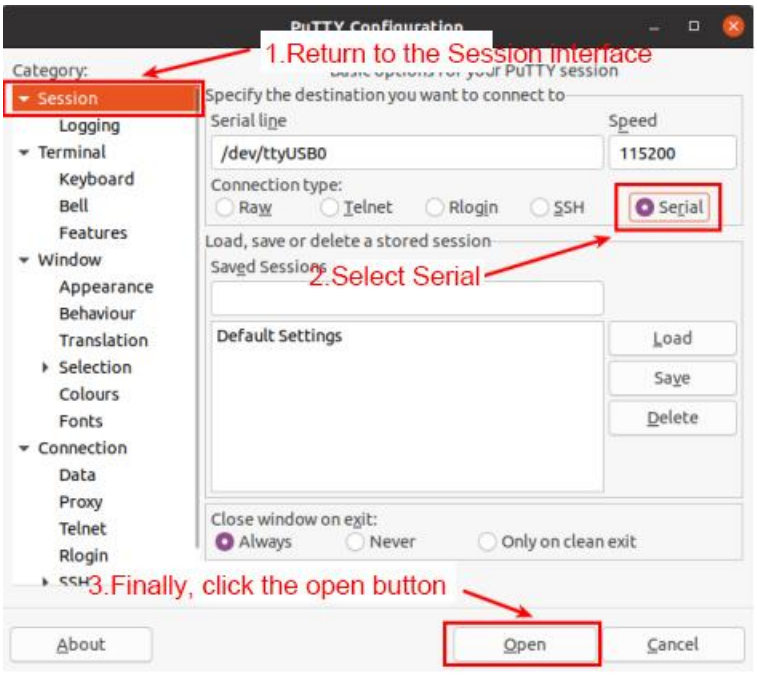
c. Set **Flow control** to **None**



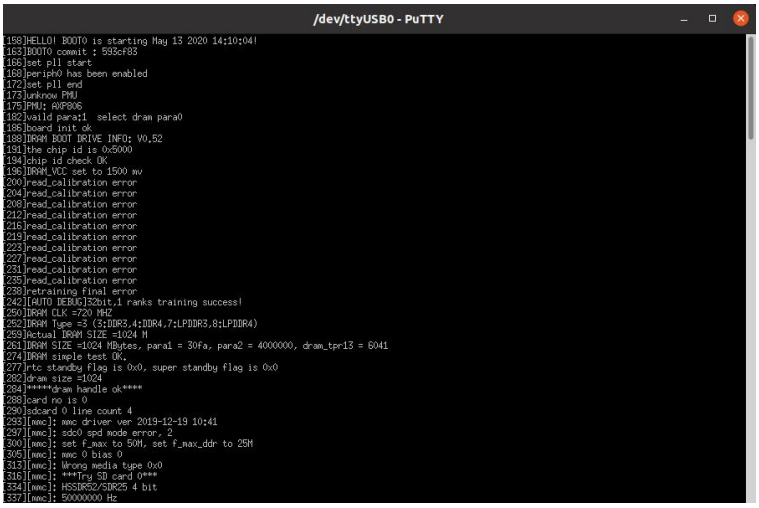
7) After setting up the serial port settings, return to the Session interface.

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

b. Then click the **Open** button to connect the serial port



8) Then start the development board, and you can see the log information output by the system from the opened serial terminal.



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

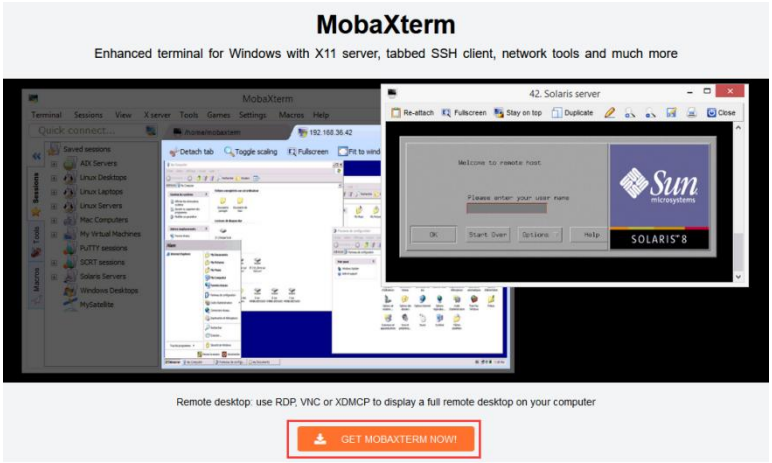
There are many serial port debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following demonstrates how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

1) Download MobaXterm

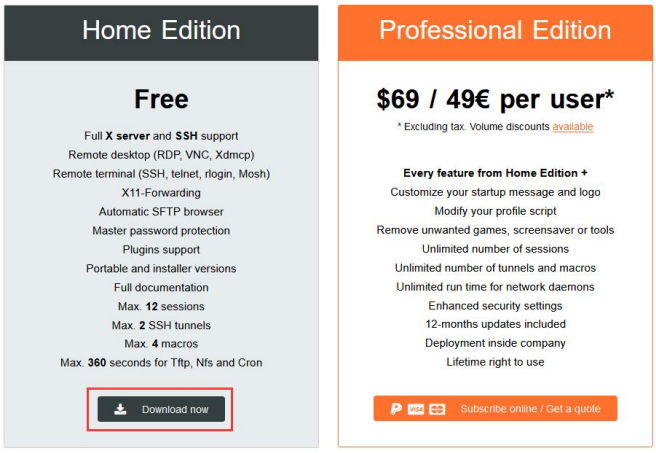
a. Download MobaXterm from the following URL

https://mobaxterm.mobatek.net/

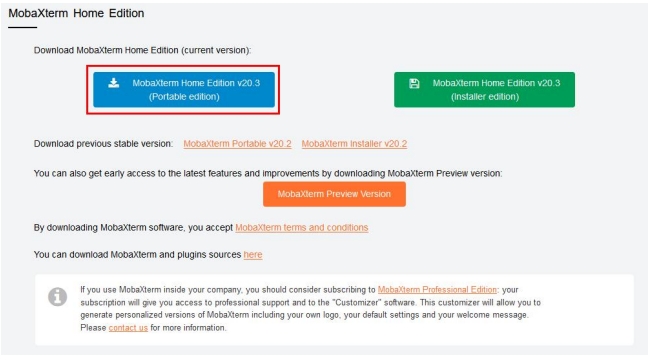
b. Go to the MobaXterm download page and click **GET XOBATERM NOW!**



c. Then choose to download the Home version



d. Then select the portable version. After downloading, you don’t need to install it. You can use it directly by opening it.



2) Then select the portable version. After downloading, you don’t need to install it. You can use it directly by opening it.

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

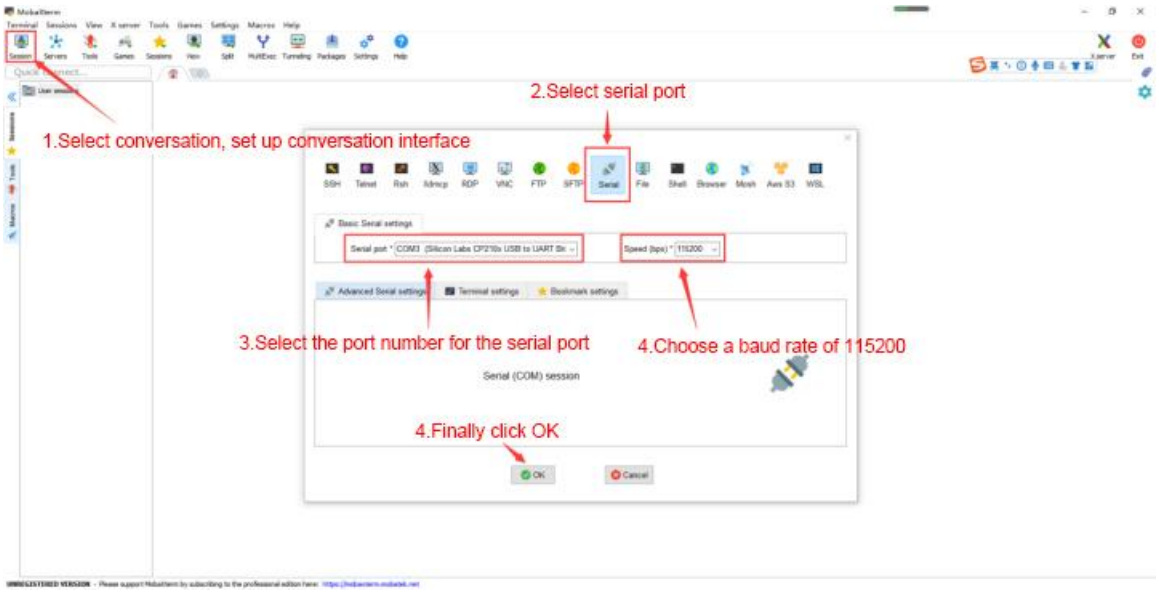
a. Open the session settings interface

b. Select the serial port type

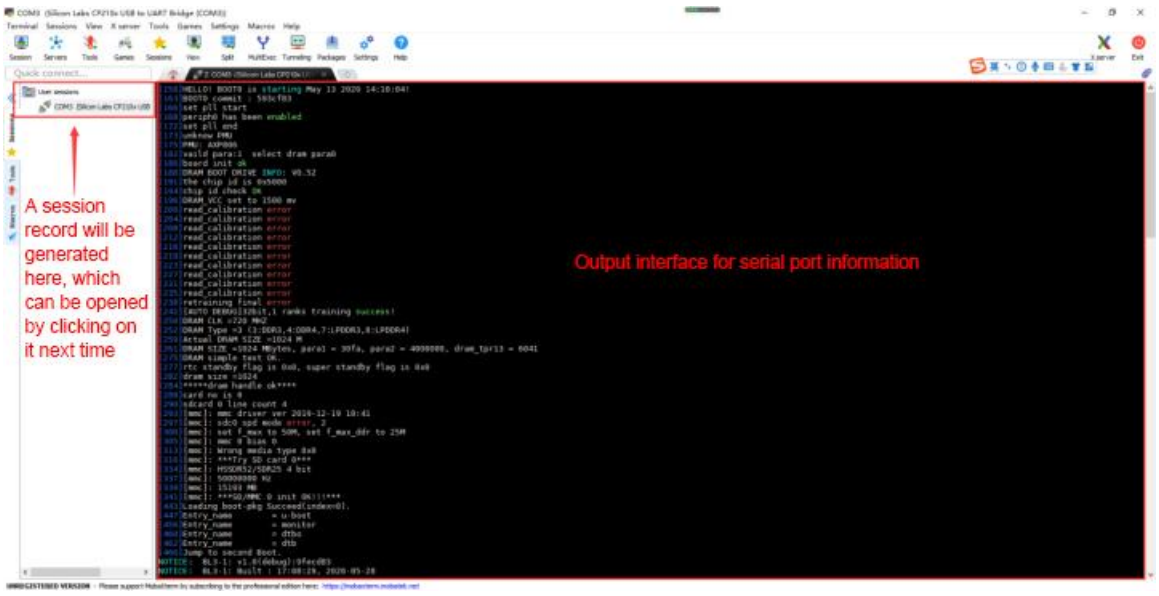
c. Select the serial port number (select the corresponding port number according to the actual situation). If you cannot see the port number, please use **360 Driver Master** to scan and install the USB to TTL serial port chip driver.

d. Select the serial port baud rate as **115200**

e. Finally, click the **"OK"** button to complete the settings



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



2. 11. Instructions for using the 5V pin in the 40-pin interface of the development board to power

We recommend using a 5V/5A Type-C power cable plugged into the board's Type-C power connector to power the board. If you need to use the 5V pin on the 40-pin connector to power the board, ensure that the power cable you use meets the board's power requirements. If you experience instability, switch back to a Type-C power supply.

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

The power cord shown in the picture above can be purchased on Taobao. Please search and purchase it yourself.

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

a. The USB A port of the power cable shown in the figure above needs to be plugged into the 5V/3A power adapter connector **(it is not recommended to plug it into the USB port of the computer for power supply. If too many peripherals are connected to the development board, it will be unstable**).
b. The red DuPont line needs to be plugged into the 5V pin of the 40-pin interface on the development board
c. The black DuPont line needs to be plugged into the GND pin of the 40-pin interface
d. The positions of the 5V and GND pins of the 40-pin interface on the

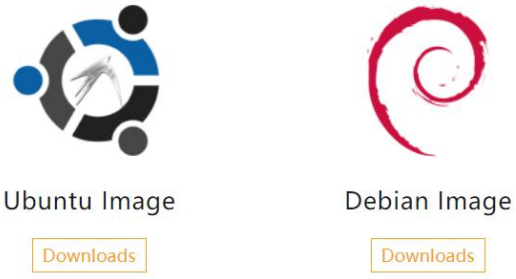
development board are shown in the figure below. **Be sure not to connect them in reverse**.

3. Debian/Ubuntu Server and Xfce Desktop System Instructions

3. 1. Supported Linux image types and kernel versions

Linux image types	Kernel versions	Desktop version
Ubuntu 22.04 - Jammy	Linux5.15	support
Debian 12 - Bookworm	Linux5.15	support
Debian 11 - Bullseye	Linux5.15	support

On the **Orange Pi download page**, enter the download page for the corresponding development board and you'll see the following download options. In the following descriptions, **Ubuntu and Debian images are generally referred to as Linux images.**



The naming rules for Linux images are:

Development board model_version number_Linux distribution type_distribution

code_server or desktop_kernel version

a. **Development board model:** All are **OrangePI4Pro**. Different development boards generally have different model names. Before burning the image, please ensure that the model name of the selected image matches the development board.

b. **Version number:** For example, **1.x.x.** This version number will increase as the image function is updated. In addition, the last digit of the version number of the development board Linux image is always an even number.

c. **Linux distribution type:** Currently, **Ubuntu** and **Debian** are supported. Since Ubuntu is derived from Debian, the two operating systems generally differ in usage. However, there are some minor differences in the default configurations of some software and the usage of commands. Ubuntu and Debian each maintain their own supported software repositories, resulting in slight differences in supported installable packages. A thorough understanding of these requires personal experience. For more details, please refer to the official documentation provided by Ubuntu and Debian.

d. **Release codenames:** are used to distinguish different versions of specific Linux distributions, such as Ubuntu or Debian. **Jammy** is an Ubuntu release, representing Ubuntu 22.04. The biggest difference between versions is that newer Ubuntu releases often maintain repositories with newer software, such as Python and the GCC compilation toolchain. **Bookworm** is a specific Debian release codename, representing Debian 12.

e. **Server or Desktop:** This indicates whether the system comes with a desktop environment. If "**server**" is selected, it means the system does not have a desktop environment installed. The image occupies less storage space and resources, and the system is primarily controlled using the command line. If "**desktop_gnome**" is selected, the system comes with the GNOME desktop environment installed by default. The image occupies more storage space and resources, and the system can be operated through a monitor, mouse, and keyboard. Of course, the desktop version of the system can also be operated through the command line, just like the server version.

f. **Kernel version:** used to indicate the version number of the Linux kernel, currently supporting **Linux 5.15**.

3. 2. Linux kernel driver adaptation

Function	Linux5.15
HDMI Video	OK
HDMI Audio	OK
USB2.0 x 3	OK
USB3.0 x 1	OK
TF card Boot	OK
eMMC	OK
NVME SSD Identification	OK
SPI Flash	OK
Gigabit Network	OK
WIFI	OK
Bluetooth	OK
Headphone Audio	OK
Onboard MIC	OK
Speaker Audio	OK
LCD Screen	OK
CAM1	Driver OK, 3A not compatible
CAM2	Driver OK, 3A not compatible
LED Light	OK
40 pin GPIO	OK
40 pin I2C	OK
40 pin SPI	OK
40 pin UART	OK
40 pin PWM	OK
Power Button	OK
Temperature Sensor	OK
Hardware Watchdog	OK
GPU	Only supported on Debian Bullseye
NPU	OK
Video Codec	Only supported on Debian

	Bullseye

3. 3. Linux command format description in this manual

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

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

2) Description of the prompt type before the command

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

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

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

c. **root@orangepi:~#** The prompt indicates that this command is entered in **the Linux system** of the development board. The # at the end of the prompt indicates that the current user of the system is the root user and can execute any command he wants.

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

e. **root@test:~#** The prompt indicates that this command is entered in an Ubuntu PC or Ubuntu virtual machine, not in the Linux system of the development board. The # at the end of the prompt indicates that the current user of the system is the

root user and can execute any command he wants.

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

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

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

b. As shown below, some commands cannot be written on one line and will be placed on the next line. The bold parts are the commands that need to be entered. When these commands are entered on one line, the "\" at the end of each line needs to be removed. This is not part of the command. In addition, there are spaces between different parts of the command, so please do not miss them.

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

3. 4. Linux system login instructions

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

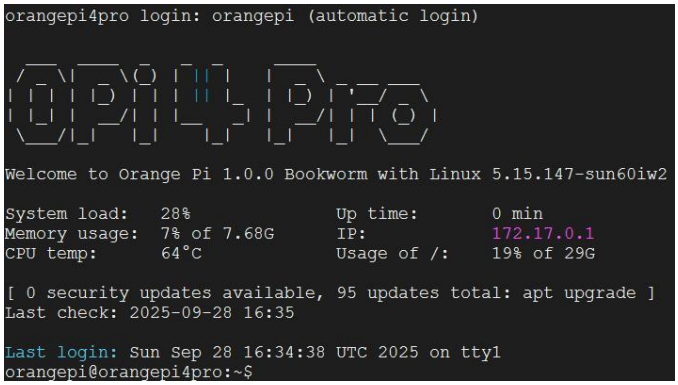
Account	Password
root	orangepi
orangepi	orangepi

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

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

3. 4. 2. How to set up automatic login for Linux system terminal

1) The default login user name for the Linux system is **orangepi**



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

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

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

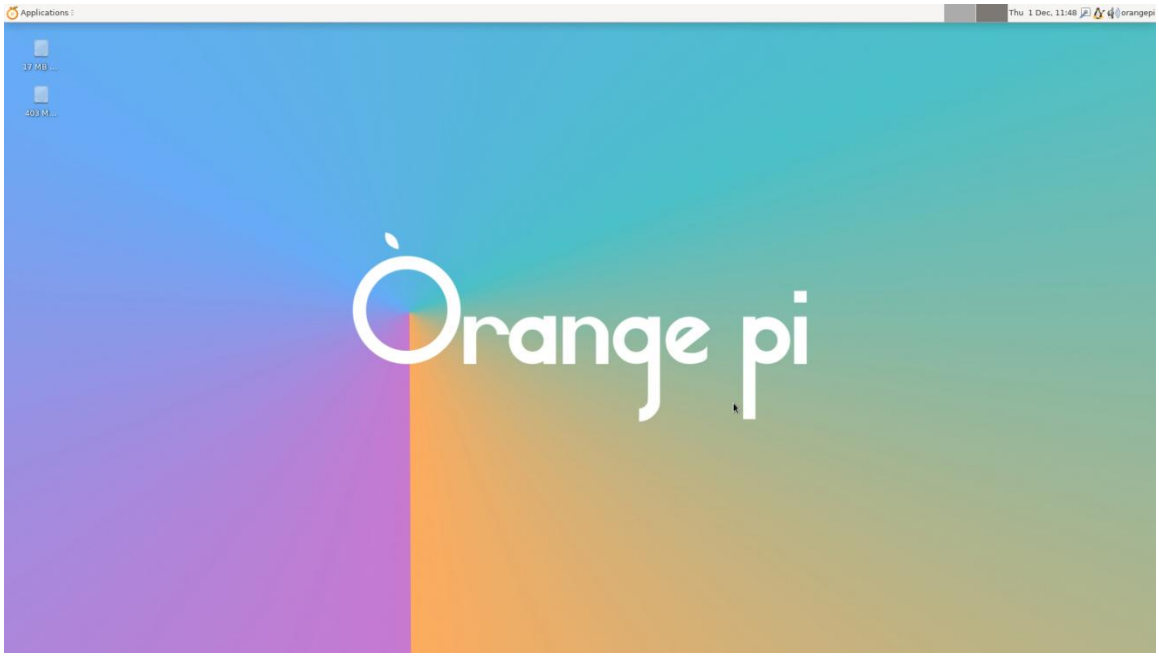
```
orangepi@orangepi:~$ sudo auto_login_cli.sh -d
```

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

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

3. 4. 3. Linux desktop system automatic login instruction

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



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

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

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

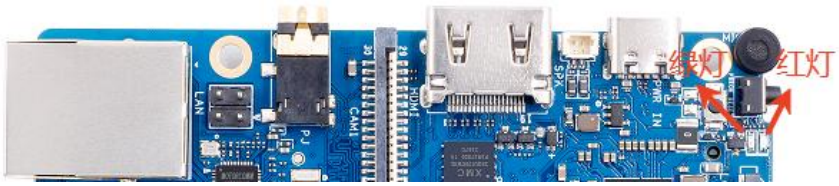
2) Then restart the system and you will automatically log in to the desktop as the root user

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

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

3. 5. Onboard LED Light Test Instructions

1) There are two LED lights on the development board, one green and one red. Their positions are shown in the figure below:



2) **As long as the development board is powered on, the red LED will be on. This is controlled by hardware and cannot be turned off by software.**

3)

4) The green LED light will keep flashing after the kernel starts, which is controlled by software.

5) The method for setting the green light on and off and flashing is as follows:

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

a. First enter the green light settings directory

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

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

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

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

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

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

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

3. 6. Instructions for the Linux system rootfs partition capacity in the TF card

3. 6. 1. The capacity of the rootfs partition in the TF card will be automatically expanded when it is first started

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

Note that not doing this step will not affect the automatic expansion of the Linux system on the development board. This is just to explain how to check the capacity

of the TF card after burning the Linux image.

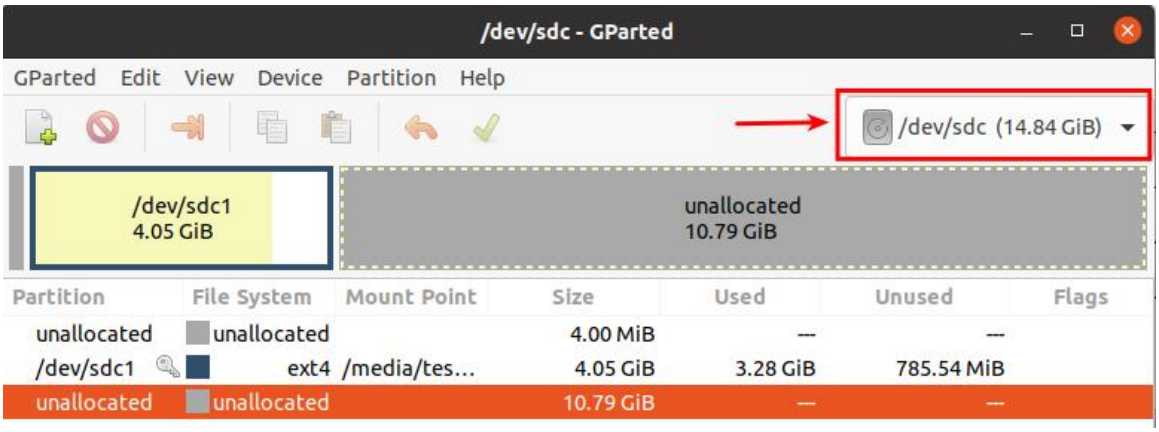
a. First install the gparted software on the Ubuntu computer

```
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 TF card after burning the Linux desktop system. You can see that although the total capacity of the TF card is 16GB (displayed as 14.84GiB in GParted), the rootfs partition (/dev/sdc1) is actually only allocated 4.05GiB, leaving 10.79GiB unallocated.

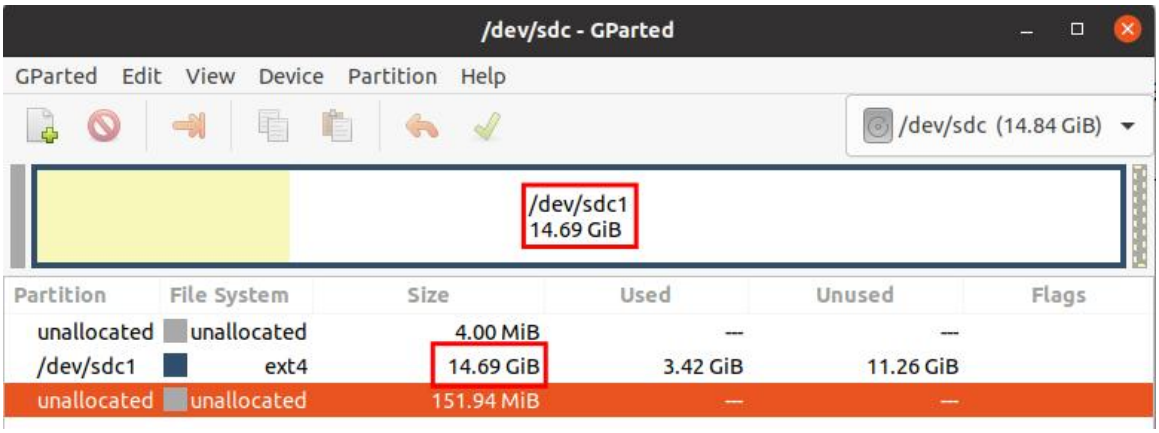
2) Then you can insert the TF card with the Linux system burned into the development board to start it. When the TF card starts the Linux system for the first time, it will call the **orangepi-resize-filesystem** script through the systemd service **orangepi-resize-filesystem.service**to automatically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

3) After logging into 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 to check the status of the TF card again. As shown in the figure below, the capacity of 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 kernel images and other files, so there is no problem of expanding the BOOT partition.

3.6.2. How to disable automatic expansion of the rootfs partition capacity in the TF card

1) First, burn the Linux image of the development board to the TF card in **the Ubuntu computer (not Windows)**, and then **unplug and replug the TF card**.卡

2) First, burn the Linux image of the development board to the TF card in the Ubuntu computer (not Windows), and then unplug and replug the TF card.

```
test@test:~$ ls /media/test/opi_root/
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 root user in 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/opi_root/
root@test:/media/test/opi_root/# cd root
root@test:/media/test/opi_root/root# touch .no_rootfs_resize
root@test:/media/test/opi_root/root# ls .no_rootfs*
.no_rootfs_resize
```

5) Then you can uninstall the TF card, unplug the TF card and insert it into the development board to start. When the Linux system starts, if it detects the file **.no_rootfs_resize** in the **/root**directory, it will no longer automatically expand the rootfs.

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

```
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 expand the capacity of the rootfs partition in the TF card, just execute the following command and then restart the Linux system of the development board.

Note: Please execute the following commands as the root user.

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

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

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

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

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

If the total capacity of the TF card is large, for example, 128GB, and you do not want the Linux system rootfs partition to use all of the TF card capacity, but only want to allocate a portion of the capacity, for example, 16GB, for the Linux system, and then the remaining capacity of the TF card can be used for other purposes, then you can use the information in this section to manually expand the capacity of the rootfs partition in the TF card.

1) First, burn the Linux image of the development board to the TF card in **the Ubuntu computer (not Windows)**, and then **unplug and replug the TF card.**

2) Then the Ubuntu computer will usually automatically mount the TF card partition. If the automatic mounting is normal, you can use the ls command to see the following output

```
test@test:~$ ls /media/test/opi_root/
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 root user in Ubuntu computer

```
test@test:~$ sudo -i
[sudo] test 的密码：
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/opi_root/
root@test:/media/test/opi_root/# cd root
root@test:/media/test/opi_root/root# touch .no_rootfs_resize
root@test:/media/test/opi_root/root# ls .no_rootfs*
.no_rootfs_resize
```

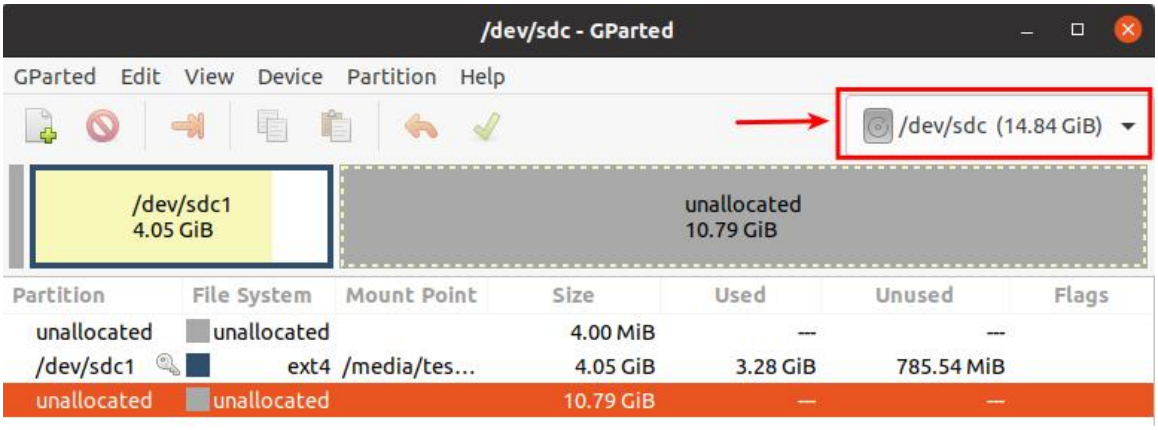
5) Then install the gparted software in the Ubuntu computer

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

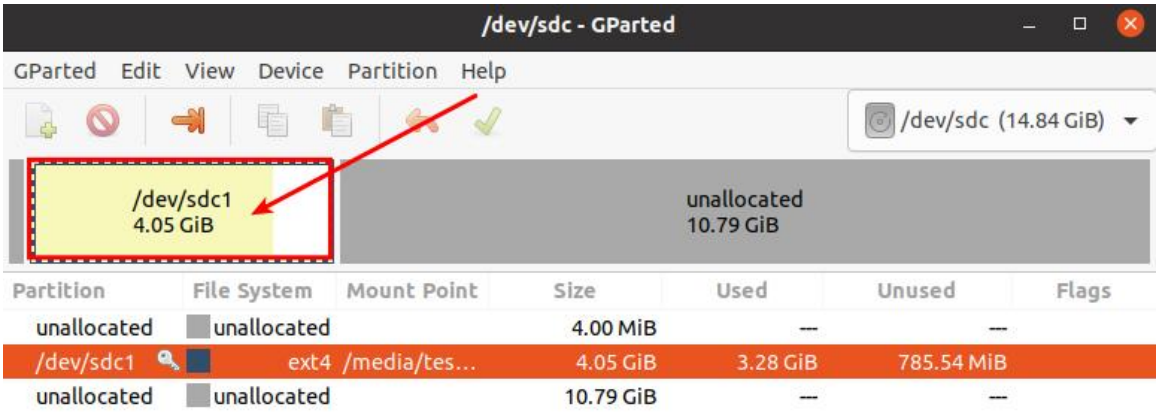
6) Then open gparted

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

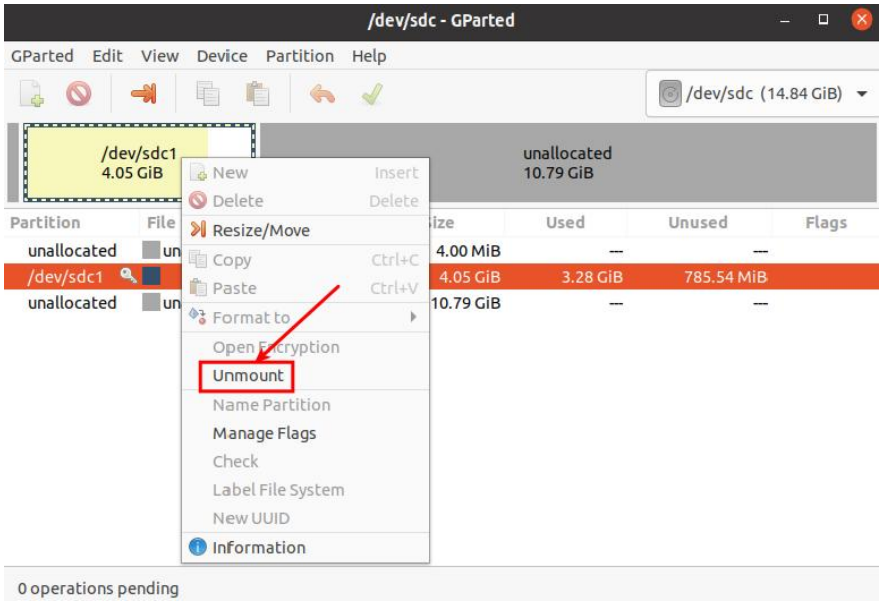
7) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity. The figure below shows the TF card after burning the Linux desktop system. You can see that although the total capacity of the TF card is 16GB (displayed as 14.84GiB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



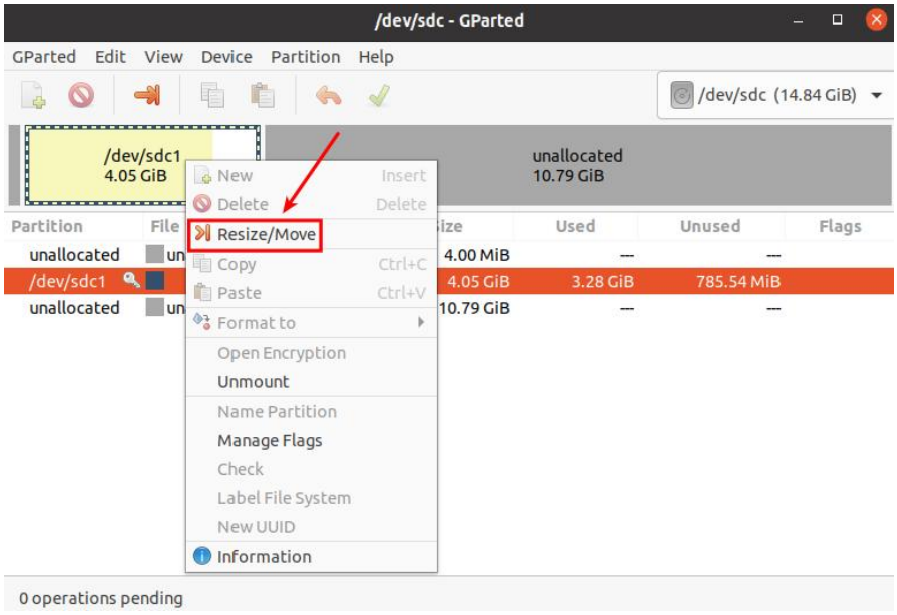
8) Then select the rootfs partition (/dev/sdc1)



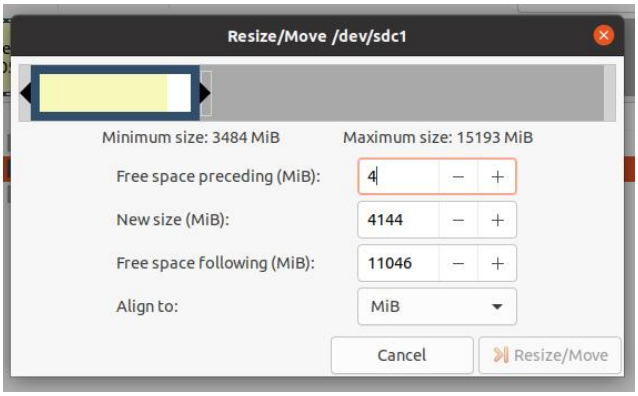
9) Click the right button of the mouse again to see the operation options shown in the figure below. If the TF card has been mounted, you need to Umount the rootfs partition of the TF card first.



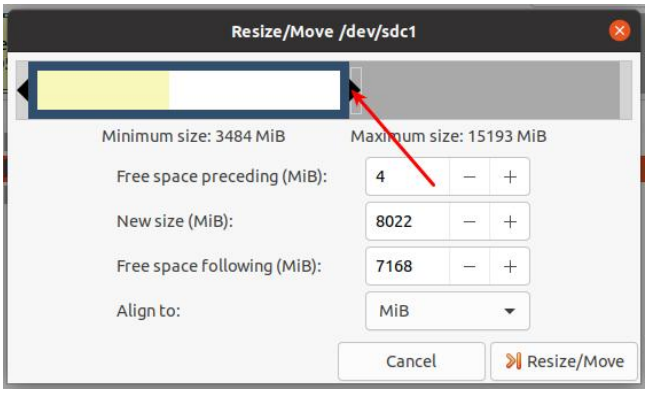
10) Then select the rootfs partition again, right-click, and select **Resize/Move** to start expanding the size of the rootfs partition



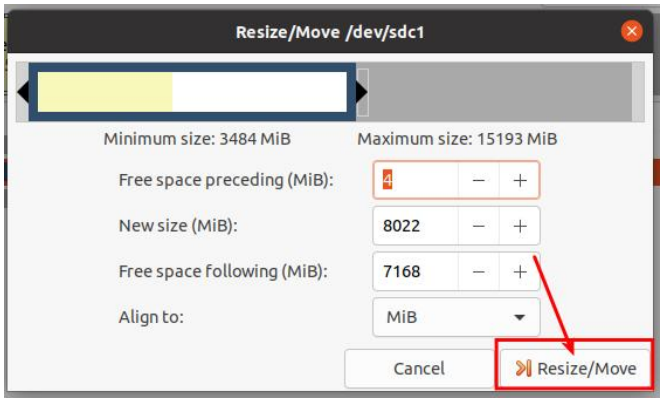
11) After the **Resize/Move** option is turned on, the following setting interface will pop up



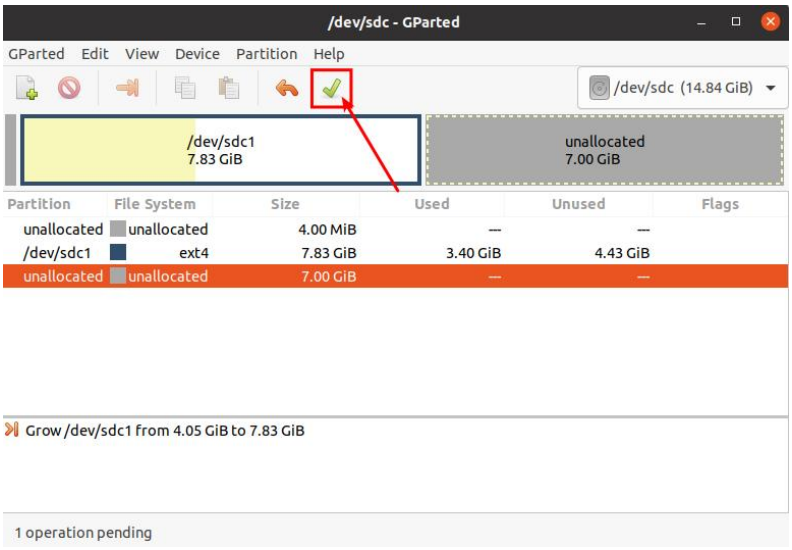
12) You can then directly drag the position shown in the figure below to set the capacity, or you can set the size of the rootfs partition by setting the number in **New size(MiB)**



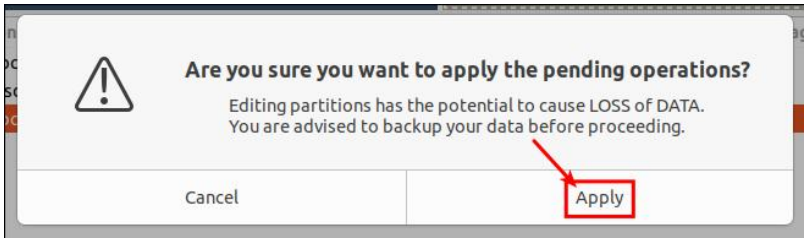
13) After setting the capacity, click **Resize/Move** in the lower right corner.



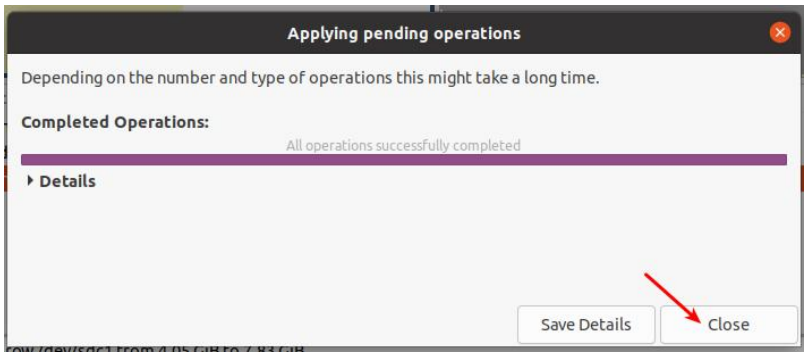
14) After final confirmation, click the **green** √ as shown below



15) Then select **Apply** to officially start expanding the capacity of the rootfs partition



16) After the expansion is completed, click **Close**.



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

```
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. How to reduce the capacity of the rootfs partition in the TF card

After configuring the application or other development environment in the Linux system on the TF card, if you want to back up the Linux system on 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 into the **Ubuntu computer** (not Windows)

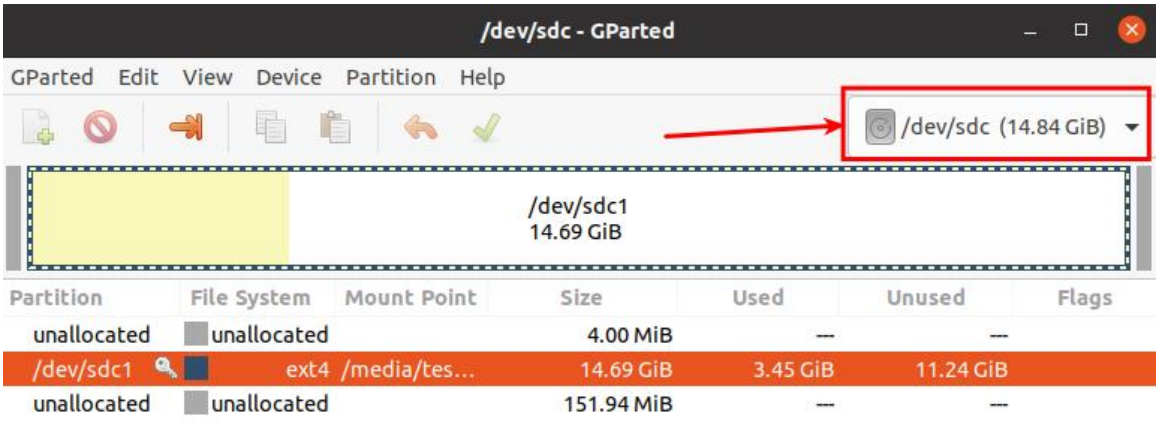
2) Then install the gparted software in the Ubuntu computer

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

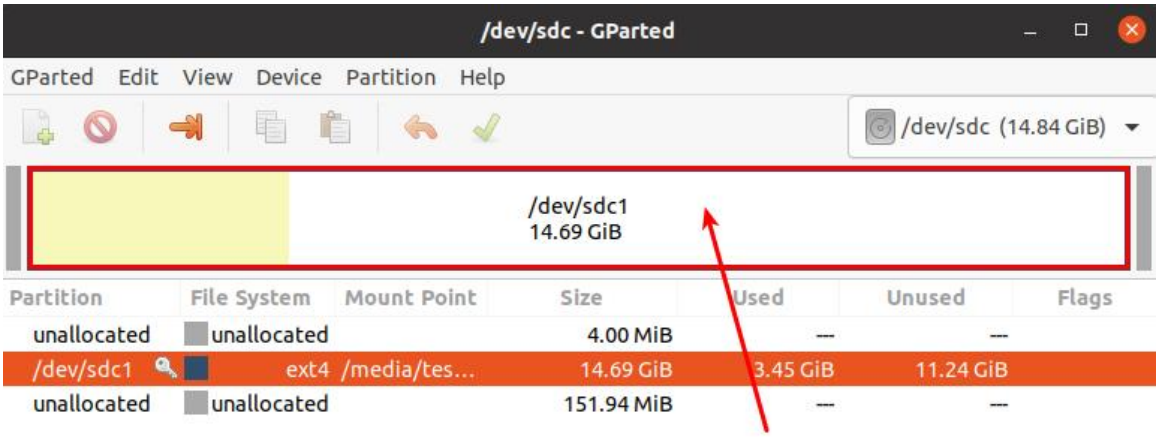
3) Then open gparted

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

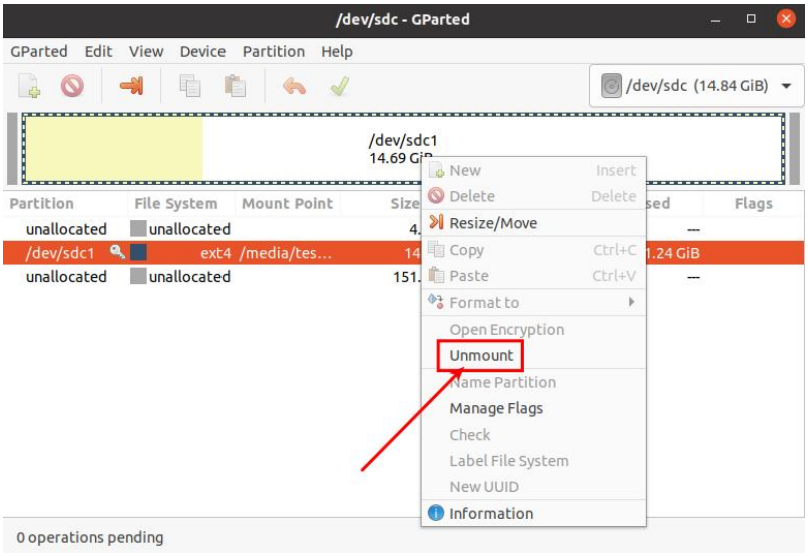
4) After opening gparted, you can select the TF card in the upper right corner, and then you can see the usage of the TF card capacity



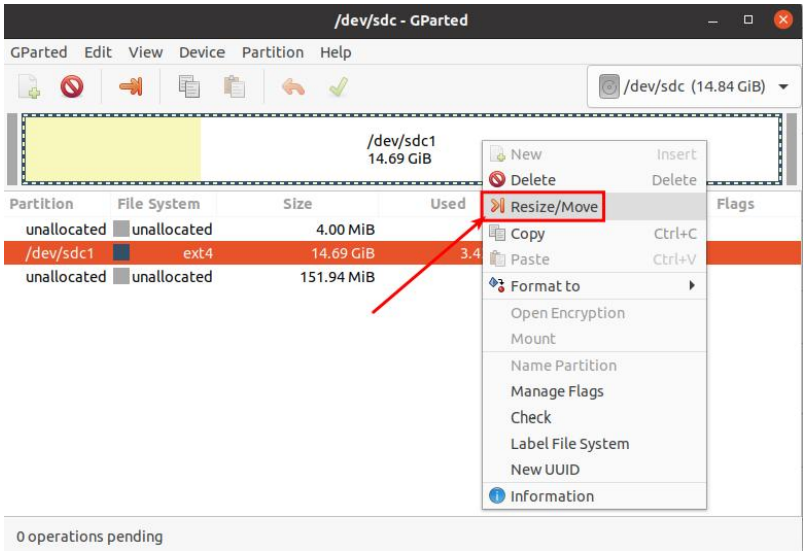
5) Then select the rootfs partition (/dev/sdc1)



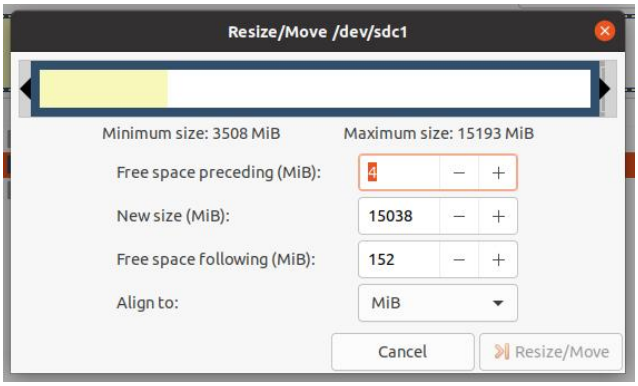
6) Click the right button of the mouse again to see the operation options shown in the figure below. If the TF card has been mounted, you need to Umount the rootfs partition of the TF card first.



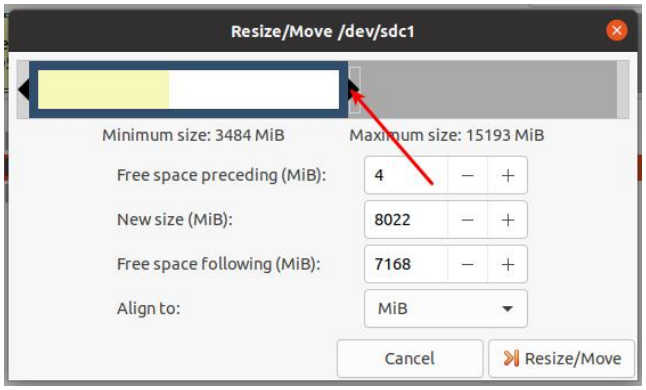
7) Then select the rootfs partition again, right-click, and select **Resize/Move** to start setting the size of the rootfs partition



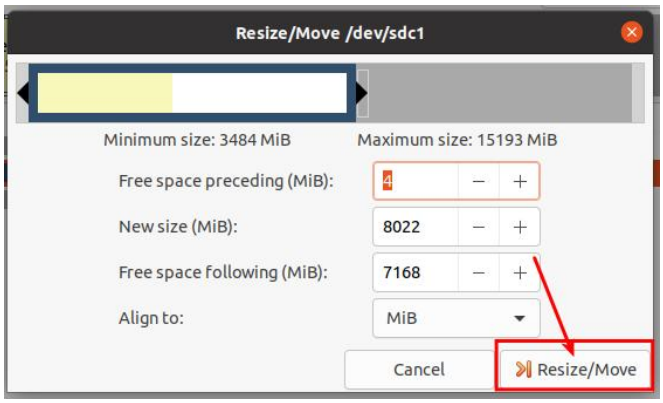
8) After the **Resize/Move** option is turned on, the following setting interface will pop up



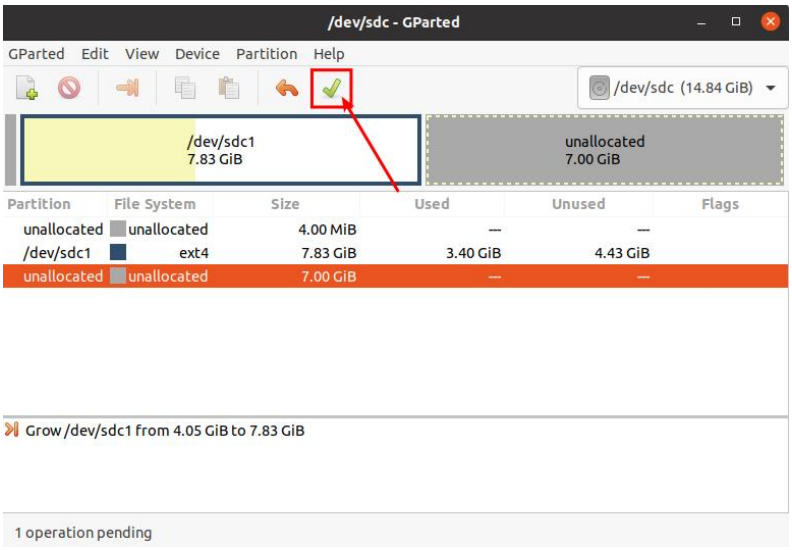
9) You can then directly drag the position shown in the figure below to set the capacity, or you can set the size of the rootfs partition by setting the number in **New sieze(MiB)**



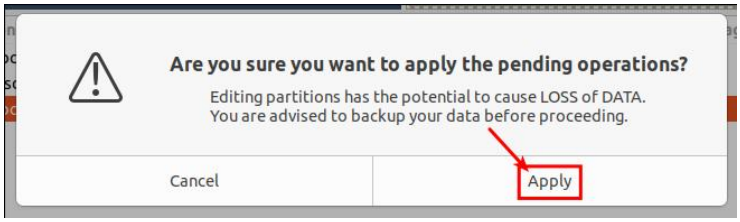
10) After setting the capacity, click **Resize/Move** in the lower right corner.



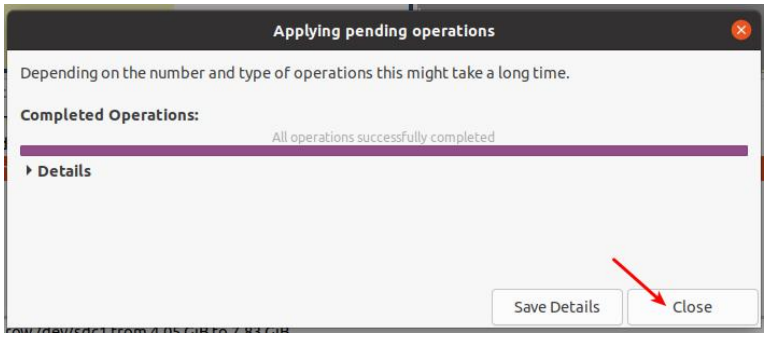
11) After final confirmation, click the **green** √ as shown below



12) Then select **Apply** to officially start expanding the capacity of the rootfs partition



13) After the expansion is completed, click **Close**.



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

```
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. Network connection test

3. 7. 1. Ethernet port test

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

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

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

Note that in the following commands, Debian 12 needs to change eth0 to end0.

```
orangepi@orangepi:~$ ip a s eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 3a:3a:57:82:eb:1f brd ff:ff:ff:ff:ff:ff
```

```
    inet 192.168.2.163/24 brd 192.168.2.255 scope global dynamic noprefixroute eth0
       valid_lft 42902sec preferred_lft 42902sec
    inet6 fdcd:e671:36f4::a39/128 scope global dynamic noprefixroute
       valid_lft 42904sec preferred_lft 42904sec
    inet6 fdcd:e671:36f4:0:7b67:e74e:f0e1:849a/64 scope global temporary dynamic
       valid_lft 604504sec preferred_lft 86095sec
    inet6 fdcd:e671:36f4:0:d098:7f17:6cea:4de4/64 scope global mngtmpaddr
noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::cc72:d313:9846:a5e0/64 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 a s eth0` command to view the IP address

2. Enter the `ip a s eth0` command in the debugging serial terminal to view the IP address

3. If you don't have a debug serial port or an HDMI display, you can also check the IP address of the development board's network port through the router's management interface. However, this method often fails to display the development board's IP address. If you can't see it, the debugging method is as follows:

A) First, check whether the Linux system has booted properly. If the green light on the development board is blinking, it's generally booted properly. If only the red light is on, or neither is on, the system has not booted properly.

B) Check that the network cable is securely plugged in, or try a different one.

C) Try a different router (we've encountered many router issues, such as routers failing to assign IP addresses, or even assigning IP addresses but not being visible on the router).

D) If you don't have a router to replace, connect an HDMI monitor or use the debug serial port to check the IP address.

It should also be noted that the development board DHCP automatically assigns an IP address and does not require any settings.

4) The command to test network connectivity is as follows. The **ping** command can be

interrupted by pressing **Ctrl+C**.

```
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
```

3.7.2. WIFI connection test

Please do not connect to WIFI by modifying the /etc/network/interfaces configuration file. Connecting to the WIFI network in this way may cause problems.

3.7.2.1. Server version image connects to WIFI through command

If the development board is not connected to Ethernet or an HDMI display, but only to a serial port, it is recommended to use the commands demonstrated in this section to connect to a WiFi network. This is because nmtui can only display characters and cannot display a graphical interface properly in some serial port software (such as minicom). Of course, if the development board is connected to Ethernet or an HDMI display, the commands demonstrated in this section can also be used to connect to a WiFi network.

1) Log in to the Linux system first. There are three ways to do this:
   a. If the development board is connected to the network cable, you can log in to the **Linux system remotely via SSH**
   b. If the development board is connected to the debug serial port, you can use the serial terminal to log in to the Linux system
   c. If the development board is connected to an HDMI display, you can log in to the Linux system through the HDMI display terminal

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

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

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

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

a. **wifi_name** You need to change it to the name of the WIFI hotspot you want to connect to.

b. **wifi_passwd**You need to change it to the password of the WIFI hotspot you want to connect to

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

4) Use the**ip addr show wlan0** command to view the IP address of the Wi-Fi network.

```
orangepi@orangepi:~$ ip a s 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 Wi-Fi network. The **ping** command can be interrupted by pressing the **Ctrl+C**shortcut key.

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

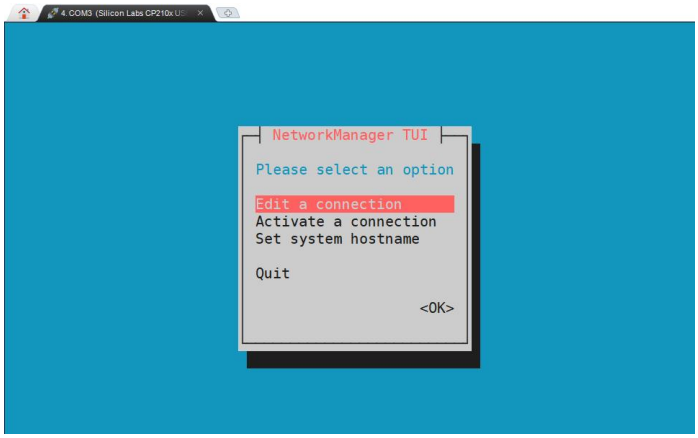
3.7.2.2. Server version image connects to WIFI through graphical mode

1) Log in to the Linux system first. There are three ways to do this:
   a. If the development board is connected to the network cable, you can log in to the **Linux system remotely via SSH**
   b. If the development board is connected to the debug serial port, you can use the serial terminal to log in to the Linux system (use MobaXterm as the serial software, minicom cannot display the graphical interface)
   c. If the development board is connected to an HDMI display, you can log in to the Linux system through the HDMI display terminal

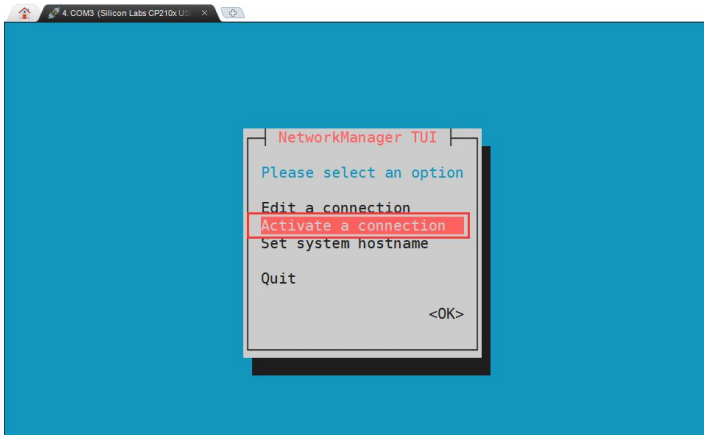
2) Then enter the nmtui command in the command line to open the wifi connection interface

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

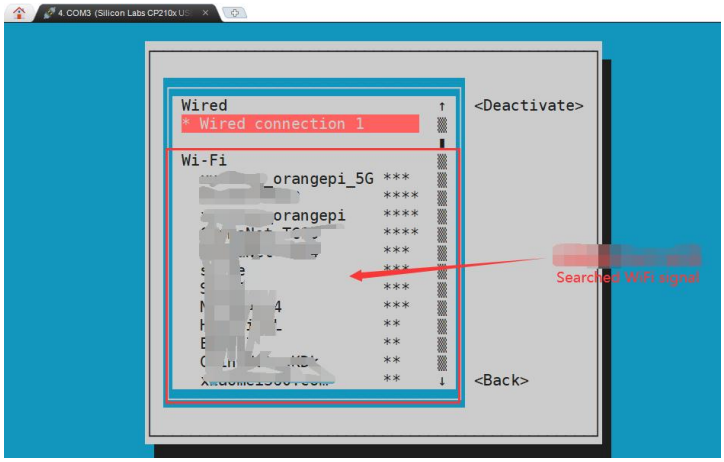
3) Enter the nmtui command to open the interface as shown below



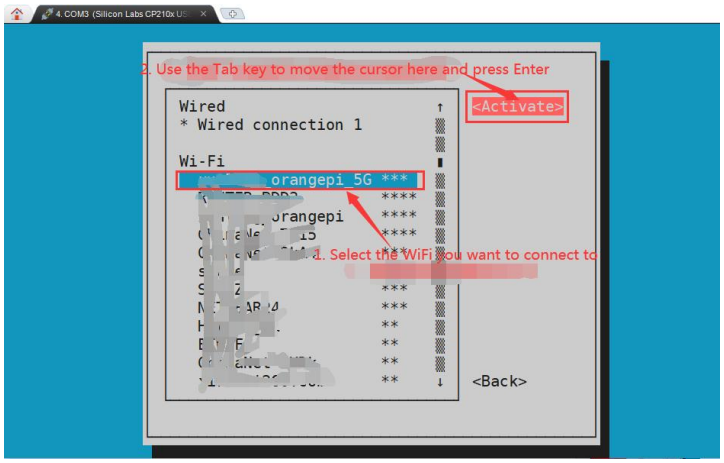
4) Select **Activate a connect** and press Enter



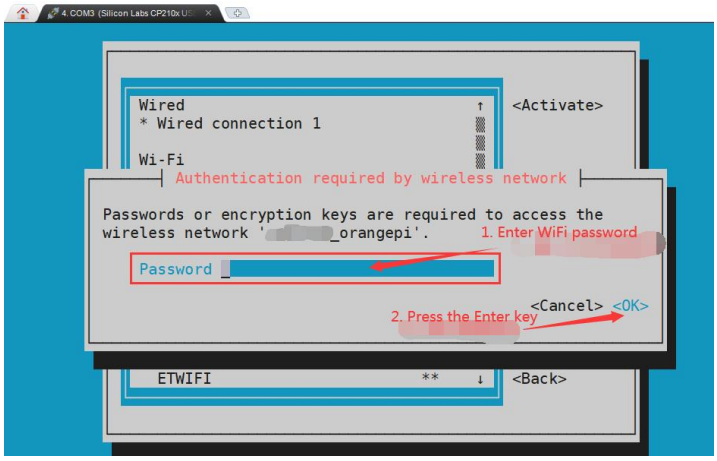
5) Then you can see all the searched WIFI hotspots



6) Select the Wi-Fi hotspot you want to connect to and use the Tab key to position the cursor on **Activate** and press Enter.



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



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

9) You can view the IP address of the WiFi through the **ip a s wlan0** command

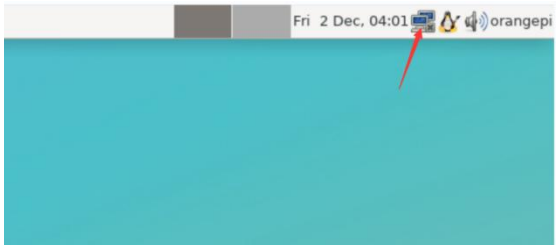
```
orangepi@orangepi:~$ ip a s wlan0
11: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 24:8c:d3:aa:76:bb brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.11/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
       valid_lft 259069sec 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 Wi-Fi network. The **ping** command can be interrupted by pressing the **Ctrl+C** shortcut key.

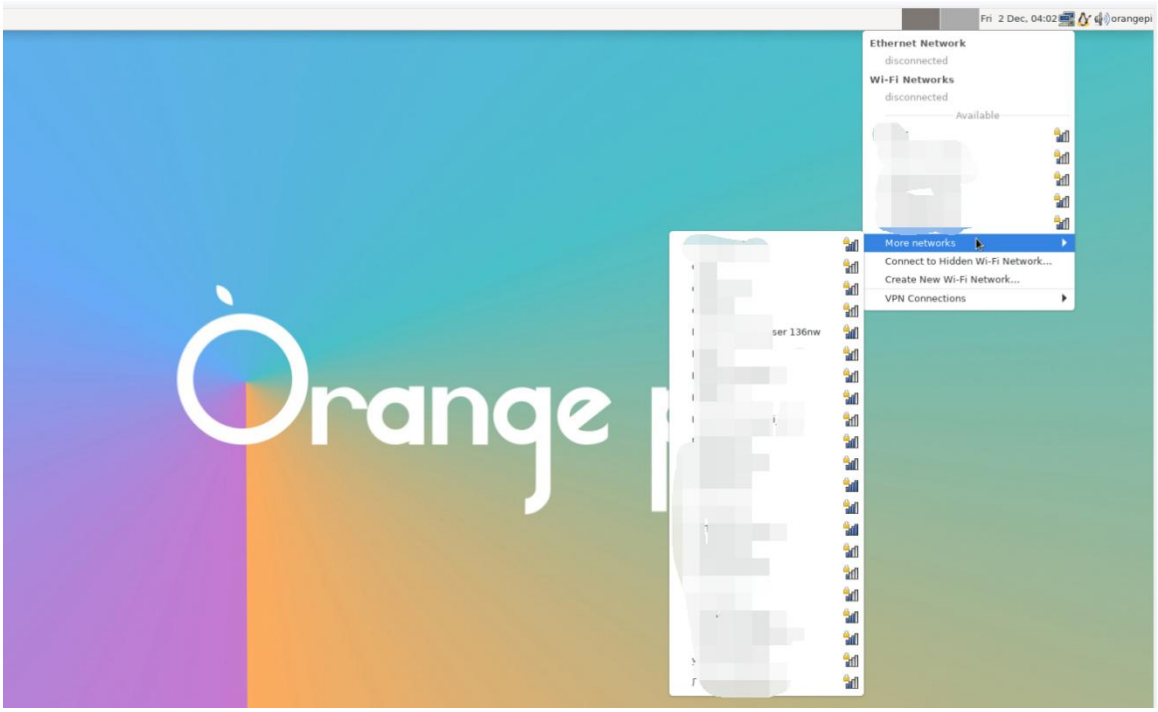
```
orangepi@orangepi:~$ ping www.orangepi.org -I wlan0
PING www.orangepi.org (182.92.236.130) from 192.168.1.49 wlan0: 56(84) bytes of data.
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=1 ttl=52 time=43.5 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=2 ttl=52 time=41.3 ms
64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

3.7.2.3. Desktop Image Testing Method

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 the scanned WIFI hotspots, and then select the WIFI hotspot you want to connect to



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

3. 7. 3. How to create a WIFI hotspot via create_ap

create_ap is a script that helps quickly create a WiFi hotspot on Linux. It supports bridge and NAT modes and can automatically combine hostapd, dnsmasq, and iptables to complete the WiFi hotspot setup, avoiding complex configuration for users. The github address is as follows:

https://github.com/oblique/create_ap

The Linux image released by OPi has pre-installed the create_ap script. You can use the create_ap command to create a WiFi hotspot. The basic command format of create_ap is as follows:

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

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

3. 7. 3. 1. create_ap method to create a WiFi hotspot in NAT mode

1）Enter the following command to create a WIFI hotspot named **orangepi** with the password **orangepi** in NAT mode

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 eth0 orangepi orangepi --no-virt
```

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

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 eth0 orangepi orangepi --no-virt
Config dir: /tmp/create_ap.wlan0.conf.TQkJtsz1
PID: 26139
Network Manager found, set wlan0 as unmanaged device... DONE
```

```
Sharing Internet using method: nat
hostapd command-line interface: hostapd_cli -p
/tmp/create_ap.wlan0.conf.TQkJtsz1/hostapd_ctrl
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

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

4）The display after successful connection is as shown below

5）In NAT mode, the IP addresses assigned to the network port and WIFI are in two different network segments. For example, the IP address of the development board network port is 192.168.1.X

```
orangepi@orangepi:~$ sudo ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.150  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::938f:8776:5783:afa2  prefixlen 64  scopeid 0x20<link>
        ether 4a:a0:c8:25:42:82  txqueuelen 1000  (Ethernet)
        RX packets 25370  bytes 2709590 (2.7 MB)
        RX errors 0  dropped 50  overruns 0  frame 0
        TX packets 3798  bytes 1519493 (1.5 MB)
```

```
        TX errors 0    dropped 0 overruns 0    carrier 0    collisions 0
        device interrupt 83
```

The default IP address of the development board's WIFI is 192.168.12.1.

```
orangepi@orangepi:~$ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.12.1  netmask 255.255.255.0  broadcast 192.168.12.255
        inet6 fe80::be17:96ff:fe91:c53c  prefixlen 64  scopeid 0x20<link>
        ether bc:17:96:91:c5:3c  txqueuelen 1000  (Ethernet)
        RX packets 5973  bytes 1129156 (1.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7213  bytes 6453949 (6.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

By default, the development board's DHCP service assigns IP addresses in the **192.168.12.0/24** network segment to devices connected to the hotspot. Clicking the connected **orangepi** hotspot on your phone will show that its IP address is **192.168.12.X**.

6）If you want to specify a different network segment for the connected device, you can use the -g parameter to specify it. For example, the command to specify the network segment of the access point AP as 192.168.2.1 using the -g parameter is as follows:

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 eth0 orangepi orangepi -g 192.168.2.1 --no-virt
```

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

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

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 eth0 orangepi orangepi --freq-band 5    --no-virt
```

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

```
orangepi@orangepi:~$ sudo create_ap -m nat wlan0 eth0 orangepi orangepi --hidden    --no-virt
```

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

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

1）Enter the following command to create a Wi-Fi hotspot in bridge mode with the name **orangepi**i and the password **orangepi**

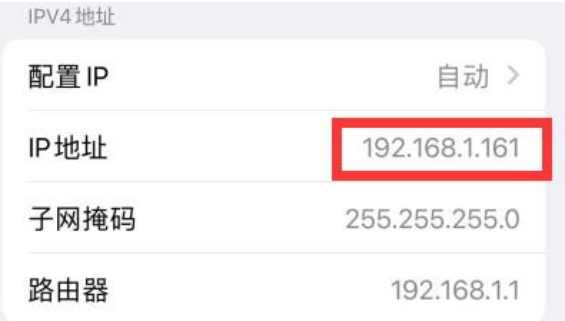
```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 eth0 orangepi orangepi --no-virt
```

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

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 eth0 orangepi orangepi --no-virt
Config dir: /tmp/create_ap.wlan0.conf.zAcFlYTx
PID: 27707
Network Manager found, set wlan0 as unmanaged device... DONE
Sharing Internet using method: bridge
Create a bridge interface... br0 created.
hostapd command-line interface: hostapd_cli -p
/tmp/create_ap.wlan0.conf.zAcFlYTx/hostapd_ctrl
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

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

4）The display after successful connection is as shown below



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

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 eth0 orangepi orangepi --freq-band 5 --no-virt
```

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

```
orangepi@orangepi:~$ sudo create_ap -m bridge wlan0 eth0 orangepi orangepi --hidden --no-virt
```

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

3. 7. 4. How to set a static IP address

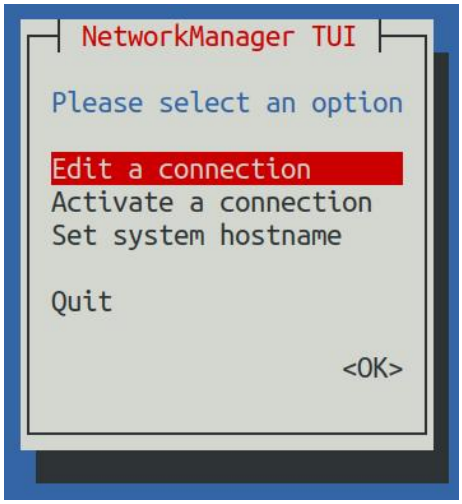
Please do not set a static IP address by modifying the /etc/network/interfaces configuration file.

3. 7. 4. 1. Using nmtui to set a static IP address

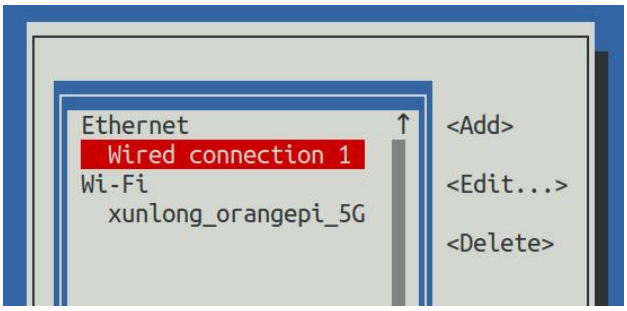
1) First run the **nmtui** command

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

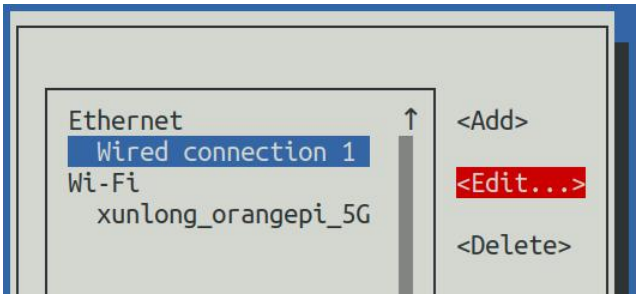
2) Then select **Edit a connection** and press Enter



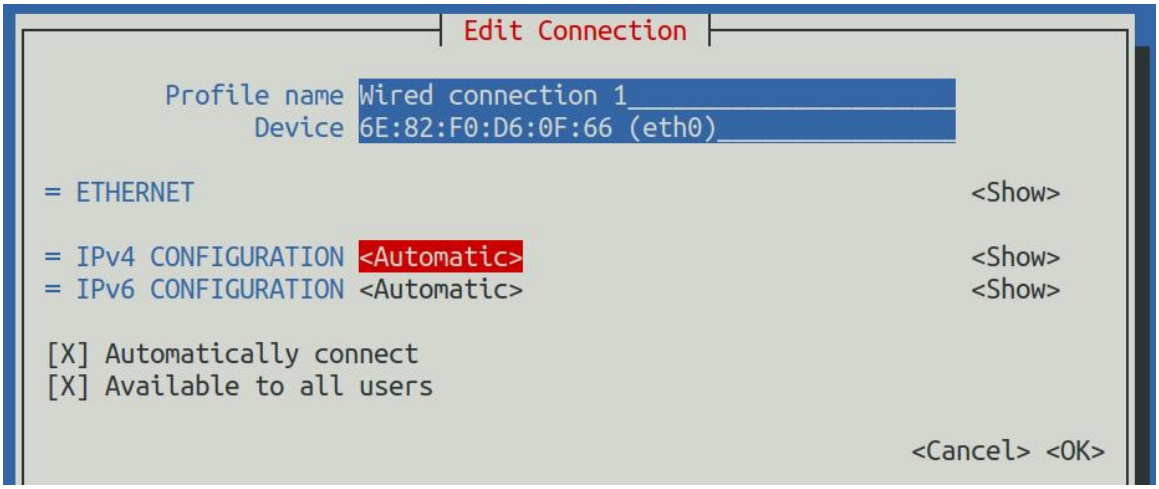
3) Then select the network interface for which you want to set a static IP address. For example, to set the static IP address of the Ethernet interface, select **Wired connection 1**.



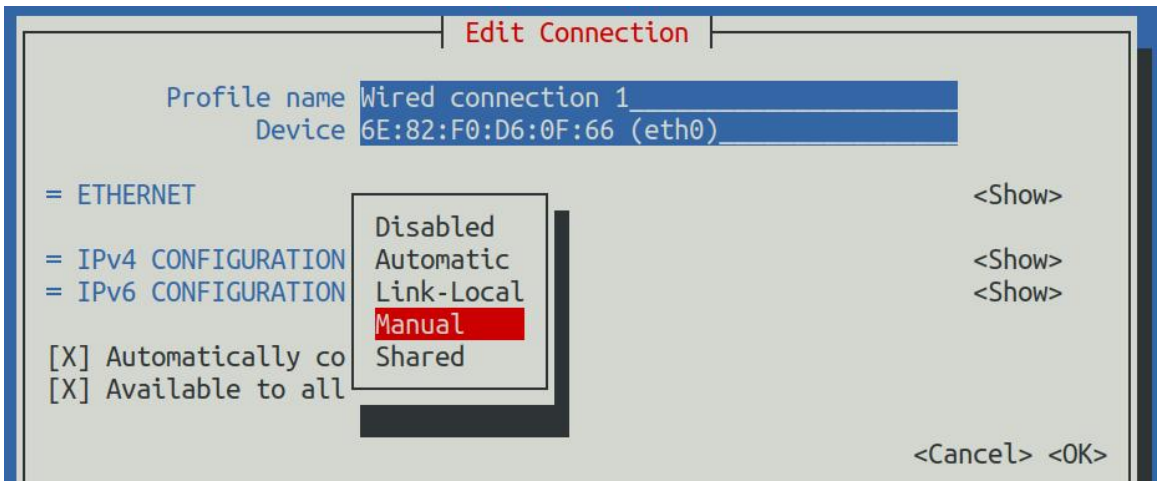
4) Then select **Edit** using the **Tab** key and press Enter



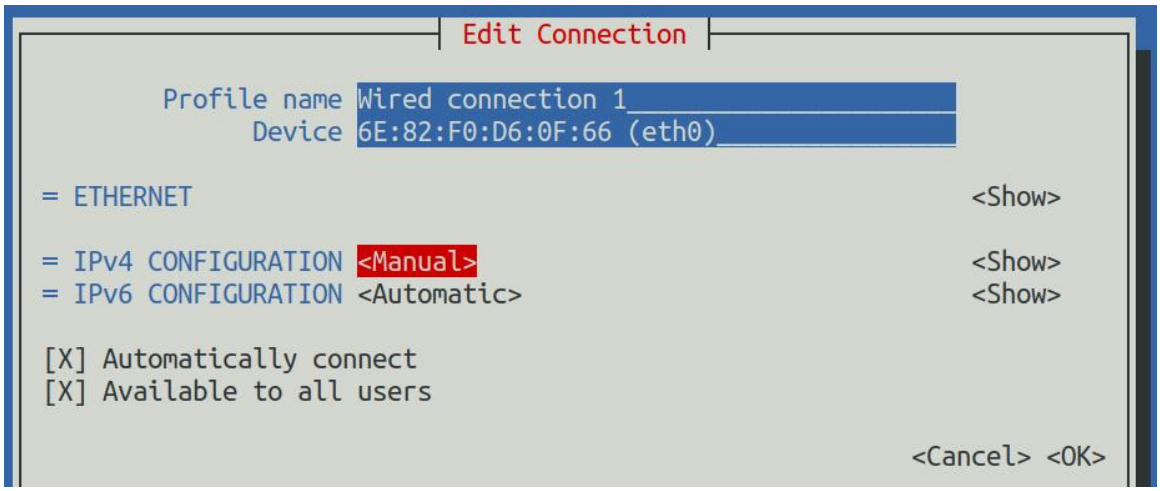
5) Then use the Tab key to move the cursor to the **<Automatic>** position shown in the figure below to configure IPv4



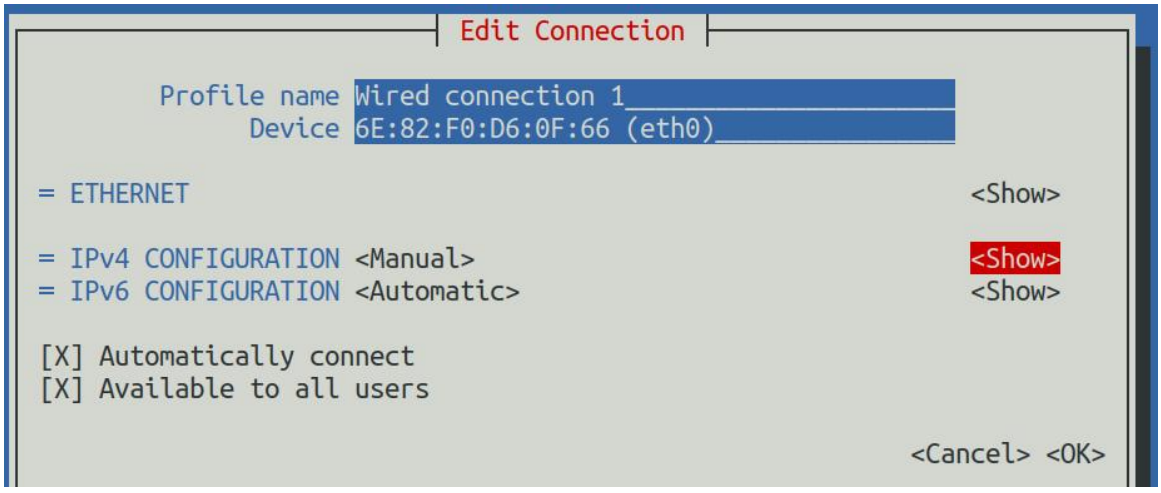
6) Then press Enter, use the up and down arrow keys to select **Manual**, and then press Enter to confirm.



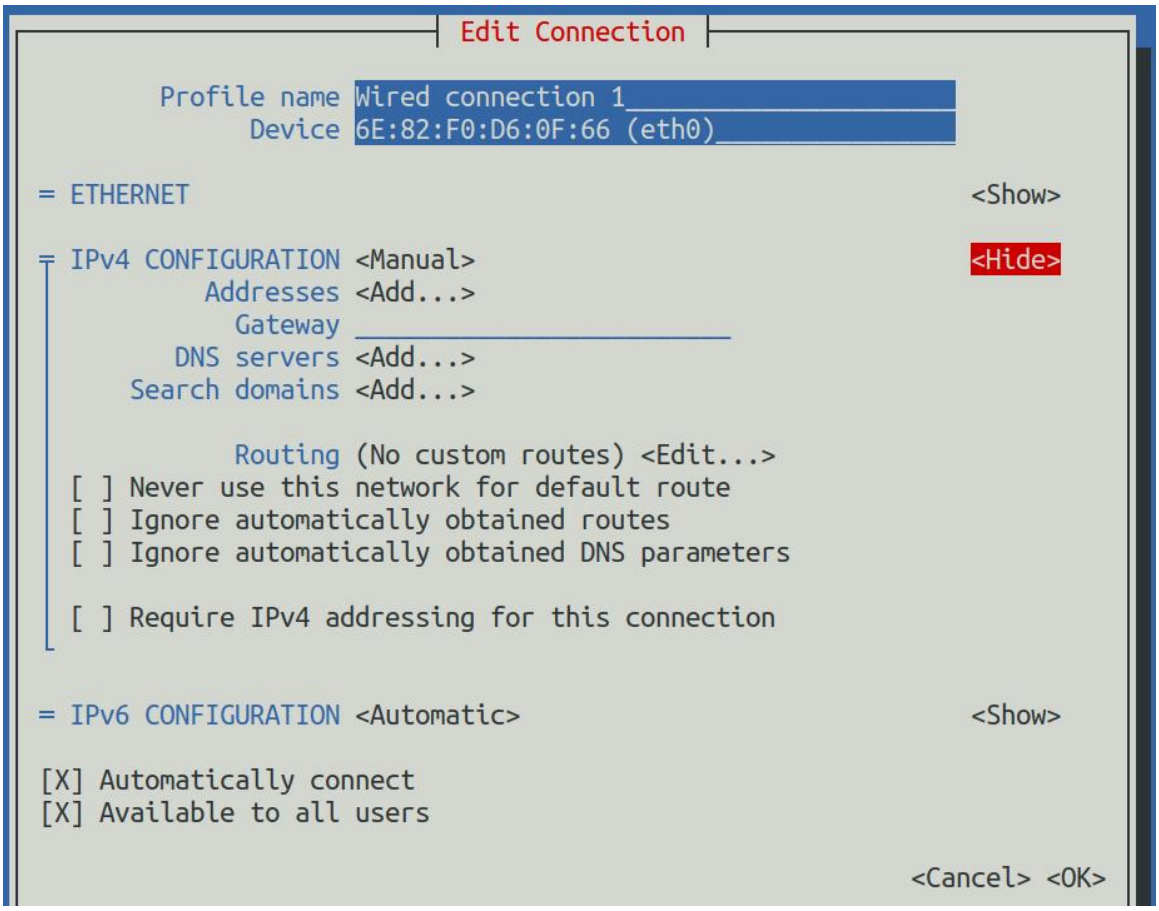
7) The display after selection is as shown below



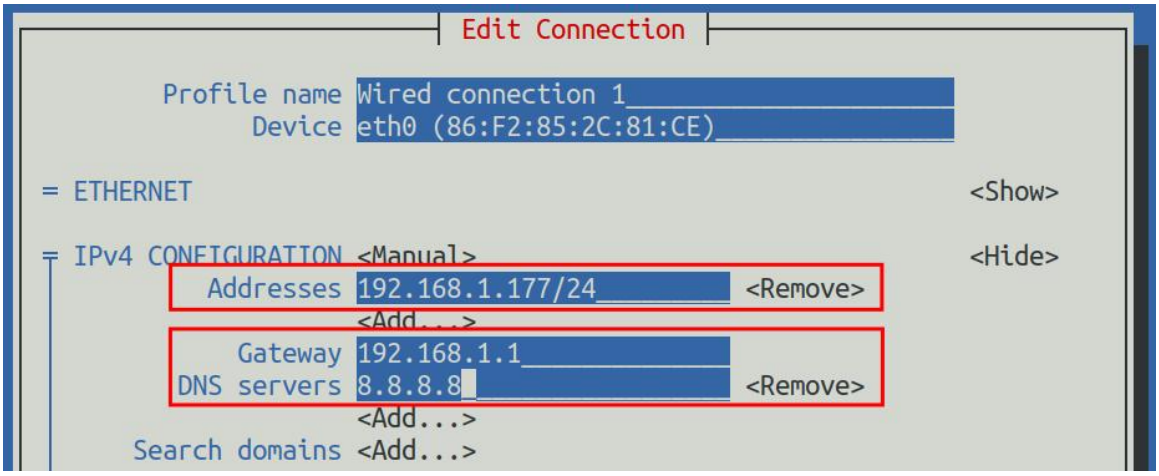
8) Then use the Tab key to move the cursor to **<Show>**



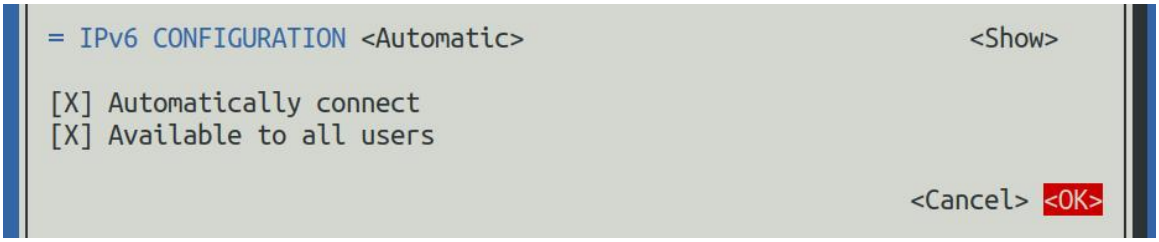
9) Then press Enter, and the following setting interface will pop up after pressing Enter



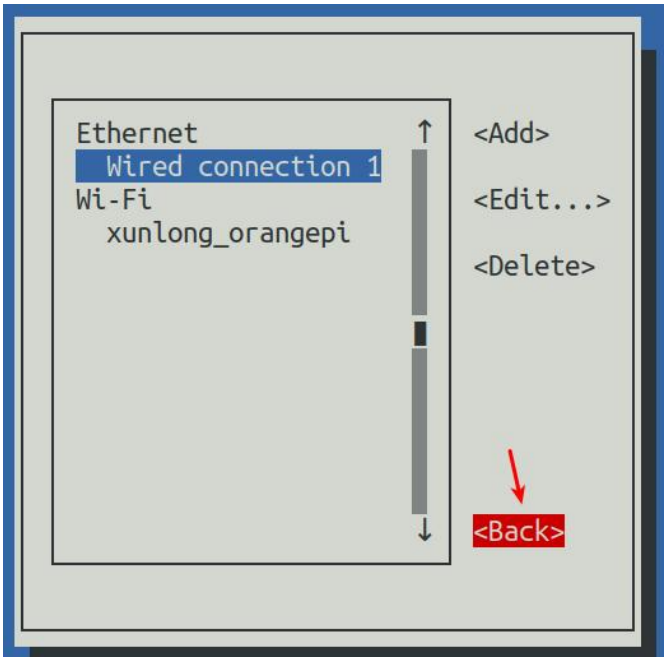
10) Then you can set the IP address (Addresses), gateway (Gateway) and DNS server address as shown in the figure below (there are many other setting options, please explore them yourself). **Please set them according to your specific needs. The values set in the figure below are just an example.**



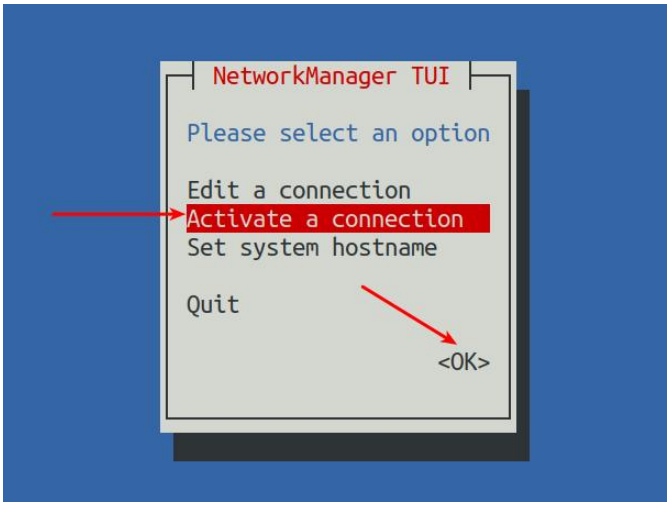
11) After setting, move the cursor to **<OK>** in the lower right corner and press Enter to confirm.



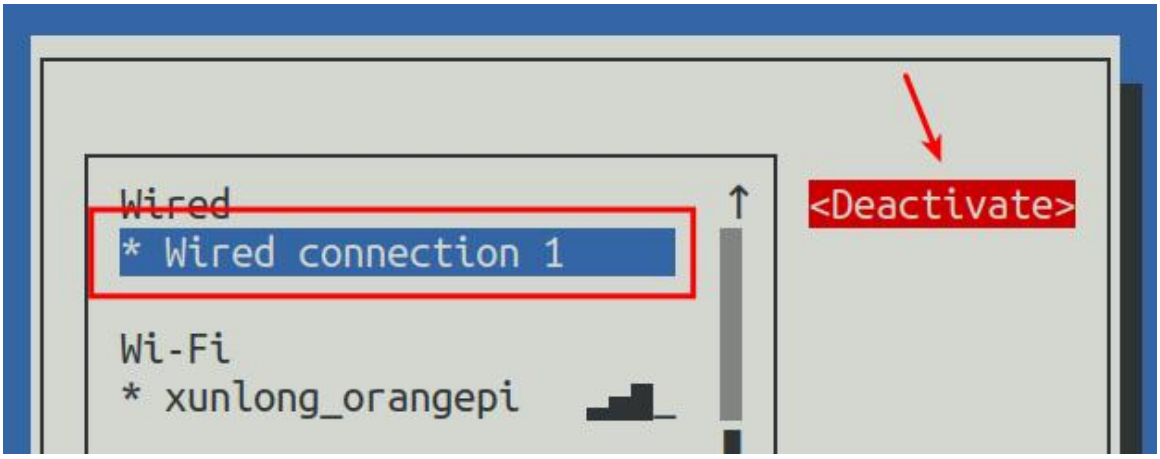
12) Then click **<Back>** to return to the previous selection interface



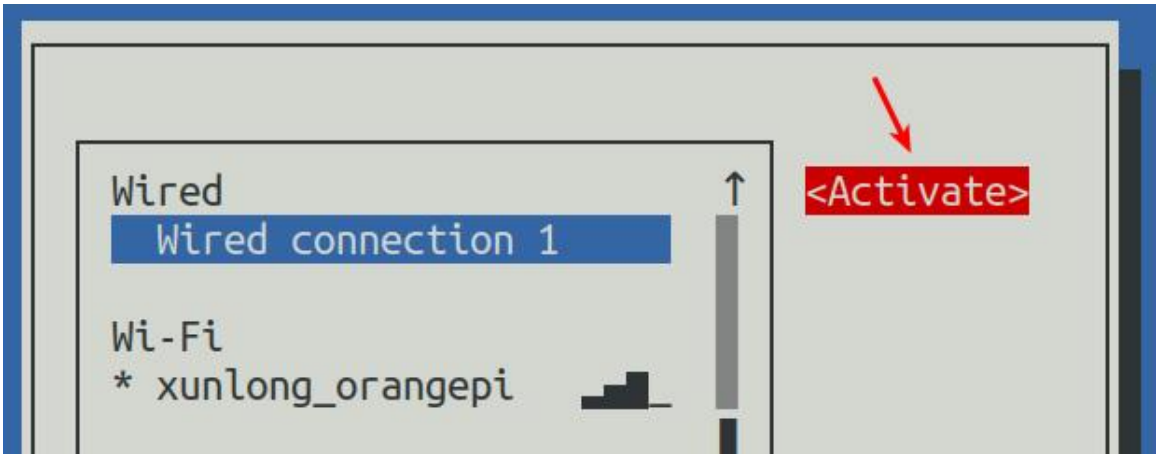
13) Then select **Activate a connection**, move the cursor to **<OK>**, and finally press Enter



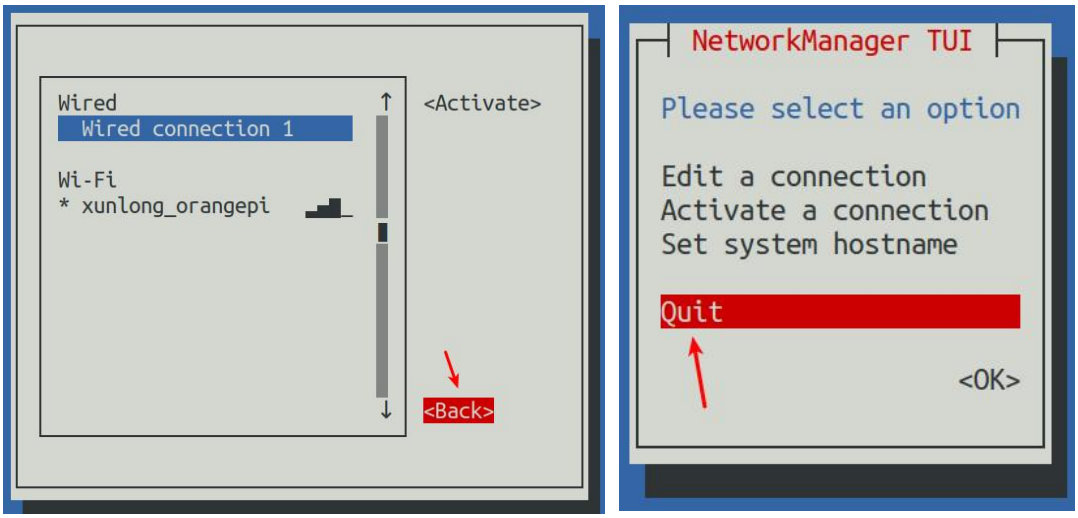
14) Then select the network interface you want to set, such as **Wired connection 1**, then move the cursor to **<Deactivate>**, and press Enter to disable **Wired connection 1**



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



16) Then you can exit nmtui by pressing the **<Back>** and **Quit** buttons



17) Then use **ip a s eth0** see that the IP address of the network port has become the static IP address set previously.

```
orangepi@orangepi:~$ ip a s 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 network connectivity to check if the IP address is configured correctly. The **ping** command can be interrupted by pressing the **Ctrl+C** shortcut key.

```
orangepi@orangepi:~$ ping 192.168.1.177 -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
```

3.7.4.2. Using nmcli to set a static IP address

1) If you want to set a static IP address for the network port, please first plug the network cable into the development board. **If you need to set a static IP address for WiFi, please connect WiFi first and then start setting the static IP address.**

2) Then use the **nmcli con show** command to view the name of the network device, as shown below

a. **orangepi** is the name of the Wi-Fi network interface (the name may not be the same)

b. **Wired connection 1** is 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, where

a. **"Wired connection 1"** means setting a static IP address for the Ethernet port. If you need to set a static IP address for Wi-Fi, please change it to the name of the

Wi-Fi network interface (which can be obtained through the **nmcli con show** command)

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

c. **ipv4.gateway** Indicates the gateway address

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

3.7.5. How to set up the Linux system to automatically connect to the network when it starts for the first time

The development board has an Ethernet port. To remotely log in to the Linux system on the development board through the Ethernet port, simply plug a network cable into the Ethernet port. After the Linux system boots, the Ethernet port will automatically assign an IP address via DHCP. You can then obtain the Ethernet port's IP address through an HDMI monitor, the serial port, or by checking the router's backend, and then remotely log in to the Linux system.

The development board also has wireless Wi-Fi. To remotely log in to the Linux system on the development board through Wi-Fi, you need to use SSH to remotely log in to the Linux system using the Ethernet port's IP address and then connect to Wi-Fi using a command, or connect to Wi-Fi using a command through the HDMI monitor or serial port.

However, if you don't have an HDMI monitor or serial port module, even if you have an Ethernet cable, you won't be able to check the development board's IP address through the router's backend. Alternatively, if you don't have an HDMI monitor, serial port module, or Ethernet cable, and only have Wi-Fi available, you can use the method described in this section to automatically connect to Wi-Fi and set a static IP address for Wi-Fi or the Ethernet port.

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

The reason you need a Linux machine is that the Linux root file system on the development board burned into the TF card is in ext4 format. A Linux machine can mount it normally and modify the configuration files.

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

In addition, if you encounter any problems when using Paragon ExtFS for Windows, please solve them yourself. We will not answer your questions.

1) First, burn the Linux image of the development board you want to use to a TF card. Then use a card reader to insert the TF card with the burned Linux image of the development board into a machine with a Linux system installed (such as a computer with an Ubuntu system installed. The following demonstration uses an Ubuntu computer as an example).

2) When the TF card is inserted into the Ubuntu computer, the Ubuntu computer will generally automatically mount the Linux root file system partition in the TF card. From the following command, we can know that **/media/test/opi_root** is the path where the Linux root file system in the TF card is mounted.

```
test@test:~$ df -h | grep "media"
```

```
/dev/sdd1  1.4G  1.2G  167M  88% /media/test/opi_root
test@test:~$ ls /media/test/opi_root
bin  boot  dev  etc  home  lib  lost+found  media  mnt  opt  proc  root  run
sbin  selinux  srv  sys  tmp  usr  var
```

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

```
test@test:~$ cd /media/test/opi_root/boot/
```

4) Then copy the **orangepi_first_run.txt.template** file to **orangepi_first_run.txt**. You can use the orangepi_first_run.txt configuration file to set the development board to automatically connect to a WIFI hotspot when the Linux system starts for the first time, or set a static IP address for the WIFI or Ethernet port.

```
test@test:/media/test/opi_root/boot$ sudo cp orangepi_first_run.txt.template orangepi_first_run.txt
```

5) Use the following command to open the orangepi_first_run.txt file, and then you can view and modify the contents

```
test@test:/media/test/opi_root/boot$ sudo vim orangepi_first_run.txt
```

6) Instructions for using variables in the orangepi_first_run.txt file

a. The **FR_general_delete_this_file_after_completion** variable is used to set whether to delete the orangepi_first_run.txt file after the first startup. The default value is 1, which means deletion. If it is set to 0, the orangepi_first_run.txt will be renamed to orangepi_first_run.txt.old after the first startup. Generally, keep the default value.

b. The **FR_net_change_defaults** variable is used to set whether to change the default network settings. This must be set to 1, otherwise all network settings will not take effect.

c. The **FR_net_ethernet_enabled** variable is used to control whether the Ethernet port configuration is enabled. If you need to set a static IP address for the Ethernet port, please set it to 1

d. The **FR_net_wifi_enabled** variable is used to control whether WiFi is enabled. If you need to set the development board to automatically connect to WiFi hotspots, you must set it to 1. Also, please note that if this variable is set to 1, the Ethernet port settings will be invalid. In other words, WiFi and Ethernet ports cannot be set at the same time (why, because there is no need...)

e. **FR_net_wifi_ssid** variable is used to set the name of the WIFI hotspot you want to connect to.
f. **FR_net_wifi_key** variable is used to set the password of the WIFI hotspot you want to connect to
g. **FR_net_use_static** variable is used to set whether to set a static IP address for the WIFI or Ethernet port
h. **FR_net_static_ip** variable is used to set the static IP address. Please set it according to your actual situation.
i. **FR_net_static_gateway** variable is used to set the gateway. Please set it according to your actual situation.

7) Here are some specific setting examples:
   a. For example, if you want the Linux system of the development board to automatically connect to the WiFi hotspot after the first startup, you can set it like this:
      a) Set **FR_net_change_defaults** to **1**
      b) Set **FR_net_wifi_enabled** to **1**
      c) Set **FR_net_wifi_ssid** to the name of the Wi-Fi hotspot you want to connect to.
      d) Set **FR_net_wifi_key** to the password of the WIFI hotspot you want to connect to
   b. For example, if you want the Linux system on the development board to automatically connect to the WIFI hotspot after the first startup, and set the WIFI IP address to a specific static IP address (so that when the Linux system starts, you can directly use the set static IP address to remotely log in to the development board through SSH, without having to check the IP address of the development board through the router backend), you can set it like this:
      a) Set **FR_net_change_defaults** to 1
      b) Set **FR_net_wifi_enabled** to 1
      c) Set **FR_net_wifi_ssid** to the name of the Wi-Fi hotspot you want to connect to
      d) Set **FR_net_wifi_key** to the password of the Wi-Fi hotspot you want to connect to
      e) Set **FR_net_use_static** to 1
      f) Set **FR_net_static_ip** to the desired IP address
      g) Set **FR_net_static_gateway** to the desired gateway address

c. For example, if you want the Linux system on the development board to automatically set the IP address of the Ethernet port to the desired static IP address after the first boot, you can set it like this:
   a) Set **FR_net_change_defaults** to 1
   b) Set **FR_net_ethernet_enabled** to 1
   c) Set **FR_net_use_static** to 1
   d) Set **FR_net_static_ip** to the desired IP address
   e) Set **FR_net_static_gateway** to the desired gateway address

8) After modifying the orangepi_first_run.txt file, you can exit the /boot directory of the Linux system on the TF card, unmount the TF card, and then insert the TF card into the development board to start it.

9) If you do not set a static IP address, you still need to check the IP address through the router background. If you set a static IP address, you can ping the static IP address set on the computer. If you can ping, it means that the system has started normally and the network has been set correctly. Then you can use the set IP address ssh remote login to the Linux system of the development board

After the development board's Linux system is started for the first time, orangepi_first_run.txt will be deleted or renamed to orangepi_first_run.txt.old. At this time, even if you reset the orangepi_first_run.txt configuration file and restart the development board's Linux system, the configuration in orangepi_first_run.txt will not take effect again, because this configuration only takes effect on the first startup after burning the Linux system. Please pay special attention to this.

3. 8. SSH remote login to the development board

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

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

1) Get the IP address of the development board

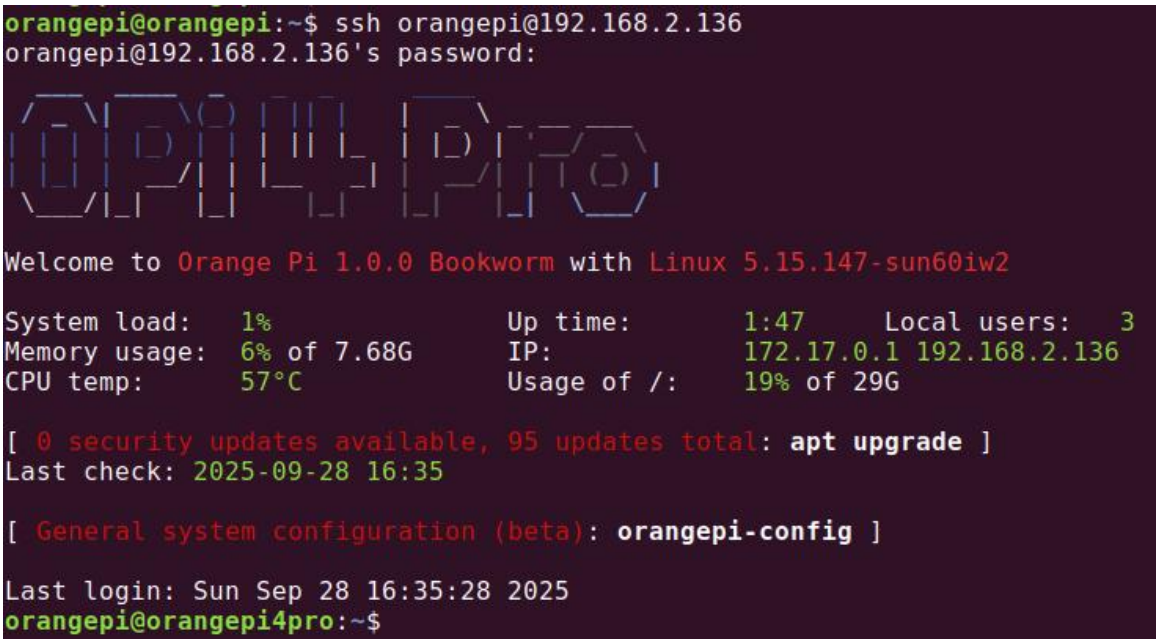
2) Then you can remotely log in to the Linux system through the ssh command

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

Note that the exact password will not be displayed on the screen when you enter it. Do not assume there is a malfunction; simply press Enter after entering it.

If you receive a connection rejection message, as long as you are using an Orange Pi image, do not suspect that the password "orangepi" is incorrect; instead, look for other possible causes.

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



If SSH fails to log in to the Linux system, first check whether the IP address of the development board can be pinged. If the ping is successful, you can log in to the Linux system through the serial port or HDMI display and then enter the following command on the development board to try to connect again:

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

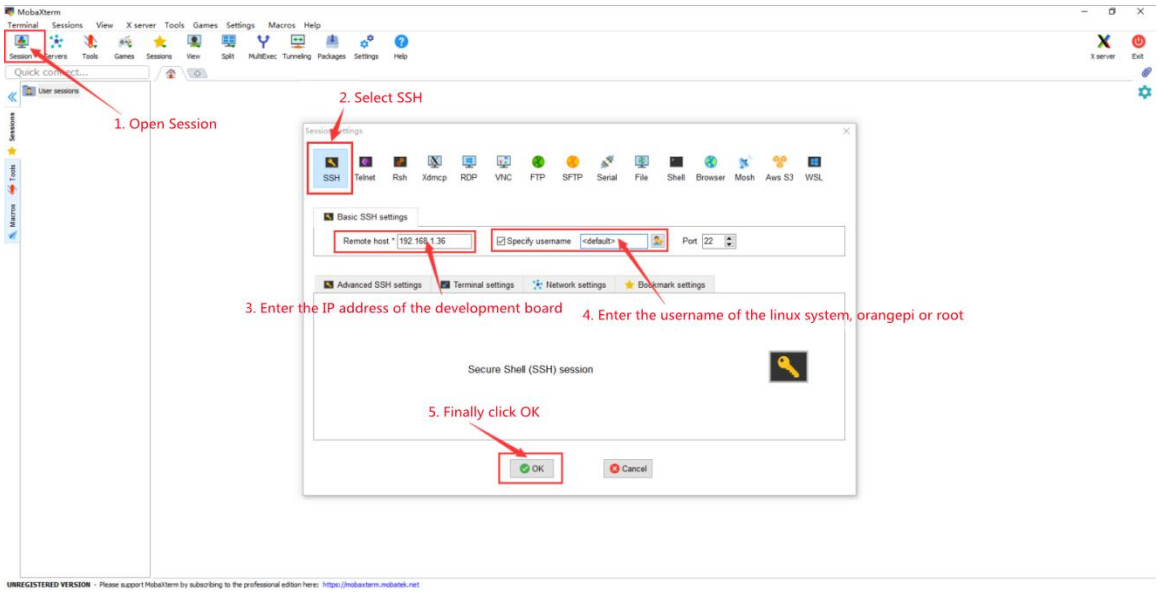
If it still doesn't work, please try reburning the system.

3. 8. 2. SSH remote login to the development board under Windows

1) First obtain the IP address of the development board

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

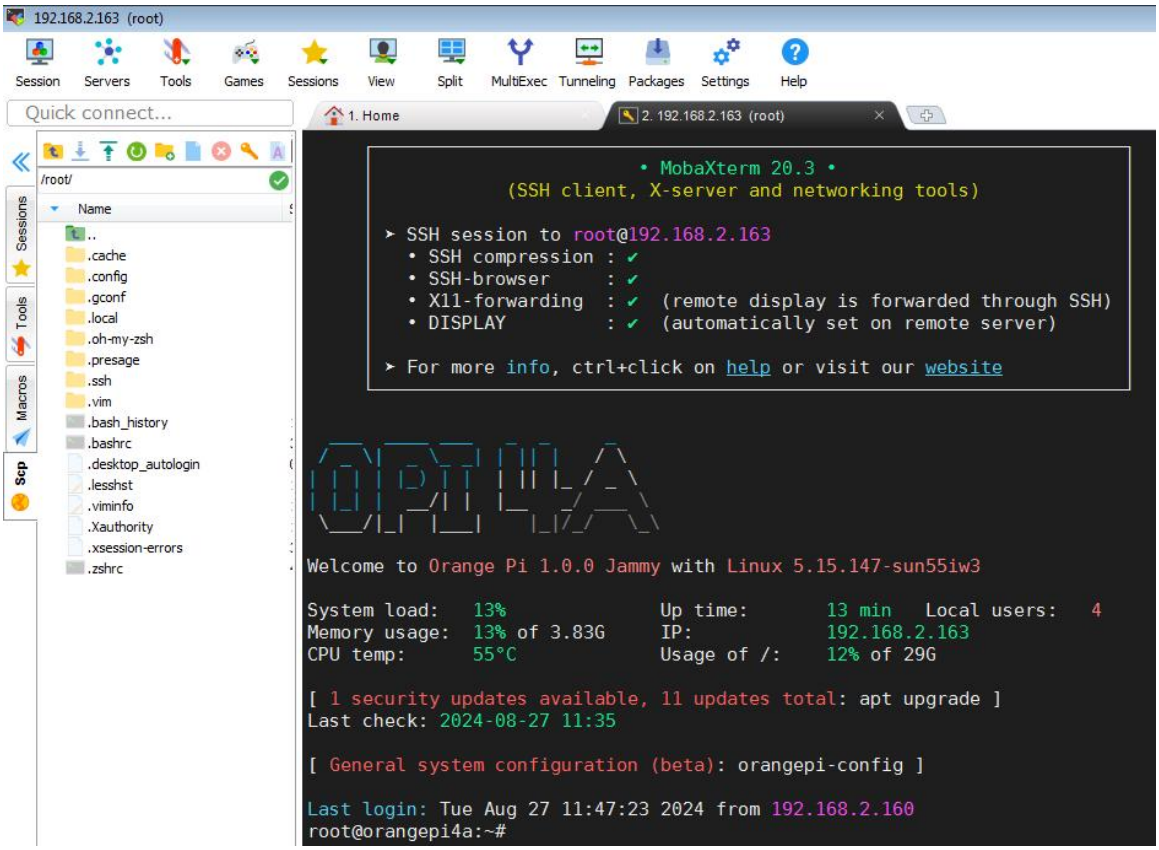
a. Open **Session**.
b. Select **SSH** in **Session Setting**.
c. Enter the development board's IP address in **Remote host**.
d. Enter your Linux username (**root or orangepi**) in **Specify usernam**.
e. Click **OK**.



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

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

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



3. 9. How to use ADB

3. 9. 1. How to use network adb

1) After the system starts, please execute the following command

```
orangepi@orangepi:~$ sudo /etc/adb_conf.sh
```

2) Then please make sure adbd has been started

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

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

4) Then install the adb tool on your Ubuntu PC

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

5) Then use the following command to connect to the network adb

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

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

```
test@test:~$ adb shell
root@orangepi4pro:/#      <--- After seeing this prompt, it means that you have successfully logged into the development board
```

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

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

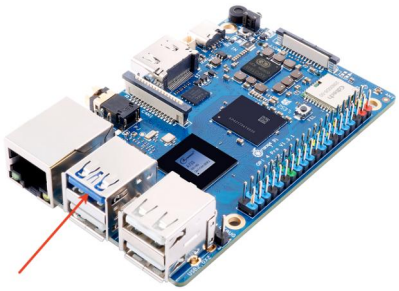
8) Use adb to restart the development board as follows

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

3.9.2. Connect adb using a Type-C cable

1) First prepare a good quality USB 2.0 male to male data cable

2) Then connect the development board to the Ubuntu PC via a USB 2.0 male-to-male data cable. The USB port on the development board that supports USB Device mode is located as shown in the figure below:



3) Then run the following command to set the Type-C interface to **device** mode

```
orangepi@orangepi:~$ sudo /etc/adb_conf.sh
```

4) Then please confirm that adbd has been started

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

5) Then install the adb tool on the Ubuntu PC

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

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

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

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

```
test@test:~$ adb shell
root@orangepi4pro:/#      <--- After seeing this prompt, it means that you have successfully logged into the development board
```

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

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

3.10. HDMI Test

3.10.1. HDMI Display Test

1) Connect the Orange Pi development board and HDMI display using an HDMI to HDMI cable

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

Note that while many laptops have HDMI ports, these ports typically only function as outputs and lack HDMI input functionality. This means you cannot display the HDMI output of other devices on the laptop screen.

When connecting the development board's HDMI port to a laptop's HDMI port, please ensure that your laptop supports HDMI input.

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

3. 10. 2. HDMI to VGA display test

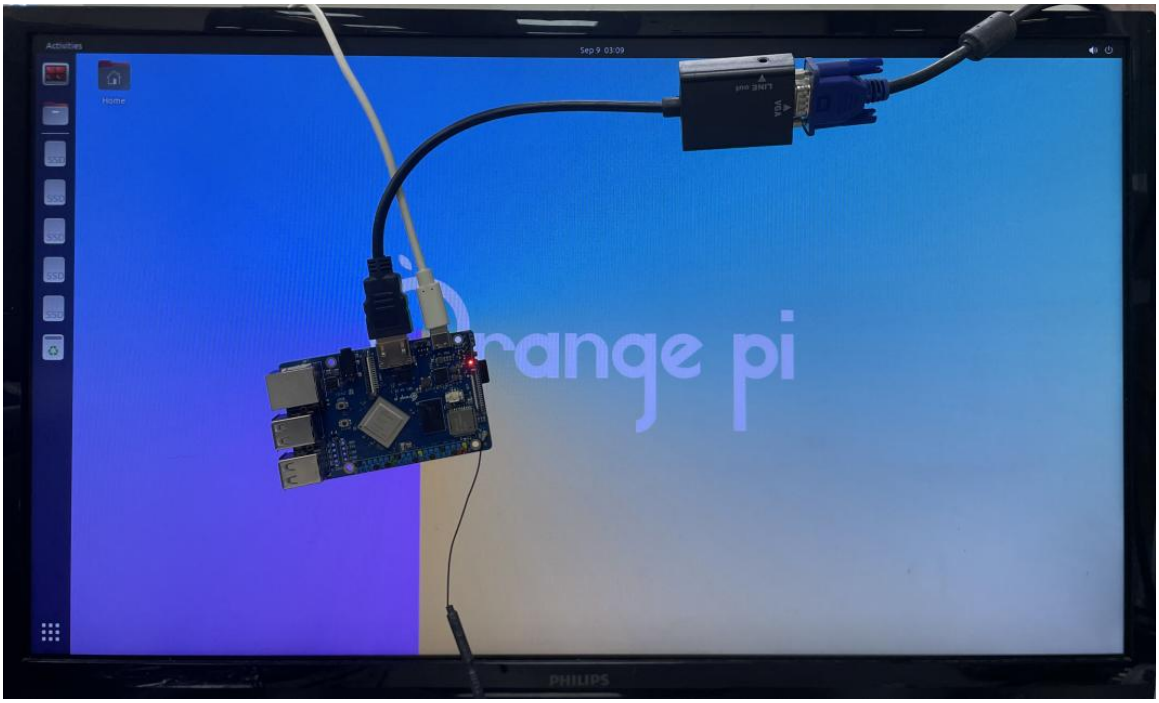
1) First, you need to prepare the following accessories

a. HDMI to VGA converter

b. A VGA cable

c. A monitor or TV that supports VGA interface

2) HDMI to VGA display test is as follows

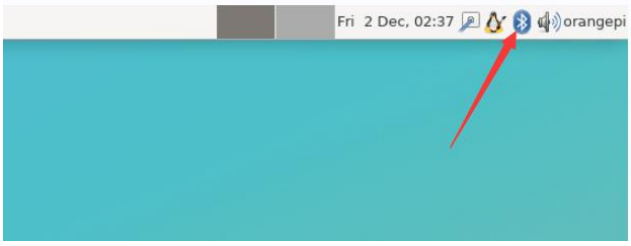


When using HDMI to VGA display, the development board and its Linux system do not require any configuration. It only requires the development board's HDMI interface to display normally. Therefore, if there are any problems with the test, please check the HDMI to VGA converter, VGA cable, and monitor for any problems.

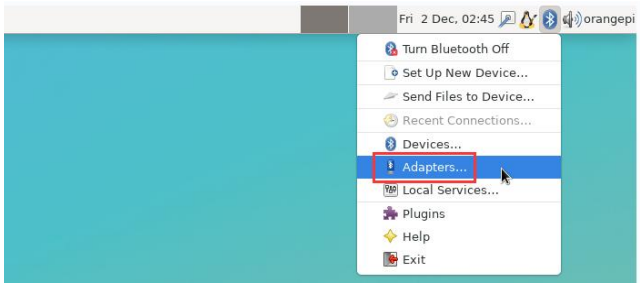
3. 11. How to use Bluetooth

3. 11. 1. Desktop Image Testing Method

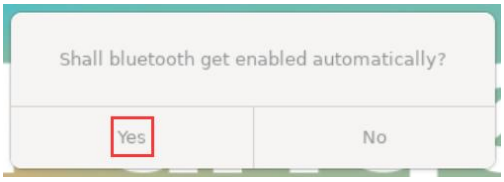
1) First click the Bluetooth icon in the upper right corner of the desktop



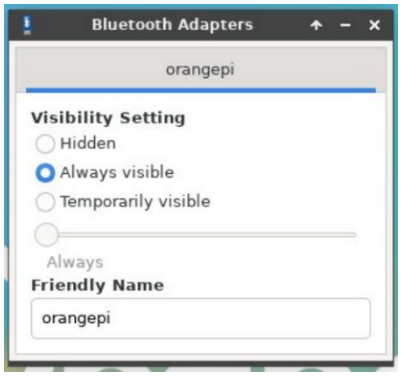
2) Then select the adapter



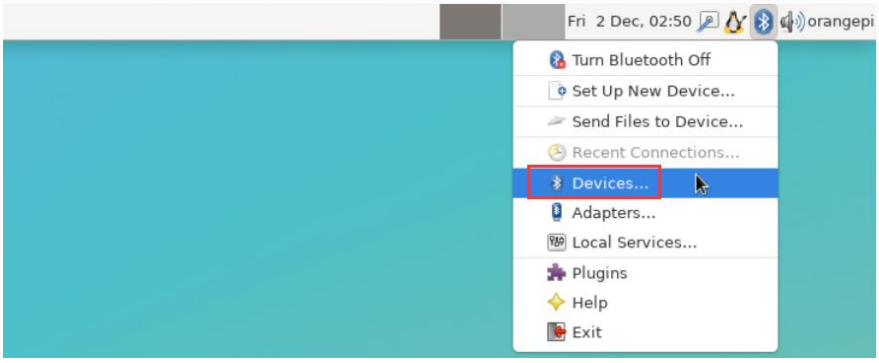
3) If the following interface appears, please select **Yes**



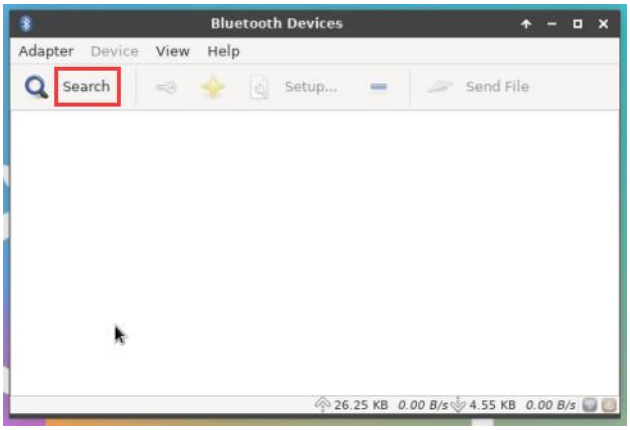
4) Then set the **Visibility Setting** to **Always visible** in the Bluetooth adapter settings interface, and then turn it off.



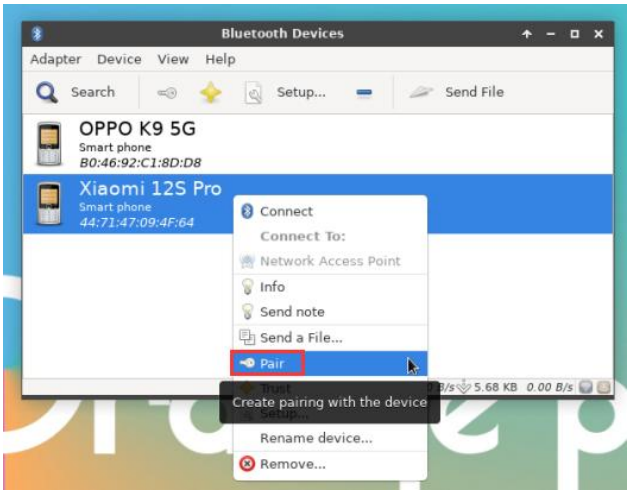
5) Then open the configuration interface of the Bluetooth device



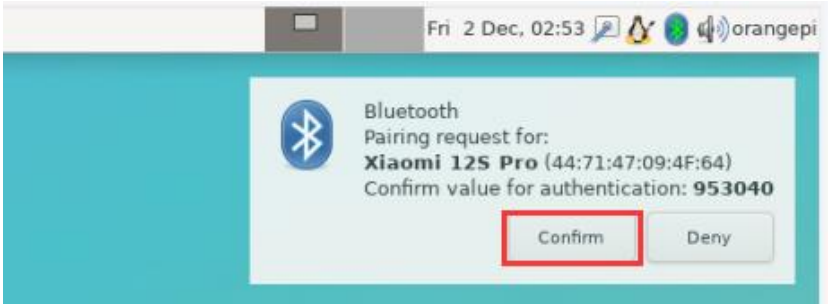
6) Click **Search** to start scanning for surrounding Bluetooth devices



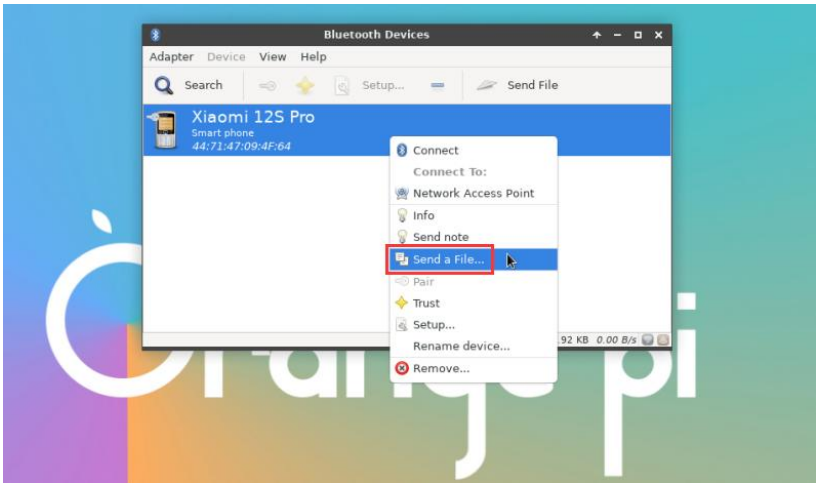
7) Then select the Bluetooth device you want to connect to, and then right-click the mouse to pop up the operation interface of the Bluetooth device. Select **Pair** to start pairing. The demonstration here is pairing with an Android phone.



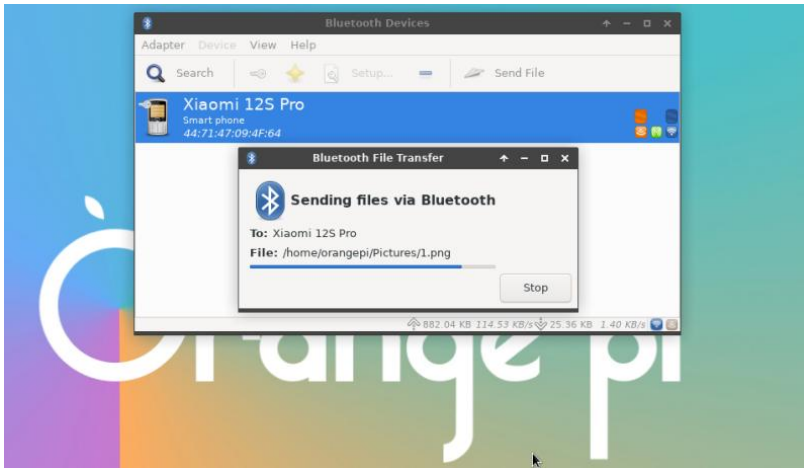
8) During pairing, a pairing confirmation box will pop up in the upper right corner of the desktop. Select **Confirm** to confirm. You will also need to confirm on your phone.



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



10) The interface for sending pictures is as follows



3. 11. 2. How to use the server version image

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

```
orangepi@orangepi:~$ sudo apt update && sudo apt install -y bluez
orangepi@orangepi:~$ hciconfig -a
hci0:   Type: Primary  Bus: UART
        BD Address: 3E:61:3D:19:0E:52  ACL MTU: 1021:8  SCO MTU: 240:3
        UP RUNNING
        RX bytes:925 acl:0 sco:0 events:72 errors:0
        TX bytes:5498 acl:0 sco:0 commands:72 errors:0
        Features: 0xbf 0xff 0x8d 0xfe 0xdb 0x3d 0x7b 0xc7
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT
        Name: 'orangepi'
        Class: 0x3c0000
        Service Classes: Rendering, Capturing, Object Transfer, Audio
        Device Class: Miscellaneous,
        HCI Version: 5.0 (0x9)  Revision: 0x400
        LMP Version: 5.0 (0x9)  Subversion: 0x400
        Manufacturer: Spreadtrum Communications Shanghai Ltd (492)
```

2) Use **bluetoothctl** to scan for Bluetooth devices

```
orangepi@orangepi:~$ sudo bluetoothctl
```

```
[NEW] Controller 10:11:12:13:14:15 orangepi4pro [default]
Agent registered
[bluetooth]# power on          #enable controller
Changing power on succeeded
[bluetooth]# discoverable on       #Set the controller to be 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 surrounding 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 小米手机
[NEW] Device DC:72:9B:4C:F4:CF orangepi
[bluetooth]# scan off          #After scanning to the Bluetooth device you want to connect, you can turn off scanning, 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 start pairing. Pairing requires using the device's MAC address

```
[bluetooth]# pair DC:72:9B:4C:F4:CF          #Pair using the MAC address of the scanned Bluetooth device
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 mobile 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 pairing successful
[CHG] Device DC:72:9B:4C:F4:CF ServicesResolved: no
[CHG] Device DC:72:9B:4C:F4:CF Connected: no
```

4) After successful pairing, the Bluetooth interface of the mobile phone will appear as follows:

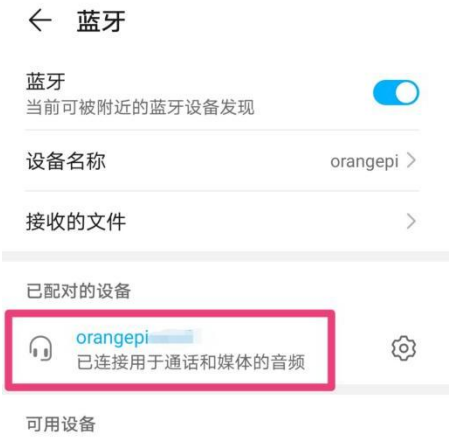
5) To connect to a Bluetooth device, you need to install the **pulseaudio-module-bluetooth** software package, and then start the **pulseaudio** service

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

6) How to connect Bluetooth devices

```
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 to Bluetooth device using 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]#                    #If this prompt appears, the connection is successful.
```

7) After connecting the Bluetooth device, you can see the prompt that **the audio for calls and media has been connected** in the Bluetooth configuration interface of the Android phone.



3. 12. USB interface test

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

3. 12. 1. Connect USB mouse or keyboard to test

1) Insert the USB interface keyboard 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 the operating system normally, it means that the USB interface is working normally (the mouse can only be used in the desktop version of the system)

3. 12. 2. Connect USB storage device for testing

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 the USB disk is successfully recognized.

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

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

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

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

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

3. 12. 3. USB Ethernet card test

1) The currently tested and usable USB Ethernet cards are as follows. Among them, the RTL8153 USB Gigabit network card can be used normally when inserted into the USB 2.0 Host interface of the development board for testing, but the speed cannot reach Gigabit. Please note this.

Serial Number	Model
1	RTL8152B USB 100M network card
2	RTL8153 USB Gigabit network card

2) First insert the USB network card into the USB interface 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
eth1: 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 eth1
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=56 time=6.74 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=56 time=6.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=56 time=6.26 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=56 time=7.27 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 6.260/6.770/7.275/0.373 ms
```

3. 12. 4. USB camera test

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

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

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

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

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

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

Also, the video number may not always be "video0." Please refer to your actual video number.

4) Use fswebcam to test the USB camera

a. Install fswebcam

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

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

a) The -d option is used to specify the device node of the USB camera

b) --no-banner is used to remove watermarks from photos

c) -r option is used to specify the resolution of the photo

d) The -S option is used to skip the previous number of frames

e) /image.jpg is used to set the name and path of the generated photo

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

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

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

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

5) Use mjpg-streamer to test the USB camera

a. Download mjpg-streamer

a) Github download address:

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

b) The mirror download address of Gitee is:

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

b. Install dependent software packages

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

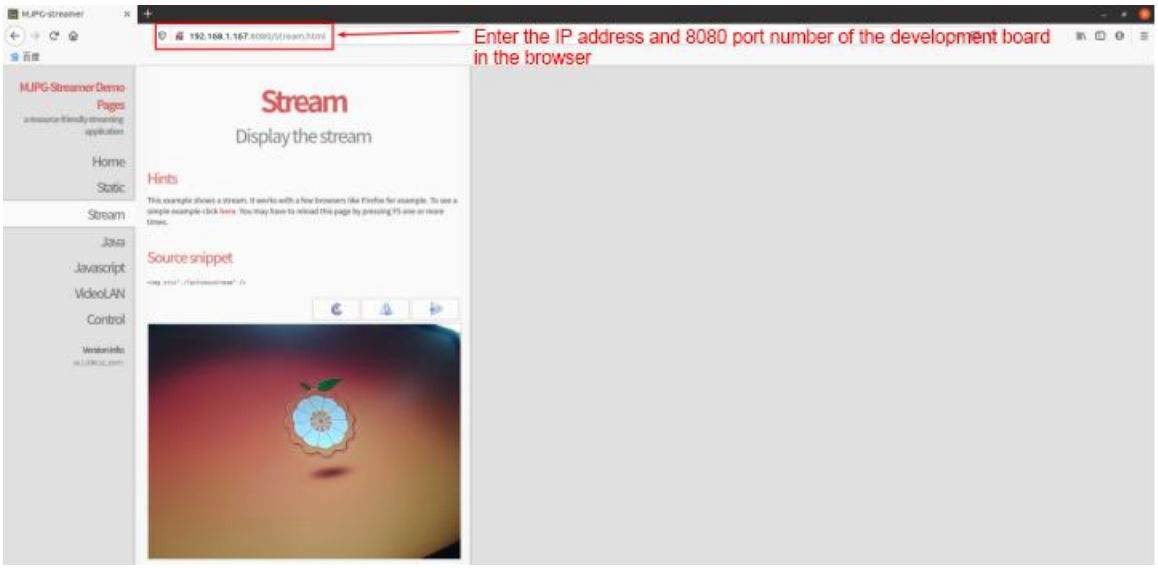
```
orangepi@orangepi:~$ cd mjpg-streamer/mjpg-streamer-experimental
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ make -j4
orangepi@orangepi:~/mjpg-streamer/mjpg-streamer-experimental$ sudo make install
```

d. Then enter the following command to start mjpg_streamer

Note that the serial number of the video is not always video0, please refer to what you actually see.

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

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

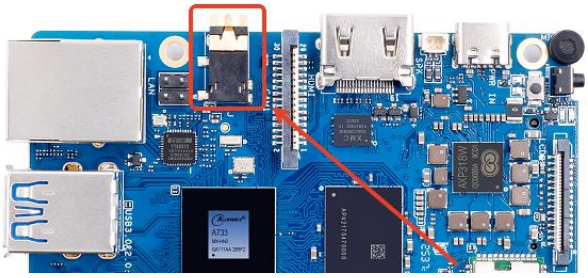


3. 13. Audio test

3. 13. 1. How to play audio using the command line

3. 13. 1. 1. Headphone interface audio playback test

1) First, plug the earphones into the earphone jack on the development board.



2) Use the**aplay -l** command to view the sound card devices supported by the Linux system, where **sndi2s4** is the sound card device required for headphone playback

```
orangepi@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: allwinnerhdmi [allwinner-hdmi], device 0: sunxi-snd-plat-i2s-sunxi-snd-codec-hdmi
soc@3000000:hdmi_codec- []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

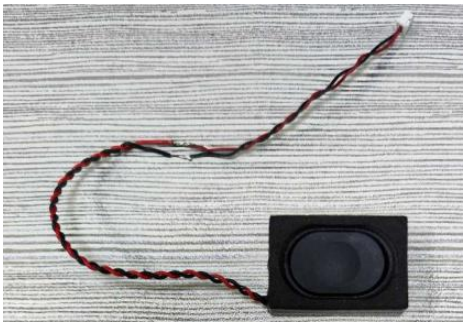
```
card 1: sndi2s4 [sndi2s4], device 0: sunxi-snd-plat-i2s-ES8323 HiFi ES8323.7-0010-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

3) Then use the **apla** command to play the audio, and the headphones will be able to hear the sound

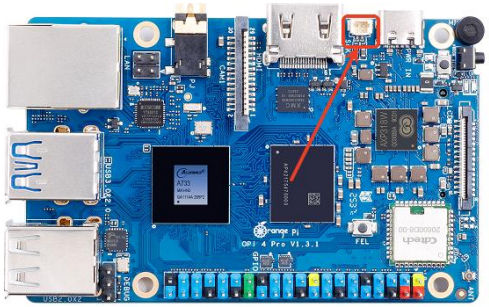
```
root@orangepi:~# aplay -D hw:1,0 /usr/share/sounds/alsa/audio.wav
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

3. 13. 1. 2. Speaker interface audio playback test

1) First, you need to prepare a speaker as shown in the figure below. The speaker socket on the development board has a **2pin 1mm** pitch.



2) The location of the speaker interface on the development board is shown below. After preparing the speaker, please insert it into the speaker interface of the development board.



3) Then use the **aplay** command to play the audio, and the speaker will hear the sound

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

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

3. 13. 1. 3. HDMI Audio Playback Test

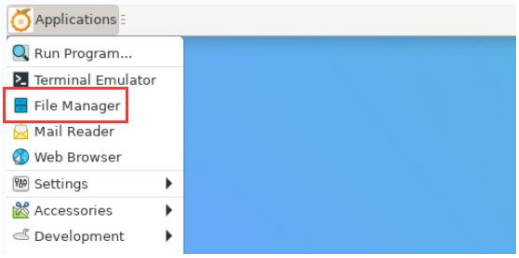
1) First, use an HDMI to HDMI cable to connect the Orange Pi development board to the TV (other HDMI displays must be able to play audio)

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

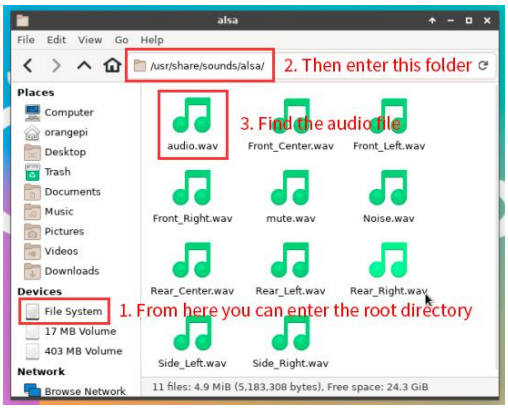
```
orangepi@orangepi:~# aplay -D hw:0,0 /usr/share/sounds/alsa/audio.wav
```

3. 13. 2. Testing Audio Methods on Desktop Systems

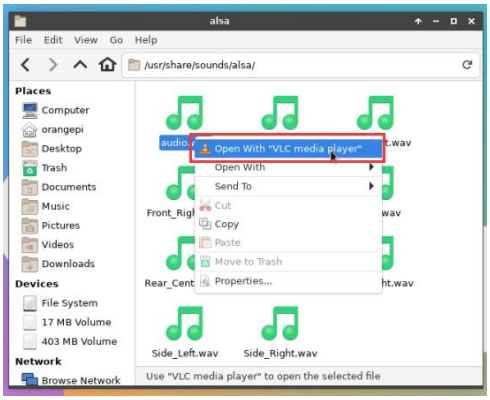
1) First open the file manager



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 yourself)

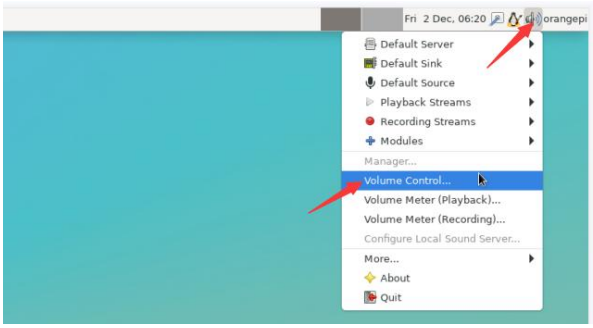


3) Then select the audio.wav file, right-click and select Open with VLC to start playing

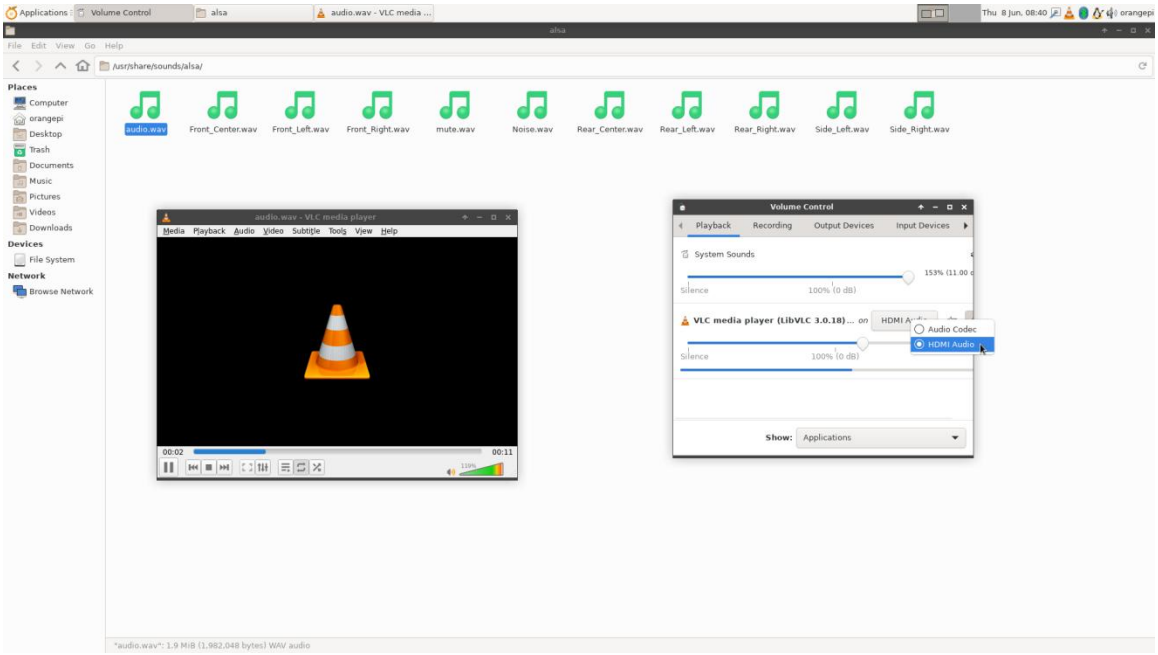


4) How to switch between different audio devices such as HDMI playback and headphone playback

a. First open the volume control interface

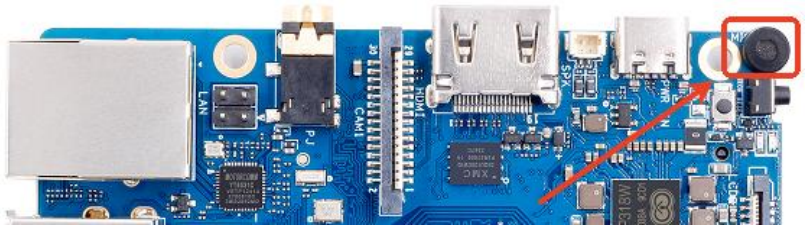


b. When playing audio, the audio device options that the playback software can use will be displayed in **Playback**, as shown in the figure below. Here you can set which audio device you want to play to.



3. 13. 3. How to test recording using commands

1) There is an onboard MIC on the development board, the location is as follows:



2) Run the following command to record an audio clip through the onboard MIC and then play it to HDMI.

```
orangepi@orangepi:~$ arecord -Dhw:1,0 -f cd |aplay -Dhw:0,0 -f cd
```

3) In addition to the onboard MIC, you can also record audio through headphones with a MIC function. After plugging a headphone with a MIC function into the development board, run the following command to record audio through the headphones and then play it to HDMI.

```
orangepi@orangepi:~$ sudo i2cset -y -f 7 0x10 0x23 0x10
orangepi@orangepi:~$ arecord -Dhw:1,0 -f cd |aplay -Dhw:0,0 -f cd
```

3. 14. Temperature sensor

A733 has a total of 5 temperature sensors. The command to check the temperature is as follows:

The displayed temperature value needs to be divided by 1000 to get the unit in Celsius.

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

```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone0/type
cpul_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone0/temp
54925
```

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

```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone1/type
cpub_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone1/temp
54990
```

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

```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone4/type
gpu_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone4/temp
55056
```

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

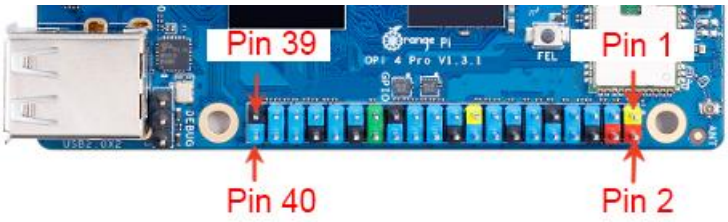
```
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone5/type
npu_thermal_zone
orangepi@orangepi:~$ cat /sys/class/thermal/thermal_zone5/temp
54686
```

e. sensor4：DDR temperature sensor, the first command is used to view the type of temperature sensor, the second command is used to view the value of the temperature sensor

```
orangepi@orangepi:~# cat /sys/class/thermal/thermal_zone6/type
ddr_thermal_zone
orangepi@orangepi:~# cat /sys/class/thermal/thermal_zone6/temp
54925
```

3. 15. 40 Pin Interface Pin Description

1) For the pinout of the Orange Pi 4 Pro development board's 40-pin interface, refer to the silkscreen on the board.



2) The functions of the 40-pin interface pins of the development board are shown in the following table

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

3) There are 28 GPIO ports in the 40-pin interface, and the voltage of all GPIO ports is **3.3v**

3. 16. How to install wiringOP

Note that wiringOP is pre-installed in the Linux image released by Orange Pi. Unless the wiringOP code is updated, there is no need to download, compile and install it again. You can use it directly.

The compiled wiringOP deb package is stored in the orangepi-build directory: orangepi-build/external/cache/debs/arm64/wiringpi_x.xx.deb

After logging into the system, run the gpio readall command. If you see the following output, it means wiringOP has been pre-installed and can be used normally.

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |    SDA.0 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |    SCL.0 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
 |  361 |   5 |      PL9 |    OFF | 0 | 11 || 12 | 0 | OFF    | PL8      | 6   | 360  |
 |  364 |   7 |     PL12 |    OFF | 0 | 13 || 14 |   |        | GND      |     |      |
 |  344 |   8 |     PK24 |    OFF | 0 | 15 || 16 | 0 | OFF    | PL5      | 9   | 357  |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | OFF    | PK25     | 10  | 345  |
 |  130 |  11 |   MOSI.3 |    OFF | 0 | 19 || 20 |   |        | GND      |     |      |
 |  131 |  12 |   MISO.3 |    OFF | 0 | 21 || 22 | 0 | OFF    | PD23     | 13  | 119  |
 |  129 |  14 |   SCLK.3 |    OFF | 0 | 23 || 24 | 1 | OUT    | CE.0     | 15  | 140  |
 |      |     |      GND |        |   | 25 || 26 | 0 | OFF    | PE4      | 16  | 132  |
 |   37 |  17 |      PB5 |    OFF | 0 | 27 || 28 | 0 | OFF    | PB4      | 18  | 36   |
 |   96 |  19 |      PD0 |    OFF | 0 | 29 || 30 |   |        | GND      |     |      |
 |   97 |  20 |      PD1 |    OFF | 0 | 31 || 32 | 0 | OFF    | PD5      | 21  | 101  |
 |   98 |  22 |      PD2 |    OFF | 0 | 33 || 34 |   |        | GND      |     |      |
 |   99 |  23 |      PD3 |    OFF | 0 | 35 || 36 | 0 | OFF    | PD6      | 24  | 102  |
 |  100 |  25 |      PD4 |    OFF | 0 | 37 || 38 | 0 | OFF    | PD7      | 26  | 103  |
 |      |     |      GND |        |   | 39 || 40 | 0 | OFF    | PL13     | 27  | 365  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+ OPI 4 PRO+---+--------+----------+-----+------+
```

1) Download the wiringOP code

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

Note that the source code needs to be downloaded from the wiringOP next branch. Please do not omit the -b next parameter.

If you have problems downloading the code from GitHub, you can directly use the wiringOP source code that comes with the Linux image, which is stored in: /usr/src/wiringOP.

2) Compile and install wiringOP

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

3) Test the output of the gpio readall command as follows

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |   SDA.0  |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |   SCL.0  |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
 |  361 |   5 |      PL9 |    OFF | 0 | 11 || 12 | 0 | OFF    | PL8      | 6   | 360  |
 |  364 |   7 |     PL12 |    OFF | 0 | 13 || 14 |   |        | GND      |     |      |
 |  344 |   8 |     PK24 |    OFF | 0 | 15 || 16 | 0 | OFF    | PL5      | 9   | 357  |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | OFF    | PK25     | 10  | 345  |
 |  130 |  11 |  MOSI.3  |    OFF | 0 | 19 || 20 |   |        | GND      |     |      |
 |  131 |  12 |  MISO.3  |    OFF | 0 | 21 || 22 | 0 | OFF    | PD23     | 13  | 119  |
 |  129 |  14 |  SCLK.3  |    OFF | 0 | 23 || 24 | 1 | OUT    | CE.0     | 15  | 140  |
 |      |     |      GND |        |   | 25 || 26 | 0 | OFF    | PE4      | 16  | 132  |
 |   37 |  17 |      PB5 |    OFF | 0 | 27 || 28 | 0 | OFF    | PB4      | 18  | 36   |
 |   96 |  19 |      PD0 |    OFF | 0 | 29 || 30 |   |        | GND      |     |      |
 |   97 |  20 |      PD1 |    OFF | 0 | 31 || 32 | 0 | OFF    | PD5      | 21  | 101  |
 |   98 |  22 |      PD2 |    OFF | 0 | 33 || 34 |   |        | GND      |     |      |
 |   99 |  23 |      PD3 |    OFF | 0 | 35 || 36 | 0 | OFF    | PD6      | 24  | 102  |
 |  100 |  25 |      PD4 |    OFF | 0 | 37 || 38 | 0 | OFF    | PD7      | 26  | 103  |
 |      |     |      GND |        |   | 39 || 40 | 0 | OFF    | PL13     | 27  | 365  |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+ OPI 4 PRO+---+--------+----------+-----+------+
```

3. 17. 40pin interface GPIO, I2C, UART, SPI and PWM test

3. 17. 1. 40pin GPIO port test

1) The following uses pin 7, which corresponds to GPIO PL4 and wPi number 2, as an example to demonstrate how to set the high and low levels of the GPIO port.

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |   SDA.0  |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |   SCL.0  |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
 |  361 |   5 |      PL9 |    OFF | 0 | 11 || 12 | 0 | OFF    | PL8      | 6   | 360  |
```

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

```
orangepi@orangepi:~/wiringOP$ gpio mode 2 out
```

3) Then set the GPIO port to output 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 setting is successful.

```
orangepi@orangepi:~/wiringOP$ gpio write 2 0
```

Using gpio readall, you can see that the value of pin 7 (V) has changed to 0

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |    SDA.0 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |    SCL.0 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OUT | 0 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
```

4) Then set the GPIO port to output 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 setting is successful.

```
orangepi@orangepi:~/wiringOP$ gpio write 2 1
```

Using gpio readall, you can see that the value of pin 7 (V) has changed to 1

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |    SDA.0 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |    SCL.0 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OUT | 1 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
```

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

3.17.2. How to set pull-up and pull-down resistors on GPIO pins

1) The following uses pin 7, which corresponds to GPIO PL4 and wPi number 2, as an example to demonstrate how to set the pull-up and pull-down resistors of the GPIO port.

```
orangepi@orangepi4pro:~$ gpio readall
 +------+-----+----------+--------+---+ OPI 4PRO +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   35 |   0 |    SDA.0 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |   34 |   1 |    SCL.0 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |  356 |   2 |  PWM0-2  |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.0    | 3   | 358  |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.0    | 4   | 359  |
 |  361 |   5 |      PL9 |    OFF | 0 | 11 || 12 | 0 | OFF    | PL8      | 6   | 360  |
```

2) First, you need to set the GPIO port to input mode. The third parameter needs to enter the wPi number corresponding to the pin.

```
orangepi@orangepi:~/wiringOP$ gpio mode 2 in
```

3) After setting to input mode, execute the following command to set the GPIO port to pull-up mode

```
orangepi@orangepi:~/wiringOP$ gpio mode 2 up
```

4) Then enter the following command to read the level of the GPIO port. If the level is 1, it means that the pull-up mode is set successfully.

```
orangepi@orangepi:~/wiringOP$ gpio read 2
1
```

5) Then execute the following command to set the GPIO port to pull-down mode

```
orangepi@orangepi:~/wiringOP$ gpio mode 2 down
```

6) Then enter the following command to read the level of the GPIO port. If the level is 0, it means that the pull-down mode is set successfully.

```
orangepi@orangepi:~/wiringOP$ gpio read 2
0
```

3.17.3. 40-pin SPI test

1) As shown in the figure below, the available SPI for Orange Pi 4 Pro is SPI3

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of SPI3 in 40pin are shown in the following table.

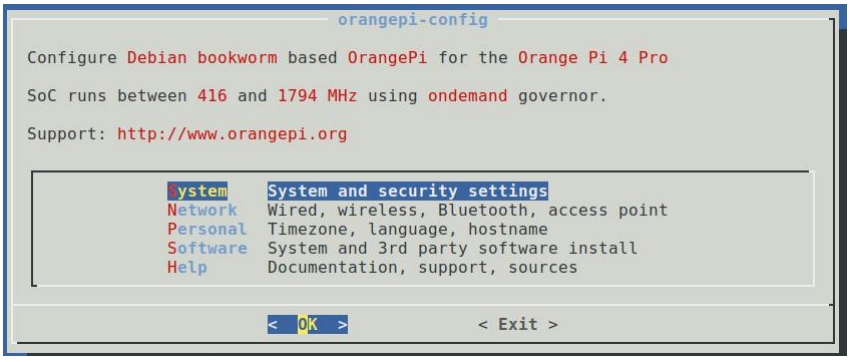
	SPI3 corresponds to 40pin
MOSI	**Pin 19**
MISO	**Pin 21**
CLK	**Pin 23**
CS0	**Pin 24**

3) In Linux systems, the SPI pin 40 is disabled by default and needs to be enabled manually. The detailed steps are as follows:

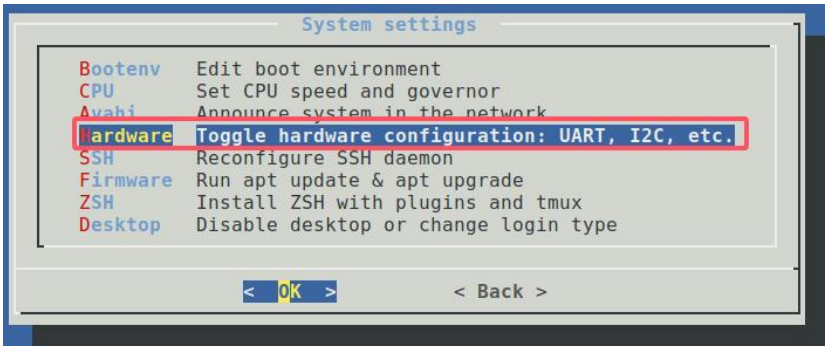
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. Select **System**



c. Then select **Hardware**

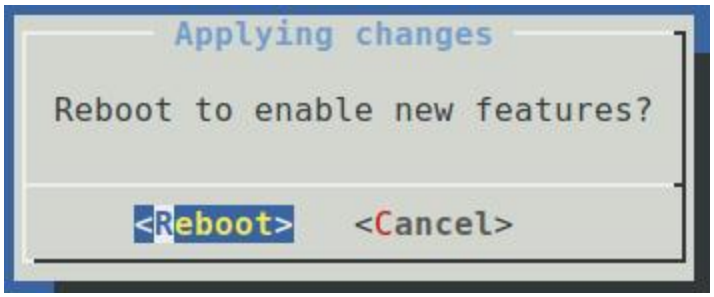


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open SPI configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) Then check whether the device node **spidevx.x** exists in the Linux system. If it exists, it means that the SPI configuration has taken effect.

```
orangepi@orangepi:~$ ls /dev/spidev*
/dev/spidev3.0
```

5) Do not short the mosi and miso pins of SPI3. The output of running spidev_test is as follows. You can see that the data of TX and RX are inconsistent.

```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev3.0
```

```
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    | ......@....▒.....................▒.
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF    | ................................
```

6) Then short the mosi and miso pins of SPI3 and run spidev_test. The output is as follows. You can see that the data sent and received are the same.

```
orangepi@orangepi:~$ sudo spidev_test -v -D /dev/spidev3.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF 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. 17. 4. 40 Pin I2C Test

1) As shown in the figure below, the Orange Pi 4 Pro has four I2C buses: i2c0, i2c1, i2c2, and i2c3.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of the four I2C bus groups in the 40-pin are shown in the following table.

I2C bus	SDA corresponds to 40pin	SCL corresponds to 40pin	dtbo corresponding configuration
I2C0	Pin 3	Pin 5	i2c0

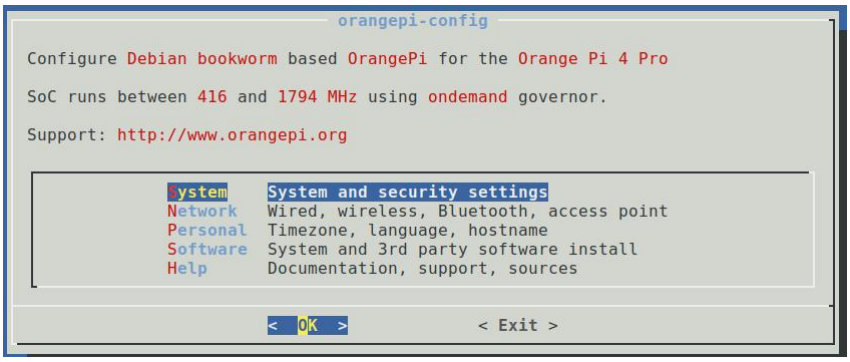
I2C1	Pin 27	Pin 28	i2c1
I2C2	Pin 19	Pin 23	i2c2
I2C3	Pin 26	Pin 21	i2c3

3) In Linux, the I2C in the 40 pins is disabled by default and needs to be enabled manually. The detailed steps are as follows

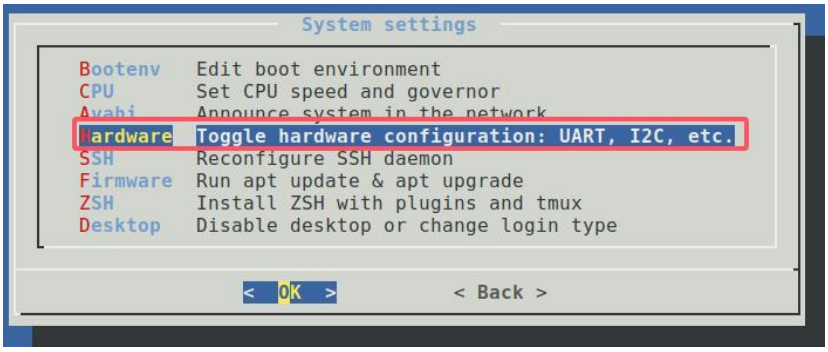
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. The select **System**



c. The select **Hardware**

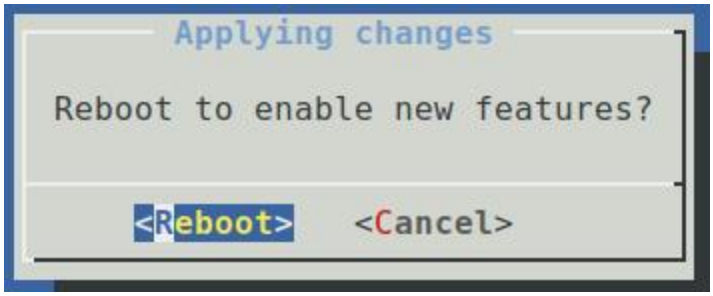


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open I2C configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) After starting the Linux system, first confirm that the i2c device node exists under /dev

```
orangepi@orangepi:~$ ls /dev/i2c-*
/dev/i2c-0  /dev/i2c-1  /dev/i2c-13  /dev/i2c-15  /dev/i2c-2  /dev/i2c-20
/dev/i2c-3  /dev/i2c-5  /dev/i2c-7  /dev/i2c-8  /dev/i2c-9
```

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

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt-get install -y i2c-tools
```

6) Then connect an i2c device to the i2c pin of the 40 pin connector

Please choose the 5V and 3.3V pins according to the specific i2c device, as different i2c devices may require different voltage values.

7) Then use the i2cdetect -y command. If the address of the connected i2c device can be detected, it means that i2c can be used normally.

```
orangepi@orangepi:~$ sudo i2cdetect -y -r 0      #i2c0 command
orangepi@orangepi:~$ sudo i2cdetect -y -r 1      #i2c1 command
```

```
orangepi@orangepi:~$ sudo i2cdetect -y -r 2      #i2c2 command
orangepi@orangepi:~$ sudo i2cdetect -y -r 3      #i2c3 command
```

3.17.5. 40 UART test of pin

1) As can be seen from the table below, the available UART for Orange Pi 4 Pro is UART7.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins in the 40 pin UART bus are shown in the table below.

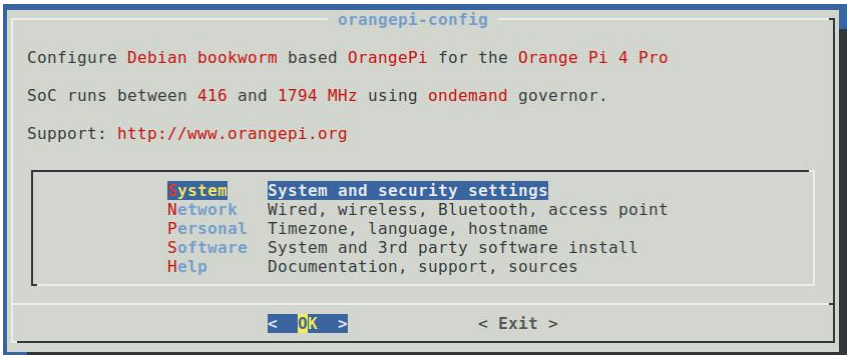
UART bus	RX corresponds to 40pin	TX corresponds to 40pin	dtbo corresponding configuration
UART7	Pin 10	Pin 8	uart7

3) In Linux systems, the UART in pin 40 is disabled by default and needs to be enabled manually. The detailed steps are as follows:

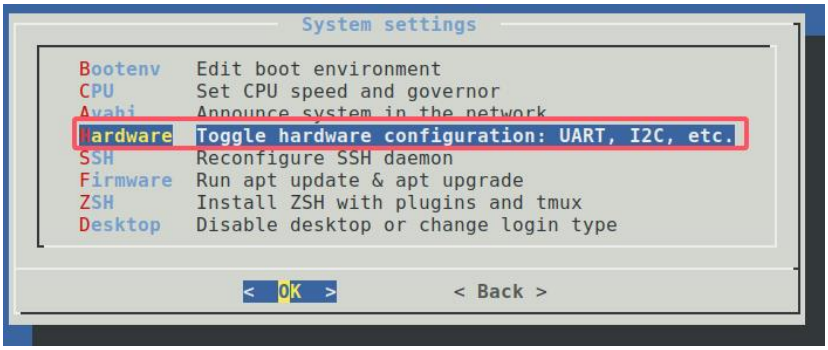
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. The select **System**



c. The select **Hardware**

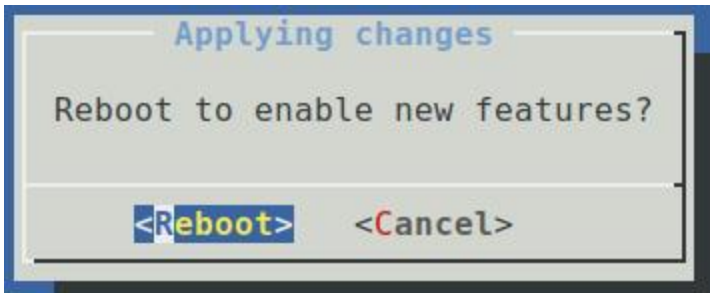


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open UART configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) After entering the Linux system, first confirm whether there is a uart device node under **/dev**

```
orangepi@orangepi:~$ ls /dev/ttyS*
/dev/ttyS0   /dev/ttyS1   /dev/ttyS6   /dev/ttyS7
```

5) Then start testing the UART interface. First use the Dupont line to short-circuit the rx and tx of the UART interface to be tested.

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

```
orangepi@orangepi:~$ gpio serial /dev/ttyS7


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

3.17.6. How to test PWM using /sys/class/pwm/

1) As can be seen from the table below, the Orange Pi 4 Pro has 8 PWM channels: spwm0_2, pwm0_0, pwm0_2, pwm0_3, pwm0_4, pwm0_5, pwm0_6, and pwm0_7

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of PWM in 40 pins are shown in the following table.

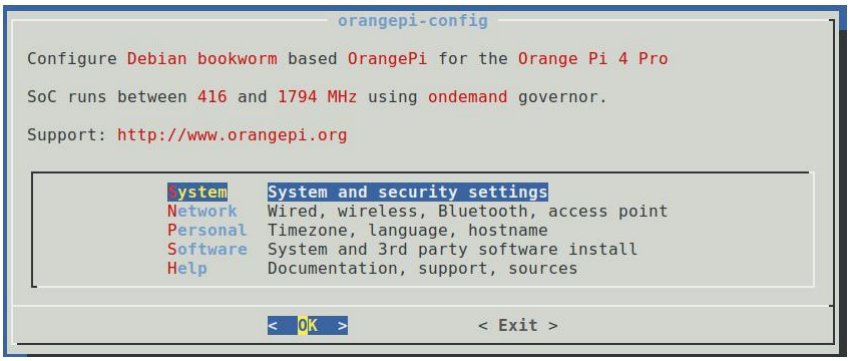
PWM Bus	Corresponding to 40pin	dtbo corresponding configuration
SPWM0-2	Pin 7	spwm2
PWM0-0	Pin 29	pwm0
PWM0-2	Pin 33	pwm2
PWM0-3	Pin 35	pwm3
PWM0-4	Pin 37	pwm4
PWM0-5	Pin 32	pwm5
PWM0-6	Pin 36	pwm6
PWM0-7	Pin 38	pwm7

3) In Linux systems, the PWM on the 40 pins is disabled by default and needs to be enabled manually. The detailed steps are as follows:

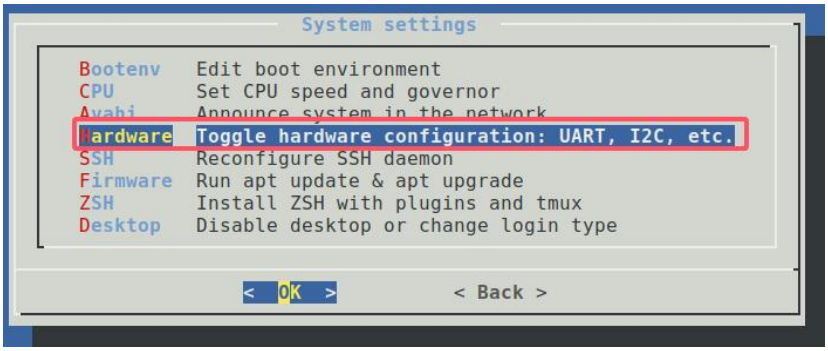
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

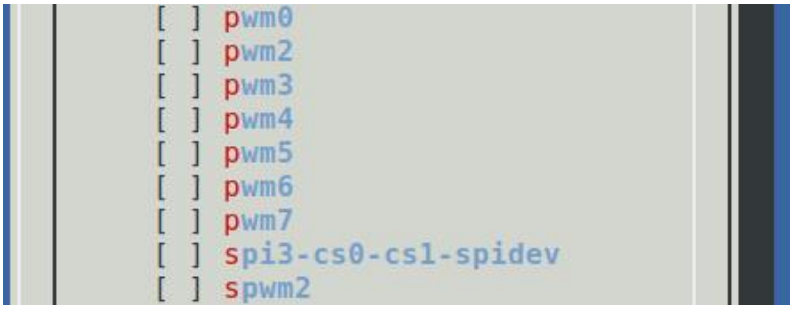
b. Then select **System**



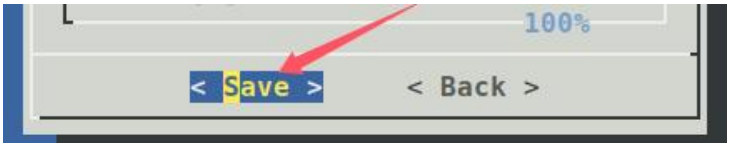
c. Then select **Hardware**



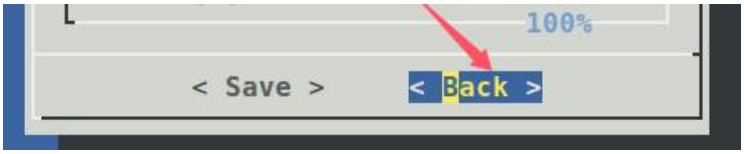
d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the PWM configuration to open



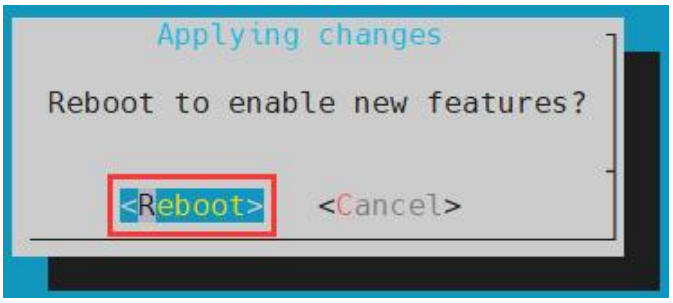
e. Then select **<Save>** to save



f. Then select **<Back>**



g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) After restarting, you can start the PWM test

Please execute the following command under the root user.

a. The command to test s_pwm0_2 is as follows

```
root@orangepi:~# echo 2 > /sys/class/pwm/pwmchip20/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip20/pwm2/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip20/pwm2/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip20/pwm2/enable
```

b. Enter the following command in the command line to make pwm0_0 output a 50Hz square wave

```
root@orangepi:~# echo 0 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm0/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable
```

c. Enter the following command in the command line to make pwm2 output a 50Hz square wave

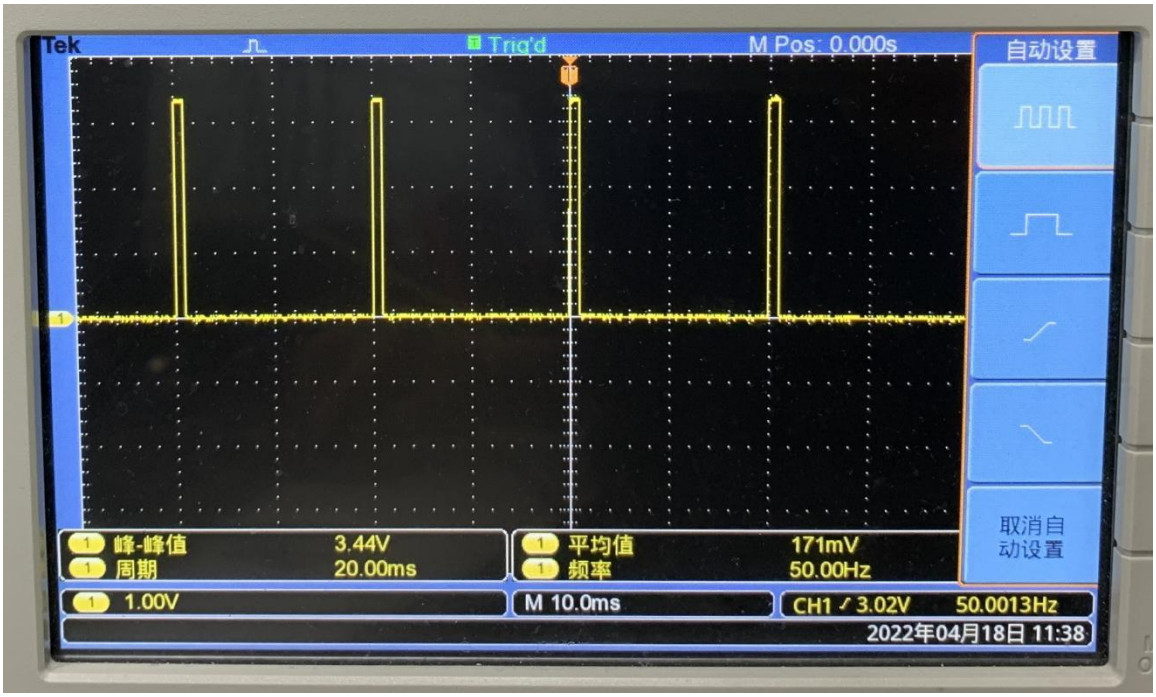
```
root@orangepi:~# echo 2 > /sys/class/pwm/pwmchip0/export
root@orangepi:~# echo 20000000 > /sys/class/pwm/pwmchip0/pwm2/period
root@orangepi:~# echo 1000000 > /sys/class/pwm/pwmchip0/pwm2/duty_cycle
root@orangepi:~# echo 1 > /sys/class/pwm/pwmchip0/pwm2/enable
```

d. Enter the following command in the command line to make pwm4 output a 50Hz square 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. Other PWM test methods are similar and will not be described here.



3. 18. Installation and Usage of wiringOP-Python

wiringOP-Python is a Python-language version of wiringOP, allowing you to access the development board's GPIO, I2C, SPI, and UART hardware resources from Python programs.

Please note that all commands below are executed as theroot user.

3. 18. 1. Installation of wiringOP-Python

1) First install the dependency package

```
root@orangepi:~# sudo apt-get update
root@orangepi:~# sudo apt-get -y install git swig python3-dev python3-setuptools
```

2) Then use the following command to download the source code of wiringOP-Python

Note that the following git clone--recursive command will automatically download the wiringOP source code, as wiringOP-Python depends on wiringOP. Please ensure that the download process does not cause errors due to network problems.

If you have problems downloading the code from GitHub, you can directly use the wiringOP-Python source code that comes with the Linux image, which is stored in: /usr/src/wiringOP-Python

```
root@orangepi:~# git clone --recursive https://github.com/orangepi-xunlong/wiringOP-Python -b next
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# git submodule update --init --remote
```

3) Then use the following command to compile wiringOP-Python and install it into the Linux system of the development board

```
root@orangepi:~# cd wiringOP-Python
root@orangepi:~/wiringOP-Python# python3 generate-bindings.py > bindings.i
root@orangepi:~/wiringOP-Python# sudo python3 setup.py install
```

4) Then enter the following command. If help information is output, it means that wiringOP-Python has been successfully installed. Press the **q** key to exit the help information interface.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi;  help(wiringpi)"
Help on module wiringpi:

NAME
    wiringpi

DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.
```

5) The steps to test whether wiringOP-Python is successfully installed in the Python command line are as follows:

a. First use the python3 command to enter the python3 command line mode

```
root@orangepi:~# python3
```

b. Then import the Python module of wiringPi

```
>>> import wiringpi;
```

c. Finally, enter the following command to view the help information of wiringOP-Python. Press the **q** key to exit the help information interface.

```
>>> help(wiringpi)
Help on module wiringpi:

NAME
    wiringpi

DESCRIPTION
    # This file was automatically generated by SWIG (http://www.swig.org).
    # Version 4.0.2
    #
    # Do not make changes to this file unless you know what you are doing--modify
    # the SWIG interface file instead.

CLASSES
    builtins.object
        GPIO
        I2C
        Serial
        nes

    class GPIO(builtins.object)
     |  GPIO(pinmode=0)
     |

>>>
```

3. 18. 2. 40 pin GPIO port test

Like wiringOP, wiringOP-Python can also determine which GPIO pin to operate by specifying the wPi number. Because there is no command to view the wPi number in wiringOP-Python, the only way to view the correspondence between the board's wPi number and the physical pin is through the gpio command in wiringOP.

```
orangepi@orangepi4a:~$ gpio readall
 +------+-----+----------+--------+---+  OPI 4A  +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  257 |   0 |    SDA.4 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |  256 |   1 |    SCL.4 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |   36 |   2 |     PWM8 |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.7    | 3   | 45   |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.7    | 4   | 46   |
 |   32 |   5 |    TXD.2 |    OFF | 0 | 11 || 12 | 0 | OFF    | PB05     | 6   | 37   |
 |   33 |   7 |    RXD.2 |    OFF | 0 | 13 || 14 |   |        | GND      |     |      |
 |   34 |   8 |     PB02 |    OFF | 0 | 15 || 16 | 0 | OFF    | PI13     | 9   | 269  |
 |      |     |     3.3V |        |   | 17 || 18 | 0 | OFF    | PI14     | 10  | 270  |
 |  260 |  11 |   MOSI.1 |    OFF | 0 | 19 || 20 |   |        | GND      |     |      |
 |  261 |  12 |   MISO.1 |    OFF | 0 | 21 || 22 | 0 | OFF    | TXD.6    | 13  | 262  |
 |  259 |  14 |   SCLK.1 |    OFF | 0 | 23 || 24 | 0 | OFF    | CE.1     | 15  | 258  |
 |      |     |      GND |        |   | 25 || 26 | 0 | OFF    | RXD.6    | 16  | 263  |
 |  265 |  17 |    SDA.5 |    OFF | 0 | 27 || 28 | 0 | OFF    | SCL.5    | 18  | 264  |
 |   35 |  19 |     PB03 |    OFF | 0 | 29 || 30 |   |        | GND      |     |      |
 |   43 |  20 |     PB11 |    OFF | 0 | 31 || 32 | 0 | OFF    | PWM12    | 21  | 267  |
 |  268 |  22 |    PWM13 |    OFF | 0 | 33 || 34 |   |        | GND      |     |      |
 |   38 |  23 |     PB06 |    OFF | 0 | 35 || 36 | 0 | OFF    | PI10     | 24  | 266  |
 |   44 |  25 |     PB12 |    OFF | 0 | 37 || 38 | 0 | OFF    | PB07     | 26  | 39   |
 |      |     |      GND |        |   | 39 || 40 | 0 | OFF    | PB08     | 27  | 40   |
 +------+-----+----------+--------+---+  OPI 4A  +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
```

1) The following uses pin 7, which corresponds to GPIO PB4 and wPi number 2, as an example to demonstrate how to set the high and low levels of the GPIO port.

```
orangepi@orangepi4a:~$ gpio readall
 +------+-----+----------+--------+---+  OPI 4A  +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |  257 |   0 |    SDA.4 |    OFF | 0 |  3 || 4  |   |        | 5V       |     |      |
 |  256 |   1 |    SCL.4 |    OFF | 0 |  5 || 6  |   |        | GND      |     |      |
 |   36 |   2 |     PWM8 |    OFF | 0 |  7 || 8  | 0 | OFF    | TXD.7    | 3   | 45   |
 |      |     |      GND |        |   |  9 || 10 | 0 | OFF    | RXD.7    | 4   | 46   |
```

2) The steps for testing directly using commands are as follows:

a. First, set the GPIO port to output mode. The first parameter of the **pinMode** function is the wPi number corresponding to the pin, and the second parameter is the GPIO mode.

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
from wiringpi import GPIO; wiringpi.wiringPiSetup() ; \
```

```
wiringpi.pinMode(2, GPIO.OUTPUT) ;  "
```

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

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.LOW)"
```

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

```
root@orangepi:~/wiringOP-Python# python3 -c "import wiringpi; \
 from wiringpi import GPIO; wiringpi.wiringPiSetup() ;\
wiringpi.digitalWrite(2, GPIO.HIGH)"
```

3) The steps for testing in the Python 3 command line are as follows:

a. First use the python3 command to enter the python3 command line mode

```
root@orangepi:~# python3
```

b. Then import the Python module of wiringPi

```
>>> import wiringpi
>>> from wiringpi import GPIO
```

c. Then set the GPIO port to output mode, where the first parameter of the **pinMode** function is the wPi number corresponding to the pin, and the second parameter is the GPIO mode

```
>>> wiringpi.wiringPiSetup()
0
>>> wiringpi.pinMode(2, GPIO.OUTPUT)
```

d. Then set the GPIO port to output 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 setting is successful.

```
>>> wiringpi.digitalWrite(2, GPIO.LOW)
```

e. Then set the GPIO port to output 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 setting is successful.

```
>>> wiringpi.digitalWrite(2, GPIO.HIGH)
```

4) WiringOP-Python: For setting the GPIO high and low levels in Python code, refer to the **blink.p** test program in the examples. The **blink.p** test program will set the voltage of all GPIO ports on the development board's 40 pins to continuously change.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# ls blink.py
blink.py
root@orangepi:~/wiringOP-Python/examples# python3 blink.py
```

3.18.3. 40 Pin SPI Test

1) As shown in the figure below, the available SPI for Orange Pi 4 Pro is SPI3.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of SPI3 on 40pin are shown in the following table

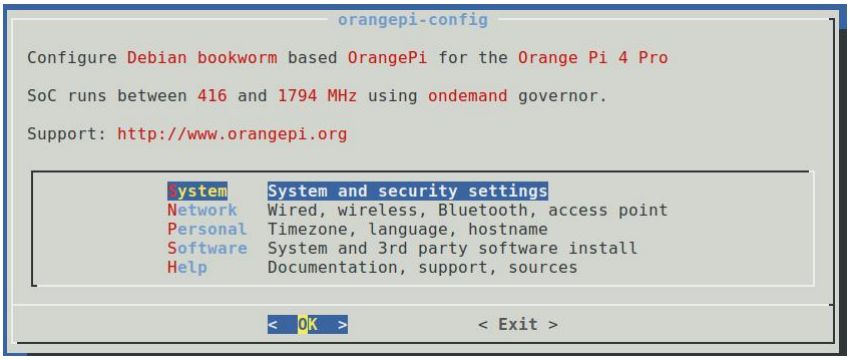
	SPI3 corresponds to 40pin
MOSI	**Pin 19**
MISO	**Pin 21**
CLK	**Pin 23**
CS0	**Pin 24**

3) In Linux systems, the SPI pin 40 is disabled by default and needs to be enabled manually. The detailed steps are as follows:

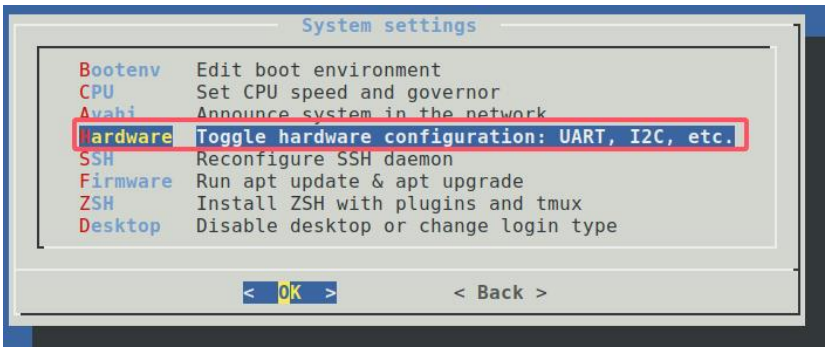
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. Then Select **System**



c. Then Select **Hardware**

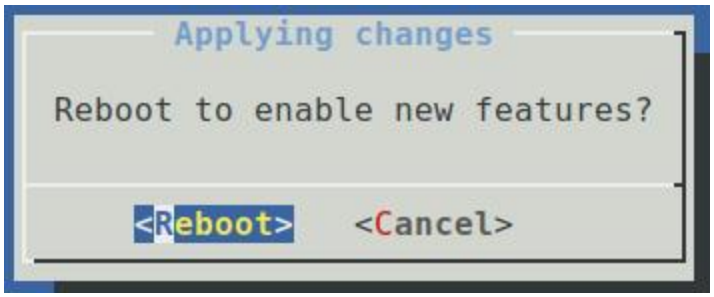


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open SPI configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) Then check whether the device node **spidevx.x** exists in the Linux system. If it exists, it means that the SPI configuration has taken effect.

```
orangepi@orangepi:~$ ls /dev/spidev*
/dev/spidev3.0
```

5) Then you can use the **spidev_test.py** program in the examples to test the SPI loopback function. The **spidev_test.py** program needs to specify the following two parameters:

a. **--channel：** Specify the SPI channel number
b. **--port：** Specify the SPI port number

6) Without shorting the mosi and miso pins of the SPI, the output of running spidev_test.py is as follows. You can see that the TX and RX data are inconsistent.

The x after the --channel and --port parameters needs to be replaced with the specific SPI channel number and SPI port number.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py \
--channel x --port x
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev1.1
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    |................|
```

7) Then use a DuPont line to short the SPI's TXD and RXD pins and run spidev_test.py. The output is as follows: you can see that the sent and received data are the same, indicating that the SPI loopback test is normal.

The x after the --channel and --port parameters needs to be replaced with the

specific SPI channel number and SPI port number.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 spidev_test.py \
--channel x --port x
spi mode: 0x0
max speed: 500000 Hz (500 KHz)
Opening device /dev/spidev1.1
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    |......@.....|
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D    |......@.....|
```

3. 18. 4. 40 Pin I2C Test

1) As can be seen from the table below, the Orange Pi 4 Pro has four i2c buses: i2c0, i2c1, i2c2, and i2c3.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of the three I2C buses in the 40-pin are shown in the following table.

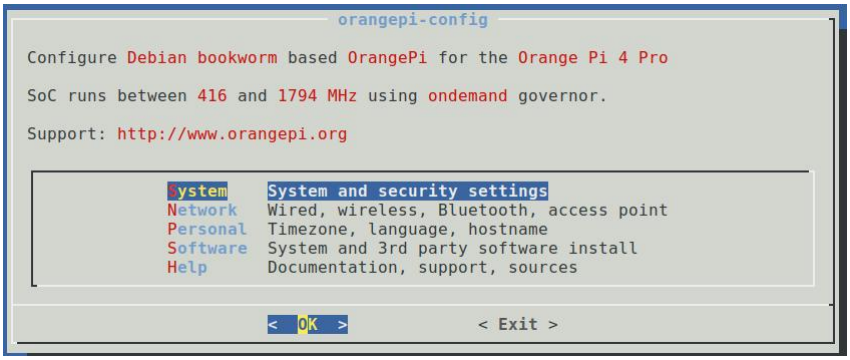
I2C bus	SDA corresponds to 40pin	SCL corresponds to 40pin	dtbo corresponding configuration
I2C0	Pin 3	Pin 5	i2c0
I2C1	Pin 27	Pin 28	i2c1
I2C2	Pin 19	Pin 23	i2c2
I2C3	Pin 26	Pin 21	i2c3

3) In Linux systems, the I2C pins in the 40 pins are disabled by default and need to be enabled manually. The detailed steps are as follows:

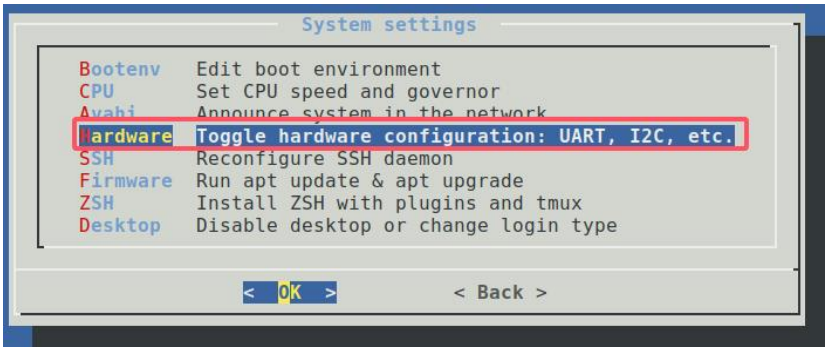
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. Then selecct **System**



c. Then selecct **Hardware**

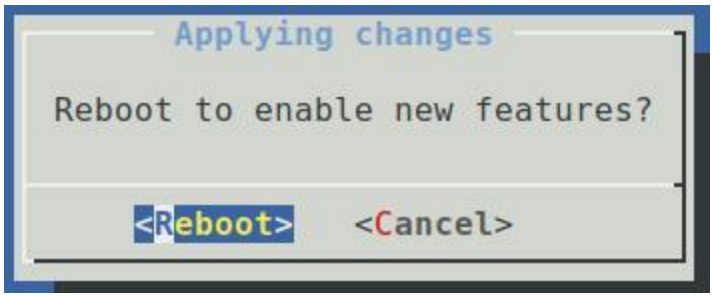


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open I2C configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) After starting the Linux system, first confirm that the i2c device node exists under /dev

```
orangepi@orangepi:~$ ls /dev/i2c-*
/dev/i2c-0  /dev/i2c-1  /dev/i2c-13  /dev/i2c-15  /dev/i2c-2  /dev/i2c-20
/dev/i2c-3  /dev/i2c-5  /dev/i2c-7  /dev/i2c-8  /dev/i2c-9
```

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

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt-get install -y i2c-tools
```

6) Then connect an i2c device to the i2c pin of the 40-pin connector. Here we use the DS1307 RTC module as an example.

7) Then use the **i2cdetect -y** command. If the address of the connected i2c device can be detected, it means that the i2c device is connected correctly.

```
orangepi@orangepi:~$ sudo i2cdetect -y -r 0      #Command for i2c0
orangepi@orangepi:~$ sudo i2cdetect -y -r 1      #Command for i2c1
orangepi@orangepi:~$ sudo i2cdetect -y -r 2      #Command for i2c2
orangepi@orangepi:~$ sudo i2cdetect -y -r 3      #Command for i2c3
```

8) Then you can run the **ds1307.py** test program in the **examples** to read the RTC time.

```
root@orangepi:~/wiringOP-Python# cd examples
```

```
root@orangepi:~/wiringOP-Python/examples# python3 ds1307.py --device \
"/dev/i2c-1"
Thu 2022-06-16  04:35:46
Thu 2022-06-16  04:35:47
Thu 2022-06-16  04:35:48
^C
exit
```

3.18.5. 40 UART test of pin

1) As can be seen from the table below, the available UART for Orange Pi 4 Pro is UART7.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) The corresponding pins of the six UART bus groups in the 40 pins are shown in the following table.

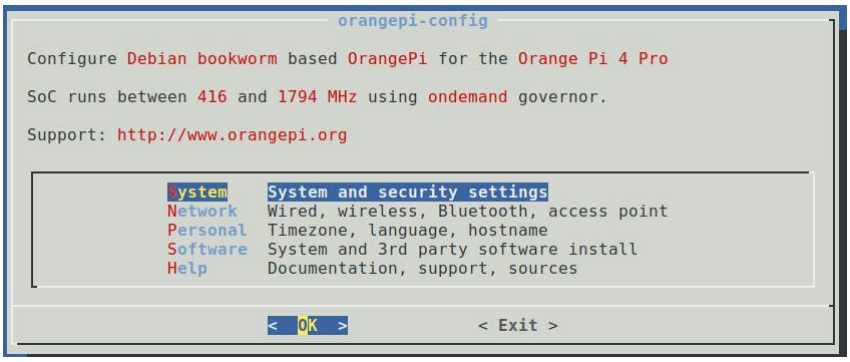
UART bus	RX corresponds to 40pin	TX corresponds to 40pin	dtbo corresponding configuration
UART7	Pin 10	Pin 8	uart7

3) In Linux systems, the UART in pin 40 is disabled by default and needs to be enabled manually. The detailed steps are as follows:

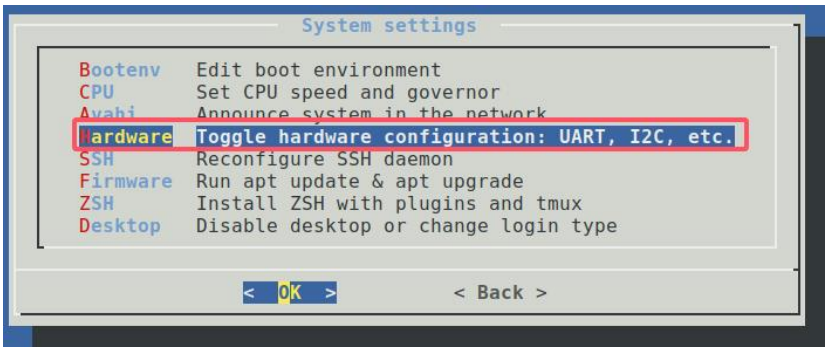
a. First run **orangepi-config**, and remember to add **sudo** permissions as a normal user

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

b. Then select **System**



c. Then select **Hardware**

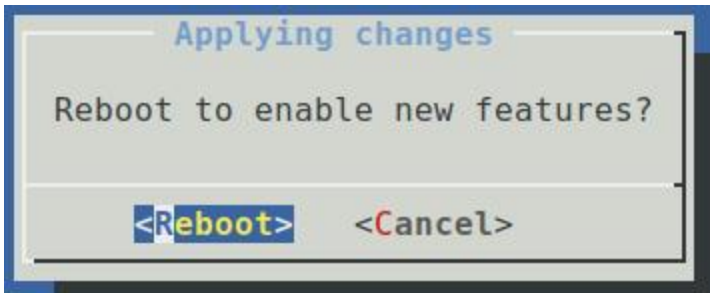


d. Then use the arrow keys on the keyboard to locate the position shown in the figure below, and then use the spacebar to select the open UART configuration

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

f. Then select **<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



4) After entering the Linux system, first confirm whether there is a uart device node under **/dev**

```
orangepi@orangepi:~$ ls /dev/ttyS*
/dev/ttyS0    /dev/ttyS1    /dev/ttyS6    /dev/ttyS7
```

5) Then start testing the UART interface. First use the Dupont line to short-circuit the rx and tx of the UART interface to be tested.

6) Finally, you can run the **serialTest.py**program in the examples to test the loopback function of the serial port. If you can see the following print, it means that the serial port loopback test is normal.

```
root@orangepi:~/wiringOP-Python# cd examples
root@orangepi:~/wiringOP-Python/examples# python3 serialTest.py --device /dev/ttyS7
Out:    0: ->    0
Out:    1: ->    1
Out:    2: ->    2
Out:    3: ->    3
Out:    4:^C
exit
```

3. 19. Hardware watchdog test

The Linux system released by Orange Pi has the watchdog_test program pre-installed, which can be used for direct testing.

The method to run the watchdog_test program is as follows:

a. The second parameter 10 represents the watchdog count time. If the watchdog is not fed within this time, the system will restart.

b. We can feed the dog by pressing any key on the keyboard (except ESC). After

feeding the dog, the program will print a line of keep alive to indicate that the dog was successfully fed.

```
orangepi@orangepi:~$ sudo watchdog_test 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
put_usr return,if 0,success:0
keep alive
keep alive
keep alive
```

3.20. Check the chipid of A733 chip

The command to view the chipid of the A733 chip is as follows. The chipid of each chip is different, so the chipid can be used to distinguish multiple development boards.

```
orangepi@orangepi:~# cat /sys/class/sunxi_info/sys_info |grep sunxi_serial
sunxi_serial            : 147d208d0161172000007c0000000000
```

3.21. Python related instructions

3.21.1. Python source code compilation and installation method

If the Python version in the Ubuntu or Debian system repository doesn't meet your development requirements and you want to use the latest version of Python, you can download the Python source code package and compile and install the latest version using the following method.

The following demonstrates compiling and installing the latest version of Python 3.9. If you want to compile and install other versions of Python, the method is similar (you will need to download the source code for the Python you want to install).

1) First install the dependency packages required to compile Python

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt-get 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 Python 3.9 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 configuration command

```
orangepi@orangepi:~$ cd Python-3.9.10
orangepi@orangepi:~/Python-3.9.10$ ./configure --enable-optimizations
```

4) Then compile and install Python 3.9. The compilation time takes about half an hour.

```
orangepi@orangepi:~/Python-3.9.10$ make -j4
orangepi@orangepi:~/Python-3.9.10$ sudo make altinstall
```

5) After installation, you can use the following command to view the version number of Python just installed

```
orangepi@orangepi:~/Python-3.9.10$ python3.9 --version
Python 3.9.10
```

6) Then update pip

```
orangepi@orangepi:~$ /usr/local/bin/python3.9 -m pip install --upgrade pip
```

3.21.2. How to change the pip source in Python

The default source used by pip on Linux systems is the official Python source. However, accessing the official Python source in China is very slow, and Python package installation often fails due to network issues. Therefore, when using pip to install Python libraries, please remember to change the pip source.

1) First install **python3-pip**

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

```
orangepi@orangepi:~$ sudo apt-get install -y python3-pip
```

2) How to permanently change 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

```
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 use pip3 to install the Python library, which will be very fast.

3) How to temporarily change the pip source under Linux, where **<packagename>** needs to be replaced with the specific package name

```
orangepi@orangepi:~$ pip3 install <packagename> -i    \
https://pypi.tuna.tsinghua.edu.cn/simple --trusted-host pypi.tuna.tsinghua.edu.cn
```

3. 22. How to install Docker

The Linux image provided by Orange Pi has Docker pre-installed, but the Docker service is disabled by default. Use the **enable_docker.sh** script to enable the Docker service. You can then use Docker commands, and the Docker service will automatically start the next time you boot the system.

```
orangepi@orangepi:~$ enable_docker.sh
```

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.
.....
```

When using the docker command, if the prompt "**permission denied**" appears, please add the current user to the docker user group so that you can run the docker command without sudo.

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

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

3. 23. How to install Home Assistant

Note that this article only provides instructions for installing Home Assistant on Ubuntu or Debian systems. For detailed instructions on using Home Assistant, please refer to the official documentation or corresponding books.

3. 23. 1. Installation via Python

Before installation, please change the source of pip to a domestic source to speed up the installation of the Python package. For configuration methods, see the instructions in the section "How to change the 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
```

If it is Debian 12, please use the following command:

```
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 libturbojpeg0-dev tzdata
```

2) Then you need to compile and install Python 3.9. For instructions, please refer to **the section on compiling and installing Python** source code.

The default Python version for Ubuntu Jammy is Python 3.10, so no compilation or installation is required.

The default Python version for Debian Bookworm is Python 3.11, so no compilation or installation is required either.

3) Then create a Python virtual environment

Debian Bookworm uses Python 3.11. Please remember to replace the corresponding commands.

```
orangepi@orangepi:~$ sudo mkdir /srv/homeassistant
orangepi@orangepi:~$ sudo chown orangepi:orangepi /srv/homeassistant
orangepi@orangepi:~$ cd /srv/homeassistant
orangepi@orangepi:/srv/homeassistant$ python3.9 -m venv .
orangepi@orangepi:/srv/homeassistant$ 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

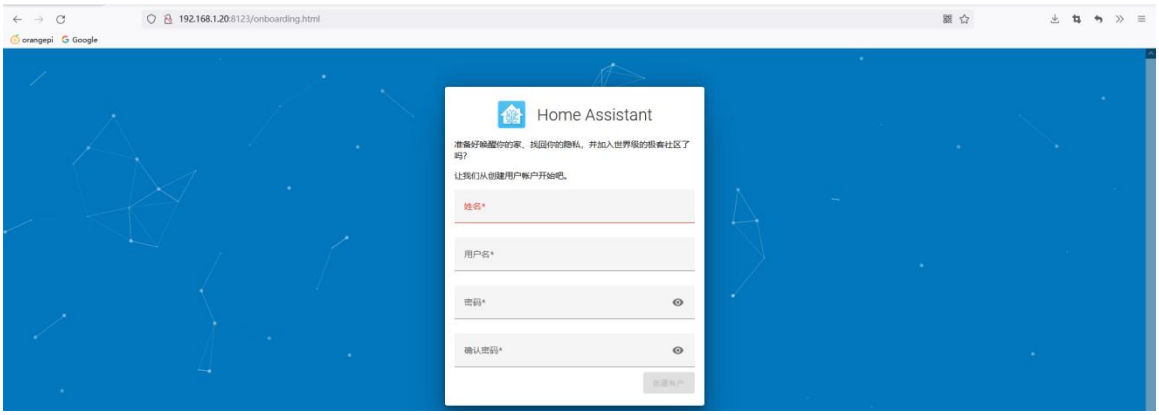
```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ pip3 install homeassistant
```

6) Then enter the following command to run Home Assistant Core

```
(homeassistant) orangepi@orangepi:/srv/homeassistant$ hass -v
```

7) Then enter **[IP address of the development board: 8123]** in the browser to see the Home Assistant interface

The first time you run the hass command, it will download, install, and cache some necessary libraries and dependencies. This process may take several minutes. Note that the Home Assistant interface will not be visible in your browser at this point. Please wait a while before refreshing the browser.



3. 24. OpenCV installation method

3. 24. 1. Using apt to install OpenCV

1) The installation command is as follows

```
orangepi@orangepi:~$ sudo apt-get update
orangepi@orangepi:~$ sudo apt-get install -y libopencv-dev python3-opencv
```

2) Then use the following command to print the version number of OpenCV and output it normally, indicating successful installation of OpenCV

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 Debian12 is as follows:

```
orangepi@orangepi:~$ python3 -c "import cv2; print(cv2.__version__)"
4.6.0
```

3. 25. QT installation method

1) Use the following script to install QT5 and QT Creator

```
orangepi@orangepi:~$ install_qt.sh
```

2) After installation, the QT version number will be automatically printed

a. The QT version that comes with Ubuntu 22.04 is **5.15.3**

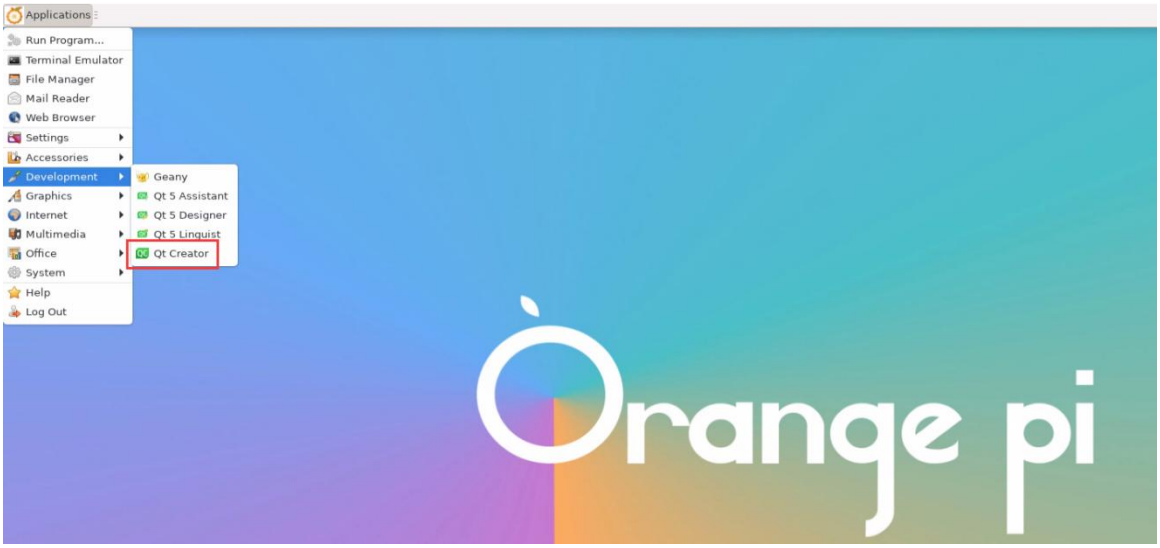
```
orangepi@orangepi:~$ install_qt.sh
```

```
.....
QMake version 3.1
Using Qt version 5.15.3 in /usr/lib/aarch64-linux-gnu
```

b. The QT version that comes with Debian12 is **5.15.8**

```
orangepi@orangepi:~$ install_qt.sh
.....
QMake version 3.1
Using Qt version 5.15.8 in /usr/lib/aarch64-linux-gnu
```

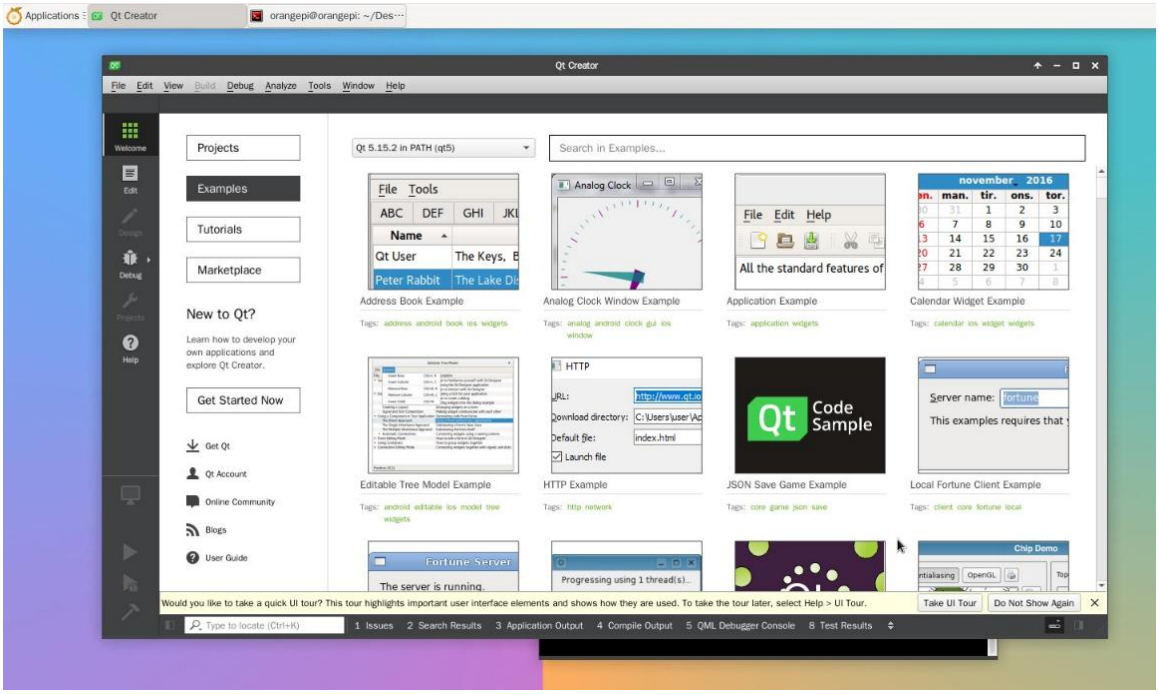
3) Then you can see the QT Creator startup icon in the **Applications** list



You can also use the following command to open QT Creator

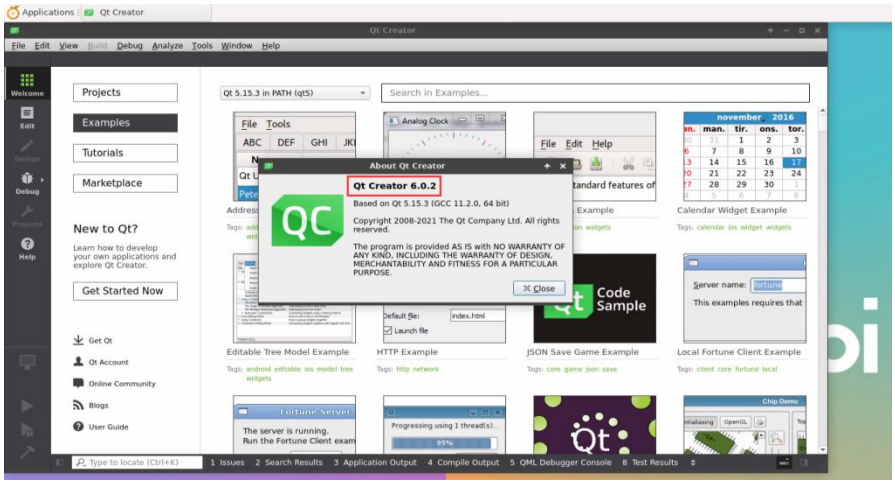
```
orangepi@orangepi:~$ qtcreator
```

4) The interface after QT Creator is opened is as follows

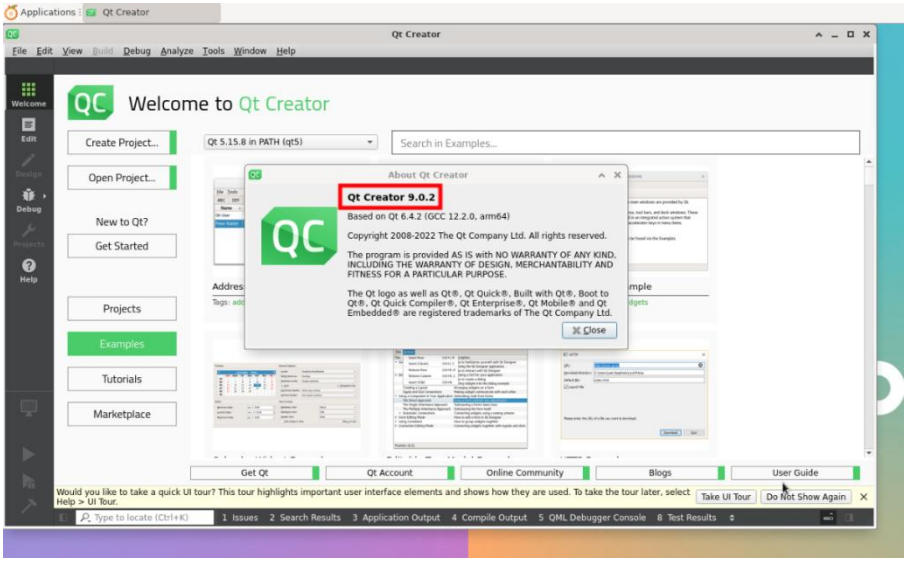


5) The version of QT Creator is as follows

a. The default version of QT Creator in **Ubuntu22.04** is as follows

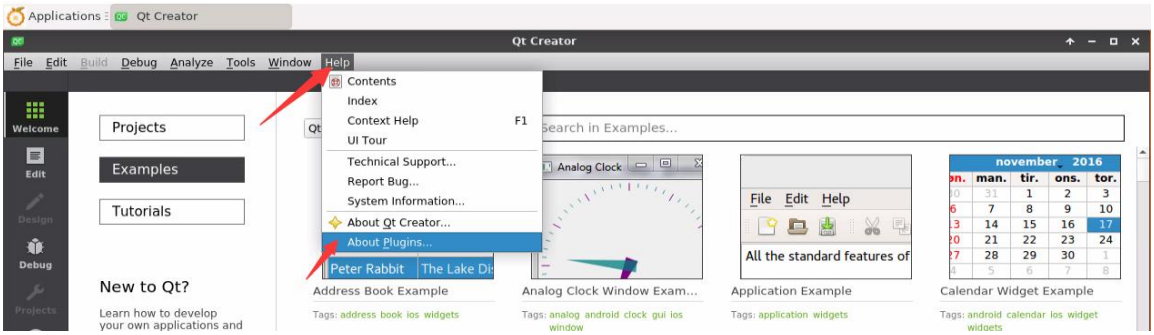


b. The default version of QT Creator in **Debian12** is as follows

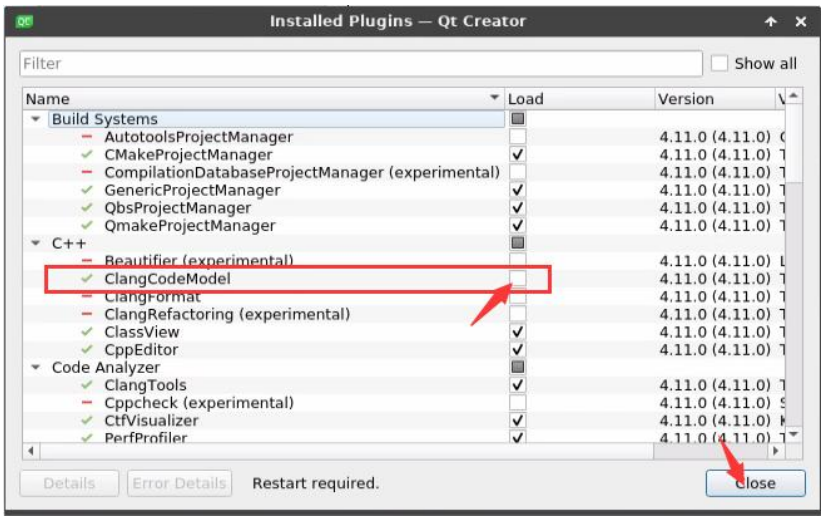


6) Then set up QT

a. First, open **Help->About Plugins...**



b. Then remove the check mark of **ClangCodeModel**

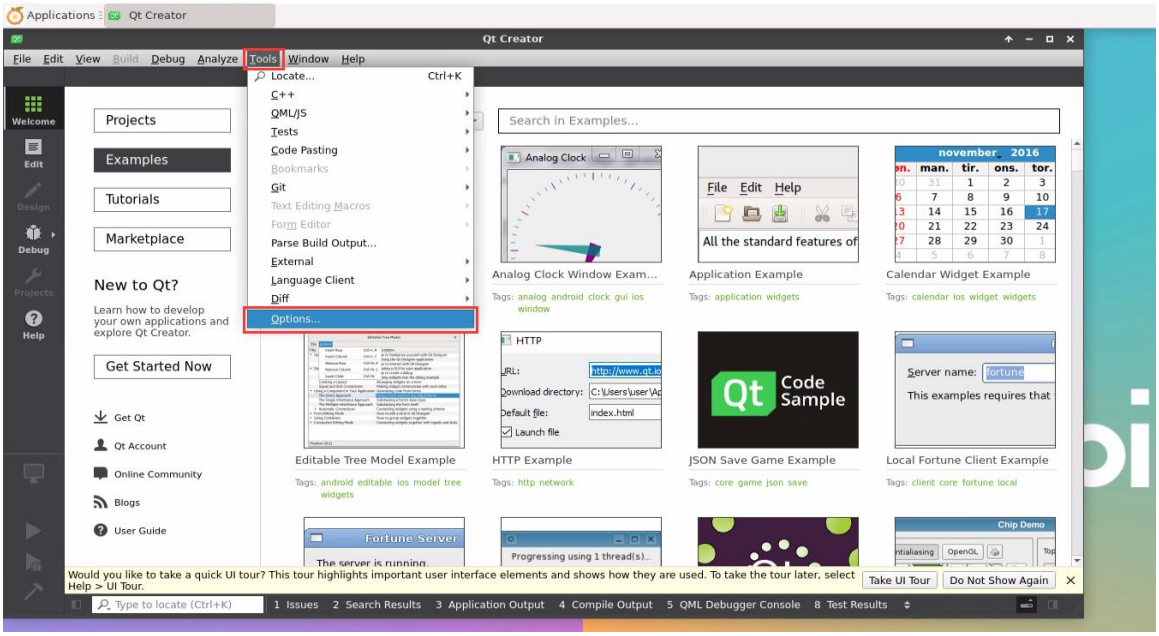


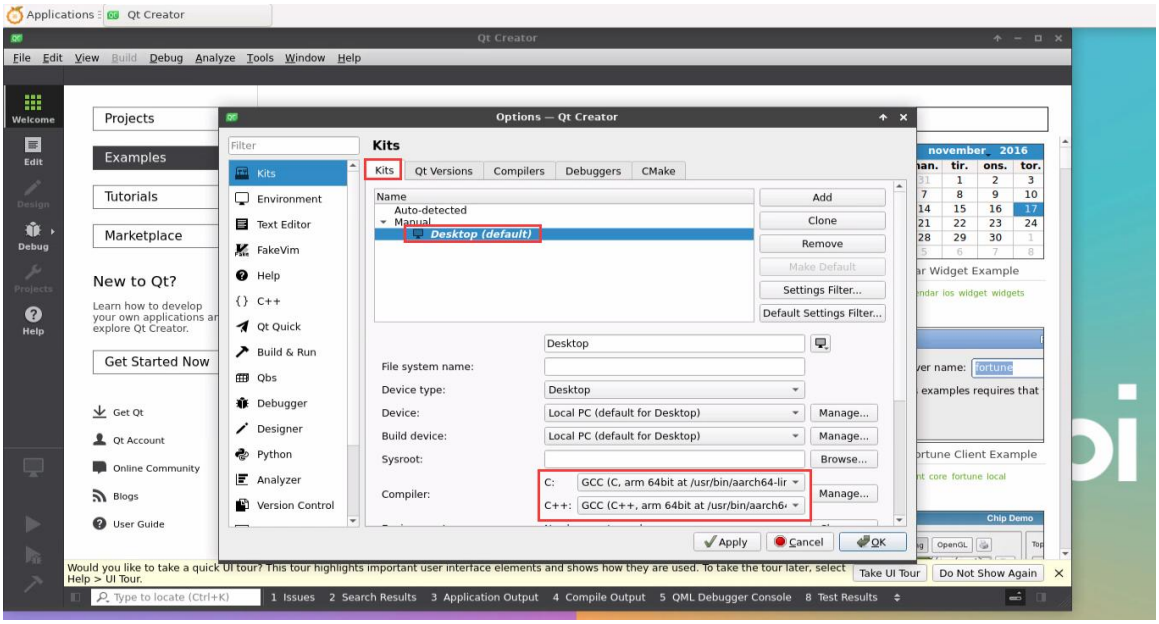
c. **After setting, you need to restart QT Creator**

d. Then make sure that QT Creator uses the GCC compiler. If the default is Clang,

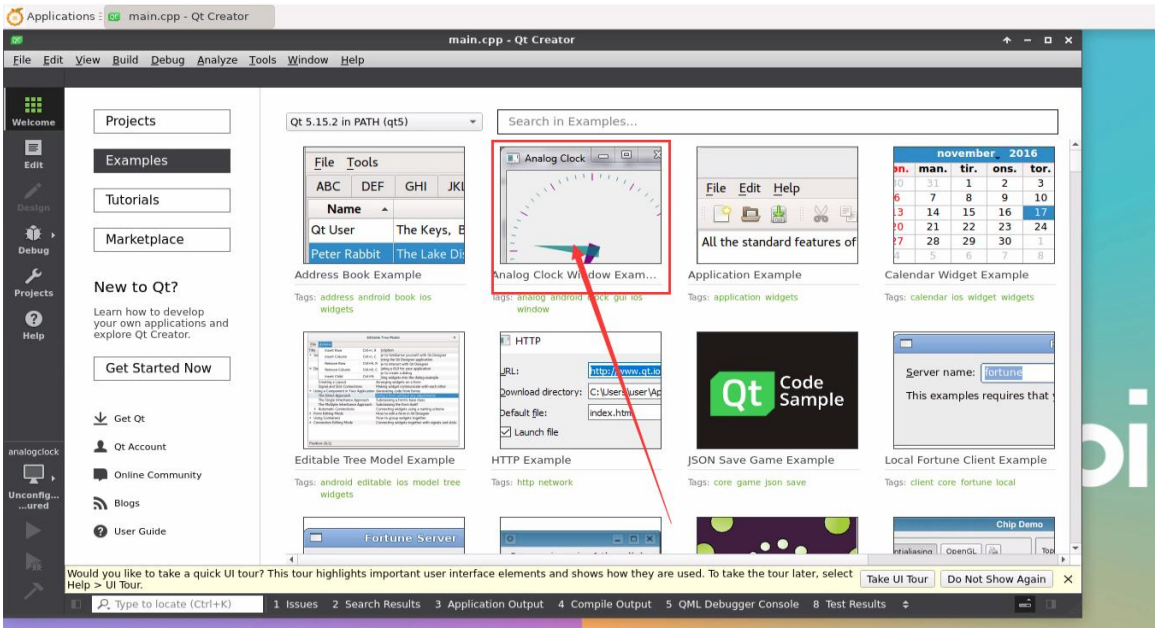
please change it to GCC

For Debian 12, please skip this step.

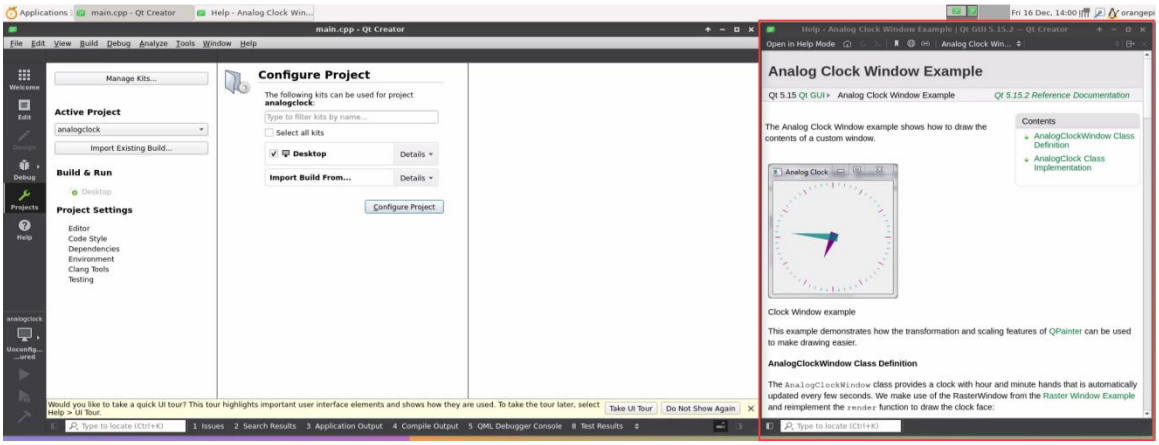




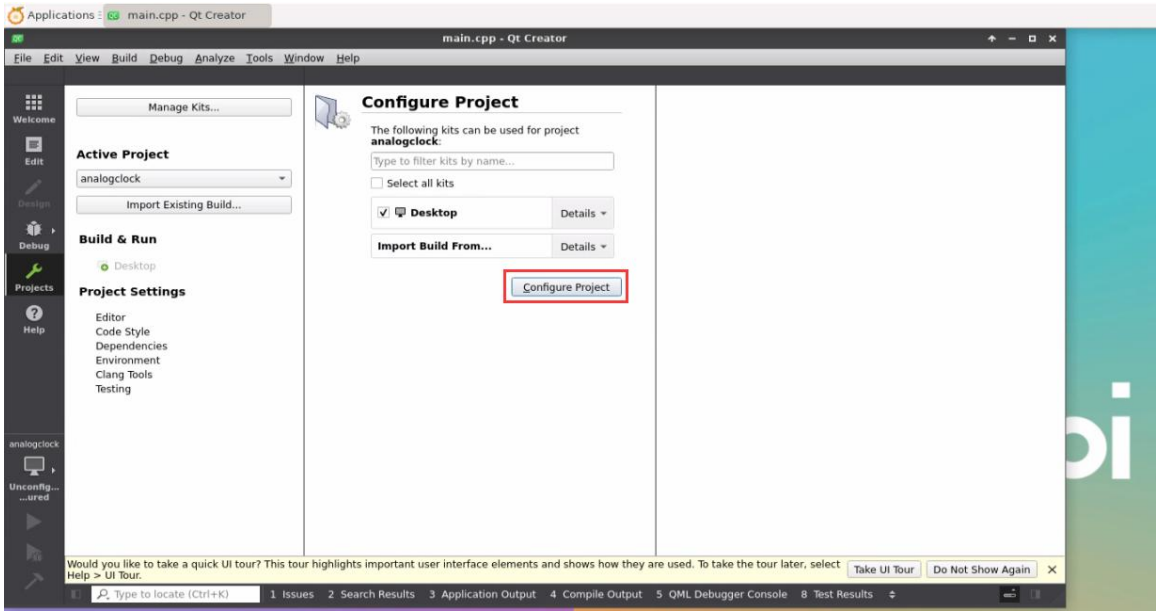
7) Then you can open a sample code



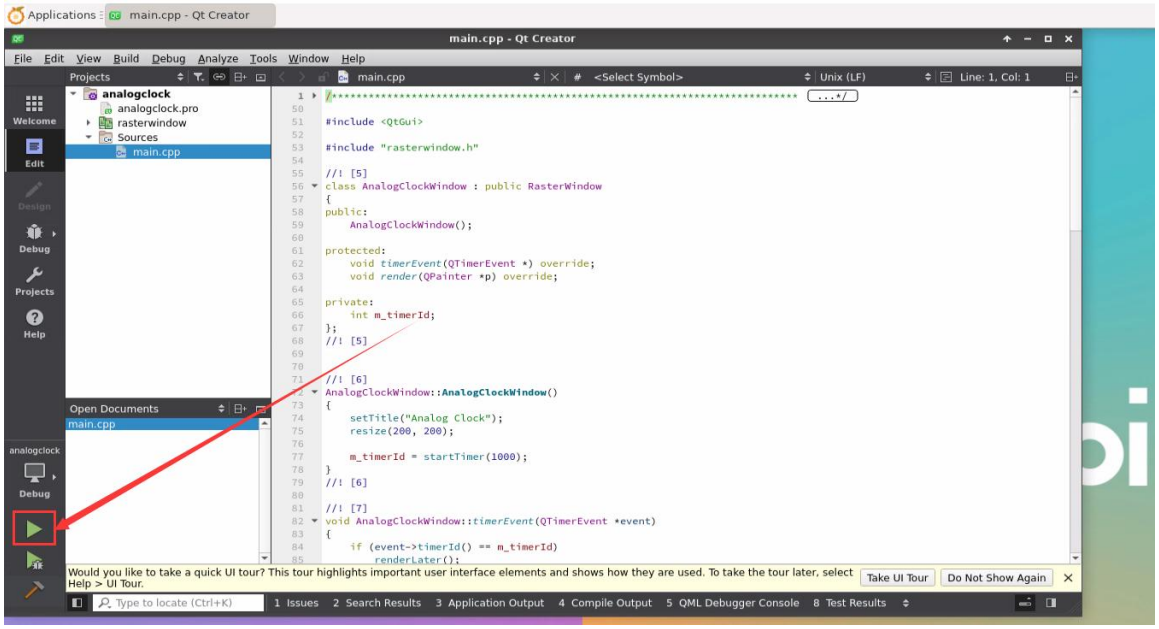
8) Clicking on the sample code will automatically open the corresponding documentation. Please read the instructions carefully.



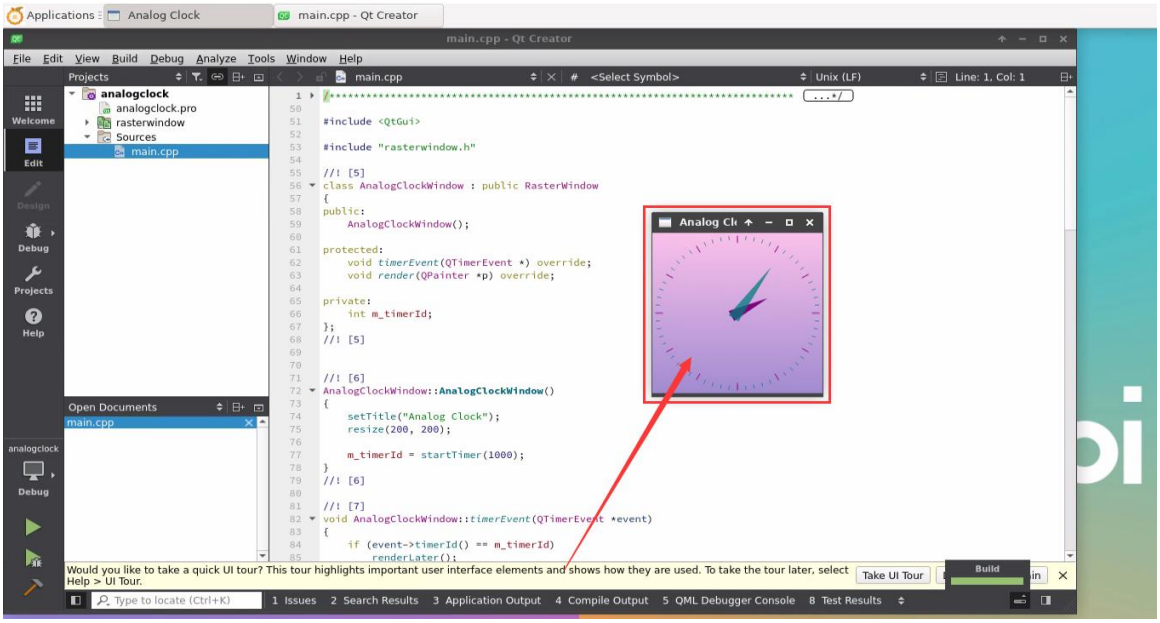
9) Then click **Configure Project**



10) Then click the green triangle in the lower left corner to compile and run the sample code



11) After waiting for a while, the interface shown below will pop up, which means that QT can compile and run normally



12) References

https://wiki.qt.io/Install_Qt_5_on_Ubuntu
https://download.qt.io/archive/qtcreator
https://download.qt.io/archive/qt

3. 26. ROS Installation Method

3. 26. 1. How to install ROS 2 Humble on Ubuntu 22.04

1) Use the **install_ros.sh** script to install ros2

```
orangepi@orangepi:~$ install_ros.sh ros2
```

2) After the **install_ros.sh** script installs ros2, it will automatically run the **ros2 -h** command. If you can see the following print, it means that the ros2 installation is complete.

```
usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...

ros2 is an extensible command-line tool for ROS 2.

optional arguments:
  -h, --help            show this help message and exit
```

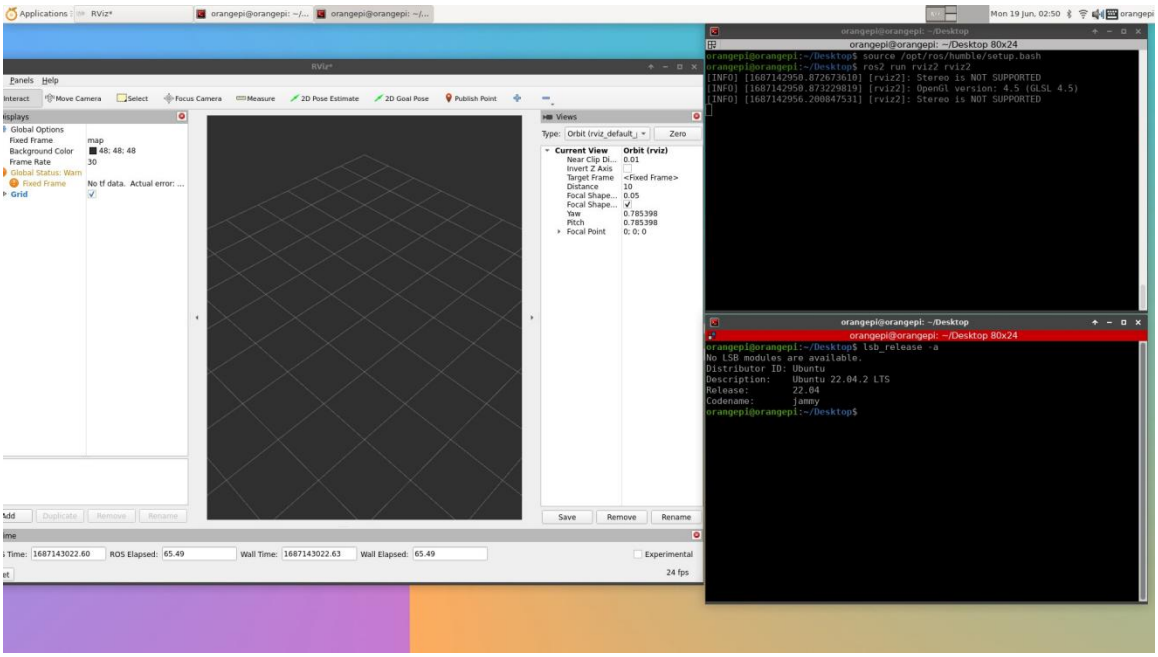
```
Commands:
  action         Various action related sub-commands
  bag            Various rosbag related sub-commands
  component      Various component related sub-commands
  daemon         Various daemon related sub-commands
  doctor         Check ROS setup and other potential issues
  interface      Show information about ROS interfaces
  launch         Run a launch file
  lifecycle      Various lifecycle related sub-commands
  multicast      Various multicast related sub-commands
  node           Various node related sub-commands
  param          Various param related sub-commands
  pkg            Various package related sub-commands
  run            Run a package specific executable
  security       Various security related sub-commands
  service        Various service related sub-commands
  topic          Various topic related sub-commands
  wtf            Use `wtf` as alias to `doctor`

  Call `ros2 <command> -h` for more detailed usage.
```

3) Then you can use the **test_ros.sh** script to test whether ROS 2 is installed successfully. If you can see the following print, it means that ROS 2 is running normally.

```
orangepi@orangepi:~$ test_ros.sh
[INFO] [1671174101.200091527] [talker]: Publishing: 'Hello World: 1'
[INFO] [1671174101.235661048] [listener]: I heard: [Hello World: 1]
[INFO] [1671174102.199572327] [talker]: Publishing: 'Hello World: 2'
[INFO] [1671174102.204196299] [listener]: I heard: [Hello World: 2]
[INFO] [1671174103.199580322] [talker]: Publishing: 'Hello World: 3'
[INFO] [1671174103.204019965] [listener]: I heard: [Hello World: 3]
```

4) Run the following command to open rviz2

```
orangepi@orangepi:~$ source /opt/ros/humble/setup.bash
orangepi@orangepi:~$ ros2 run rviz2 rviz2
```



5) Reference Documents

http://docs.ros.org/en/humble/index.html
http://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html

3. 27. How to install kernel header files

1) The Linux image released by OPi comes with the kernel header file deb package by default, which is stored in **/opt/**

```
orangepi@orangepi:~$ ls /opt/linux-headers*
/opt/linux-headers-xxx-sun60iw2_x.x.x_arm64.deb
```

2) Use the following command to install the kernel header file deb package

```
orangepi@orangepi:~$ sudo dpkg -i /opt/linux-headers*.deb
```

3) After installation, you can see the folder where the kernel header files are located under **/usr/src**

```
orangepi@orangepi:~$ ls /usr/src
linux-headers-x.x.x
```

4) Then you can compile the source code of the hello kernel module that comes with the

Linux image. The source code of the hello module is in **/usr/src/hello**. After entering this directory, use the make command to compile it.

```
orangepi@orangepi:~$ cd /usr/src/hello/
orangepi@orangepi:/usr/src/hello$ sudo make
make -C /lib/modules/5.15.147-sun60iw2/build M=/usr/src/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.147-sun60iw2'
  CC [M]  /usr/src/hello/hello.o
  MODPOST /usr/src/hello/Module.symvers
  CC [M]  /usr/src/hello/hello.mod.o
  LD [M]  /usr/src/hello/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.15.147-sun60iw2'
```

5) After compilation, the **hello.ko** kernel module will be generated

```
orangepi@orangepi:/usr/src/hello$ ls *.ko
hello.ko
```

6) Use the **insmod** command to insert the **hello.ko** kernel module into the kernel

```
orangepi@orangepi:/usr/src/hello$ sudo insmod hello.ko
```

7) Then use the **demsg** command to view the output of the **hello.ko** kernel module. If you can see the following output, it means that the **hello.ko** kernel module is loaded correctly.

```
orangepi@orangepi:/usr/src/hello$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
```

8) Use the **rmmod** command to uninstall the **hello.ko** kernel module

```
orangepi@orangepi:/usr/src/hello$ sudo rmmod hello
orangepi@orangepi:/usr/src/hello$ dmesg | grep "Hello"
[ 2871.893988] Hello Orange Pi -- init
[ 3173.800892] Hello Orange Pi -- exit
```

3. 28. Instructions for using OV13850 and IMX219 MIPI cameras

3. 28. 1. Installation method of camera

At present, the development board supports two MIPI cameras, OV13850 and IMX219. The specific pictures are shown below:

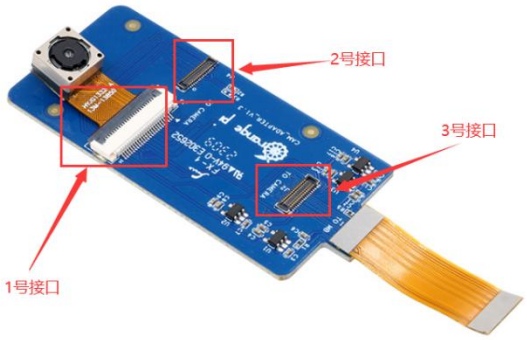
a. OV13850 camera with 13 million MIPI interface

b. 8 million IMX219 camera（**No image available**）

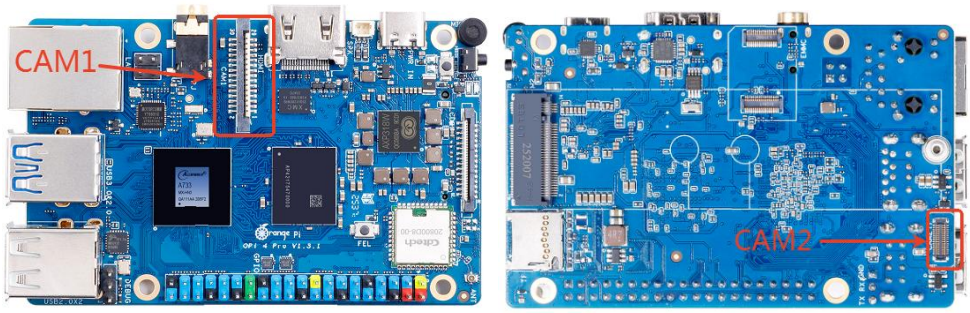
The FPC cable of OV13850 is shown in the following figure. Please note that the FPC cable has direction. The end marked as **TO MB** needs to be plugged into the camera interface of the development board, and the end marked as **TO CAMERA** needs to be plugged into the camera adapter board.

There are a total of 3 camera interfaces on the camera adapter board, and only one can be connected for use at a time, as shown in the following figure. Among them:

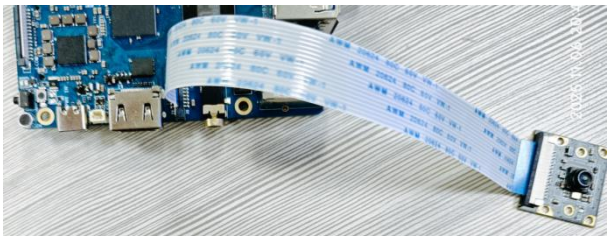
a. **Connect interface 1 to OV13850 camera**
b. **Connect interface 2 to OV13855 camera**
c. Interface 3 is not in use, just ignore it



There are a total of 2 camera interfaces on the Orange Pi 4 Pro development board, and the positions of CAM1 and CAM2 are as follows:



The method of inserting the IMX219 camera into the CAM1 interface of the development board is as follows:



The method of inserting the OV13850 camera into the CAM2 interface of the development board is as follows:

3. 28. 2. Method of opening MIPI camera through OpenCV program

1) First, you need to install the dependency package.

```
orangepi@orangepi:~$ sudo apt install python3-pip python3-opencv libopencv-dev
```

```
orangepi@orangepi:~$ sudo apt install python3-pybind11 python3-dev
```

2) Then enter the directory where **v4l2_opencv_demo** is located, and you can see that it contains the following files.

```
orangepi@orangepi:~$ cd /opt/v4l2_opencv_demo/
orangepi@orangepi:/opt/v4l2_opencv_demo$ ls
Makefile  test_camera.cpp  test_camera.py  v4l2_camera_bind.cpp
v4l2_camera.cpp  v4l2_camera.h
```

3) Example file description:

a. **Makefile**：Build the compilation rule file for the C++sample program.

b. **test_camera.cpp**：C++testing program, using OpenCV to preview the camera in real-time.

c. **test_camera.py** ： Python sample program, using OpenCV to preview camera images.

d. **v4l2_camera_bind.cpp**： Use pybind11 to encapsulate the V4L2Camera class for Python to call.

e. **v4l2_camera.cpp** ： V4L2Camera class implementation, encapsulating camera initialization and frame capture logic.

f. **v4l2_camera.h** ： V4L2Camera class header file, defining interfaces and member variables.

4) Then install the camera driver

```
orangepi@orangepi:~$ sudo modprobe vin_v4l2
```

5) Then execute the following command to confirm the device node of the camera. Currently, two MIPI cameras are connected, so there will be two nodes displayed.

```
orangepi@orangepi:~$ ls /dev/video*
/dev/video0  /dev/video8
```

6) The steps to open a MIPI camera using the C++example are as follows:

a. First, execute the following command to compile the C++example.

```
orangepi@orangepi:~$ cd /opt/v4l2_opencv_demo/
orangepi@orangepi:/opt/v4l2_opencv_demo$ make clean && make
```

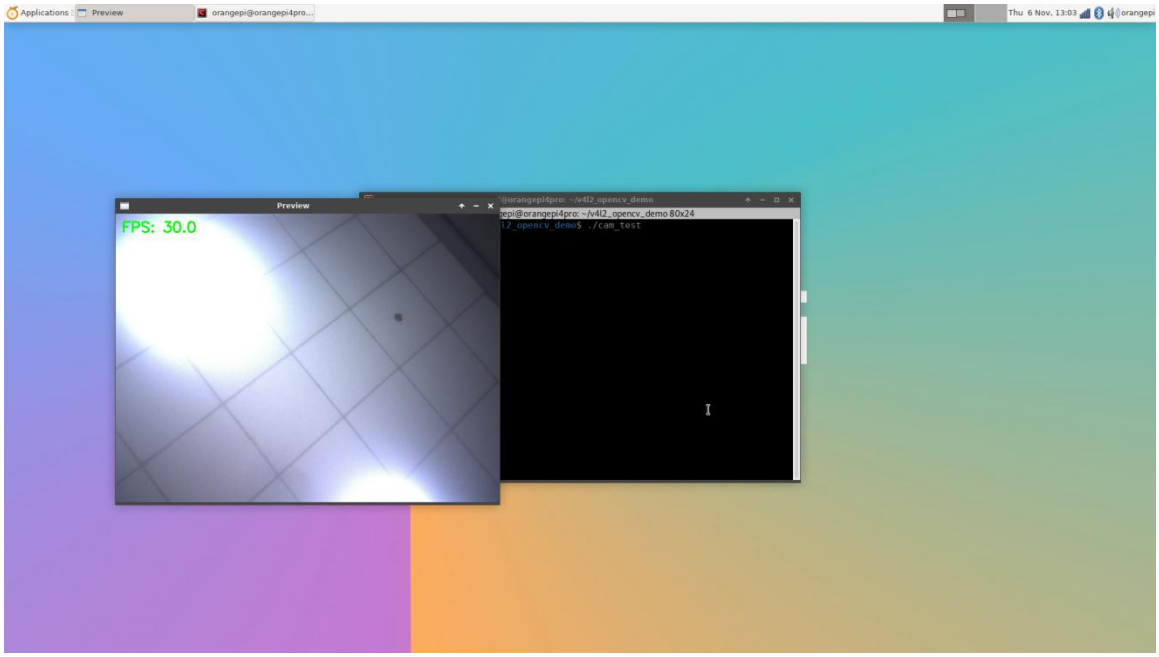
b. After compilation, the **cam_test** executable file will be generated in the current

directory.

```
orangepi@orangepi:/opt/v4l2_opencv_demo$ ls cam_test
cam_test
```

c. Then execute the **cam_test** program to open the MIPI camera (note that the camera node should be selected according to the actual situation):

```
orangepi@orangepi:/opt/v4l2_opencv_demo$ ./cam_test /dev/video8
```



d. The camera resolution can be set by modifying the following code in the **test_camera.cpp** file. After modification, please note that the code needs to be recompiled.

```
V4L2Camera cam(device_str, 640, 480);
```

7) The steps to open a MIPI camera using a Python example are as follows:

a. First, execute the following command to compile the **python_module**

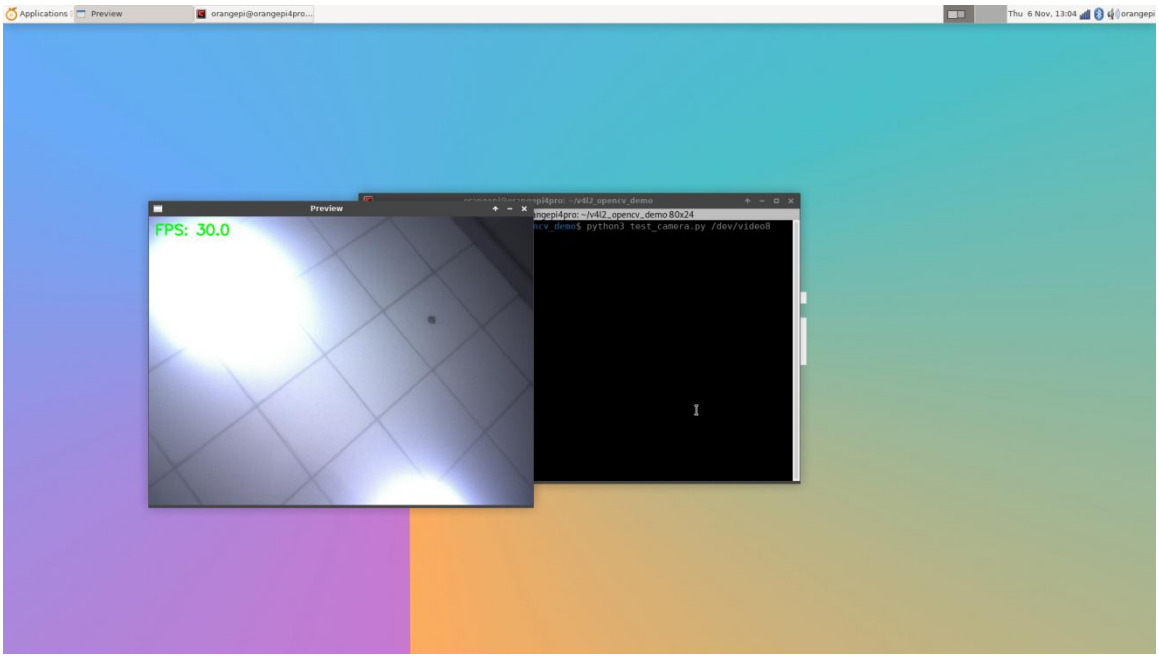
```
orangepi@orangepi:~$ cd /opt/v4l2_opencv_demo/
orangepi@orangepi:/opt/v4l2_opencv_demo$ make python_module
```

b. Then set the node and resolution of the camera by modifying the following code in the test_camera.py file:

```
cam = v4l2cam.V4L2Camera("/dev/video8", 640, 480)
```

c. Then execute the test_camera.py script to open the MIPI camera.

```
orangepi@orangepi:/opt/v4l2_opencv_demo$ python3 test_camera.py
```



3. 29. How to use 3.27.10.1 inch MIPI LCD screen

3. 29. 1. 10.1 inch MIPI screen assembly method

1) First prepare the necessary accessories

a. 10.1-inch MIPI LCD display + touch screen

b. Screen adapter board + 31pin to 40pin cable

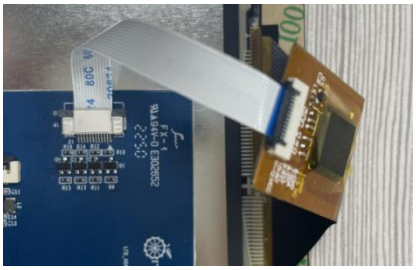
c. 30-pin MIPI cable

d. 12-pin touch screen cable

2) Connect the 12-pin touchscreen cable, 31-pin to 40-pin cable, and 30-pin MIPI cable to the screen adapter board as shown below. Note that the **blue insulation side of the touchscreen cable should face down**, while the insulation sides of the other two cables should face up. If connected incorrectly, it will result in no display or inability to touch.

3) Place the adapter board with the connected cables on the MIPI LCD screen as shown below, and connect the MIPI LCD screen and the adapter board via a 31-pin to 40-pin cable.

4) Then connect the touch screen to the adapter board via the 12-pin touch screen cable, paying attention to the direction of the insulating surface.

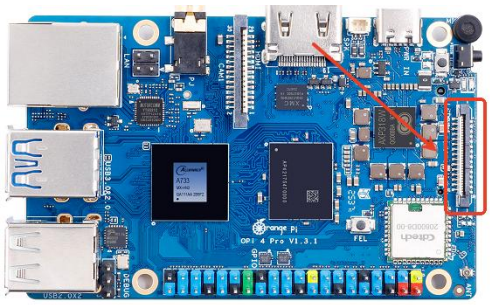


5) Finally, connect it to the LCD interface of the development board via a 30-pin MIPI cable.

3. 29. 2. How to open the 10.1-inch MIPI LCD screen configuration

1) The Linux image does not have the MIPI LCD screen configuration turned on by default. If you need to use the MIPI LCD screen, you need to turn it on manually.

2) The interface of the mipi lcd screen on the development board is shown in the figure below:

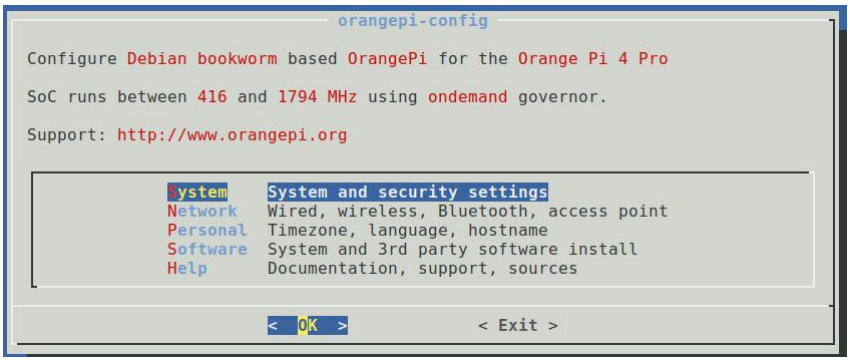


3) The method to open the mipi lcd configuration is as follows:

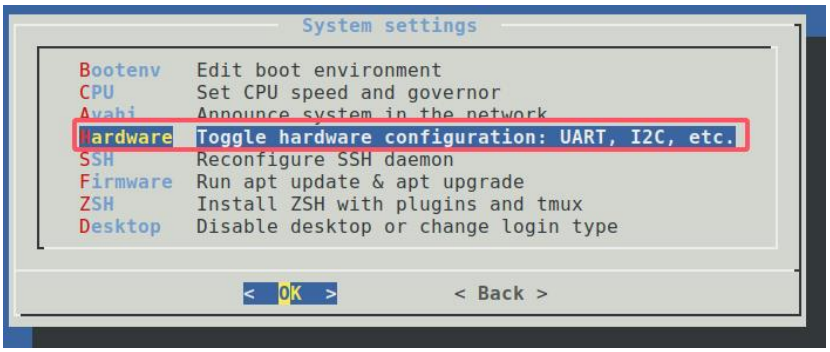
a. First run **orangepi-config**. Ordinary users should remember to add **sudo** permissions.

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

b. Then select **System**

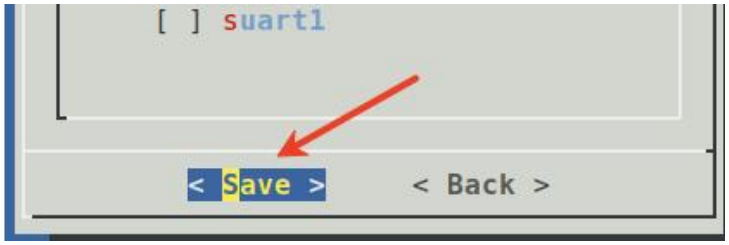


c. Then select **Hardware**



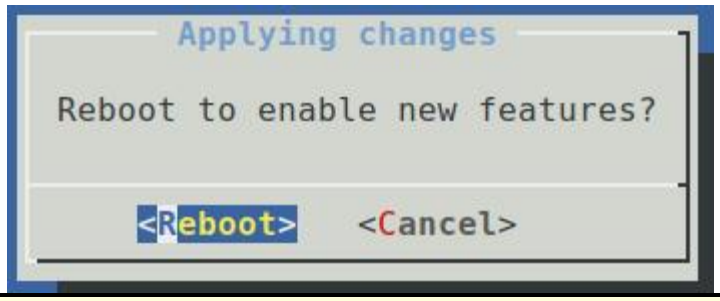
d. Then use the arrow keys on the keyboard to locate the **lcd**, and then use the **spacebar** to select

e. Then select **<Save>**



f. Then selec**<Back>**

g. Select **<Reboot>** to restart the system for the configuration to take effect.



The above configuration will eventually add overlays=lcd to /boot/orangepiEnv.txt. After setting it, you can check it first. If this line does not exist, then there is a problem with the configuration.

If you find it troublesome to use orangepi-config, you can also use the vim editor to open /boot/orangepiEnv.txt and add the line overlays=lcd.

orangepi@orangepi:~$ **cat /boot/orangepiEnv.txt | grep "lcd"**
overlays=lcd #Example Configuration

4) **Then restart the Linux system**

5) After restarting, you can see the display of the LCD screen as shown below (the default is vertical screen)：

3. 29. 3. How to rotate the display direction of the server version image

1) Add **extraargs=fbcon=rotate: direction to rotate** in **/boot/orangepiEnv.txt** to set the display direction of the server version of Linux system. The number after **fbcon=rotate:** can be set to:

a. 0: Normal screen (portrait by default)
b. 1: Rotate 90 degrees clockwise
c. 2: Flip 180 degrees
d. 3: Rotate 270 degrees clockwise

```
orangepi@orangepi:~$ sudo vim /boot/orangepiEnv.txt
overlays=lcd
extraargs=fbcon=rotate:3
```

2) Then **restart** the Linux system and you will see that the direction of the LCD screen display has rotated

3. 30. Test of some programming languages supported by Linux system

3. 30. 1. Debian Bookworm System

1) Debian Bookworm is installed with the gcc compilation tool chain by default, which can compile C language programs directly in the Linux system of the development board

a. The version of gcc is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Debian 12.2.0-14) 12.2.0
Copyright (C) 2022 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Debian Bookworm has Python 3 installed by default

a. The specific version of Python is as follows

```
orangepi@orangepi:~$ python3
Python 3.11.2 (main, Apr 28 2025, 14:11:48) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Use the Ctrl+D shortcut key to exit Python's interactive mode.

b. Write the **hello_world.py** program in Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) Debian Bookworm does not install Java compilation tools and runtime environment by default

a. You can use the following command to install openjdk. The latest version in Debian Bookworm is openjdk-17.

```
orangepi@orangepi:~$ sudo apt install -y openjdk-17-jdk
```

b. After installation, you can check the Java version

```
orangepi@orangepi:~$ java --version
```

c. Write a Java version 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!
```

3.30.2. Ubuntu Jammy system

1) Ubuntu Jammy is installed with the gcc compilation tool chain by default, which can compile C language programs directly in the Linux system of the development board

a. The version of gcc is as follows

```
orangepi@orangepi:~$ gcc --version
gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.
```

b. Write the **hello_world.c** program in C language

```
orangepi@orangepi:~$ vim hello_world.c
#include <stdio.h>

int main(void)
{
        printf("Hello World!\n");

        return 0;
}
```

c. Then compile and run **hello_world.c**

```
orangepi@orangepi:~$ gcc -o hello_world hello_world.c
orangepi@orangepi:~$ ./hello_world
Hello World!
```

2) Ubuntu Jammy has Python 3 installed by default

a. The specific version of Python3 is as follows

```
orangepi@orangepi:~$ python3
Python 3.10.12 (main, Jul 29 2024, 16:56:48) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Use the Ctrl+D shortcut key to exit Python's interactive mode.

b. Write the **hello_world.py** program in Python

```
orangepi@orangepi:~$ vim hello_world.py
print('Hello World!')
```

c. The result of running **hello_world.py** is as follows

```
orangepi@orangepi:~$ python3 hello_world.py
Hello World!
```

3) 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 Java version

```
orangepi@orangepi:~$ java --version
openjdk 18.0.2-ea 2022-07-19
OpenJDK Runtime Environment (build 18.0.2-ea+9-Ubuntu-222.04)
OpenJDK 64-Bit Server VM (build 18.0.2-ea+9-Ubuntu-222.04, mixed mode, sharing)
```

c. Write a Java version 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. 31. How to upload files to the Linux system of the development board

3. 31. 1. How to upload files from Ubuntu PC to the Linux system of the development board

3. 31. 1. 1. How to upload files using the scp command

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

a. **file_path：** Need to be replaced with the path of the file to be uploaded
b. **orangepi：** This is the user name of the development board's Linux system. You can also replace it with other names, such as root.
c. **192.168.xx.xx:** The IP address of the development board. Please modify it according to the actual situation.
d. **/home/orangepi:** The path in the Linux system of the development board can also be modified to other paths

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

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

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

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

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

3. 31. 1. 2. How to upload files using FileZilla

1) First install filezilla in your Ubuntu PC

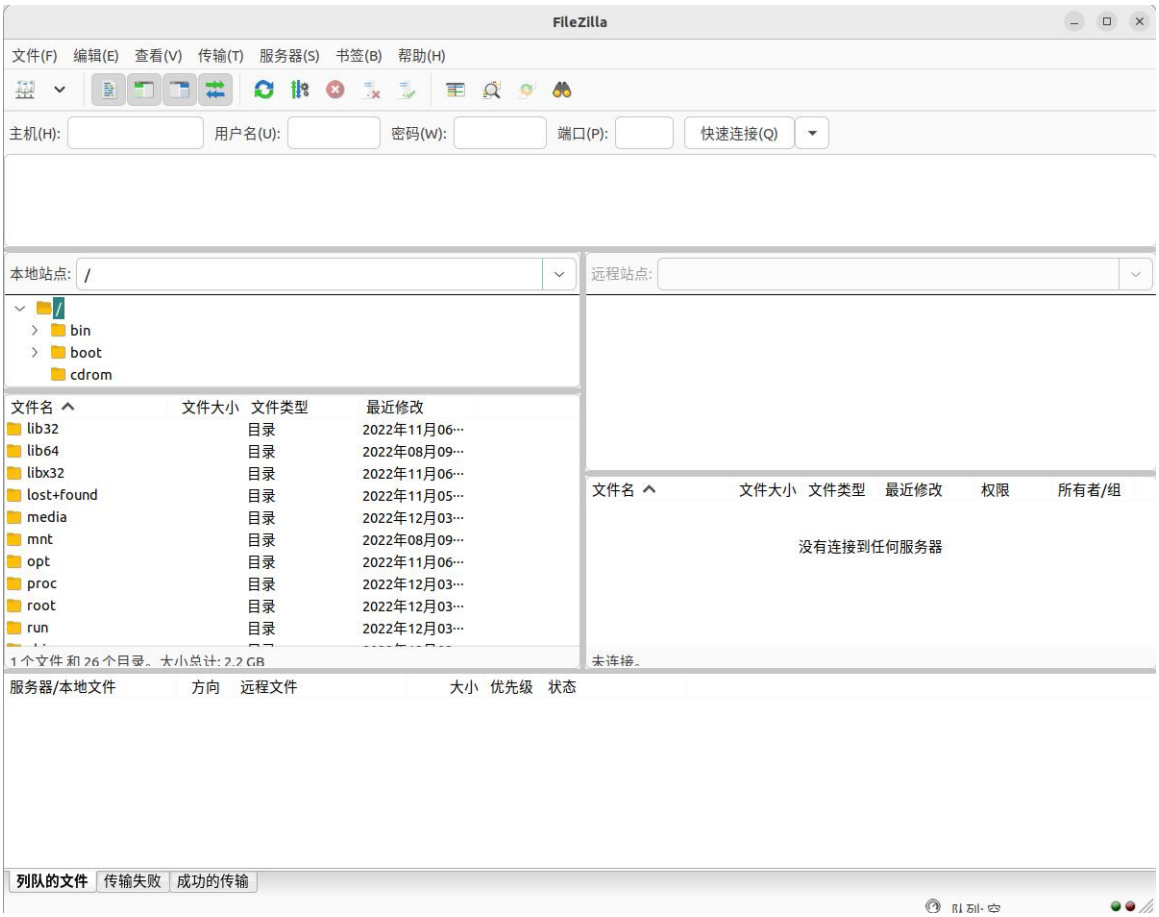
```
test@test:~$ sudo apt install -y filezilla
```

2) Then open filezilla using the following command

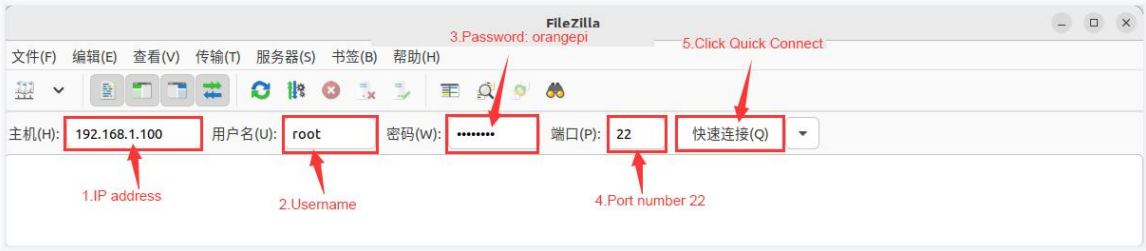
```
test@test:~$ filezilla
```

3) The interface after opening FileZilla is as follows. At this time, the remote site on the

right is empty.

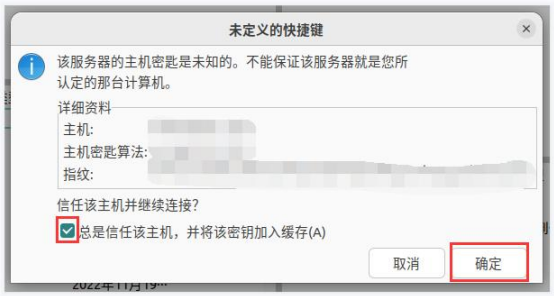


4) The method of connecting the development board is shown in the figure below

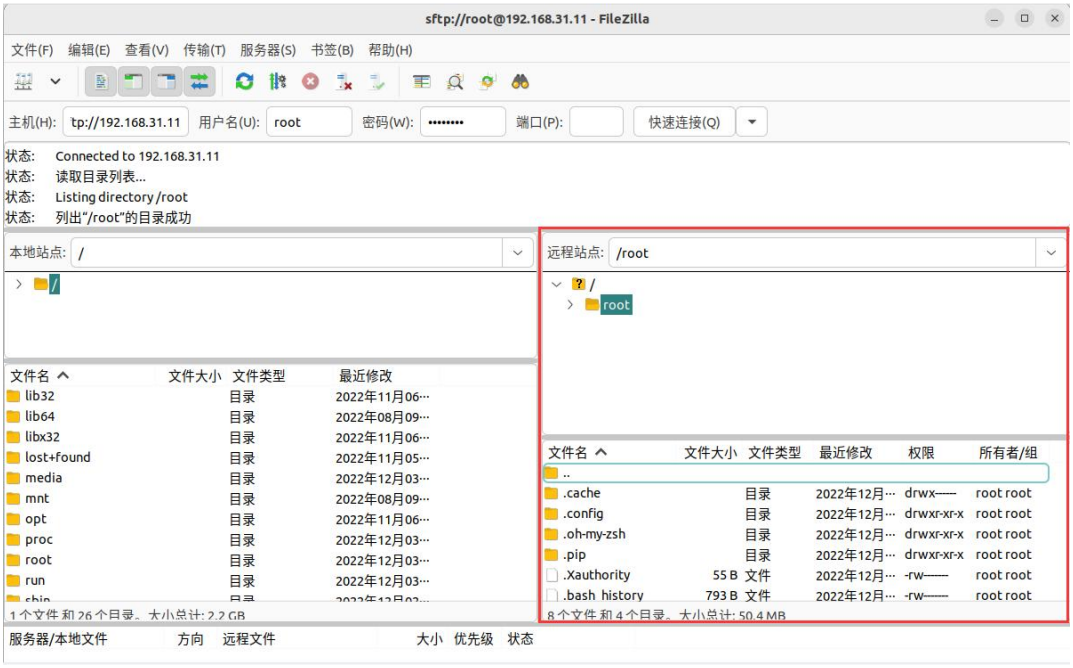


5) Then select **Save Password** and click **OK**

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

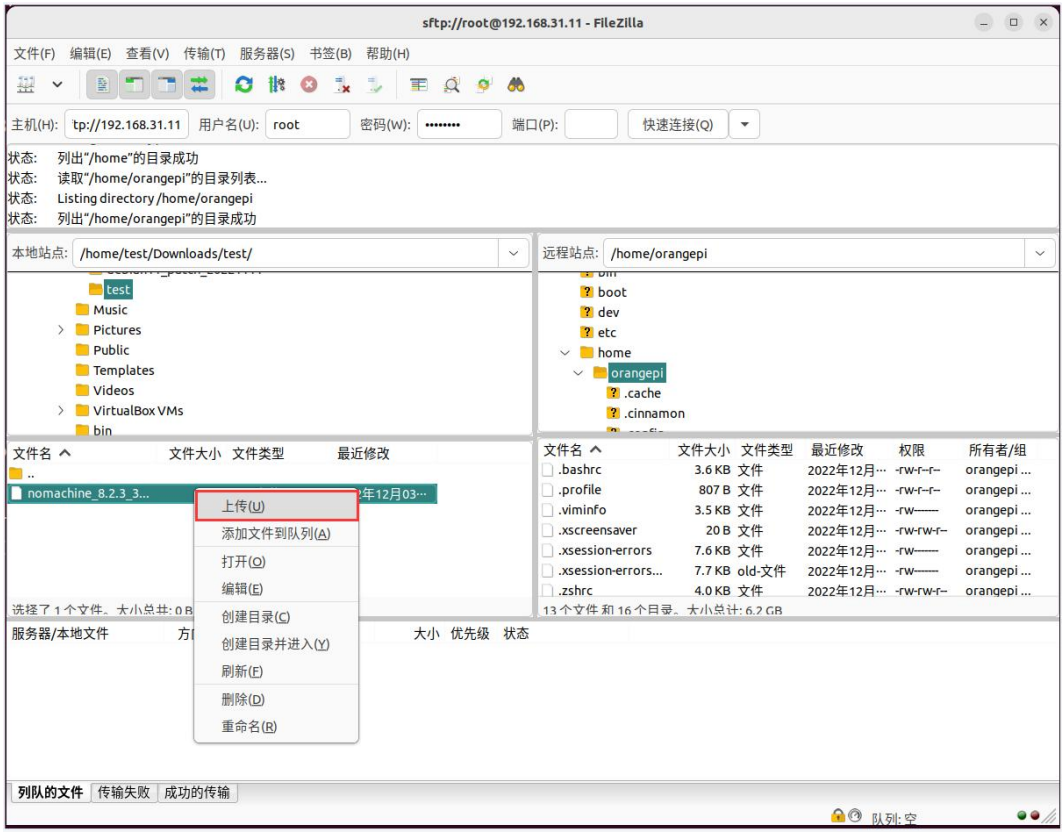


7) After the connection is successful, you can see the directory structure of the development board's Linux file system on the right side of the filezilla software



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

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



9) After uploading, you can check the uploaded files in the corresponding path of the development board Linux system.

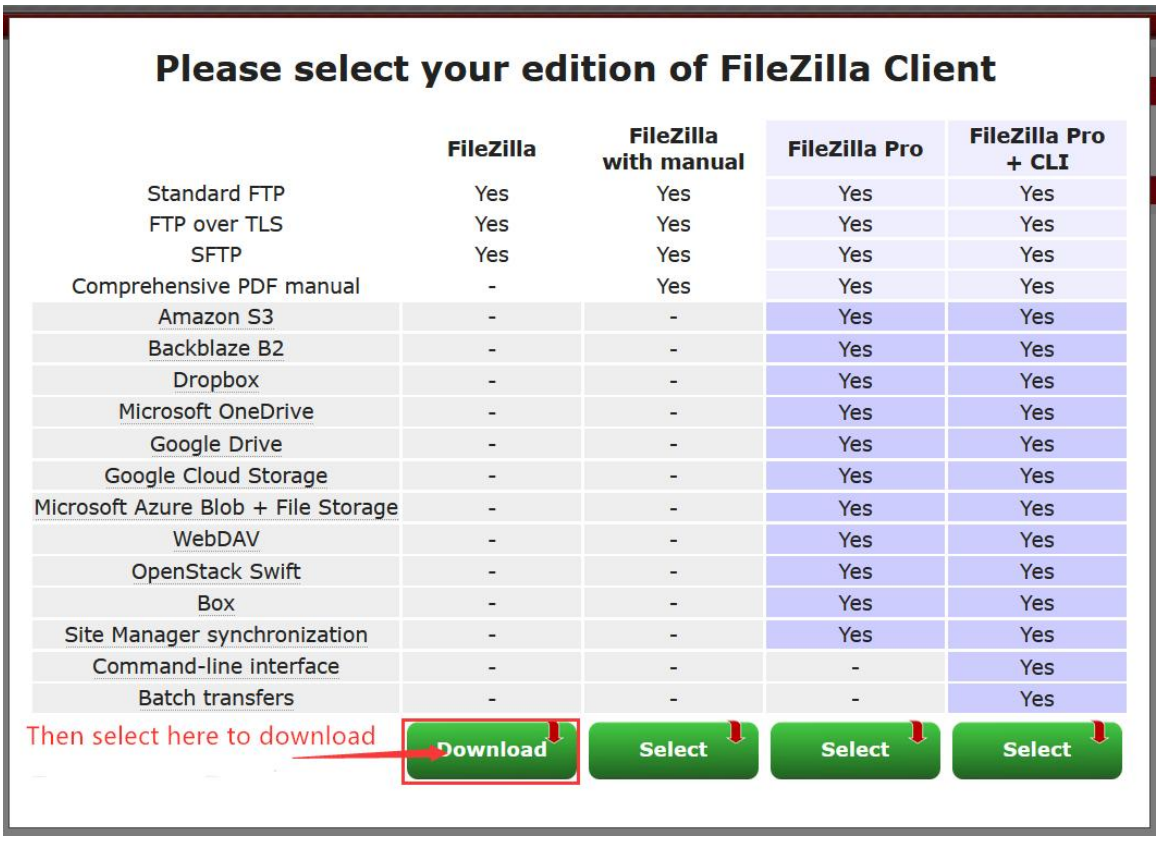
10) The method of uploading folders is the same as that of uploading files, so I will not go into details here.

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

3.31.2.1. How to upload files using FileZilla

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

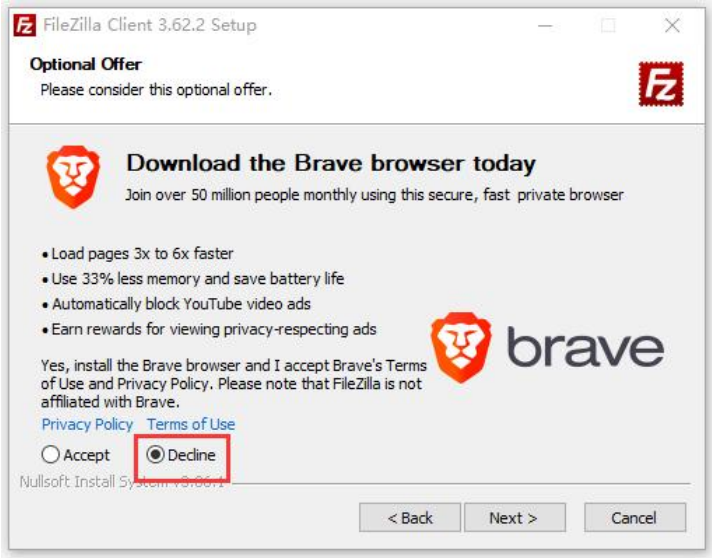
https://filezilla-project.org/download.php?type=client



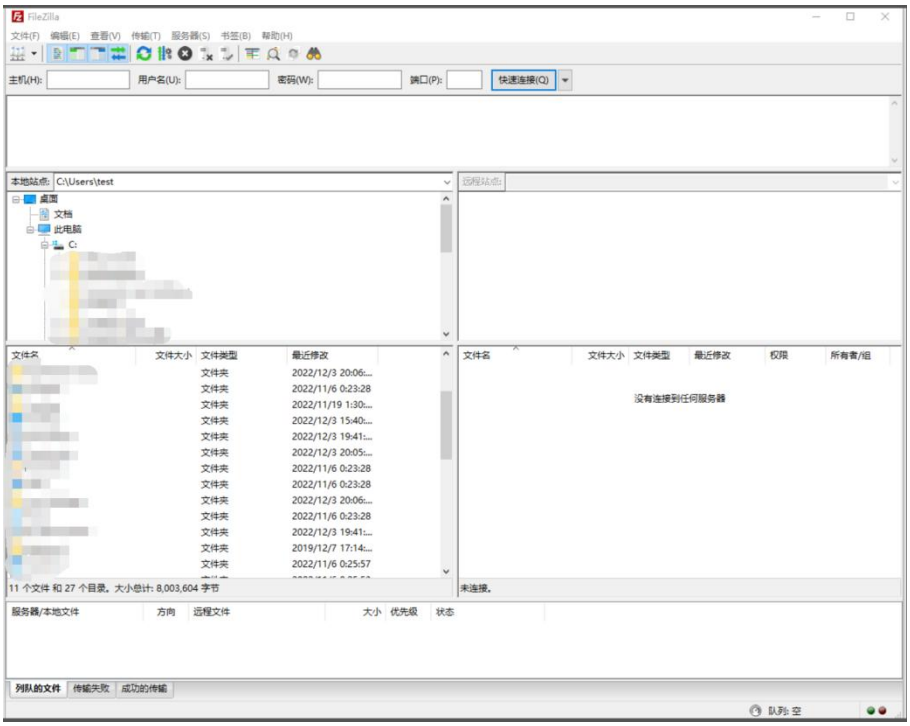
2) The downloaded installation package is as follows, then double-click to install it directly

FileZilla_Server_1.5.1_win64-setup.exe

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

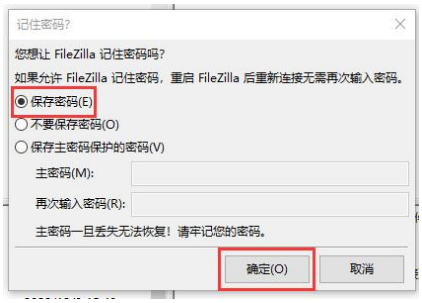


3) The interface after opening FileZilla is as follows. At this time, the remote site on the right is empty.

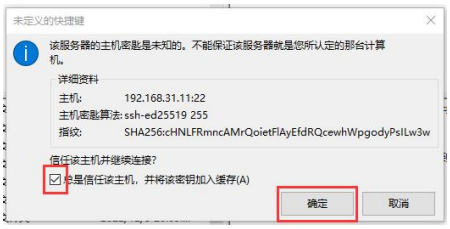


4) The method of connecting the development board is shown in the figure below:

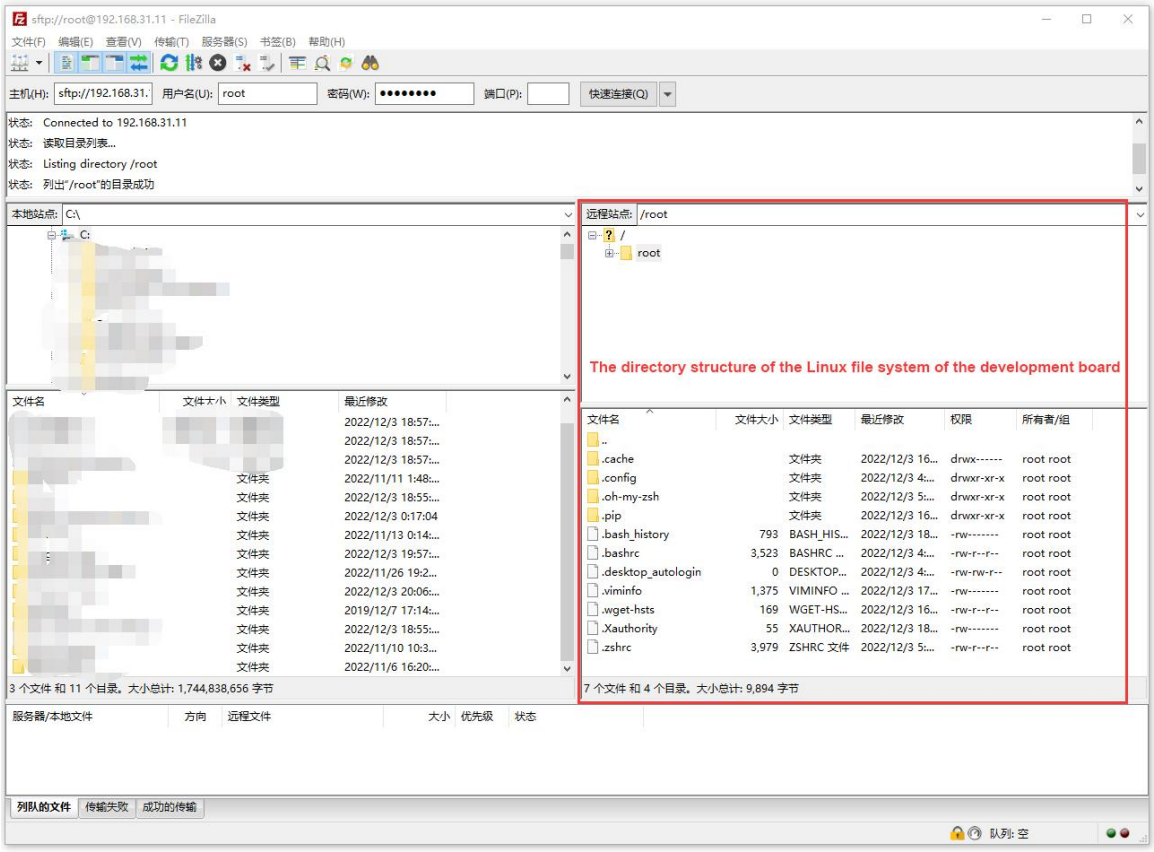
5) Then select **Save Password** and click **OK**



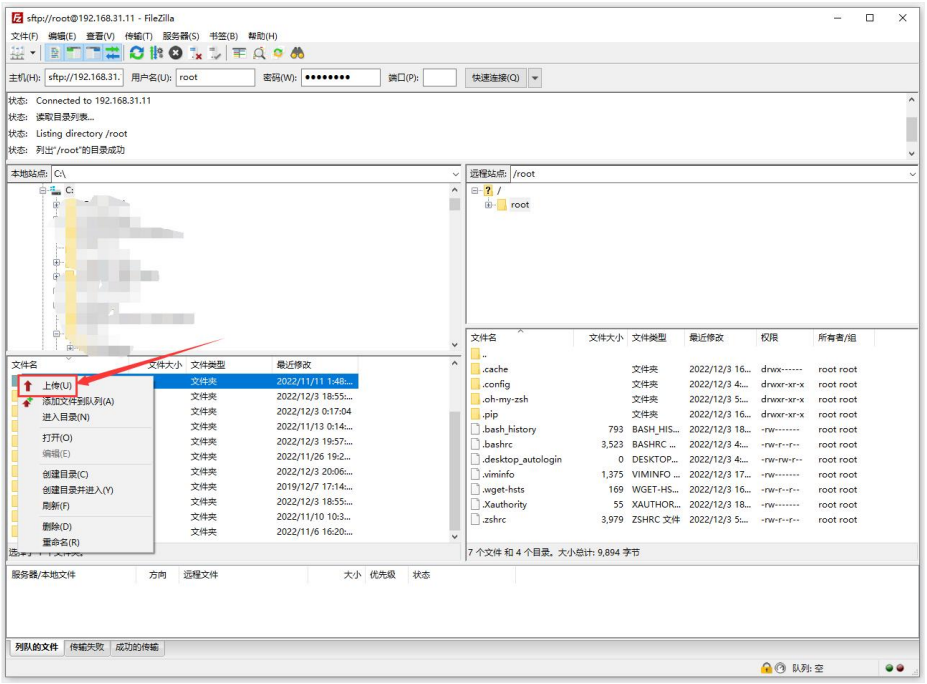
6) Then select **Always trust this host** and click **OK**



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



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



9) After uploading, you can check the uploaded files in the corresponding path of the development board Linux system.

10) The method of uploading folders is the same as that of uploading files, so I will not go into details here.

3. 32. Debian Bullseye System Hard Decoding Test Instructions

Note that currently only the Debian bullseye system supports hard decoding, and other versions of the system are not supported yet.

1) First prepare the test video, which can be downloaded from the following website.

http://bbb3d.renderfarming.net/download.html

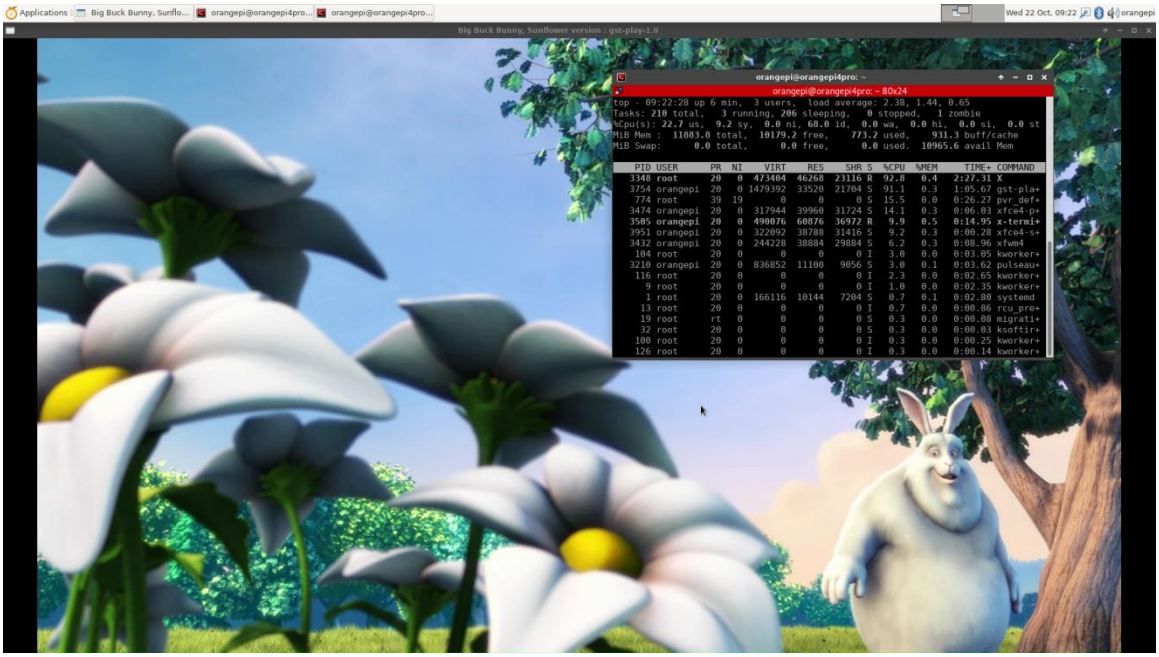
2) Then open a command line terminal on the Debian system desktop and run the following gstreamer command.

```
orangepi@orangepi:~$ gst-play-1.0 bbb_sunflower_1080p_60fps_normal.mp4
```

3) Alternatively, you can use the following command.

```
orangepi@orangepi:~$ gst-launch-1.0 filesrc location=bbb_sunflower_1080p_60fps_normal.mp4 ! qtdemux ! h264parse ! omxh264dec ! xvimagesink
```

4) At this time, you can open the terminal and enter the top command to check the CPU usage. If the CPU usage is relatively low, it means that hard decoding is being used.



3. 33. Debian Bullseye System GPU Test Instructions

Note that currently only the Debian bullseye system supports GPU acceleration, and other versions of the system do not support it yet.

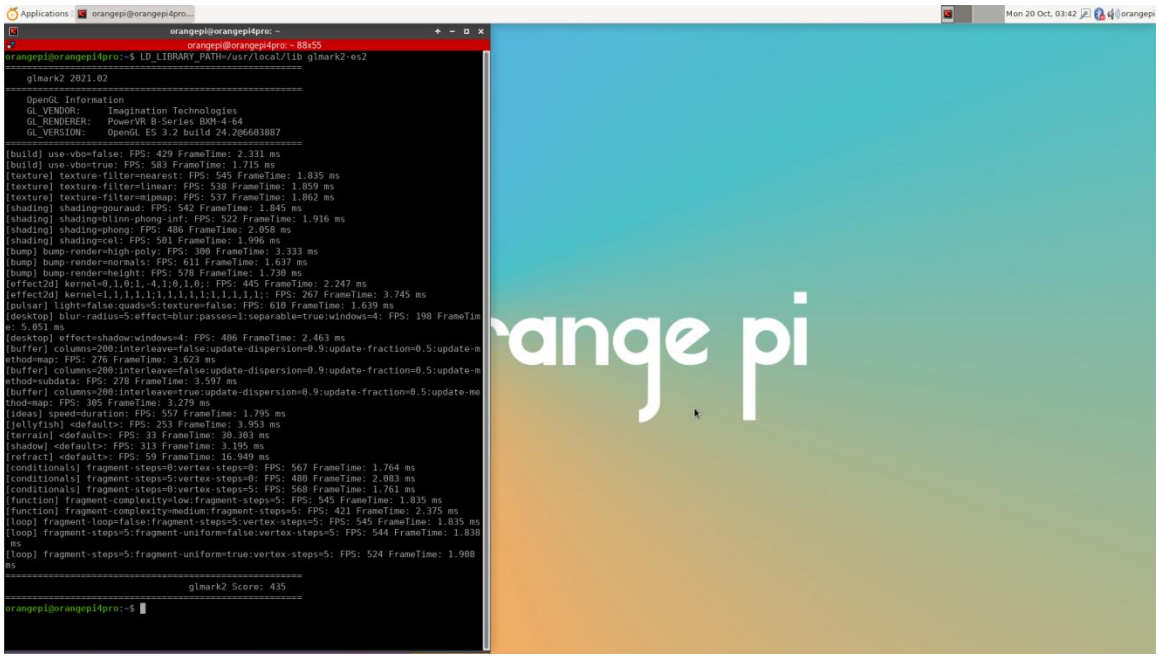
5) First, open a command line terminal on the Debian system desktop and run **glmark2-es2**. Choose one of the following commands:

Note that you need to run the following command on the desktop.

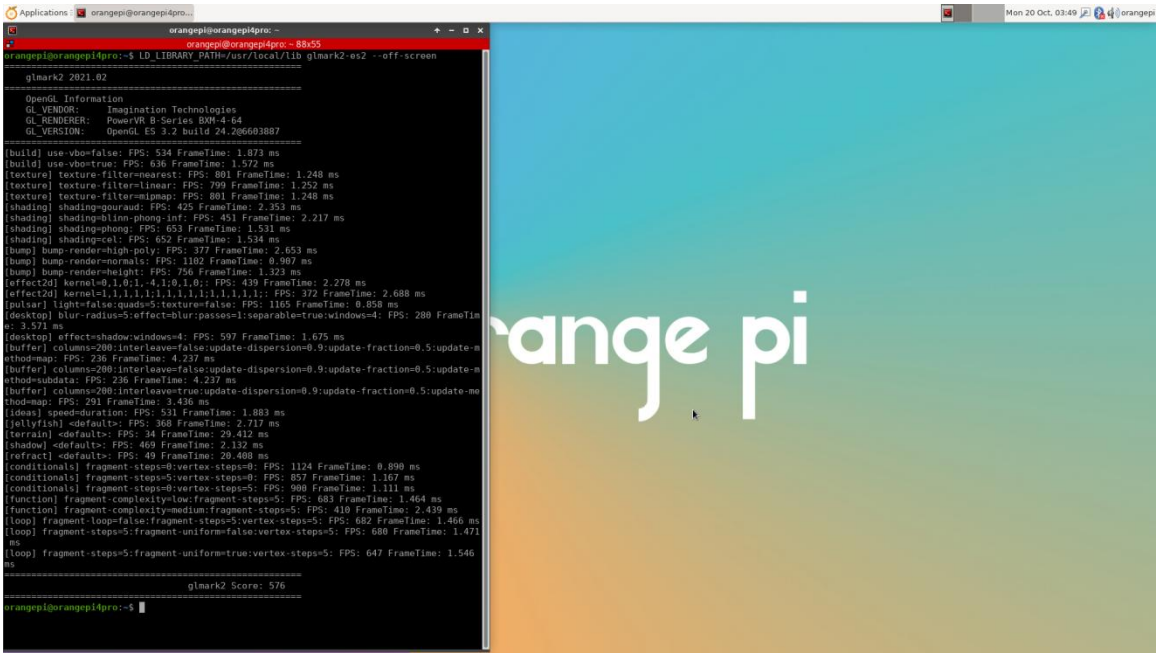
```
orangepi@orangepi:~$ LD_LIBRARY_PATH=/usr/local/lib glmark2-es2
orangepi@orangepi:~$ LD_LIBRARY_PATH=/usr/local/lib glmark2-es2 --off-screen
```

6) **glmark2-es2** is a benchmark tool for OpenGL (ES) 2.0. You can use glmark2 to test the performance of GPU OpenGL ES 2.0.

a. The test scores of the **glmark2-es2** command are as follows



b. The test scores of the **glmark2-es2 --off-screen** command are as follows



3. 34. NPU Usage Instructions

3. 34. 1. PC development environment configuration

3. 34. 1. 1. Installing the Docker Environment

1) First, install the Docker software on your Ubuntu computer. After installing Docker, if you can see the version number using the docker -v command, it means the installation is successful.

```
$ sudo apt update
$ sudo apt install -y docker.io
$ docker -v
```

2) Download the ACUITY Docker compressed package from the **Allwinner network disk** and decompress it.

```
$ unzip docker_images_v2.0.x.zip
```

3) Then execute the following command to import the Docker image containing the Allwinner NPU development environment.

```
$ cd docker_images_v2.0.x
$ unzip ubuntu-npu_v2.0.10.tar.zip
$ sudo docker load -i ubuntu-npu_v2.0.10.tar
```

4) Then download the **ai-sdk.tar.gz** compressed package from the official tool on the Orange Pi website and unzip it to the docker_data directory.

```
$ mkdir docker_data
$ tar -xvf ai-sdk.tar.gz -C docker_data
```

5) Then create and enter the docker container.

```
$ cd docker_data
$ sudo docker run --ipc=host -itd -v ${PWD}:/workspace --name allwinner_v2.0.10 ubuntu-npu:v2.0.10 /bin/bash
$ sudo docker exec -it allwinner_v2.0.10 /bin/bash
```

3.34.1.2. Model conversion and quantization

1) Before starting the model conversion, you need to configure the model compilation script

```
# cd /workspace/ai-sdk/models
# source env.sh v3
# cp ../scripts/* .
```

2) The **ai-sdk/models** directory provides conversion examples for multiple framework models. The following uses the **yolov5s** model as an example to explain how to use the model deployment script.

3) Directory structure of **yolov5s**:

```
# tree yolov5s-sim
yolov5s-sim
|-- channel_mean_value.txt
|-- convert_export.sh
|-- dataset.txt
|-- images
|      |-- COCO_train2014_000000000529.jpg
|      |-- COCO_train2014_000000001183.jpg
|      |-- COCO_train2014_000000002349.jpg
|      |-- COCO_train2014_000000003685.jpg
|      |-- COCO_train2014_000000004463.jpg
|      `-- dog.jpg
|-- inference_compare.sh
|-- inputs_outputs.txt
`-- yolov5s-sim.onnx        #The onnx model file of yolov5s has fixed the input size to (1,3,640,640)

1 directory, 12 files
```

4) Generate uint8 quantized NBG file

a. Import the model.

```
# ./pegasus_import.sh yolov5s-sim
```

b. Modify the **scale** of the yolov5s-sim_inputmeta.yml file to 0.00392157

```
# vim yolov5s-sim/yolov5s-sim_inputmeta.yml
```

c. Perform quantization, using uint8 quantization by default.

```
# ./pegasus_quantize.sh yolov5s-sim uint8
```

d. Run simulation verification.

```
# ./pegasus_inference.sh yolov5s-sim uint8
```

e. Export NBG model files available for NPU

```
# ./pegasus_export_ovx.sh yolov5s-sim uint8
```

f. Export NBG model files that can be used by NPU

```
yolov5s-sim/wksp/yolov5s-sim_uint8_nbg_unify/network_binary.nb
```

3.34.2. Board-side model deployment

3.34.2.1. Board Environment Preparation

1) First, you need to install opencv and cmake on the development board.

```
orangepi@orangepi:~$ sudo apt update
orangepi@orangepi:~$ sudo apt install libopencv-dev cmake
```

2) Then download the **ai-sdk.tar.gz** compressed package from the official tool on the Orange Pi official website.

3) Then upload the compressed package to the development board and decompress it.

```
orangepi@orangepi:~$ tar -xvf ai-sdk.tar.gz
```

4) After the compressed package is decompressed, the directory structure is as follows.

```
orangepi@orangepi:~$ cd ai-sdk
orangepi@orangepi:~/ai-sdk$ ls
examples machinfo Makefile models scripts tests tools unified-android
unified-tina viplite-android viplite-tina
```

3.34.2.2. Run the yolov5 object detection example

1）First compile the yolov5 example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/yolov5
orangepi@orangepi:~/ai-sdk/examples/yolov5$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/yolov5$ cd build
```

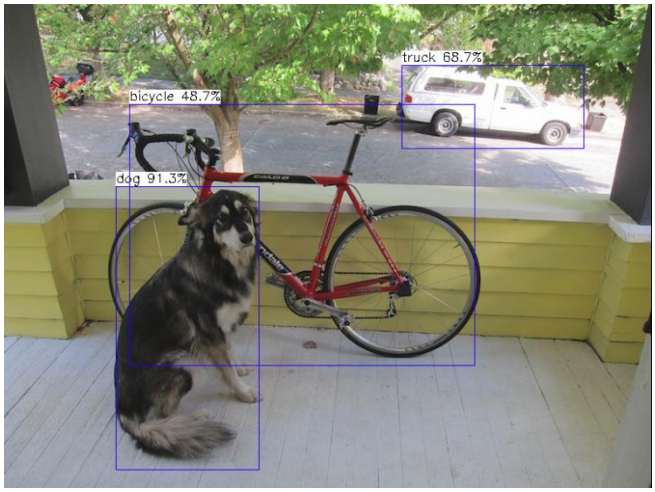
```
orangepi@orangepi:~/ai-sdk/examples/yolov5/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/yolov5/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/yolov5/build$ ./yolov5 ../model/v3/yolov5.nb ../input_data/dog.jpg ./yolov5 nbg input
VIPLite driver software version 2.0.3.2-AW-2024-08-30
yolov5_preprocess.cpp run.
yolov5_postprocess.cpp run.
detection num: 3
16:  87%, [ 134,  216,  308,  552], dog
 2:  54%, [ 463,   77,  693,  170], car
 1:  49%, [ 160,  131,  562,  423], bicycle
```

3）The output will be saved as result.png in the current directory

```
orangepi@orangepi4a:~/ai-sdk/examples/yolov5/build$ ls result.png
result.png
```



3. 34. 2. 3. Run the ResNet50 image classification example

1）First compile the resnet50 example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/resnet50
orangepi@orangepi:~/ai-sdk/examples/resnet50$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/resnet50$ cd build
orangepi@orangepi:~/ai-sdk/examples/resnet50/build$ cmake ..
```

```
orangepi@orangepi:~/ai-sdk/examples/resnet50/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/resnet50/build$ ./resnet50 ../model/v3/resnet50.nb ../input_data/dog_224_224.jpg
```

3）The output information is as follows, which outputs the top 5 predicted by the model, among which the most likely category is collie

```
orangepi@orangepi:~/ai-sdk/examples/resnet50/build$ ./resnet50 ../model/v3/resnet50.nb ../input_data/dog_224_224.jpg
...
========== top5 ==========
class id: 231, prob: 10.750525, label: collie
class id: 160, prob: 10.616143, label: Afghan hound, Afghan
class id: 169, prob: 10.481762, label: borzoi, Russian wolfhound
class id: 230, prob: 10.347381, label: Shetland sheepdog, Shetland sheep dog, Shetland
class id: 176, prob: 6.987841, label: Saluki, gazelle hound
class_postprocess success.
```

3. 34. 2. 4. Run the struct2depth depth detection example

1）First compile the struct2depth example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/struct2depth
orangepi@orangepi:~/ai-sdk/examples/struct2depth$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/struct2depth$ cd build
orangepi@orangepi:~/ai-sdk/examples/struct2depth/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/struct2depth/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/struct2depth/build$ ./struct2depth -b ../model/v3/struct2depth.nb -i ../input_data/0015.jpg
```

3）The depth information of the model inference will be saved in jpg and txt files.

```
orangepi@orangepi:~/ai-sdk/examples/struct2depth/build$ ls disp_* output_*
disp_color.jpg  disp_show.jpg  output_1.txt  output_3.txt
disp_gray.jpg   output_0.txt   output_2.txt
```

3.34.2.5. Running the transformer_cls Image Classification Example

1）First compile the transformer_cls example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/transformer_cls
orangepi@orangepi:~/ai-sdk/examples/transformer_cls$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/transformer_cls$ cd build
orangepi@orangepi:~/ai-sdk/examples/transformer_cls/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/transformer_cls/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/transformer_cls/build$ ./transformer_cls ../model/v3/transformer_cls.nb ../input_data/cat_224_224.jpg
```

3）The image classification output results are shown below.

```
orangepi@orangepi:~/ai-sdk/examples/transformer_cls/build$ ./transformer_cls ../model/v3/transformer_cls.nb ../input_data/cat_224_224.jpg
./transformer_cls nbg input
VIPLite driver software version 2.0.3.2-AW-2024-08-30
get jpeg success.
trans data success.
awnn_set_input_buffers success.
awnn_run success.
class_postprocess.cpp run.
========== top5 ==========
class id: 281, prob: 8.171444, label: tabby, tabby cat
class id: 285, prob: 6.455214, label: Egyptian cat
class id: 282, prob: 6.139260, label: tiger cat
class id: 287, prob: 3.832908, label: lynx, catamount
class id: 756, prob: 3.263942, label: rain barrel
class_postprocess success.
```

3.34.2.6. Run the YOLACT object detection and segmentation example

1）First compile the yolact example.

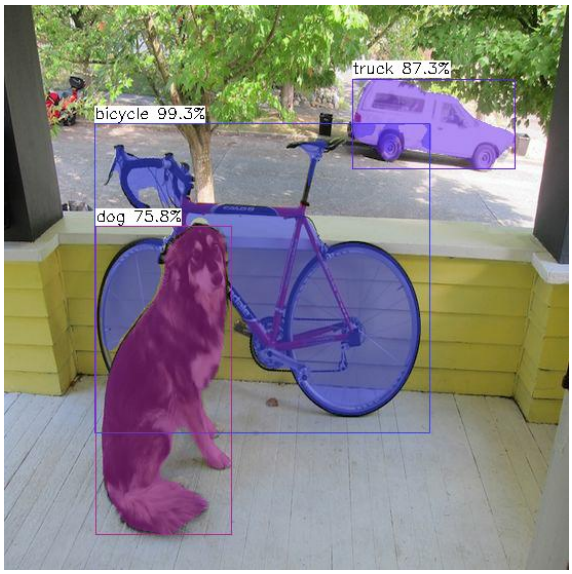
```
orangepi@orangepi:~/ai-sdk$ cd examples/yolact
orangepi@orangepi:~/ai-sdk/examples/yolact$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/yolact$ cd build
orangepi@orangepi:~/ai-sdk/examples/yolact/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/yolact/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/yolact/build$ ./yolact ../model/v3/yolact.nb ../input_data/dog_550_550.jpg
```

3）The output will be saved as result.png in the current directory

```
orangepi@orangepi:~/ai-sdk/examples/yolact/build$ ls result.png
result.png
```



3. 34. 2. 7. Run the deepspeech2 speech recognition example

4）First compile the deepspeech2 example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/deepspeech2
orangepi@orangepi:~/ai-sdk/examples/deepspeech2$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/deepspeech2$ cd build
orangepi@orangepi:~/ai-sdk/examples/deepspeech2/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/deepspeech2/build$ make
```

5）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/deepspeech2/build$ ./deepspeech2  ../model/v3/deepspeech2.nb ../input_data/1188-133604-0010.flac_756_161_1.tensor
```

6）The output is as follows.

```
orangepi@orangepi:~/ai-sdk/deepspeech2/build$ ./deepspeech2  ../model/v3/deepspeech2.nb ../input_data/1188-133604-0010.flac_756_161_1.tensor
input tensor_file = ../input_data/1188-133604-0010.flac_756_161_1.tensor
VIPLite driver software version 2.0.3.2-AW-2024-08-30
Original array: but  in  thiss  vvingyet copiiedd ffrom  turrrnerr  yoou haavve the  ttwo  prrincciiplleess  brrouught  oout perrfeccttllly
Modified array : but in this vingyet copied from turner you have the two principles brought out perfectly
```

3. 34. 2. 8. Run the ChineseOCR text recognition example

1）First compile the ChineseOCR example.

```
orangepi@orangepi:~/ai-sdk$ cd examples/chineseocr
orangepi@orangepi:~/ai-sdk/examples/chineseocr$ mkdir build
orangepi@orangepi:~/ai-sdk/examples/chineseocr$ cd build
orangepi@orangepi:~/ai-sdk/examples/chineseocr/build$ cmake ..
orangepi@orangepi:~/ai-sdk/examples/chineseocr/build$ make
```

2）Then execute the following command to run the example.

```
orangepi@orangepi:~/ai-sdk/examples/chineseocr/build$ ./chineseocr -d ../model/v3/ -1 dbnet_1024 -2 angle_net -3 crnn_lite_lstm_256 -4 keys.txt -i ../input_data/1.jpg
```

3）The output is as follows, and you can see that the text in the image has been recognized.

```
orangepi@orangepi:~/ai-sdk/examples/chineseocr/build$ ./chineseocr -d ../model/v3/ -1 dbnet_1024 -2 angle_net -3 crnn_lite_lstm_256 -4 keys.txt -i ../input_data/1.jpg
...
=====End detect=====
FullDetectTime(903.447417ms)
We at Allwinner Technology
AI
chips
take off! 12345666666!
```

```
run finished.
~CrnnNet.
~AngleNet.
~DbNet.
~NpuUint.
```

3. 35. How to burn Linux image to eMMC

Note that the development board can be booted from either a TF card or an eMMC, but the TF card has a higher priority than the eMMC. That is, if a TF card is inserted into the development board and there is a system in the TF card, the system in the TF card will be booted by default, not the system in the eMMC.

On Debian Bullseye systems, it is recommended to uninstall the GPU driver before installation. Otherwise, the system may crash or fail during the installation process.

orangepi@orangepi:~$ **sudo service lightdm stop**

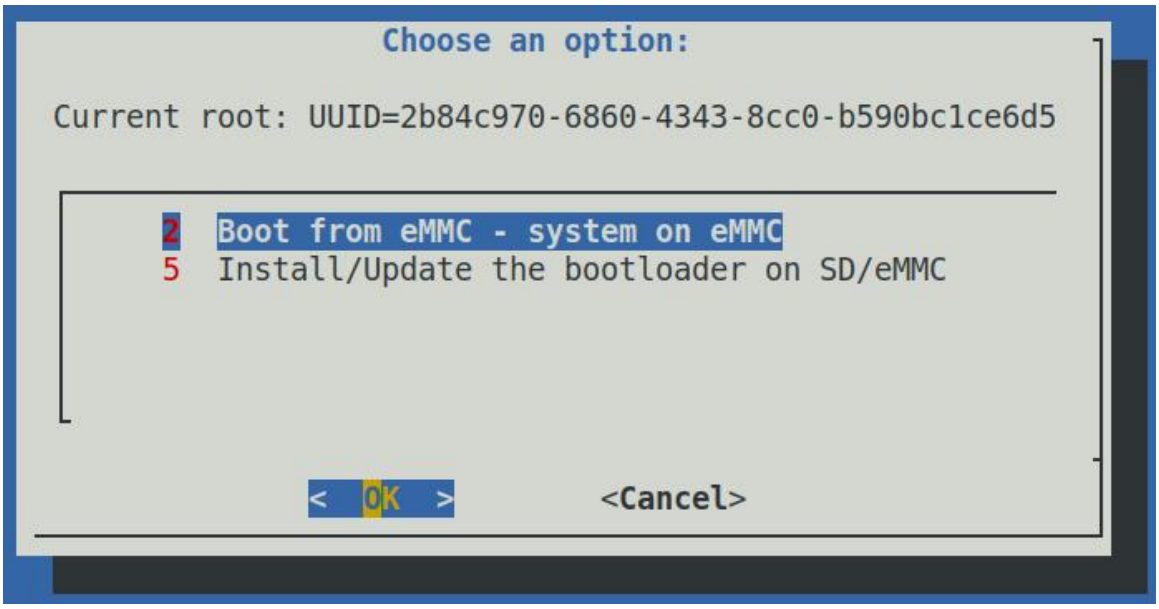
orangepi@orangepi:~$ **sudo rmmod pvrsrvkm**

1) Burning the Linux image to the eMMC requires the use of a TF card. First, burn the Linux image to the TF card, then start the development board and enter the Linux system.

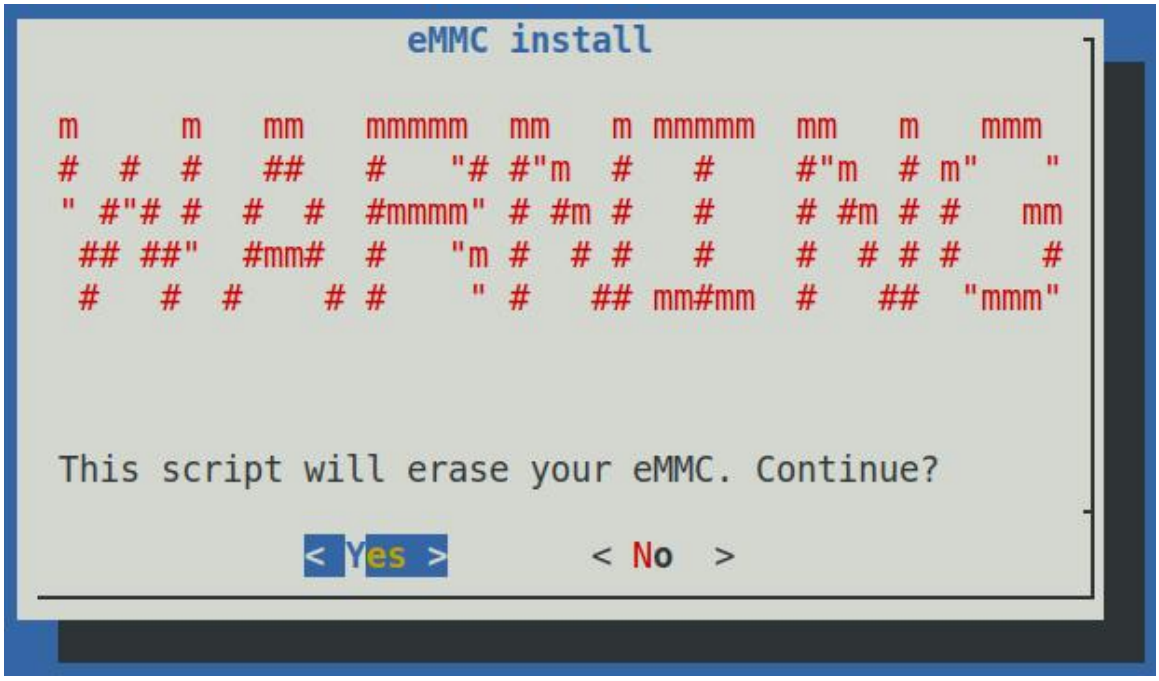
2) Then run the **nand-sata-install** script, **remember to add sudo permissions**

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

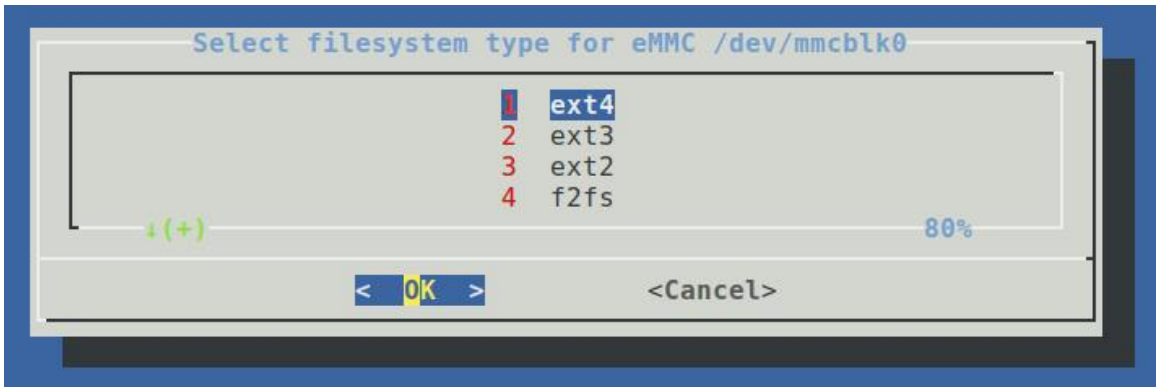
3) Then select **2 Boot from eMMC - system on eMMC**



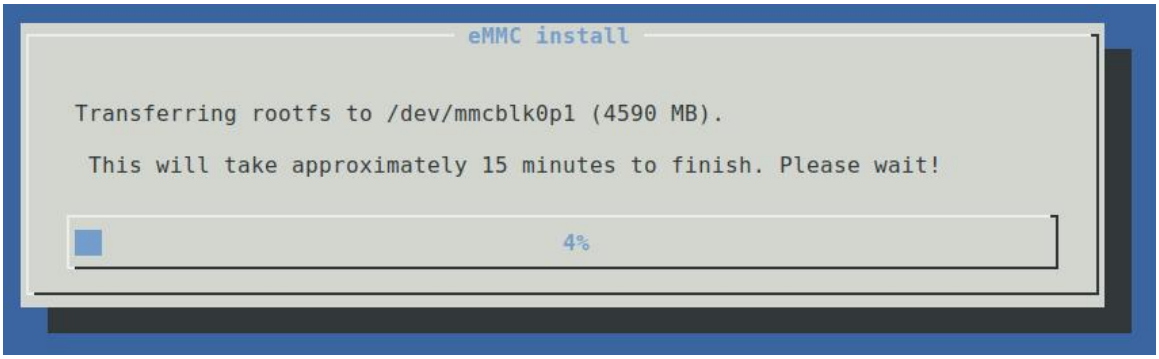
4) A warning will pop up, and the script will erase all data on the eMMC. Select **<Yes>** to continue.



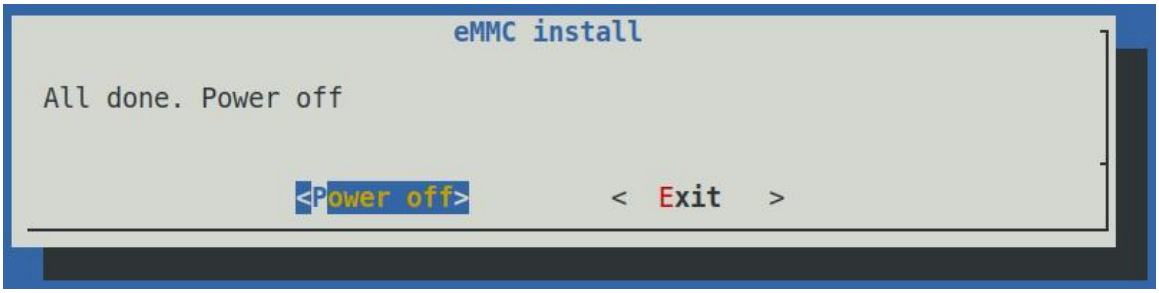
5) You will then be prompted to select the file system type. Five file systems are supported: ext2/3/4, f2fs, and btrfs.



6) Then it will start to format the eMMC. After formatting the eMMC, it will start to burn the Linux image to the eMMC.



7) After burning, the following options will be prompted. You can select **<Power off>** to shut down directly



8) Then remove the TF card and power on again, the Linux system in the eMMC will start

3. 36. How to burn the Linux image to SPIFlash+NVMe SSD

Please note that currently SPIFlash+NVMe SSD startup is only tested and confirmed to be usable on SSDs from Fanxiang and Guangcun. Other SSDs such as Kingston may have compatibility issues, causing the system to fail to start properly.

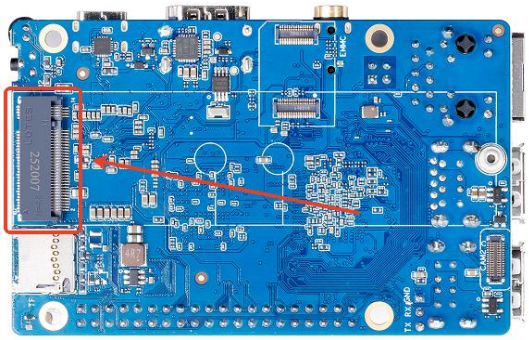
On the Debian Bullseye system, it is recommended to uninstall the GPU driver before installation, otherwise system crashes or exceptions may occur during the installation process.

orangepi@orangepi:~$ **sudo service lightdm stop**

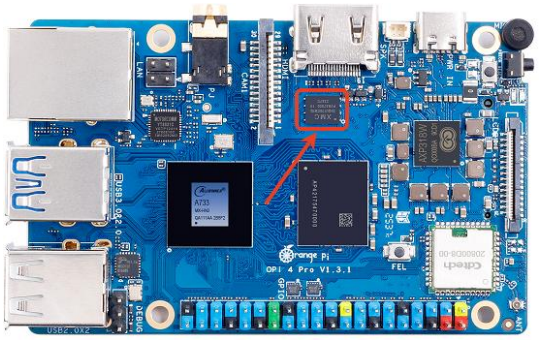
orangepi@orangepi:~$ **sudo rmmod pvrsrvkm**

1) First, you need to prepare an NVMe SSD solid-state drive. The development board's M.2 slot PCIe interface specification is PCIe3.0x1.

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



3) The location of the SPI Flash on the development board is shown in the figure below. No other settings are required before starting to burn.



4) Burning the image to the SPIFlash + NVMe SSD requires a TF card. First, burn the Linux image to the TF card, then use the TF card to boot the development board into the Linux system. For instructions on burning the Linux image to the TF card, see "**Burn the Linux image to a TF card using a Windows PC**" and "**Burn the Linux image to a TF card using an Ubuntu PC.**"

5) After booting the Linux system with the TF card, you can burn the image to the SPI Flash + NVMe SSD

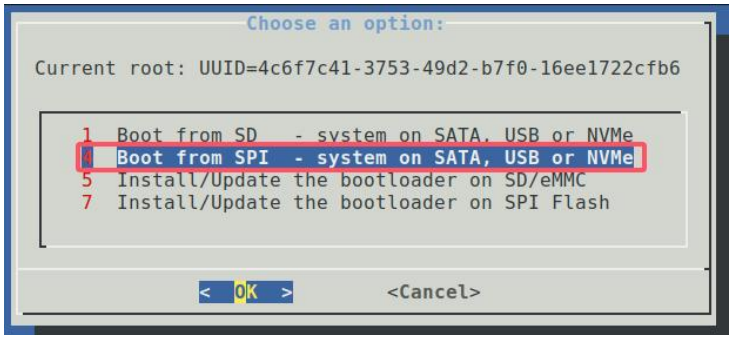
a. First, create a partition for the NVMe SSD.

```
orangepi@orangepi:~$ sudo parted --script /dev/nvme0n1 mklabel gpt mkpart primary ext4 65536s 100%
```

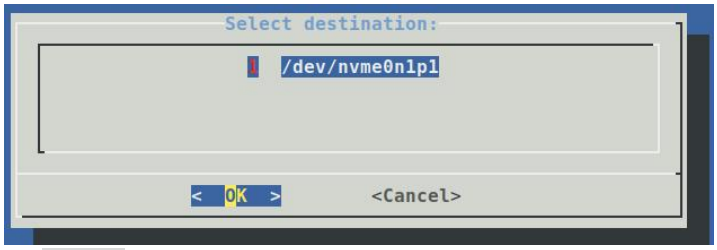
b. Then run **nand-sata-install**. Ordinary users must remember to add sudo permissions.

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

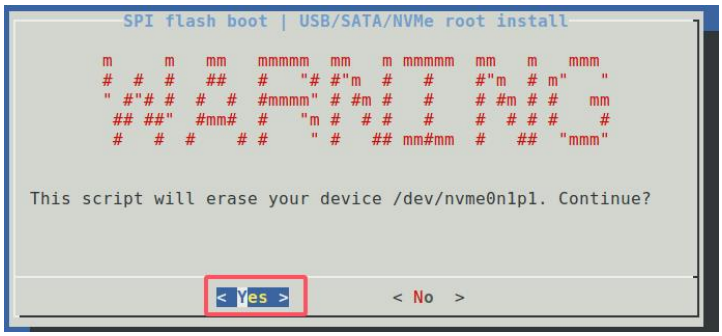
c. Then select**4 Boot from SPI - system on SATA, USB or NVMe**。



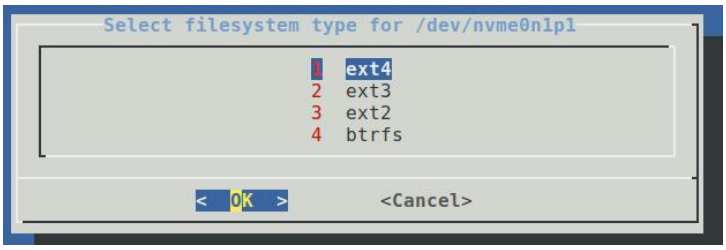
d. Then press Enter to confirm



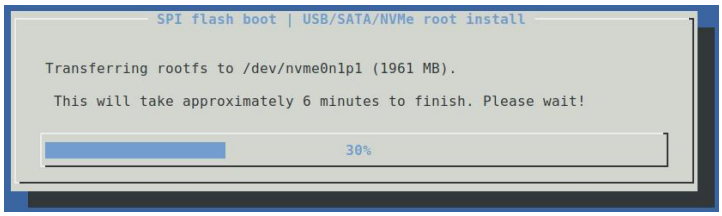
e. Then select**<Yes>**。



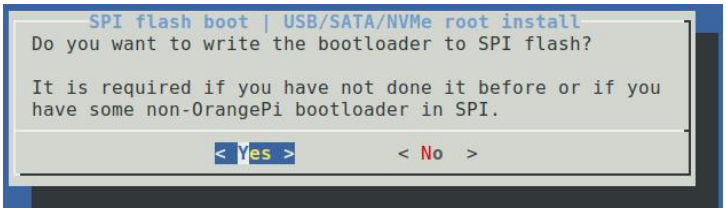
f. You will then be prompted to select the file system type.

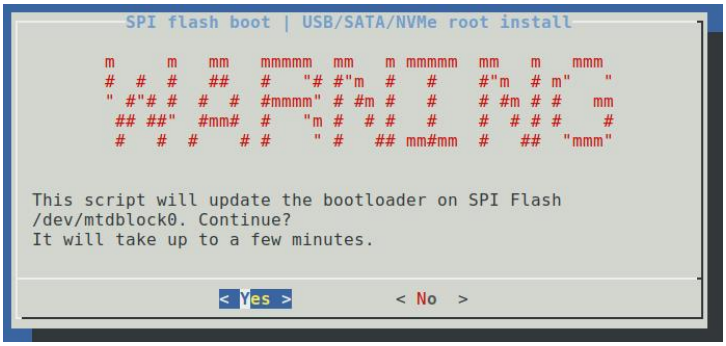


g. Then the NVMe SSD will start to be formatted. After the formatting is completed, the system will be burned to the NVMe SSD.



h. Then please wait patiently for the burning to complete. After the burning is complete, you will be prompted to choose whether to burn the bootloader to the SPI Flash. Select **<Yes>**.





i. After burning, the following options will be prompted. You can select **<Power off>** to shut down directly

3. 37. How to use the system backup script opi-bkimg

opi-bkimg is a system backup script designed for the Orange Pi series development boards. It can help users completely back up the currently running Linux system to an image file for subsequent restoration or migration to other devices.

Before starting the backup, please make sure there is enough space in the system, otherwise the backup will fail.

1) Run the **opi-bkimg** script to start backing up the system.

```
orangepi@orangepi:~$ sudo opi-bkimg
```

2) If the**opi-bkimg** script is not available in your system, you can download it from the following repository.

```
https://gitee.com/orangepi-xunlong/opi-bkimg
```

3) The script checks the root file system size and creates a mirror file of the appropriate size. It then uses the **rsync** command to copy the system data to the mirror file. The specific process is as follows.

```
orangepi@orangepi:~$ sudo opi-bkimg
[ o.k. ] Starting backup to image [ /mnt/backup.img ]
[ o.k. ] Current rootfs size [ 2561 MiB ]
[ o.k. ] Creating blank image for rootfs [ 3500 MiB ]
[ .... ] dd: 3.42GiB [ 126MiB/s] [=========================================>] 100%
[ o.k. ] Creating partitions [ root: ext4 ]
[ .... ] Creating rootfs [ ext4 on /dev/loop1p1 ]
[ .... ] Copying file to [ / ] [################################] 100%
[ .... ] Re-enabling [ orangepi-resize-filesystem ]
[ o.k. ] Writing U-boot bootloader [ /dev/loop1 ]
[ o.k. ] Unmounting [ /mnt/opi-bkimg.iaioMi/rootfs ]
[ o.k. ] Backup completed [ /mnt/backup.img ]
orangepi@orangepi:~$
```

4) When the script displays the message "**Backup completed**", the backup is complete and the image generation path is as follows:

```
orangepi@orangepi:~$ ls /mnt/
backup.img
```

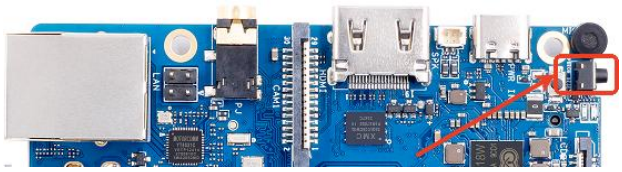
5) **backup.img** is the image file that can be burned.

3. 38. How to shut down and restart the development board

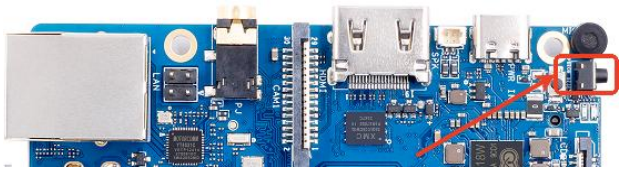
1) If you unplug the power cord while the Linux system is running, some data may be lost in the file system. It is recommended to shut down the Linux system of the development board using the **poweroff** command before unplugging the power cord.

```
orangepi@orangepi:~$ sudo poweroff
```

2) In addition, the development board is equipped with a power button, and you can also **short press** the power button on the development board to shut down



3) After shutting down, long press the power button on the development board to turn it on.



4) Use the **reboot** command to restart the Linux system in the development board

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

4. Linux SDK——orangepi-build usage instructions

4. 1. Compilation System Requirements

The Linux SDK, **orangepi-build**, only runs on x64 computers with **Ubuntu 22.04** installed. Before downloading orangepi-build, ensure that your computer is running Ubuntu 22.04. The command to check the installed Ubuntu version is as follows. If the Release field does not display **22.04**, the current Ubuntu version does not meet the requirements. Please change the operating system before proceeding with the following steps.

```
test@test:~$ lsb_release -a
No LSB modules are available.
```

```
Distributor ID: Ubuntu
Description:    Ubuntu 22.04 LTS
Release:        22.04
Codename:       jammy
```

If your computer is running Windows and does not have Ubuntu 22.04 installed, you can consider using **VirtualBox** or **VMware** to install an Ubuntu 22.04 virtual machine in your Windows system. However, please note that you should not compile orangepi-build in a WSL virtual machine, as orangepi-build has not been tested in a WSL virtual machine, so it cannot be guaranteed to work properly in WSL. In addition, please do not use orangepi-build in the Linux system of the **development board**. The installation image download address for the Ubuntu 22.04 **amd64** version is:

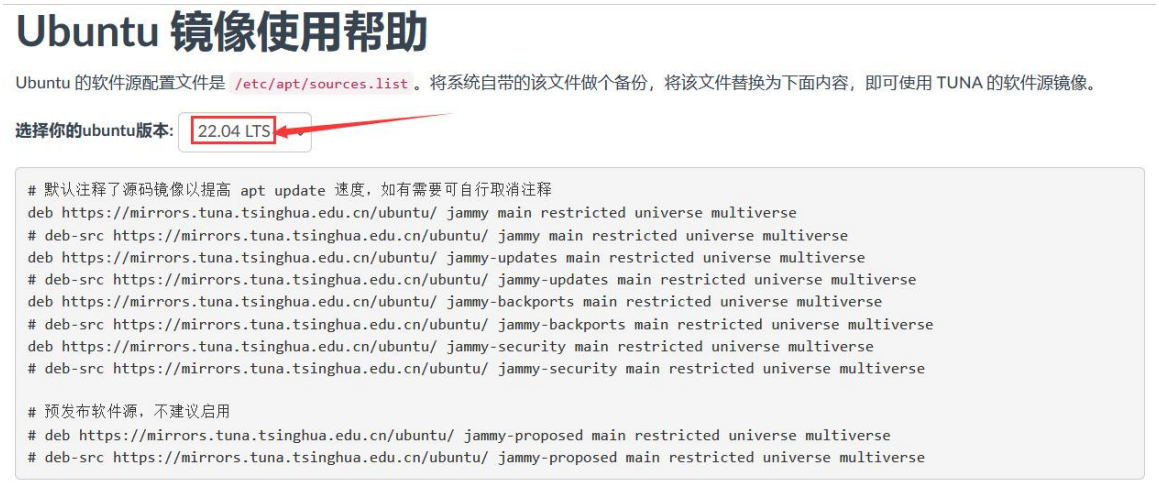
https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/22.04/ubuntu-22.04-desktop-amd64.iso

After installing Ubuntu 22.04 on your computer or in a virtual machine, please first set the software source of Ubuntu 22.04 to Tsinghua source (or other domestic sources that you think are faster). Otherwise, it is easy to encounter errors when installing software later due to network problems. The steps to replace the Tsinghua source are as follows:

a. For instructions on how to replace the Tsinghua source, please refer to this page.

https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/

b. Note that the Ubuntu version needs to be switched to 22.04.



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 mirror is commented out by default to speed up apt update. You can uncomment it if necessary.
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# Pre-release software source, not recommended to enable
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

d. After the replacement, you need to update the package information and ensure that there are no errors.

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

e. In addition, since the source code of the kernel and U-boot are all stored on GitHub, it is very important to ensure that the computer can download the code from GitHub normally when compiling the image.

4. 2. Get the source code of Linux SDK

4. 2. 1. Download orangepi-build from GitHub

The Linux SDK refers to the OrangePi-Build code. OrangePi-Build is a modified version of the Armbian Build compiler system. Using OrangePi-Build, you can compile multiple versions of Linux images. You can download the OrangePi-Build code using the following command:

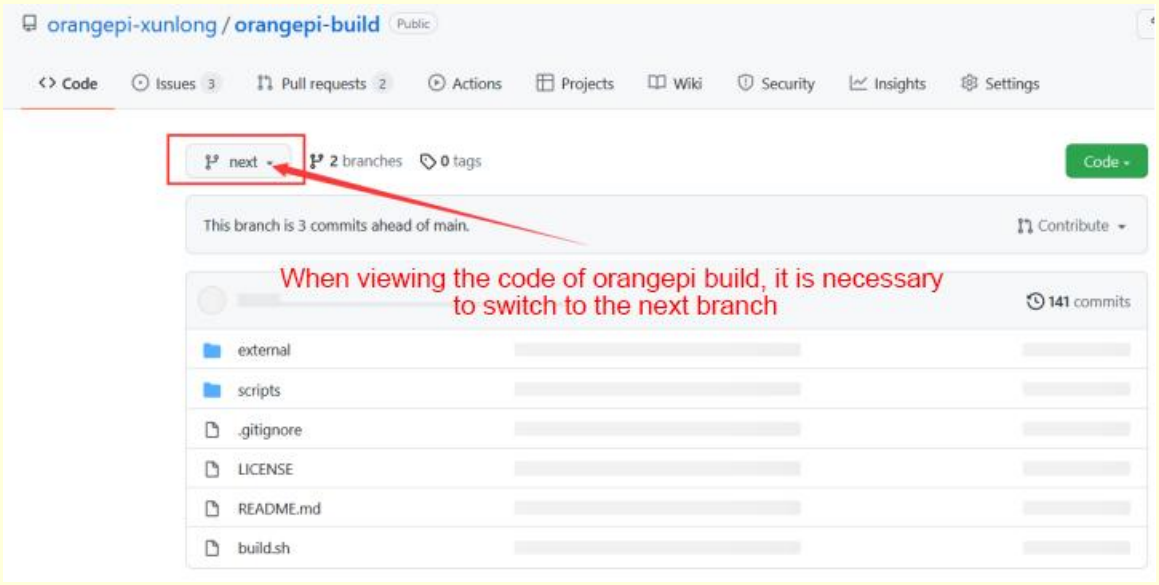
a. The command downloaded from GitHub is as follows:

```
test@test:~$ sudo apt-get update && sudo apt-get install -y git
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git -b next
```

b. The command downloaded from Gitee is as follows:

```
test@test:~$ sudo apt-get update && sudo apt-get install -y git
test@test:~$ git clone https://gitee.com/orangepi-xunlong/orangepi-build.git -b next
```

Note that to use the A733 SoC development board, you need to download the next branch source code of orangepi-build. The above git clone command needs to specify the branch of orangepi-build source code as next.



You do not need to enter the username and password of your GitHub account when downloading the orangepi-build code through the git clone command (the same applies to downloading other codes in this manual). If your Ubuntu PC prompts you to enter the username and password of your GitHub account after entering the git clone command, it usually means that the address of the orangepi-build repository after git clone is entered incorrectly. Please check the command spelling carefully for any errors, rather than thinking that we forgot to provide the username and password of our GitHub account.

The u-boot and Linux kernel versions currently used by the A733 series development boards are as follows:

branch	u-boot version	Linux kernel version
current	u-boot v2018.05	linux5.15

The branch mentioned here is not the same as the branch of the orangepi-build source code, so please don't confuse them. This branch is mainly used to distinguish different kernel source code versions.

The Linux 5.15 BSP kernel currently provided by Allwinner is defined as the

current branch.

After downloading orangepi-build, it will contain the following files and folders:

a. **build.sh**: Compile the startup script
b. **external**: Contains configuration files, specific scripts, and source code for some programs needed to compile the image.
c. **LICENSE**: GPL 2License 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 downloaded the orangepi-build code from GitHub, you might find that it doesn't include the u-boot and Linux kernel source code, nor the cross-compilation toolchain required to compile them. This is normal, as these are stored in separate GitHub repositories or on servers (their locations are detailed below). orangepi-build specifies the locations of the u-boot, Linux kernel, and cross-compilation toolchain in its scripts and configuration files. When running orangepi-build, if it detects that these are not available locally, it will automatically download them from the appropriate locations.

4.2.2. Download the cross-compilation toolchain

When orangepi-build is run for the first time, it will automatically download the cross-compilation toolchain and place it in the **toolchains** folder. Each time you run the build.sh script of orangepi-build, it will check whether the cross-compilation toolchain in **toolchains** exists. If not, it will restart the download. If it exists, it will use it directly without repeating the download.

```
[ 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
```

The mirror website of the cross-compilation tool chain in China is the open source software mirror site of Tsinghua University:

https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/

Toolchains After downloading, it will contain multiple versions of cross-compilation tool chains:

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu
gcc-linaro-4.9.4-2017.01-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-arm-11.2-2022.02-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

The cross-compilation tool chain used to compile the A733 Linux kernel source code is:

a. linux5.15

gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu

The cross-compilation tool chain used to compile the A733 u-boot source code is:

a. v2018.05

gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi

4.2.3. orangepi-build complete directory structure description

2) After downloading the orangepi-build repository, it does not contain the source code of the Linux kernel, u-boot, and the cross-compilation toolchain. The source code of the Linux kernel and u-boot are stored in independent git repositories.

a. The git repository where the Linux kernel source code is stored is as follows:

a) Github

https://github.com/orangepi-xunlong/linux-orangepi/tree/**orange-pi-5.15-sun60iw2**

b) Gitee

https://gitee.com/orangepi-xunlong/**orange-pi-5.15-sun60iw2**

b. The git repository where the u-boot source code is stored is as follows:

a) Github

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/**v2018.05-sun60iw2**

b) Gitee

https://gitee.com/orangepi-xunlong/u-boot-orangepi/tree/**v2018.05-sun60iw2**

3) When orangepi-build is run 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 visible in orangepi-build are

a. **build.sh**: Compile the startup script
b. **external**: Contains configuration files, scripts for specific functions, and source code for some programs needed to compile the image. The rootfs compressed package cached during the image compilation process is also stored in external
c. **kernel**: Store the source code of the Linux kernel
d. **LICENSE**: GPL 2 License File
e. **README.md**: orangepi-build documentation
f. **output**: Stores compiled u-boot, linux and other deb packages, compilation logs, compiled images and other files
g. **scripts**: Generic script for compiling Linux images
h. **toolchains**: Store cross-compilation tool chain
i. **u-boot**: Store u-boot source code

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
```

4. 3. Compile u-boot

If downloading code from github is slow, you can do it at Add the parameter GITEE_SERVER=yes after ./build.sh, so that it can be downloaded from the gitee website during compilation.

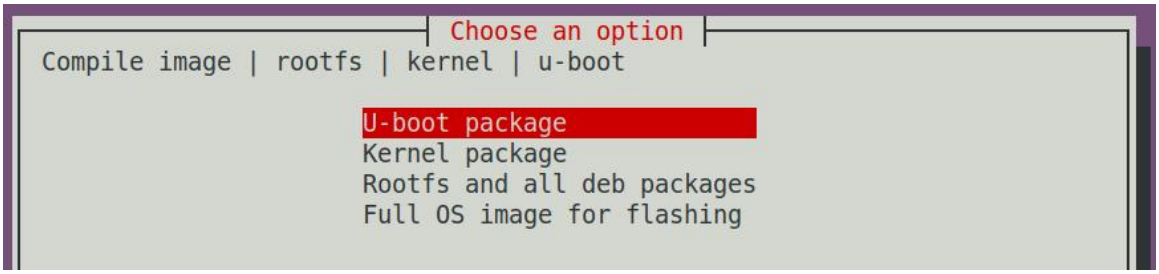
The compilation command for adding the GITEE_SERVER=yes parameter is as follows:

test@test:~/orangepi-build$ **sudo ./build.sh GITEE_SERVER=yes**

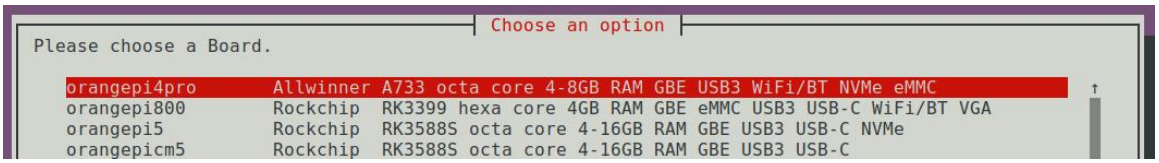
1) Run the build.sh script and remember to add sudo permissions

```
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 compiling u-boot. Some of the information prompted when compiling the current branch is as follows：

a. u-boot source code version

```
[ o.k. ] Compiling u-boot [ v2018.05 ]
```

b. Version of the cross-compilation toolchain

```
[ o.k. ] Compiler version [ aarch64-linux-gnu-gcc 11 ]
```

c. The path of the compiled u-boot deb package

```
[ o.k. ] Target directory [ orangepi-build/output/debs/u-boot ]
```

d. The package name of the compiled u-boot deb package

```
[ o.k. ] File name [ linux-u-boot-current-orangepi4pro_x.x.x_arm64.deb ]
```

e. Compilation time

```
[ 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=orangepi4pro
BRANCH=current BUILD_OPT=u-boot ]
```

5) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-current-orangepi4pro_x.x.x_arm64.deb
```

6) When the orangepi-bulid compilation system compiles the u-boot source code, it will first synchronize the u-boot source code with the u-boot source code on the GitHub server. Therefore, if you want to modify the u-boot source code, you must first turn off the source code download and update function (**you need to fully compile u-boot before turning off this function, otherwise it will prompt that the u-boot source code cannot be found**). Otherwise, all changes made 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"
.....
```

7) When debugging the u-boot code, you can use the following method to update the u-boot in the Linux image for testing

a. First 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-orangepi4pro_x.x.x_arm64.deb root@192.168.1.xxx:/root
```

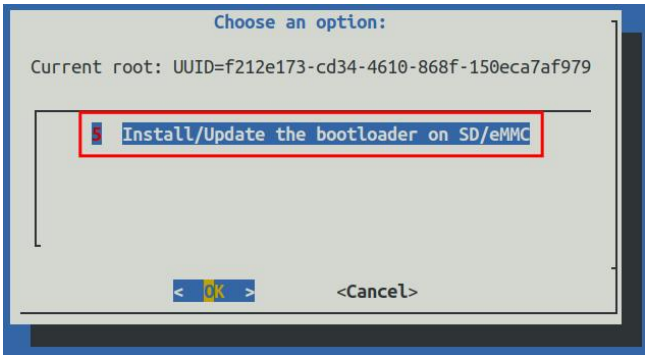
b. Install the new u-boot deb package just uploaded

```
orangepi@orangepi:~$ sudo dpkg -i linux-u-boot-current-orangepi4pro_x.x.x_arm64.deb
```

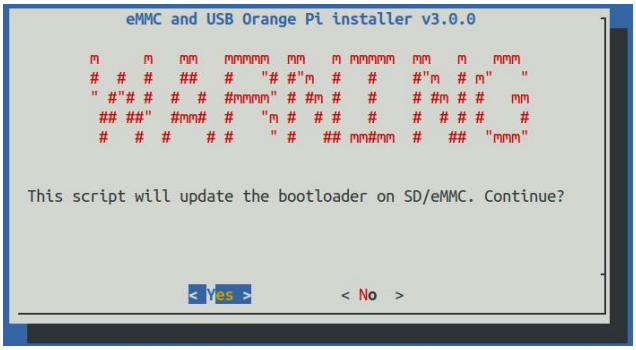
c. Then run the nand-sata-install script

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

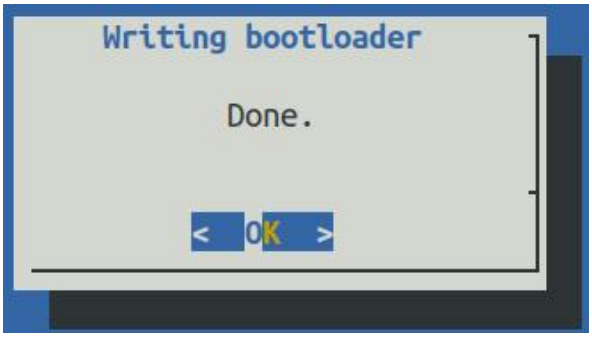
d. Then select **5 Install/Update the bootloader on SD/eMMC**



e. After pressing the Enter key, a Warning will pop up first.



f. Press the Enter key again to start updating u-boot. After the update is complete, the following information will be displayed



g. Then you can restart the development board to test whether the u-boot changes are effective

4. 4. Compile the Linux kernel

If downloading code from github is slow, you can do it at Add the parameter GITEE_SERVER=yes after ./build.sh, so that it can be downloaded from the gitee website during compilation.

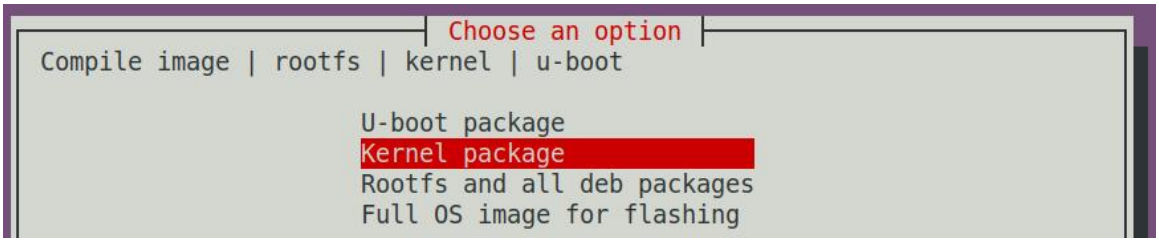
The compilation command for adding the GITEE_SERVER=yes parameter is as follows:

test@test:~/orangepi-build$ **sudo ./build.sh GITEE_SERVER=yes**

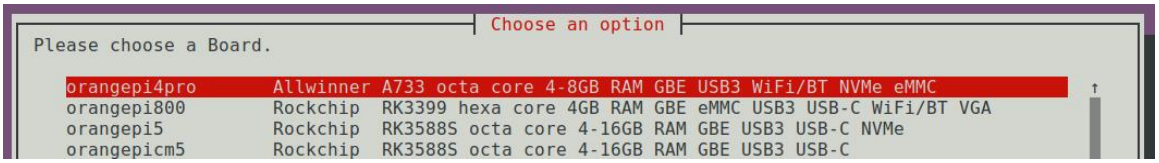
1) Run the **build.sh** script and remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

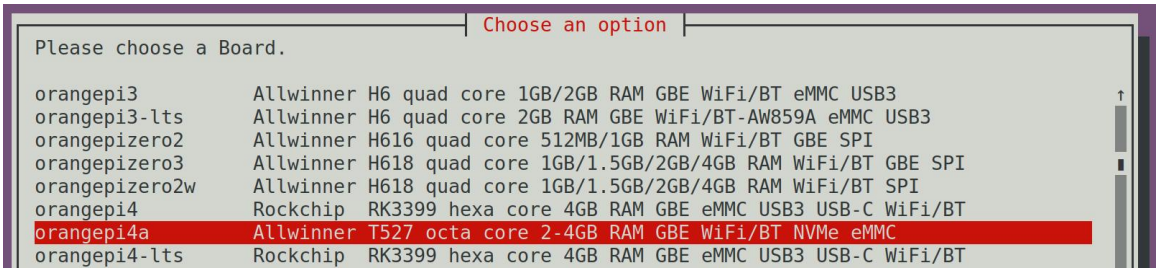
2) Select **Kernel package** and press Enter



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, select the first one. If you need to modify the kernel configuration, select the second one.

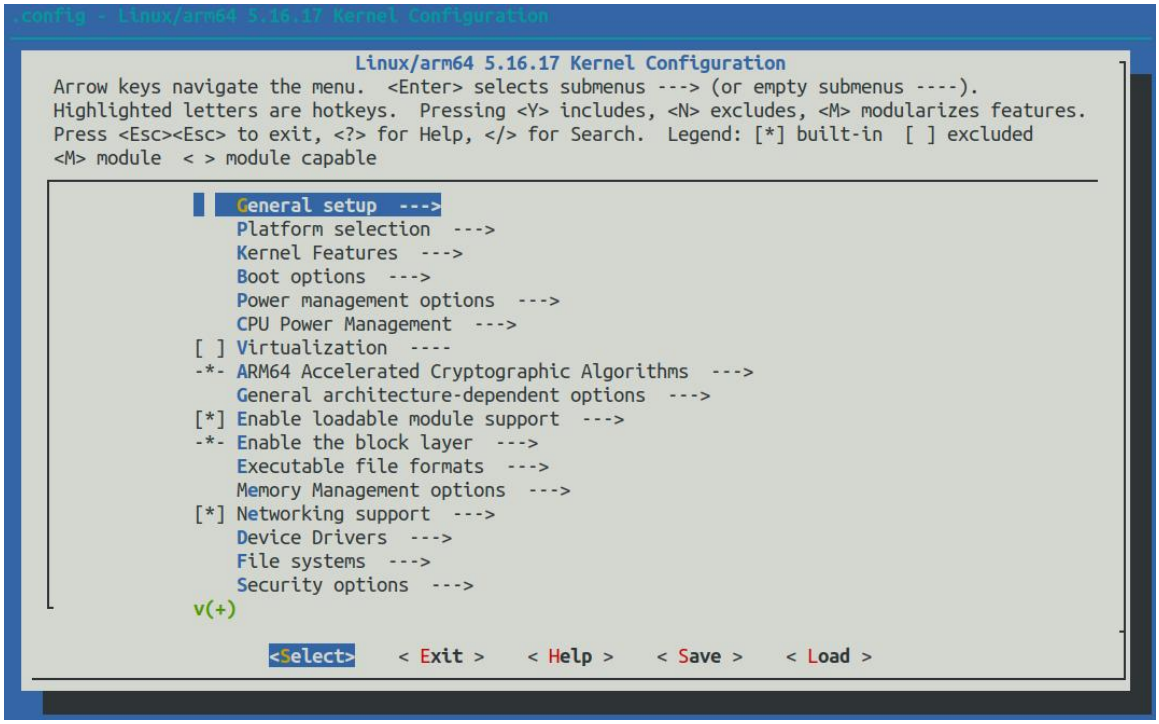


4) Then select the model of the development board



5) If you selected the second option in step 3) to display the kernel configuration menu,

the kernel configuration interface opened by **make menuconfig** will pop up. You can modify the kernel configuration directly at this time. After modifying, save and exit. After exiting, the kernel source code will be compiled.



a. If you do not need to modify the kernel configuration options, when running the build.sh script, pass in **KERNEL_CONFIGURE=no** to temporarily block the kernel configuration interface from popping up.

```
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 message appears when compiling the kernel, it is because the terminal interface of the Ubuntu PC is too small, which causes the make menuconfig interface to be unable to be displayed. Please adjust the terminal of the Ubuntu PC to the maximum size 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) Some of the information prompted when compiling the current branch kernel source code is as follows：

a. Linux kernel source code version

[o.k.] Compiling current kernel [**5.15.147**]

b. Version of the cross-compilation toolchain used

[o.k.] Compiler version [**aarch64-linux-gnu-gcc 11**]

c. The default configuration file used by the kernel and its storage path are as follows

[o.k.] Using kernel config file
[**orangepi-build/external/config/kernel/linux-5.15-sun60iw2-current.config**]

d. The path of the compiled kernel-related deb package

[o.k.] Target directory [**output/debs/**]

e. The package name of the compiled kernel image deb package

[o.k.] File name [**linux-image-current-sun60iw2_x.x.x_arm64.deb**]

f. Compilation time

[o.k.] Runtime [**10 min**]

g. Finally, the compilation command for the kernel selected last time will be displayed. Use the following command to start compiling the kernel source code directly without selecting through the graphical interface.

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepi4pro BRANCH=current BUILD_OPT=kernel KERNEL_CONFIGURE=no**]

7) Check the compiled kernel-related deb package

a. **linux-dtb-current-sun60iw2_x.x.x_arm64.deb** Contains dtb files used by the

kernel

b. **linux-headers-current-sun60iw2_x.x.x_arm64.deb** Include kernel header files

c. **linux-image-current-sun60iw2_x.x.x_arm64.deb** Contains kernel image and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-current-sun60iw2_x.x.x_arm64.deb
output/debs/linux-headers-current-sun60iw2_x.x.x_arm64.deb
output/debs/linux-image-current-sun60iw2_x.x.x_arm64.deb
```

8) When the orangepi-bulid compilation system compiles the Linux kernel source code, it will first synchronize the Linux kernel source code with the Linux kernel source code on the GitHub server. Therefore, if you want to modify the Linux kernel source code, you must first turn off the source code update function (**you need to fully compile the Linux kernel source code before turning off this function, otherwise it will prompt that the Linux kernel source code cannot be found**). Otherwise, all changes made 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) If the kernel has been modified, you can use the following method to update the kernel and kernel modules of the development board Linux system

a. Upload the compiled Linux kernel deb package to the Linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-current-sun60iw2_x.x.x_arm64.deb root@192.168.1.xxx:/root
```

b. Install the new Linux kernel deb package just uploaded

```
orangepi@orangepi:~$ sudo dpkg -i linux-image-current-sun60iw2_x.x.x_arm64.deb
```

c. Then restart the development board and check whether the kernel-related modifications have taken effect

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

4. 5. Compile rootfs

If downloading code from github is slow, you can do it at Add the parameter GITEE_SERVER=yes after ./build.sh, so that it can be downloaded from the gitee website during compilation.

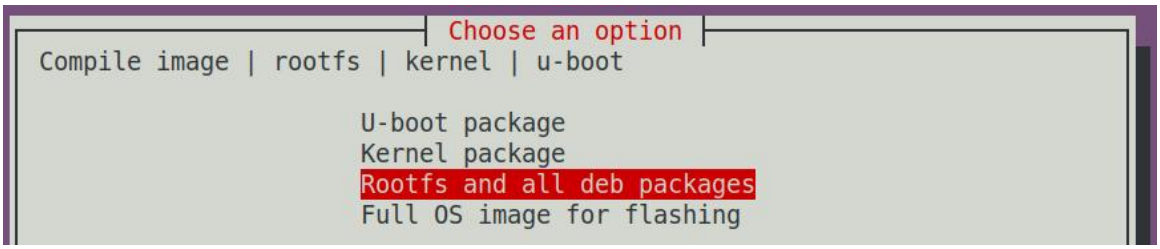
The compilation command for adding the GITEE_SERVER=yes parameter is as follows:

test@test:~/orangepi-build$ **sudo ./build.sh GITEE_SERVER=yes**

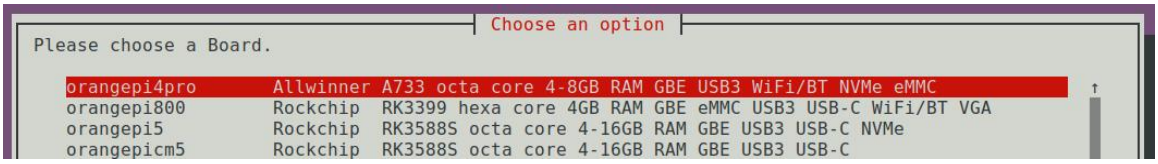
2) Run the build.sh script and remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

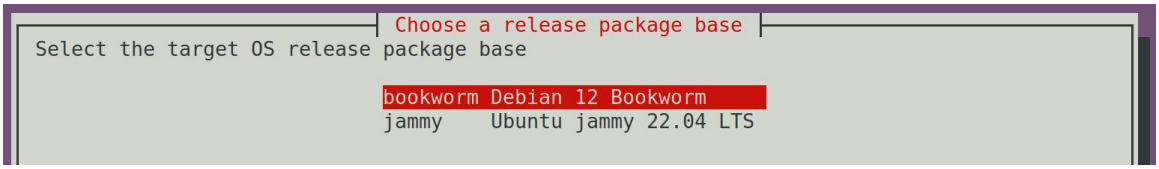
3) Select **Rootfs and all deb packages** and press Enter



4) Then select the model of the development board



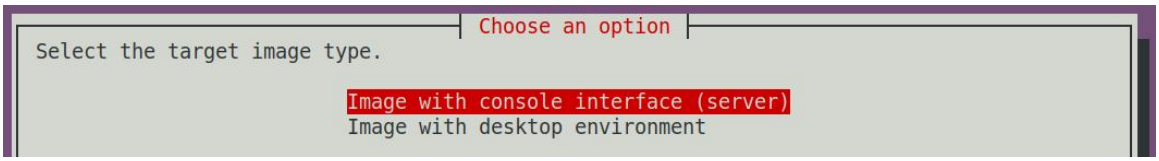
5) Then select the type of rootfs



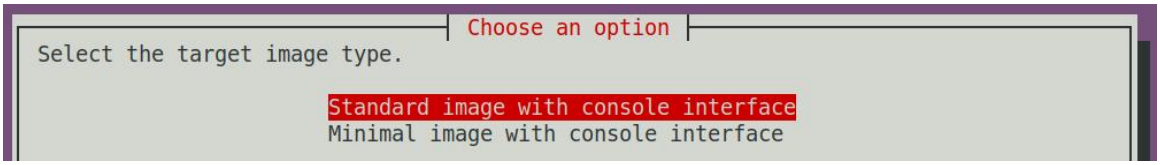
6) Then select the image type

a. **Image with console interface (server)** Indicates the server version of the image, which is relatively small in size

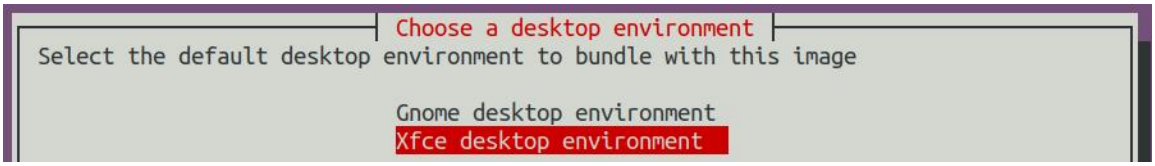
b. **Image with desktop environment** Indicates a mirror image with a desktop, which is relatively large in size.

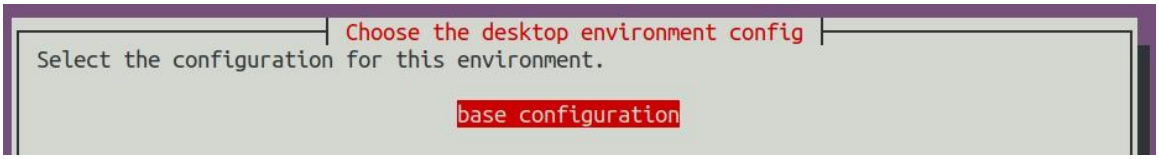


7) If you are compiling a server version image, you can also choose to compile the Standard version or the Minimal version. The Minimal version has much less pre-installed software than the Standard version. (**Please do not choose the Minimal version if you do not have special needs, because many things are not pre-installed by default and some functions may not be available**.)

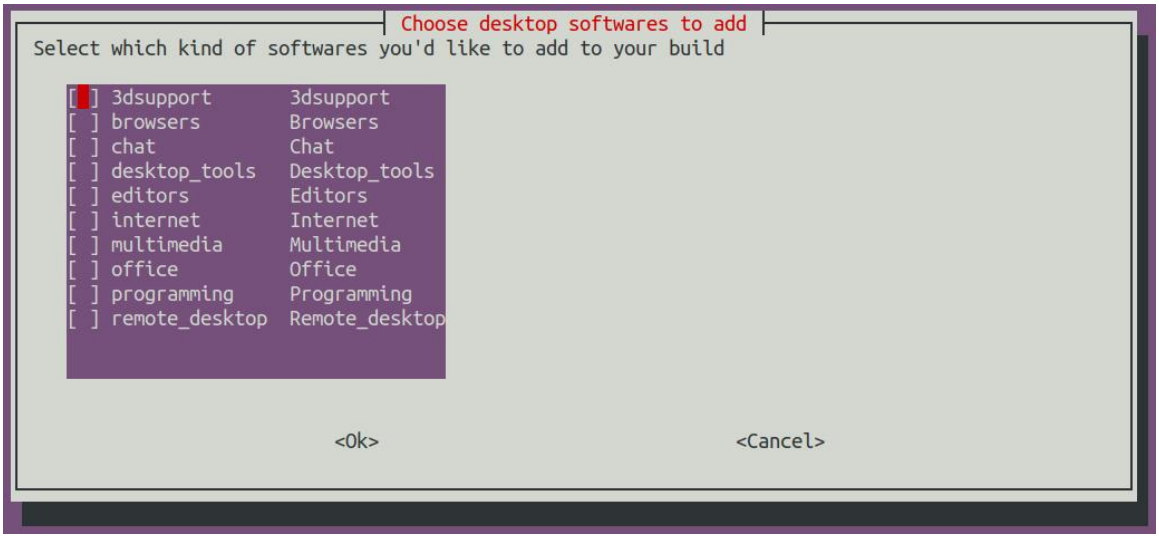


8) If you are compiling a desktop version image, you also need to select the type of desktop environment. Currently, only GNOME is maintained, so please select the GNOME type desktop





You can then select additional packages to install. Please press Enter to skip this step.



9) Then it will start compiling rootfs. Some of the information prompted during compilation is as follows

a. Type of rootfs

[o.k.] local not found [Creating new rootfs cache for **jammy**]

b. Storage path of the compiled rootfs compressed package

[o.k.] Target directory [**orangepi-build/external/cache/rootfs**]

c. The name of the rootfs compressed package generated by compilation

[o.k.] File name [**jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4**]

10) View the compiled rootfs compressed package

a. **jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4** is the compressed package of rootfs. The meaning of each field of the name is

a) **jammy** indicates the type of Linux distribution of rootfs

b) **xfce** indicates that the rootfs is a desktop version. If it is **cli**, it indicates a server version.

c) **arm64** indicates 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. The compilation script will use this MD5 hash value to determine whether the rootfs needs to be recompiled.

b. **jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list** lists the

package names of all software packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4
jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.current
jammy-xfce-arm64.5250ec7002de9e81a41de169f1f89721.tar.lz4.list
```

11) If the required rootfs already exists in **external/cache/rootfs**, then compiling rootfs again will skip the compilation process directly and will not restart the compilation. When compiling the image, it will also check whether there is a cached rootfs in **external/cache/rootfs**. If there is, it will be used directly, which can save a lot of download and compilation time.

4. 6. Compile Linux image

If downloading code from github is slow, you can do it at Add the parameter GITEE_SERVER=yes after ./build.sh, so that it can be downloaded from the gitee website during compilation.

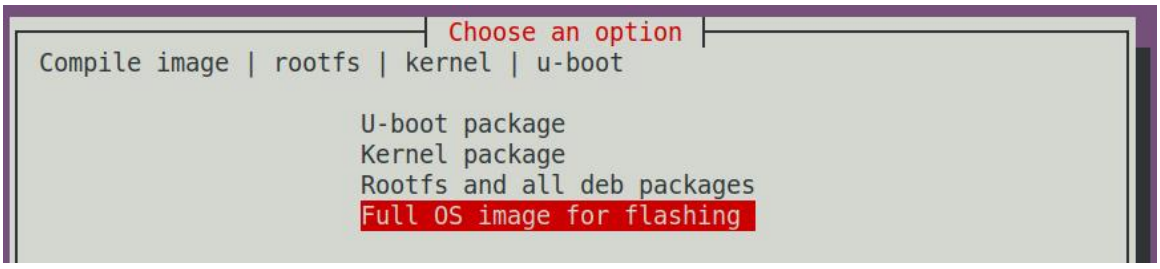
The compilation command for adding the GITEE_SERVER=yes parameter is as follows:

```
test@test:~/orangepi-build$ sudo ./build.sh GITEE_SERVER=yes
```

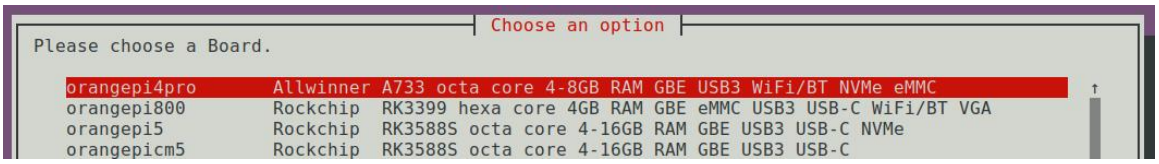
1) Run the **build.sh** script and remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

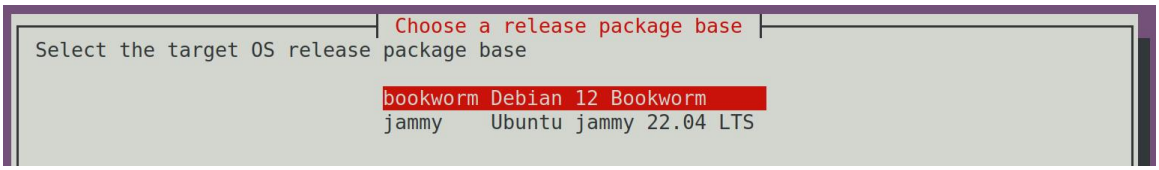
2) Select **Full OS image for flashing** and press Enter



3) Then select the model of the development board



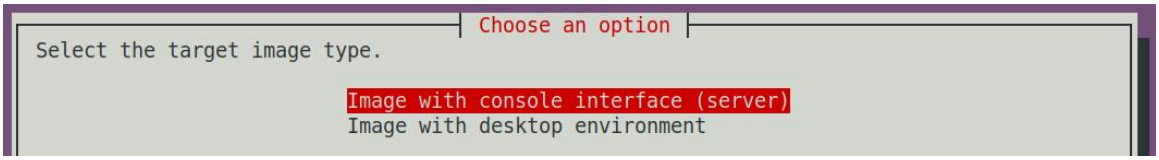
4) Then select the type of rootfs



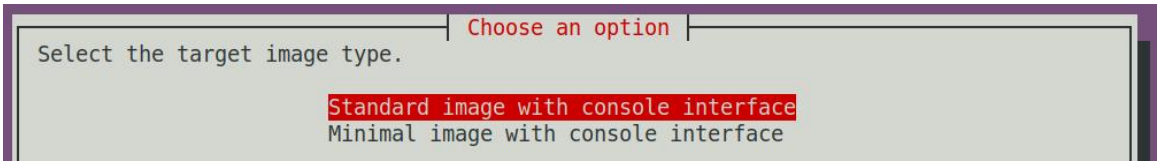
5) Then select the type of image

a. **Image with console interface (server)** Indicates the server version of the image, which is relatively small in size

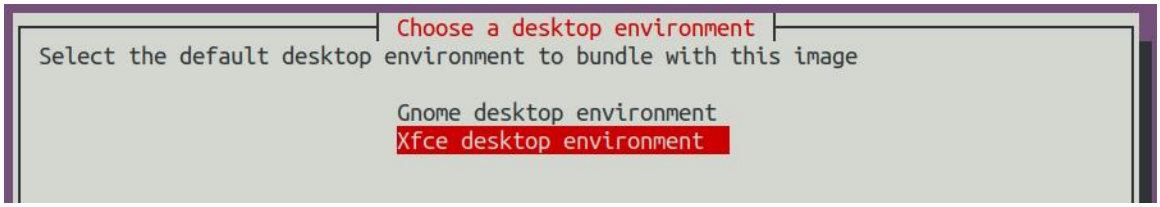
b. **Image with desktop environment** Indicates a mirror image with a desktop, which is relatively large in size.

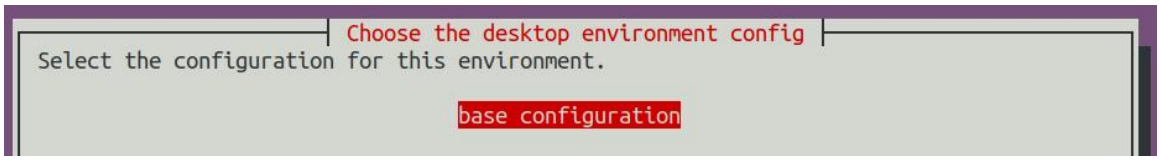


6) If you are compiling a server version image, you can also choose to compile the Standard version or the Minimal version. The Minimal version has much less pre-installed software than the Standard version. (**Please do not choose the Minimal version if you do not have special needs, because many things are not pre-installed by default and some functions may not be available.**)

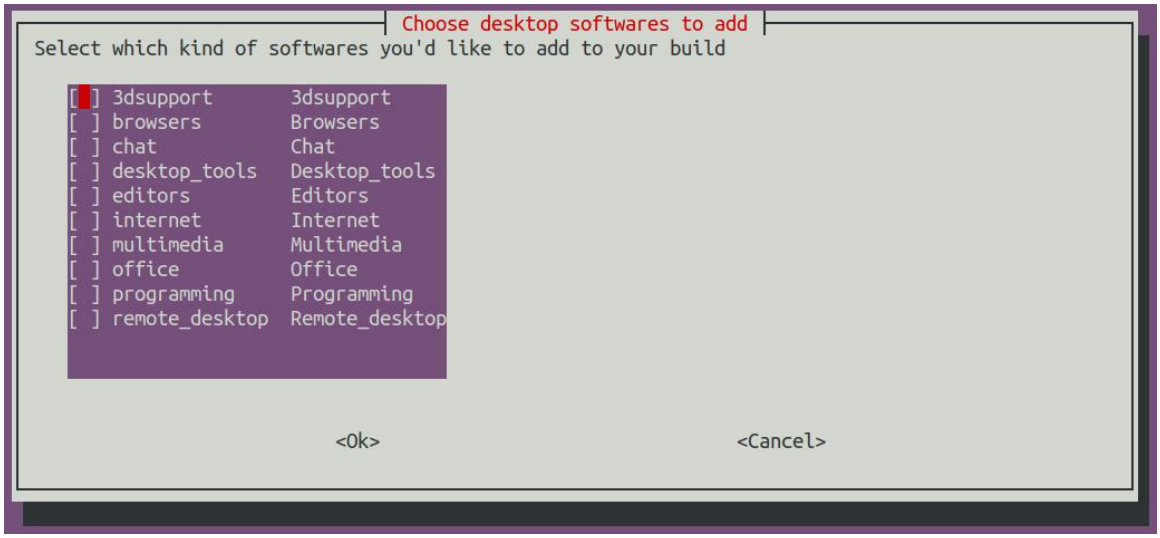


7) If you are compiling a desktop version image, you also need to select the type of desktop environment. Currently, only XFCE is maintained, so please select the XFCE type desktop





You can then select additional packages to install. Please press Enter to skip this step.



8) Then it will start compiling the Linux image. The general process of compilation is as follows

a. Initialize the compilation environment of Ubuntu PC and install the software packages required for the compilation process
b. Download the source code of u-boot and linux kernel (if it is already cached, just update the code)
c. Compile the u-boot source code and generate the u-boot deb package
d. Compile the Linux source code and generate Linux-related deb packages
e. Make a deb package for Linux firmware
f. Make the deb package of orangepi-config tool
g. Make a deb package for board-level support
h. If you compile a desktop version image, you will also create a desktop-related

deb package

i. Check if rootfs is already cached. If not, re-create rootfs. If already cached, decompress and use directly
j. Install the deb package generated previously to 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 create an image file and format the partition. The default type is ext4
m. Then copy the configured rootfs to the mirror partition
n. Then update initramfs
o. Finally, write the u-boot bin file into the image using the dd command

9) After compiling the image, the following information will be prompted

a. Storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/orangepi4pro_x.x.x_debian_jammy_linux5.15.xx_xfce_desktop/orangepi4pro_x.x.x_debian_jammy_linux5.15.xx_xfce_desktop.img ]
```

b. Compilation time

```
[ 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=orangepi4pro BRANCH=current BUILD_OPT=image RELEASE=jammy BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

5. Android 13 system usage instructions

5. 1. Supported Android versions

Android version	Kernel version
Android 13	**linux5.15**

5. 2. Android 13 function adaptation

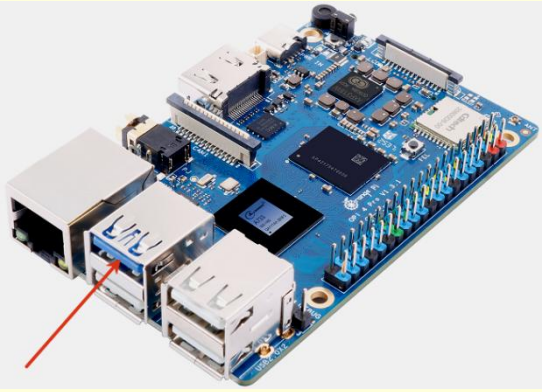
Function	Android 13
HDMI video	**OK**
HDMI Audio	**OK**
USB3.0 x 1	**OK**
USB2.0 x 3	**OK**
TF card boot	**OK**
eMMC	**OK**
NVME SSD identification	**OK**
Gigabit network card	**OK**
WIFI	**OK**
Bluetooth	**OK**

Headphone Audio	**NO**
Onboard MIC	**NO**
trumpet	**NO**
LCD Screen	**OK**
CAM1	**OK**
CAM2	**The screen is distorted when taking photos, but normal when recording videos**
LED light	**OK**
40 pin GPIO	**OK**
40 pin I2C	**OK**
40 pin SPI	**OK**
40 pin UART	**OK**
40 pin PWM	**OK**
Temperature sensor	**OK**
GPU	**OK**
Power button	**OK**

5. 3. How to use ADB

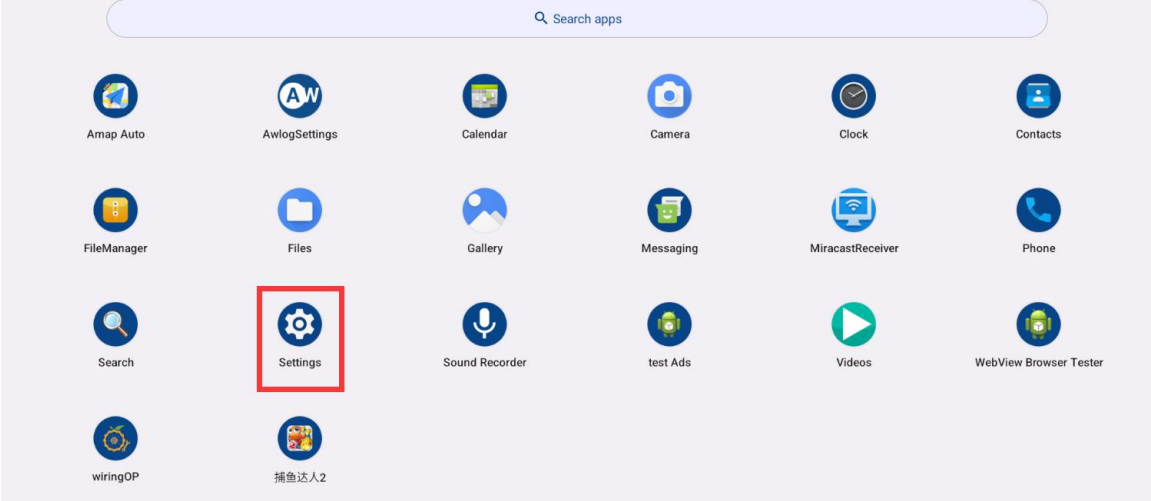
5. 3. 1. USB OTG mode switching method

The development board has 4 USB ports. The blue USB port marked with a red frame in the figure below can support both Host mode and Device mode. The other 3 USB ports only support Host mode.



The USB OTG interface is in Host mode by default and can be used to connect USB devices such as mouse and keyboard. If you want to use ADB, you need to manually switch to Device mode.

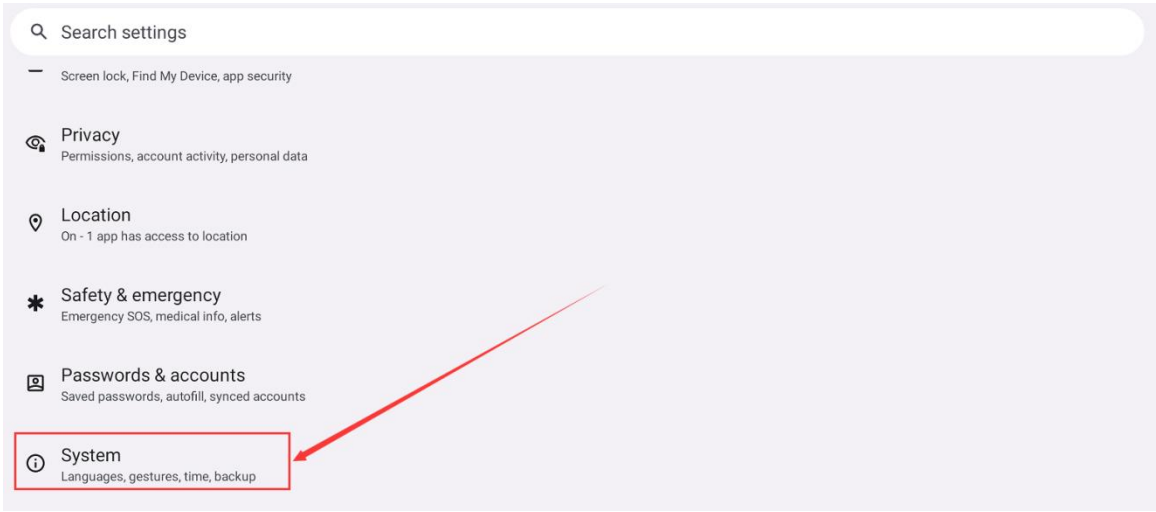
1) First open Settings



2) Then find**About tablet**

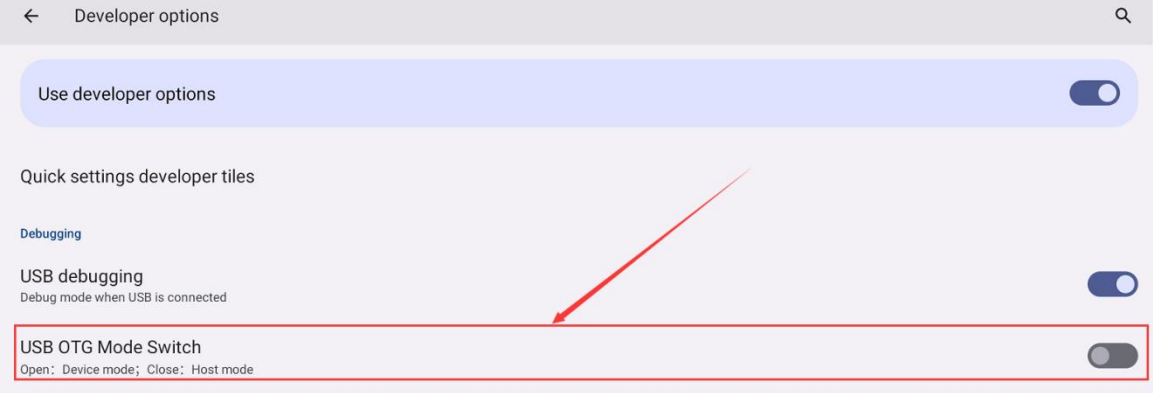
3) Then use your mouse to click the **Build number** option several times until the prompt "**You are now a developer!**" appears.

4) Then return to the previous menu and select **System**



5) Then select**Developer options**

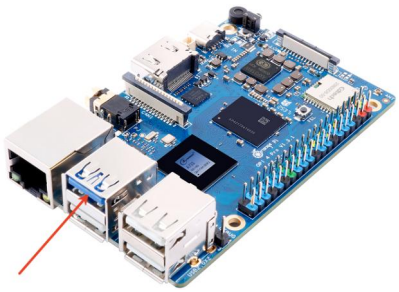
6) Finally, find the **USB OTG Mode Switch, turn it on to switch to Device mode, and turn it off to switch to Host mode.**



5. 3. 2. Use a data cable to connect adb debugging

1) Firstly, prepare a high-quality USB 2.0 male to male data cable

2) Then connect the development board to the Ubuntu PC via a USB 2.0 male to male data cable. The USB interface position of the development board that supports USB Device mode is shown in the following figure:



3) Then switch to USB Device mode by following the method of **switching to USB OTG mode**.

4) Then install adb tools on Ubuntu PC

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

5) Check that the ADB device is identified

```
test@test:~$ adb devices
List of devices attached
4c00146473c28651dd0     device
```

6) Then you can log in to the Android system through adb shell on the Ubuntu PC

```
test@test:~$ adb shell
a733-demo:/ #
```

5. 3. 3. Using adb debugging with network connection

Using network adb does not require a USB2.0 male-to-male data cable to connect the computer and the development board. Instead, communication is done over the network. So first make sure the wired or wireless network of the development board is connected, and then obtain the IP address of the development board, which will be used later.

1) Ensure that the Android system's **service.adb.tcp.port** is set to port 5555

```
console:/ # getprop | grep "adb.tcp"
[service.adb.tcp.port]: [5555]
```

2) If **service.adb.tcp.port** is not set, you can use the following command in the serial port to set the network adb port number.

```
console:/ # setprop service.adb.tcp.port 5555

console:/ # stop adbd
console:/ # start adbd
```

3) Install adb tool on Ubuntu PC

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

4) Then connect to the network adb on the Ubuntu PC

```
test@test:~$ adb connect 192.168.1.xxx:5555      (Need to be changed to the IP address of the development board)
* daemon not running; starting now at tcp:5037
* daemon started successfully
connected to 192.168.1.xxx:5555

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

5) Then you can log in to the Android system through adb shell on the Ubuntu PC

```
test@test:~$ adb shell
a733-demo:/ #
```

5. 4. HDMI to VGA display test

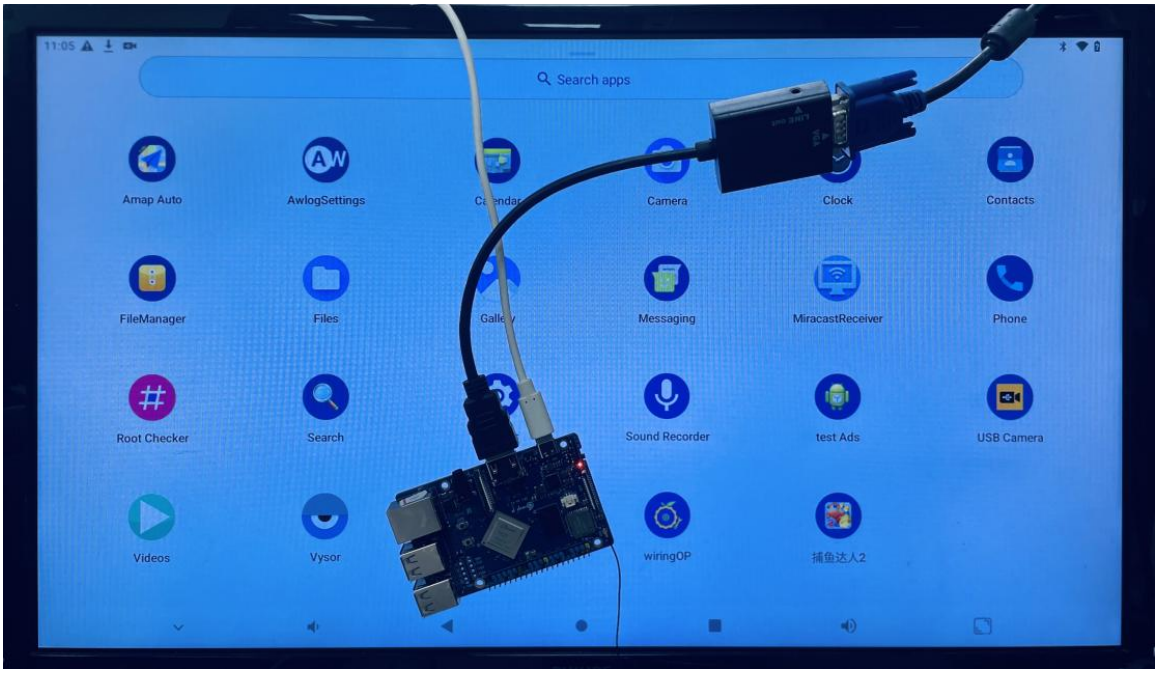
1) First, you need to prepare the following accessories

a. HDMI to VGA converter

b. A VGA cable

c. A monitor or TV that supports VGA interface

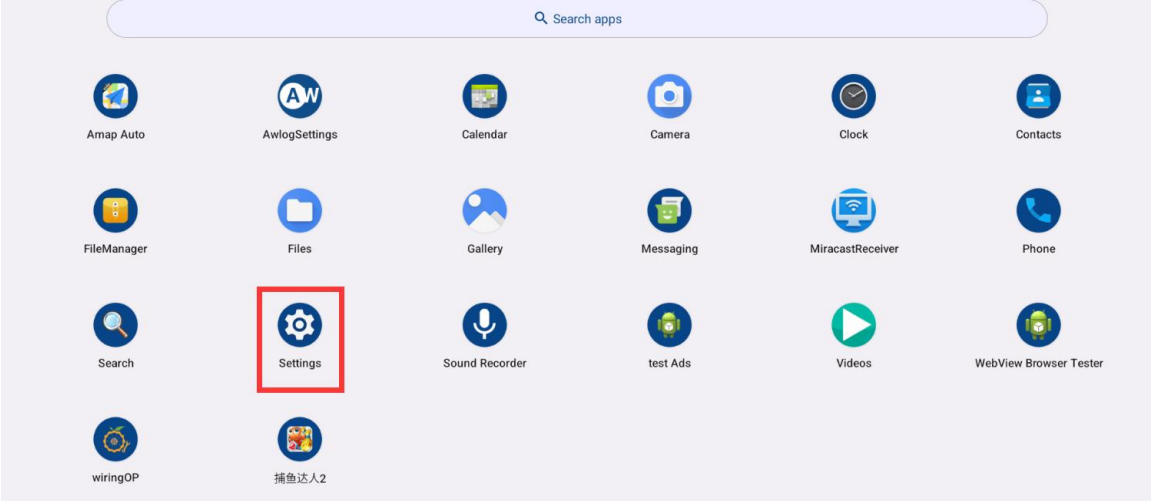
2) HDMI to VGA display test is as follows



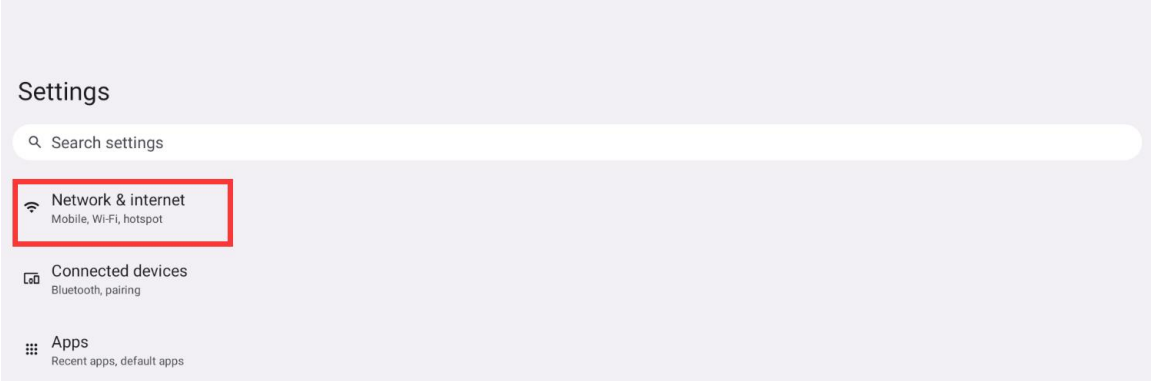
When using HDMI to VGA display, the development board and the Android system on the development board do not require any configuration. It only requires the development board's HDMI interface to display normally. Therefore, if there are any problems with the test, please check the HDMI to VGA converter, VGA cable, and monitor for any problems.

5.5. Wi-Fi connection method

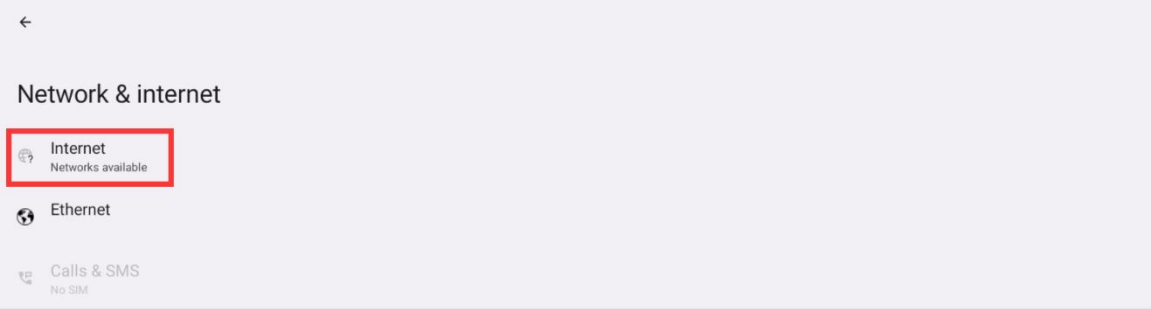
1) First select**Settings**



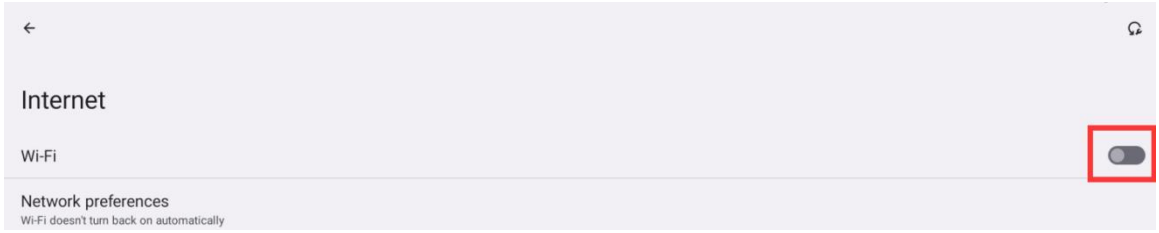
2) First select**Network & Internet**



3) First select**Internet**



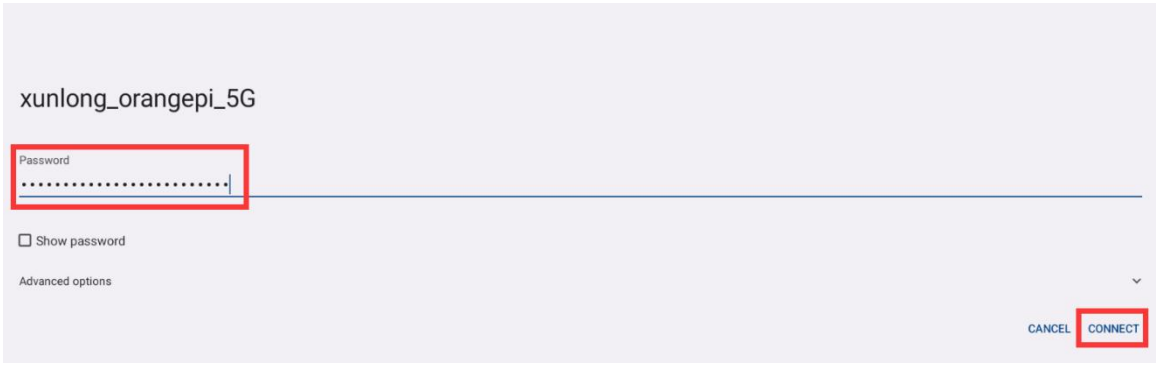
4) Then turn on Wi-Fi



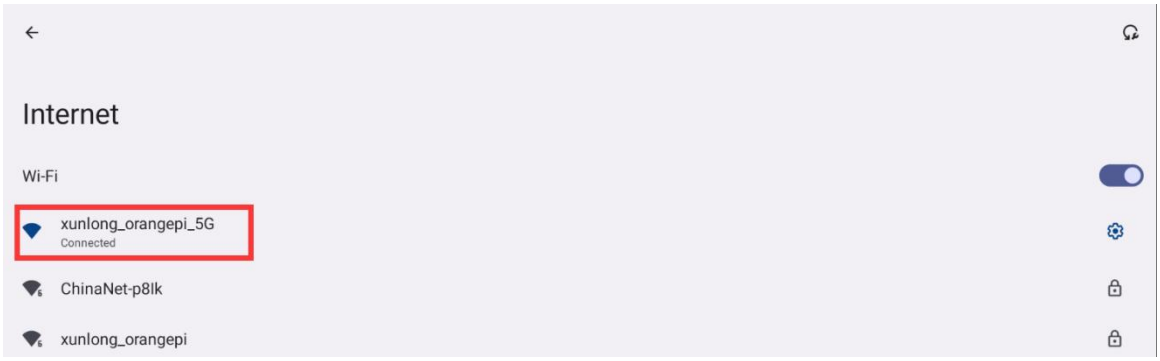
5) After turning on Wi-Fi, you can see the searched signal under **Available networks**

6) After selecting the WI-FI you want to connect to, the password input interface shown below will pop up

7) 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



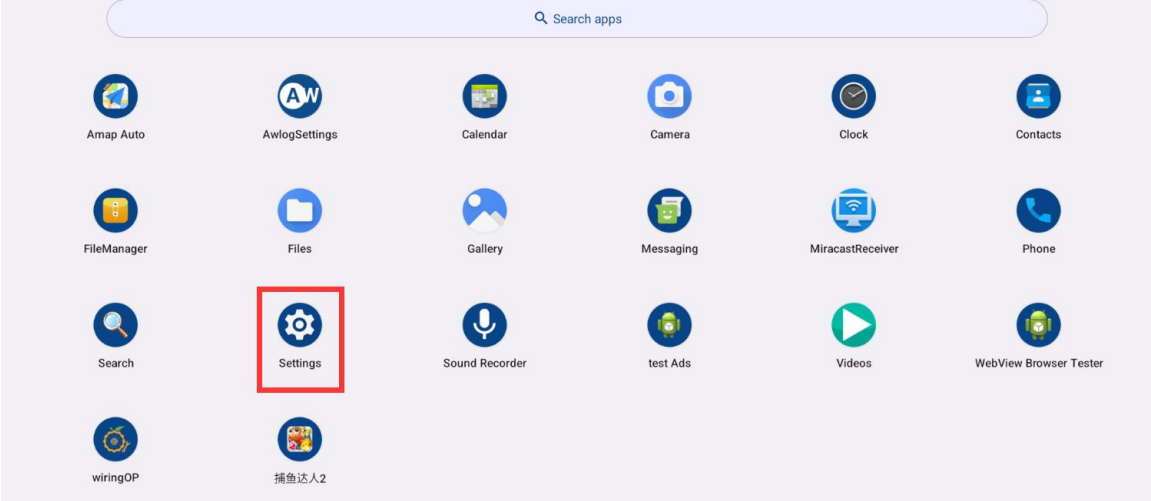
8) The display after successful WI-FI connection is as shown below



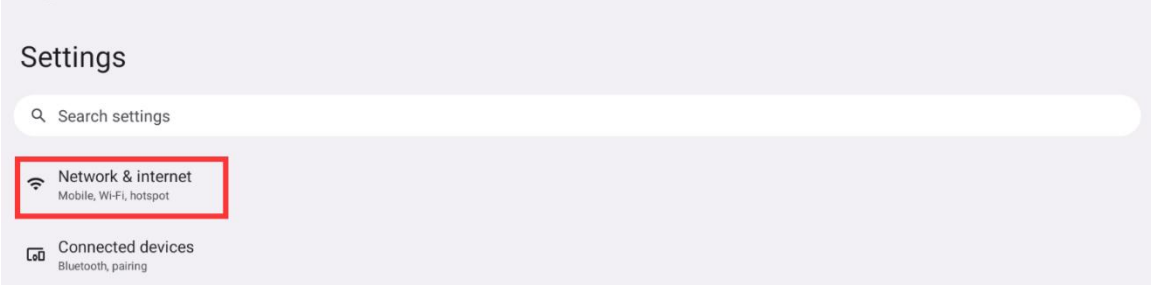
5. 6. How to use the Wi-Fi hotspot

1) First, make sure the Ethernet port is connected to the network cable and can access the Internet normally.

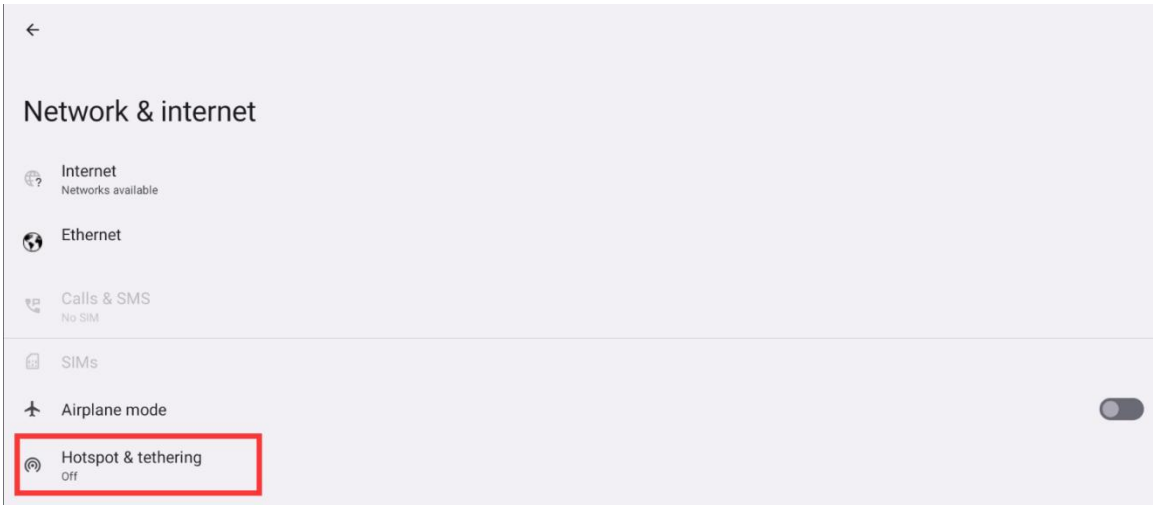
2) Then select **Settings**



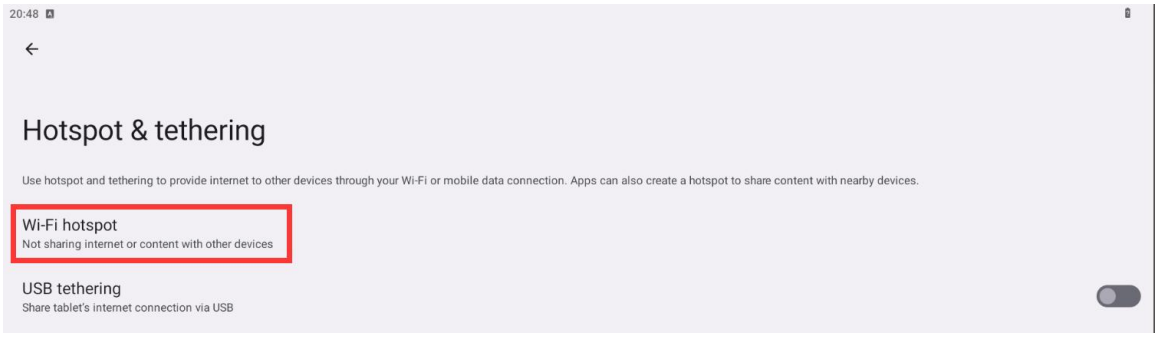
3) Then select **Network & Internet**



4) Then select **Hotspot & tethering**



5) Then select **Wi-Fi hotspot**



6) Then turn on **Wi-Fi Hotspot**. You can also see the name and password of the generated hotspot in the picture below. Remember them and use them when connecting to the hotspot. (If you need to change the hotspot name and password, you need to turn off Wi-Fi Hotspot first.)

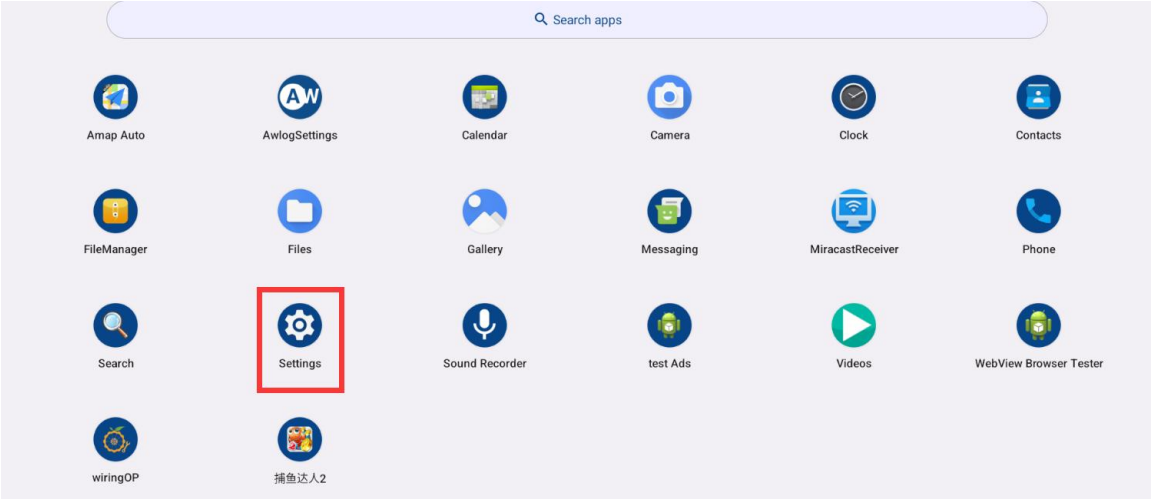
7) Now you can take out your phone. If everything is normal, you can find the WIFI hotspot with the same name (**here it is AndroidAP_4174**) shown under **Hotspot name** in the above picture in the WI-FI list searched by the phone. Then you can click **AndroidAP_4174** to connect to the hotspot. The password can be seen under Hotspot password in the above picture.

8) After the connection is successful, the following figure will be displayed (the interface may be different on different mobile phones, the specific interface is subject to the display of your mobile phone). Now you can open a web page on your mobile phone to see if you can access the Internet. If you can open the web page normally, it means that the **WI-FI Hotspot** of the development board is working properly.

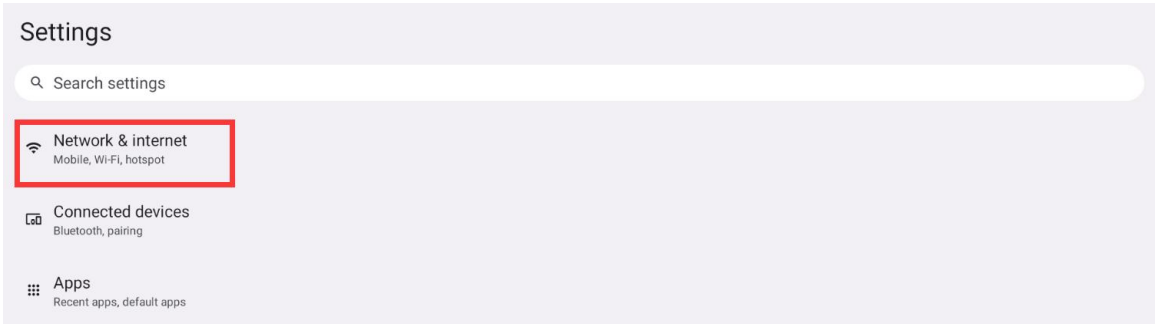
5. 7. How to check the IP address of Ethernet port

1) First, make sure the Gigabit Ethernet port of the development board is connected to a router or switch.

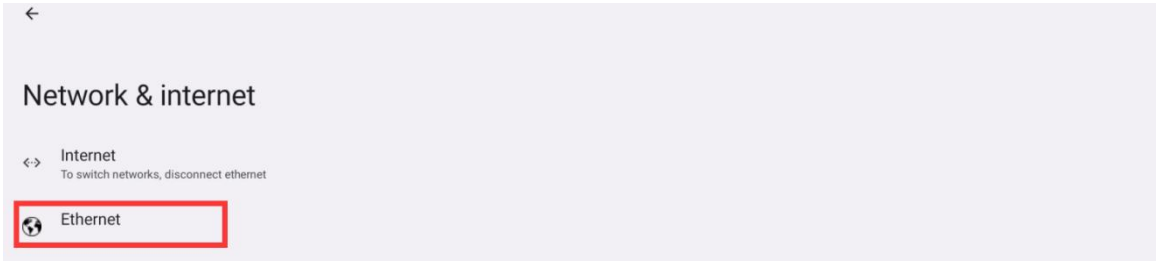
2) Then open **Settings**



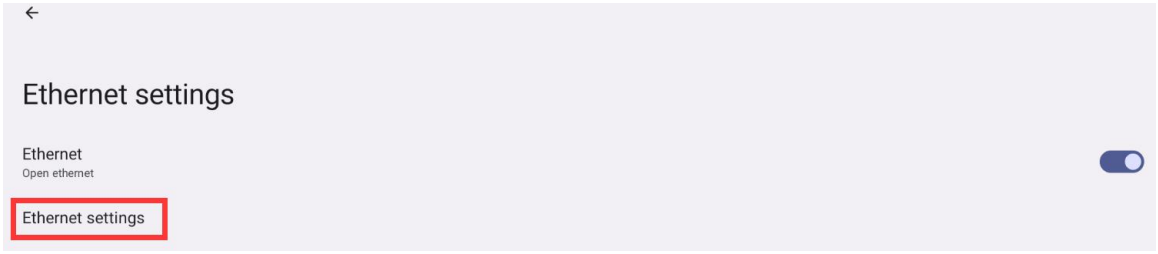
3) Then select **Network & Internet**



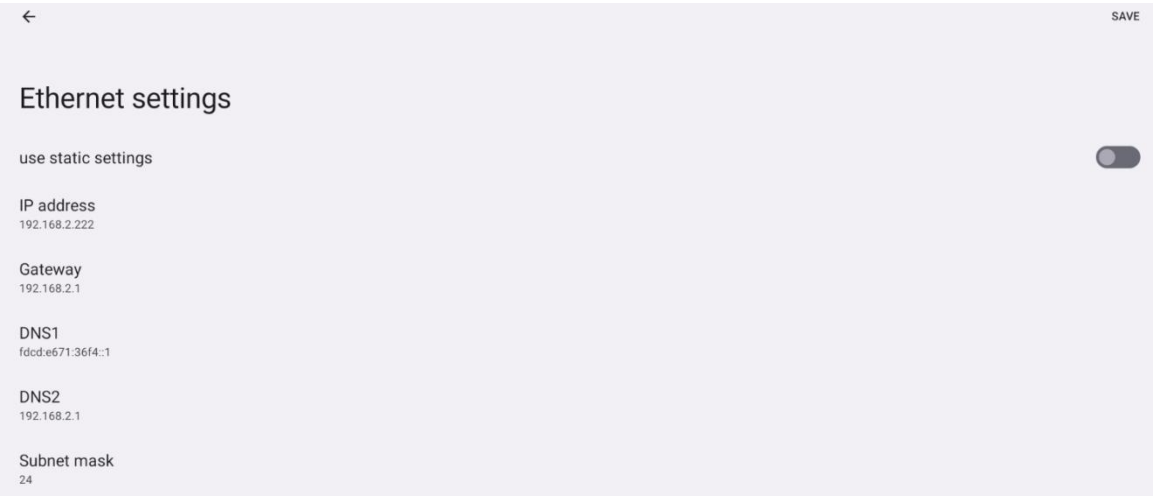
4) Then select **Ethernet**



5) Then select **Ethernet settings**

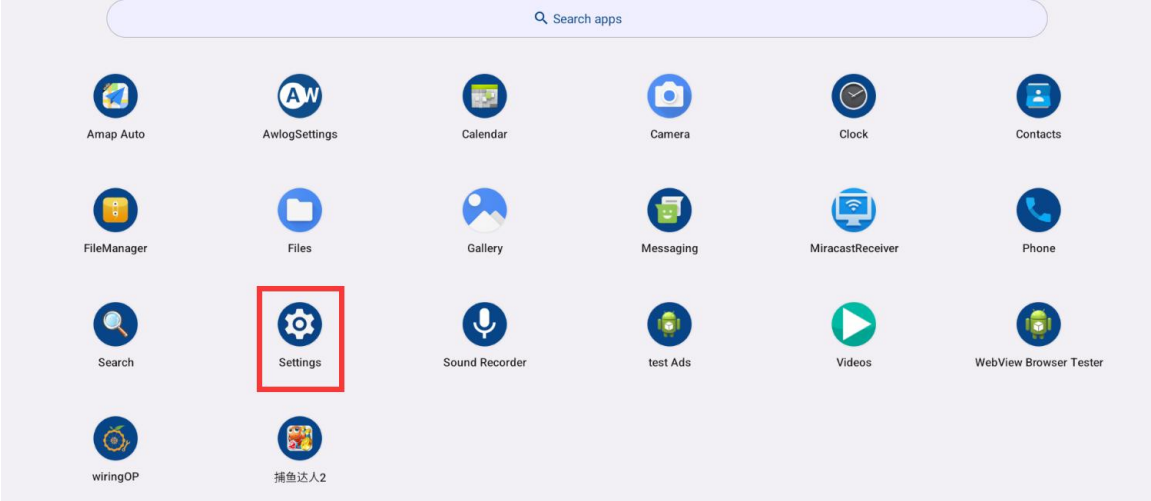


6) Then you can see the IP address information of the development board's wired network port

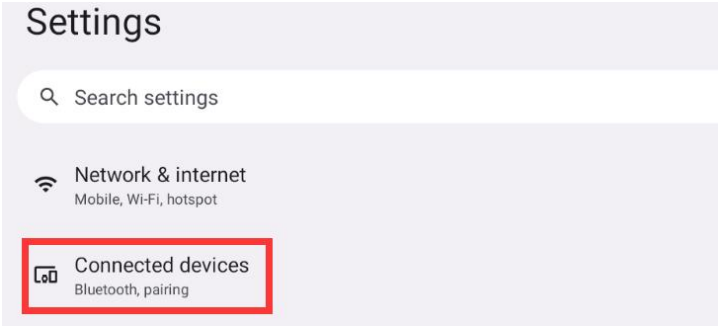


5. 8. Bluetooth connection method

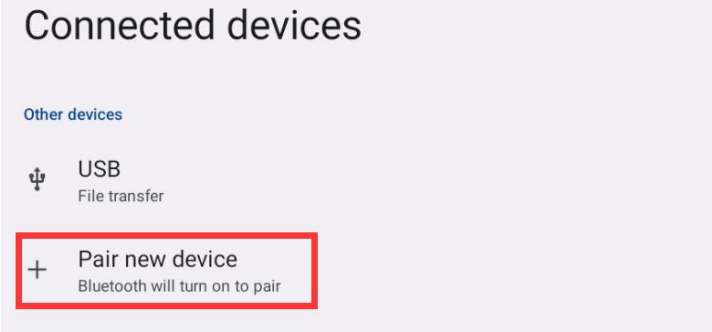
1) First select **Settings**



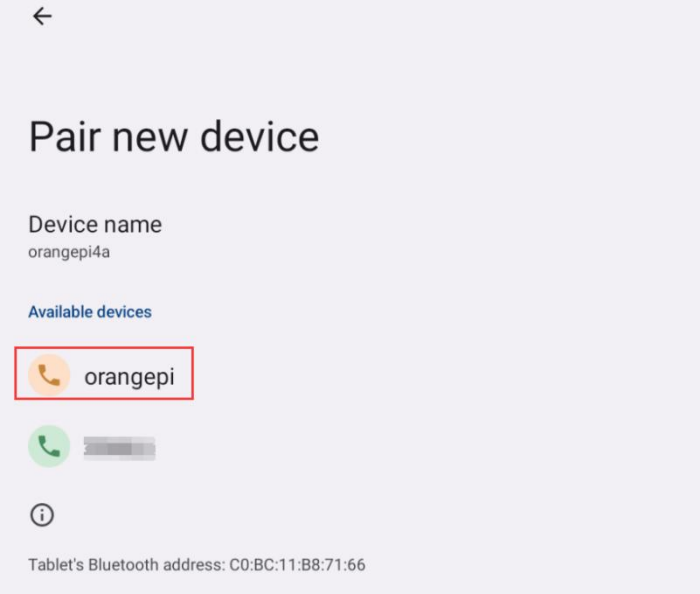
2) First select **Connected devices**



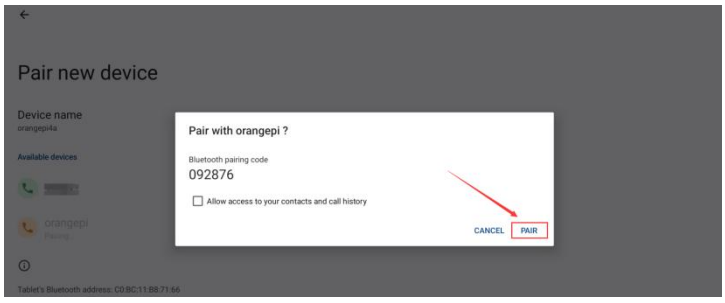
3) Then select **Pair new device** to start scanning for surrounding Bluetooth devices



4) The Bluetooth devices found will be displayed under **Available devices**



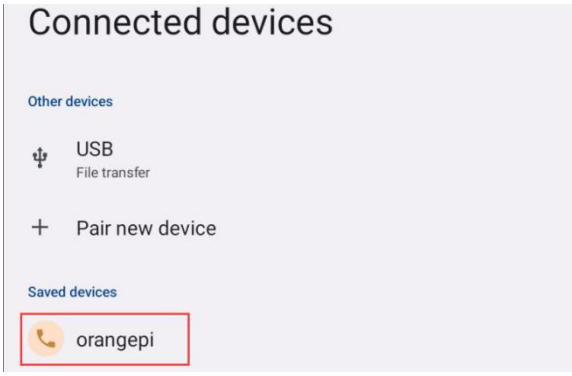
5) Then click on the Bluetooth device you want to connect to start pairing. When the following interface pops up, use the mouse to select the **Pair** option



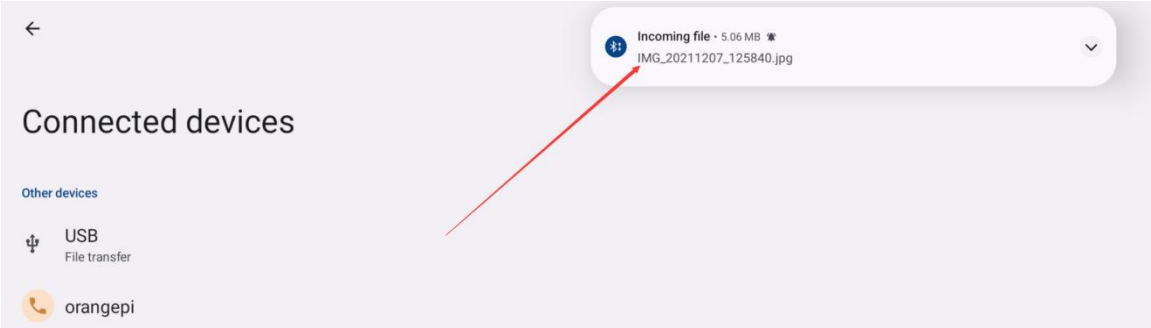
6) The test here is the configuration process of the Bluetooth of the development board

and the **Android phone**. At this time, a confirmation interface will pop up on the phone. Click the pairing button on the phone to start the pairing process.

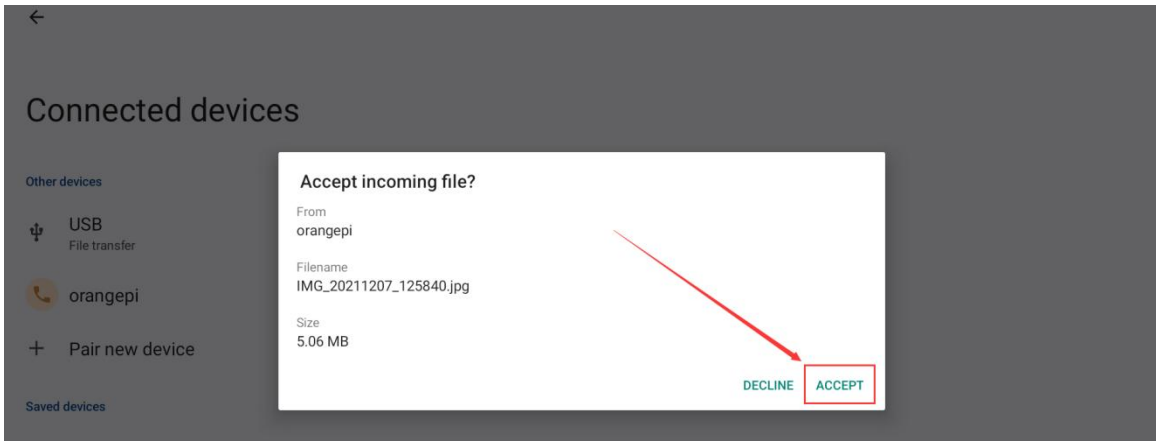
7) After pairing is complete, open **Paired devices** and you can see the paired Bluetooth devices.



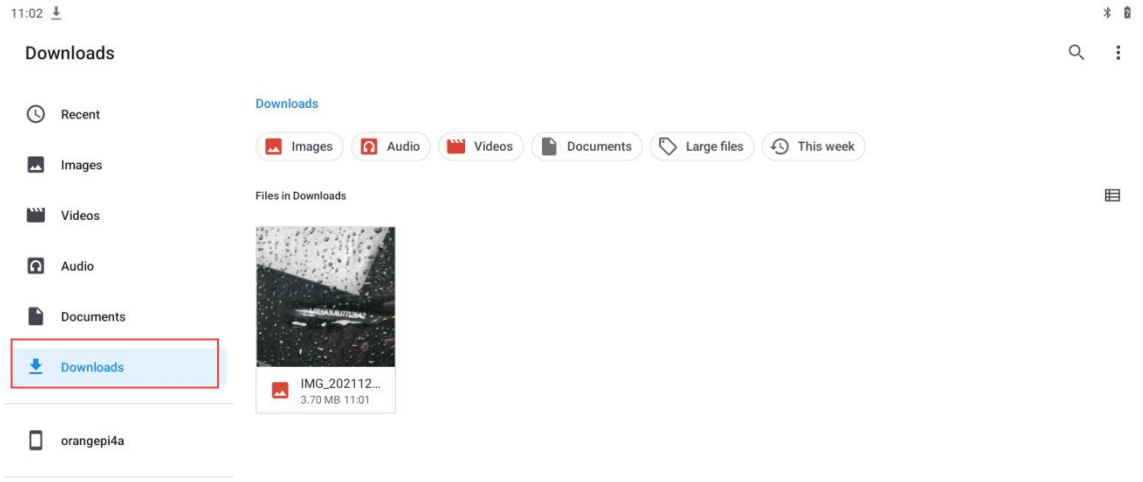
8) You can now use your phone's Bluetooth to send a picture to the development board. After sending, you will see the following prompt in the development board's Android system, then click **Incoming file**



9) Then click **Accept** in the pop-up window to start receiving pictures sent from your phone



10) The pictures received by the Android system Bluetooth on the development board can be viewed by opening the **Download** directory in the file manager

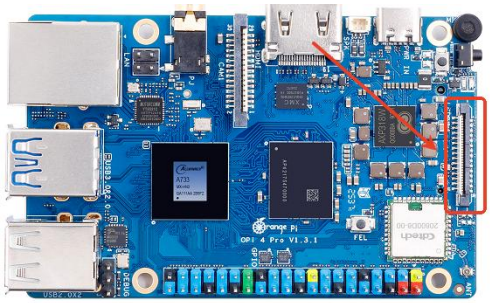


5. 9. How to use 10.1 inch MIPI screen

Please make sure that the Android image you are using is the one with the following version: OrangePi4Pro_A733_Android13_lcd_v1.x.x.img

1) First, you need to assemble the screen. Please refer to the assembly method of the 10.1-inch MIPI screen

2) The location of the MIPI LCD screen interface on the development board is shown in the figure below:



3) Connect the assembled screen to the LCD port of the development board. Unplug the HDMI port. Connect the Type-C power supply to the board and power it on. After the system starts, you can see the screen display as shown below (**the default is vertical screen**).

5. 10. How to use USB camera

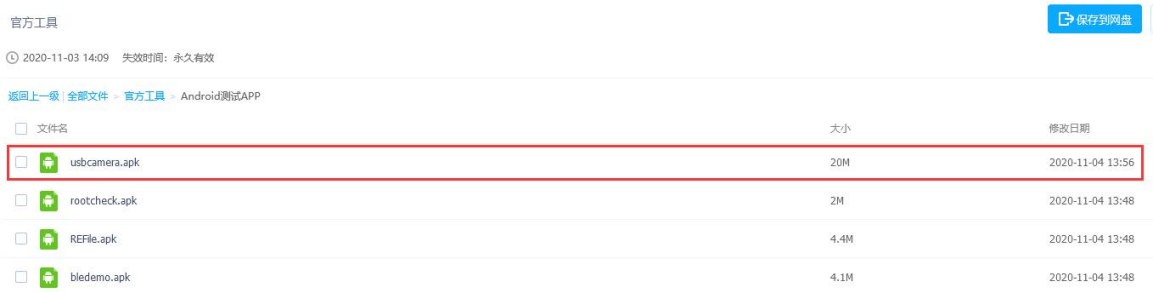
1) First, insert the USB (UVC protocol) camera into the USB port of the development board.

2) If the USB camera is recognized normally, a corresponding video device node will be generated under /dev

```
console:/ # ls /dev/video*
/dev/video0
```

3) Then make sure that the ADB connection between the Ubuntu PC and the development board is normal. For the usage of ADB, please refer to the instructions in the section "**How to use ADB**"

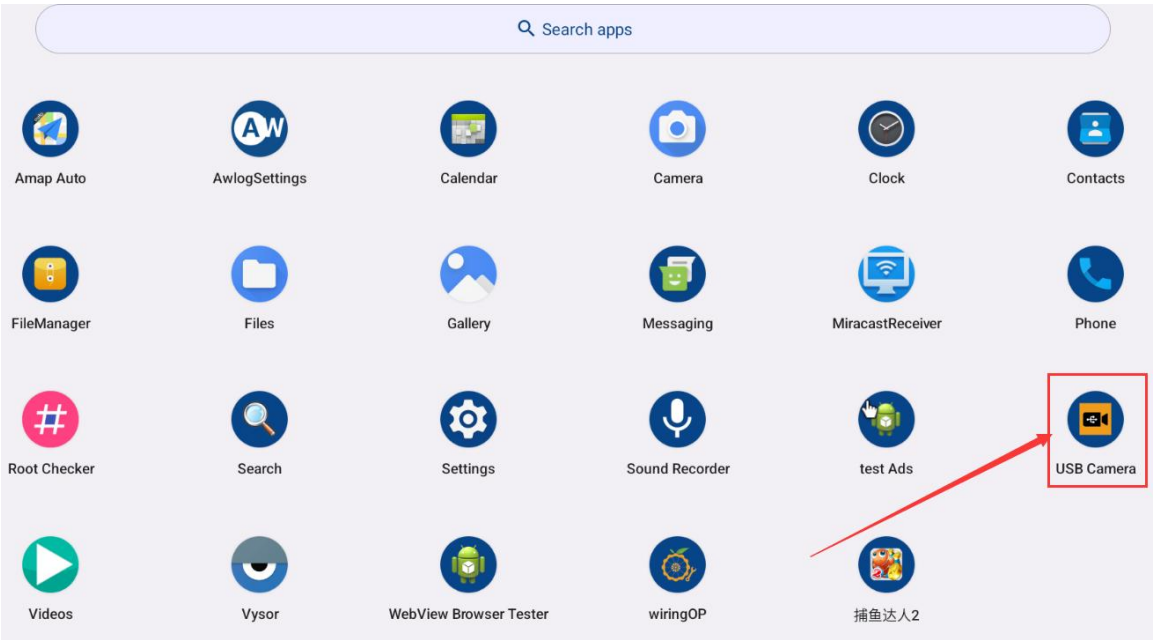
4) Download the USB camera test APP from the **official tool** on the development board download page



5) Then use the adb command to install the USB camera test APP to the Android system. Of course, you can also use a USB flash drive to copy it to install it.

```
test@test:~$ adb install usbcamera.apk
```

6) After installation, you can see the USB camera startup icon on the Android APP interface

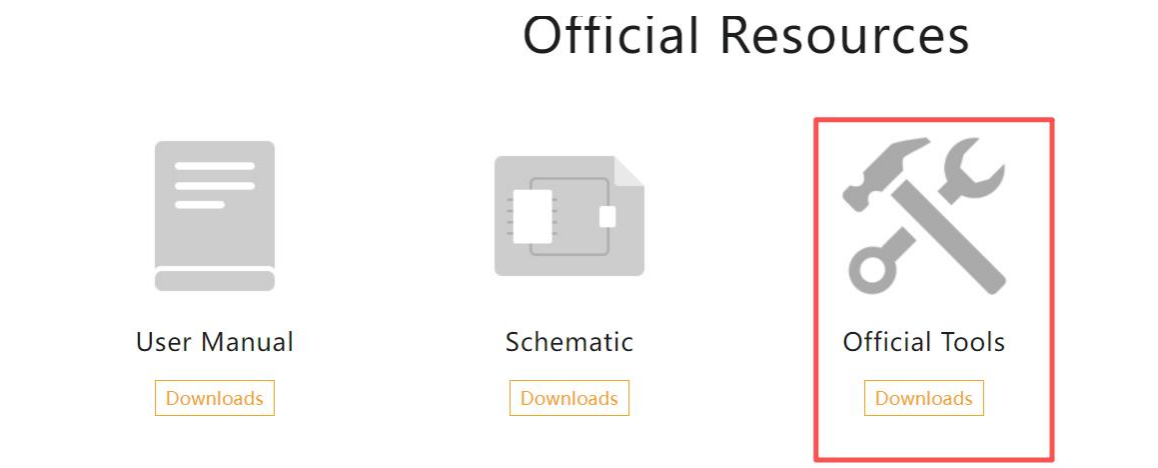


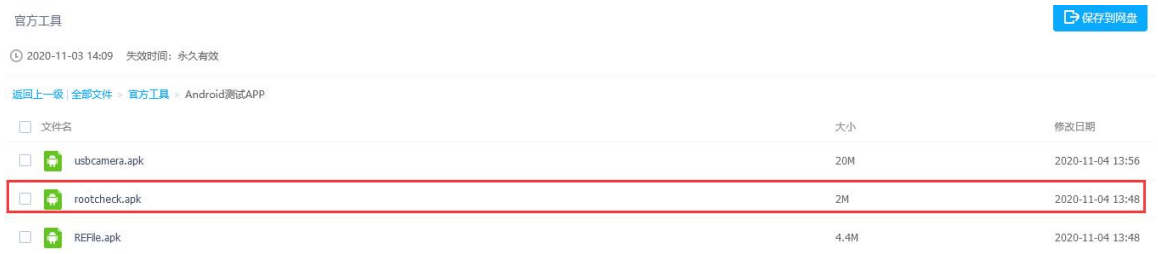
7) Then double-click to open the USB camera APP and you can see the output video of the USB camera

5. 11. Android system rooting instructions

The Android system released by Orange Pi has been rooted and can be tested using the following method.

1) Download **rootcheck.apk** from the **official tool** on the development board download page



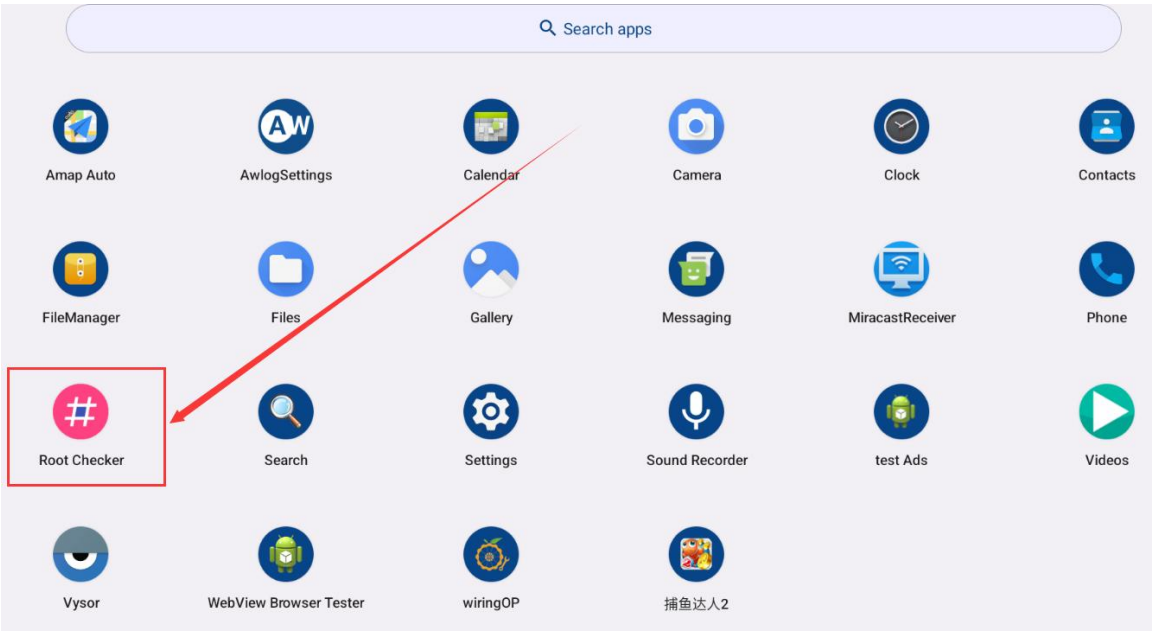


2) Then make sure that the ADB connection between the Ubuntu PC and the development board is normal. For the usage of ADB, please refer to the instructions in the section "**How to use ADB**"

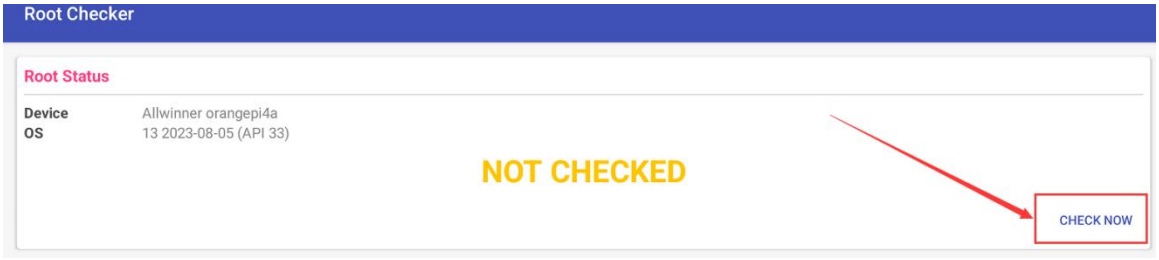
3) Then use the adb command to install rootcheck.apk to the Android system. Of course, you can also use a USB flash drive to copy and install it

```
test@test:~$ adb install rootcheck.apk
```

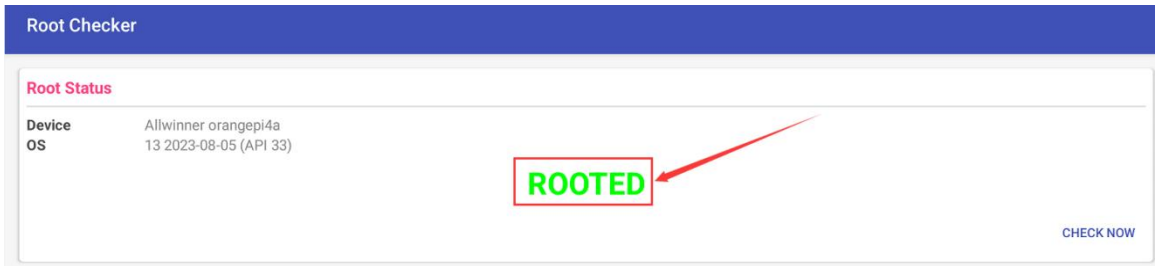
4) After installation, you can see the startup icon of the ROOT test tool on the Android APP interface



5) The display interface after opening the **ROOT test tool** for the first time is as shown below



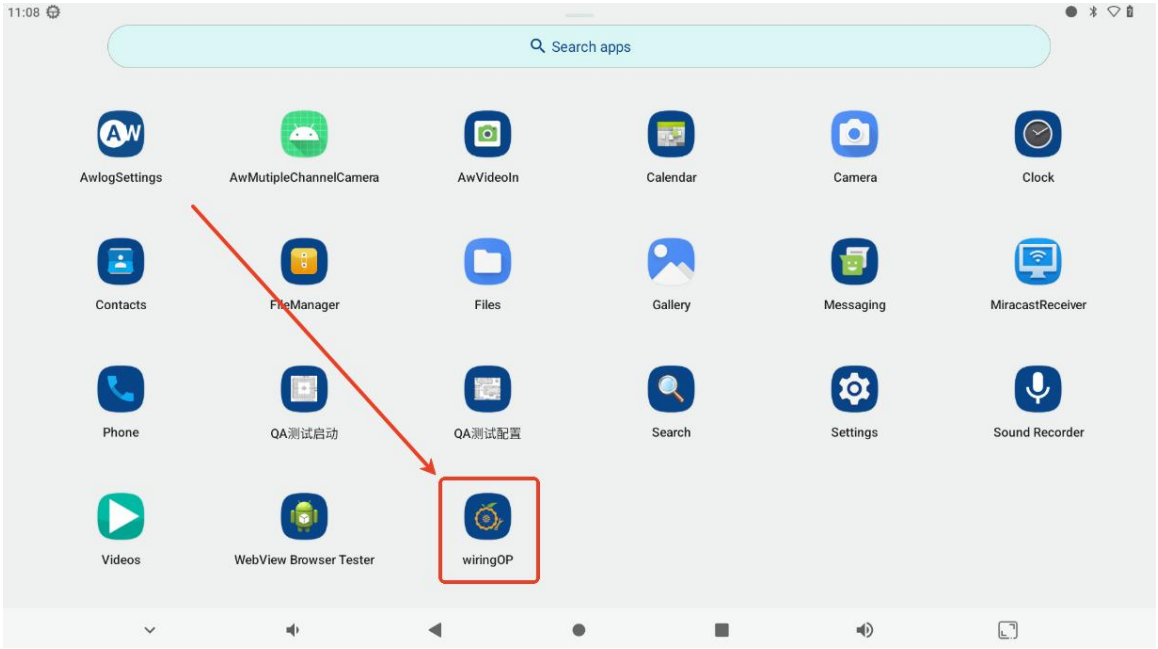
6) Then you can click **CHECK NOW** to start checking the ROOT status of the Android system. The display after the check is as follows, you can see that the Android system has obtained ROOT permissions



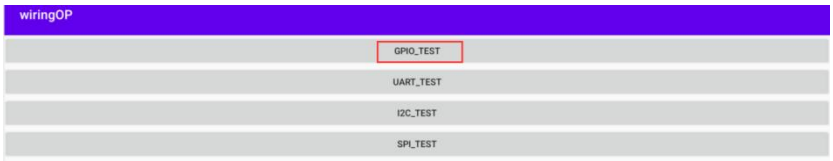
5. 12. 40 pin interface GPIO, UART, SPI test

5. 12. 1. 40 Pin GPIO Port Test Method

1) First open the wiringOP APP on your desktop



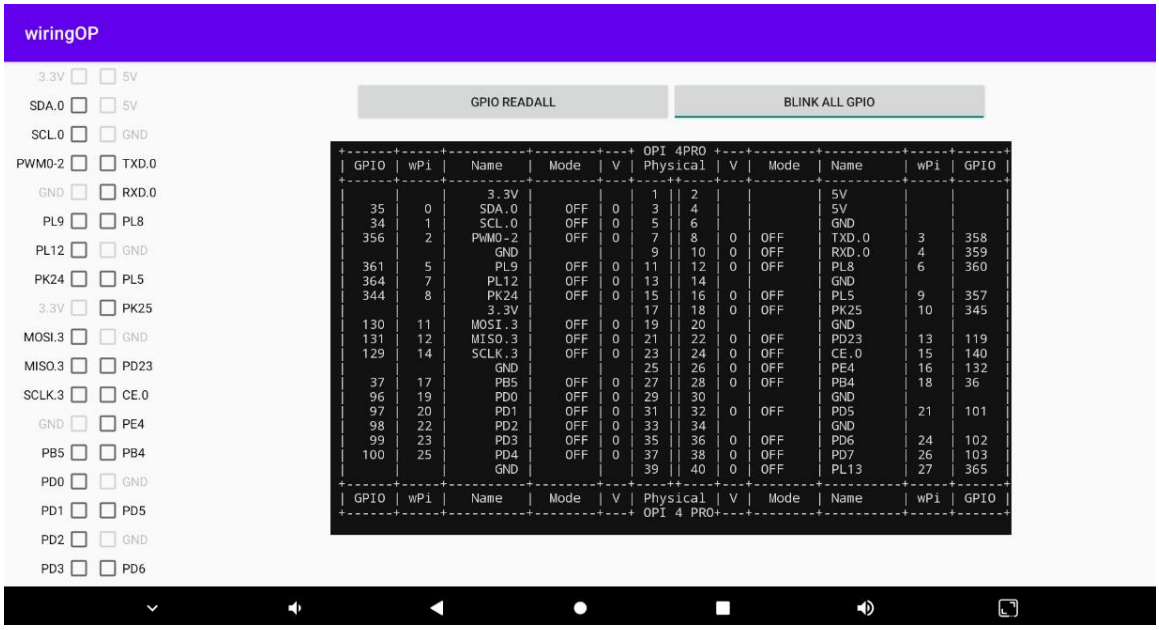
2) Then click the **GPIO_TEST** button to open the GPIO test interface



3) The GPIO test interface is shown in the figure below. The two rows of **CheckBox** buttons on the left correspond to the 40 pins. When the **CheckBox** button is checked, the corresponding GPIO pin will be set to **OUT** mode and the pin level will be set to high. When it is unchecked, the GPIO pin level will be set to low. When the **GPIO READALL** button on the right is clicked, the wPi number, GPIO mode, pin level and other information can be obtained. When the **BLINK ALL GPIO** button is clicked, all pins will continuously switch between high and low levels.

4) Then click the **GPIO READALL** button and the output information is as shown below：

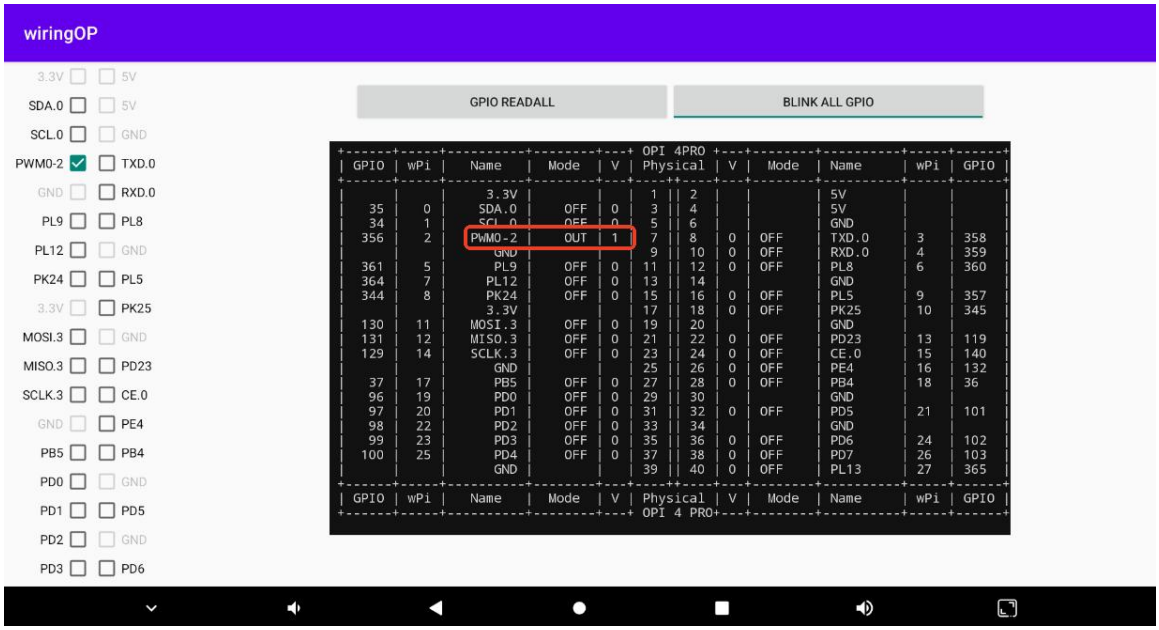
1



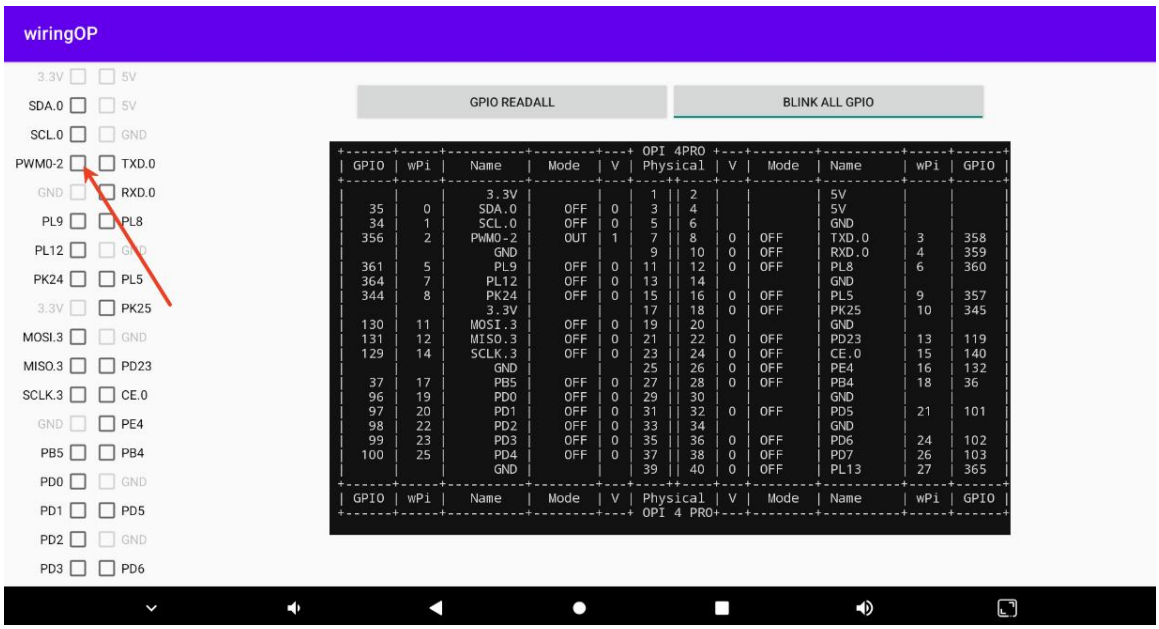
5) There are a total of 28 GPIO ports available in the development board's 40 pins. The following example uses pin 7, which corresponds to GPIO PB4 and wPi number 2, as an example to demonstrate how to set the high and low levels of the GPIO ports. First, click the **CheckBox** button corresponding to pin 7. When the button is selected, pin 7 will be set to a high level. After setting, you can use a multimeter to measure the pin voltage. If it

is **3.3v**, it means the high level setting is successful.

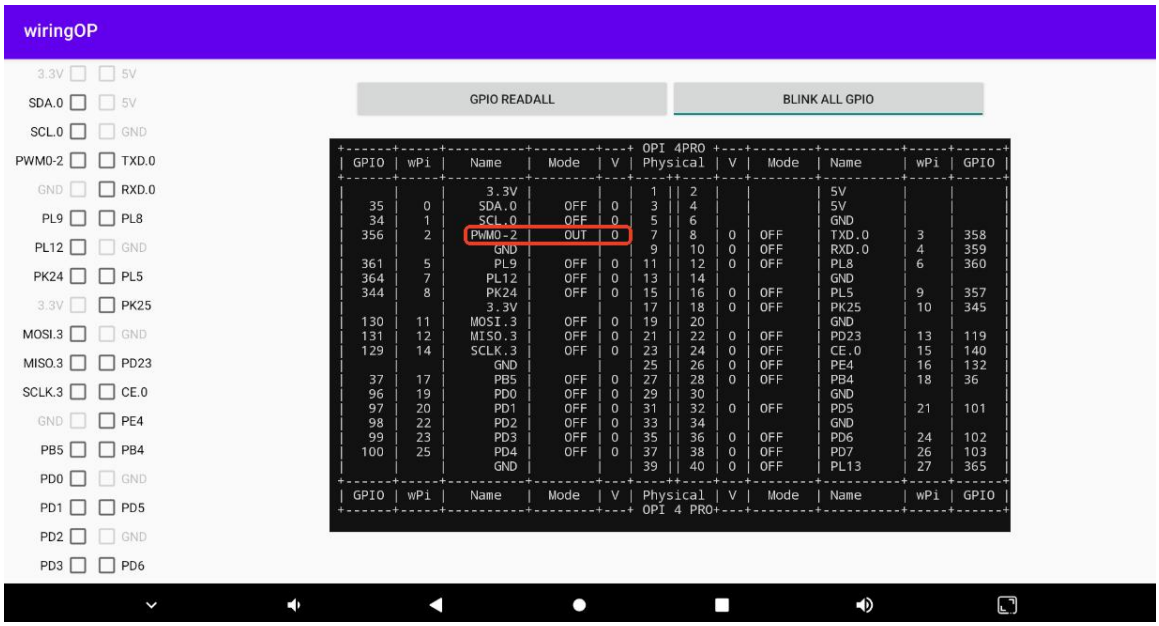
6) Then click the **GPIO READALL** button, you can see that the current pin 7 mode is **OUT** and the pin level is high



7) Click the **CheckBox** button in the figure below again to uncheck the status. Pin 7 will be set to low level. After setting, you can use a multimeter to measure the voltage value of the pin. If it is **0v**, it means that the low level is set successfully.



8) Then click the **GPIO READALL** button, you can see that the current pin 7 mode is OUT and the pin level is low

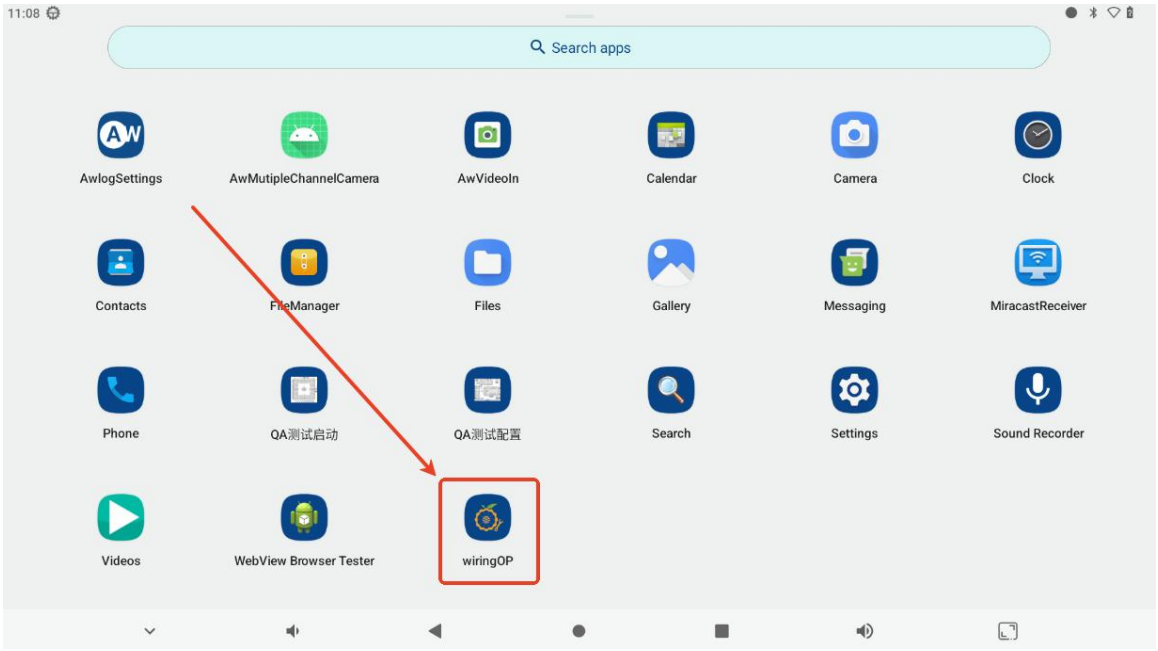


5. 12. 2. 40 Pin UART Test Method

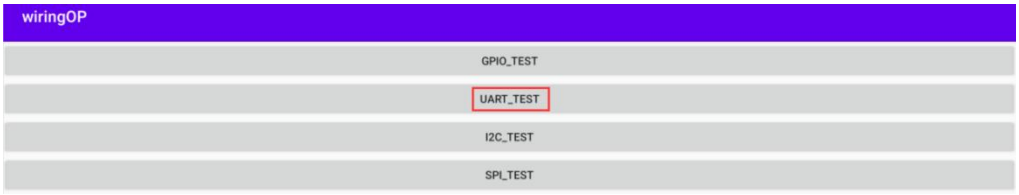
1) **UART7**and **UART8** are enabled by default in Android, and the corresponding device nodes are **/dev/ttyAS7** and **/dev/ttyAS8**

```
a733-demo:/ $ ls /dev/ttyAS*
/dev/ttyAS0  /dev/ttyAS1  /dev/ttyAS6  /dev/ttyAS7  /dev/ttyAS8
```

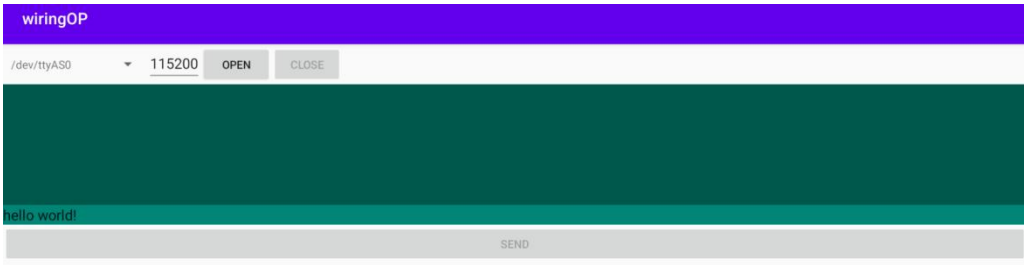
2) First open the wiringOP APP on your desktop



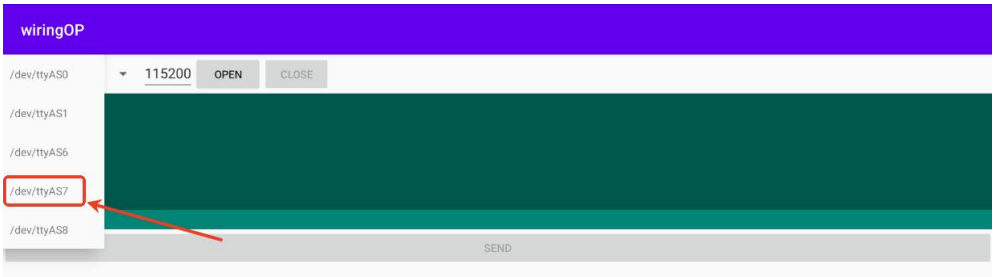
3) Then click the **UART_TEST** button to open the UART test interface



4) The serial port test interface of wiringOP is shown in the figure below



5) Take UART7 as an example, select **/dev/ttyAS7** node in the selection box

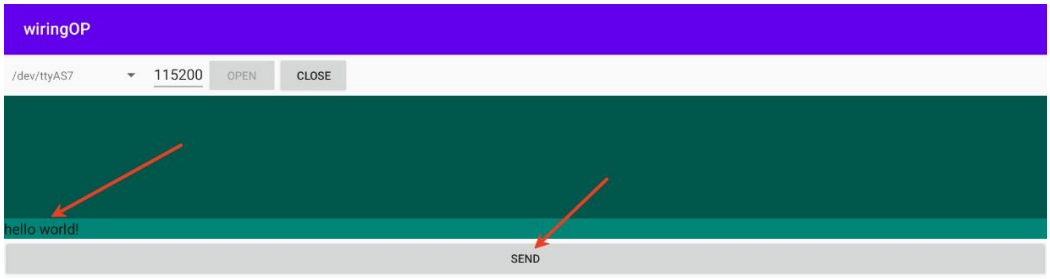


6) Enter the baud rate you want to set in the edit box, and then click the **OPEN** button to open the **/dev/ttyAS7** node. After opening successfully, the **OPEN** button becomes unselectable, and the **CLOSE** button and **SEND** button become selectable.

7) Then use Dupont wire to short the rx and tx pins of uart7

	uart7	uart8
tx pin	Corresponding to pin 8 of 40 pin	Corresponding to pin 12 of 40 pin
rx pin	Corresponding to pin 10 of 40 pin	Corresponding to pin 11 of 40 pin

8) Then you can enter a string of characters in the send edit box below and click the **SEND** button to start sending.

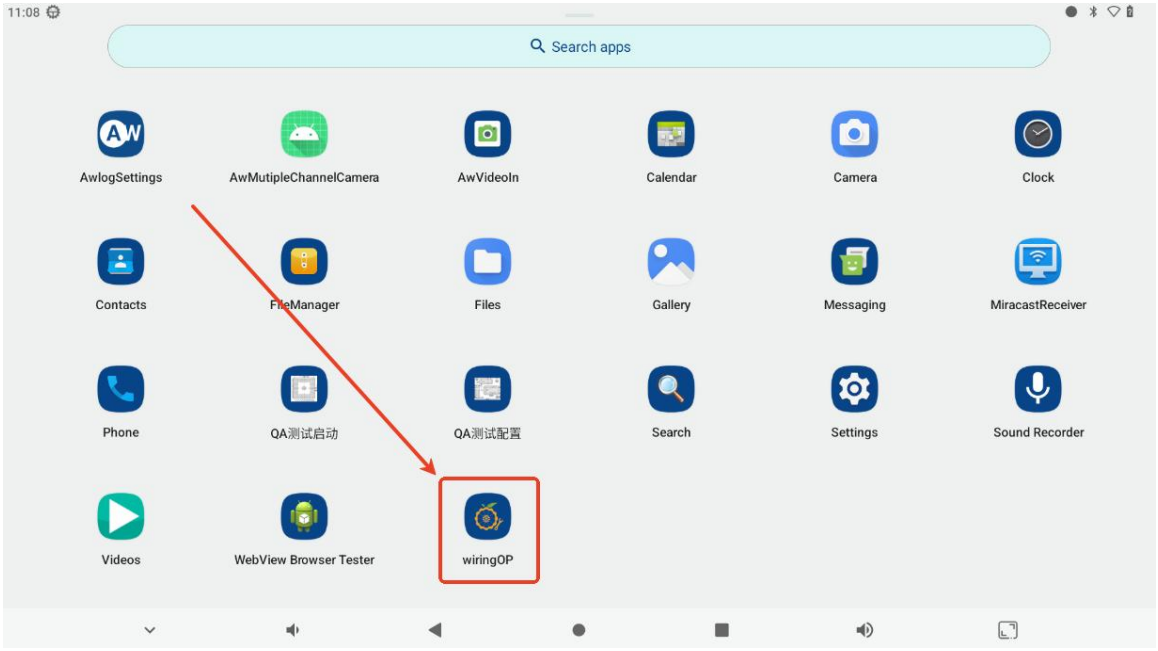


9) If everything is normal, the received string will be displayed in the receiving box

5. 12. 3. 40 pin SPI test method

1) The SPI that can be used in 40 pins is SPI3, and the corresponding device node is

/dev/spidev3.0

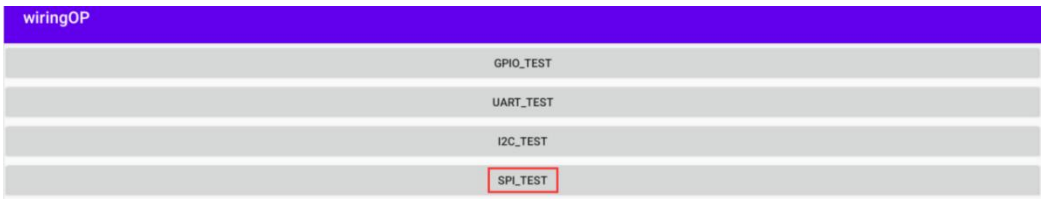


2) Here we demonstrate how to test the SP3 interface through the **w25q64** module. First, connect the w25q64 module to the SPI3 interface.

It doesn't matter if you don't have the w25q64 module, because the development board has a SPIFlash connected to SPI0. The SPI0 configuration is also turned on by default in Android, so we can also use the onboard SPIFlash for testing.

3) Then open the wiringOP APP on the desktop

4) Then click the **SPI_TEST** button to open the SPI test interface

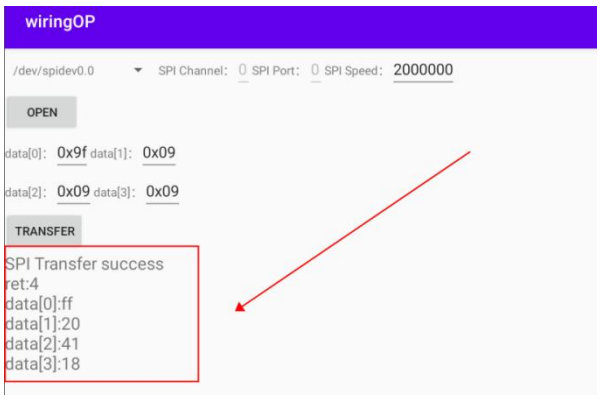


5) Then select the SPI device node in the upper left corner. If you are testing the onboard SPIFlash directly, keep the default **/dev/spidev0.0**. If you are connecting the **w25q64** module to the 40-pin SPI3, then select **/dev/spidev3.0**

6) Then click the **OPEN** button to initialize SPI

7) Then fill in the bytes to be sent, such as reading the ID information of the onboard SPIFlash, fill in the address 0x9f in data[0], and then click the **TRANSFER** button

8) Finally, the APP will display the ID information of the onboard SPI Flash.



9) If you are reading the w25q64 module connected to the 40-pin SPI3, the ID information read is as shown below

10) The MANUFACTURER ID of the w25q64 module is EFh, and the Device ID is 4017h, which corresponds to the values read above (h represents hexadecimal)

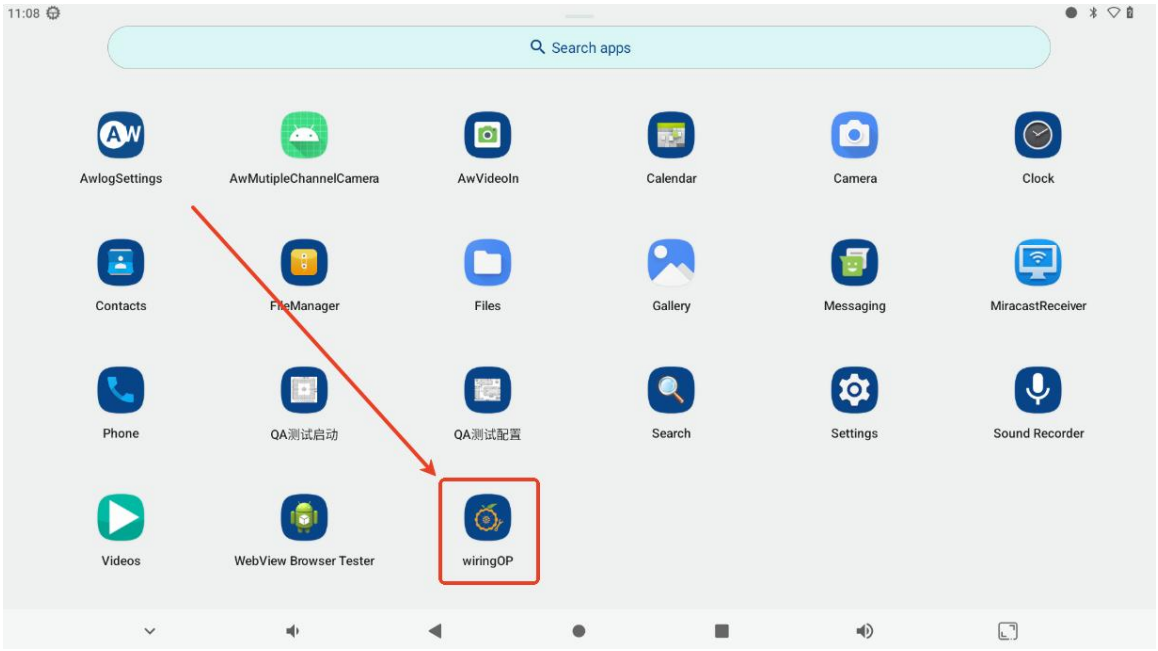
MANUFACTURER ID	**(MF7 - MF0)**	
Winbond Serial Flash	EFh	
Device ID	**(ID7 - ID0)**	**(ID15 - ID0)**
Instruction	**ABh, 90h, 92h, 94h**	**9Fh**
W25Q64FV (SPI)	16h	4017h
W25Q64FV (QPI)	16h	6017h

5. 12. 4. 40 Pin I2C Test Method

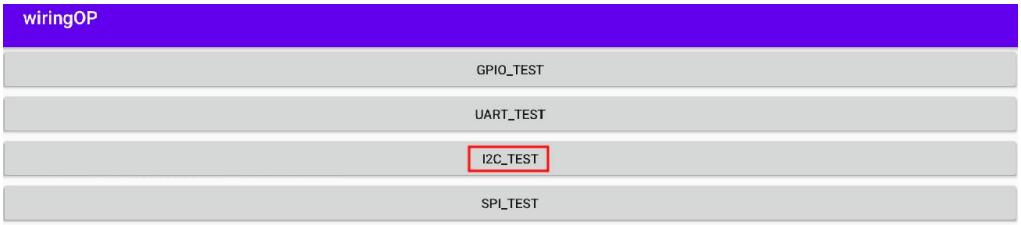
1) In Android, i2c0 and i2c1 are enabled by default in 40 pins, and the corresponding device nodes are **/dev/i2c-0** and **/dev/i2c-1** respectively.

```
console:/ # ls /dev/i2c-*
/dev/i2c-0    /dev/i2c-1
```

2) First open the wiringOP APP on your desktop

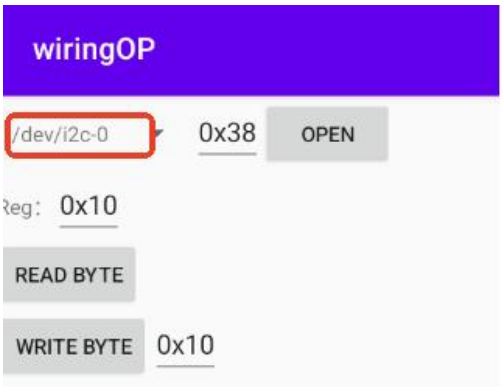


3) Then click the **I2C_TEST** button to open the i2c test interface

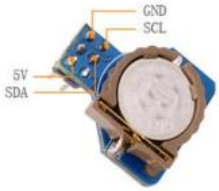


4) The i2c test interface of wiringOP is shown in the figure below

5) Take i2c0 as an example, the default node **in the selection box is /dev/i2c-0**



6) Then connect an i2c device to the i2c0 pin of pin 40. Here we take the ds1307 rtc module as an example.



RTC module pins	Development board 40 pin corresponding pin
5V	**Pin 2**
GND	**Pin 6**
SDA	**Pin 3**
SCL	**Pin 5**

7) The I2C address of the DS1307 RTC module is 0x68. After connecting the cables, we can use the **i2cdetect -y -r 0** command in the serial port command line to check whether the I2C address of the DS1307 RTC module can be scanned. As shown in the figure below, if the address 0x68 is seen, it means that the DS1307 RTC module is wired correctly.

```
console:/ # i2cdetect -y 0
```

```
     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: -- -- -- -- -- -- -- --
t527-demo:/ #
```

8) Then set the i2c address to 0x68 in wiringOP, and then click the **OPEN** button to open i2c0

9) Click the **OPEN** button to open i2c0 and the display is as follows：

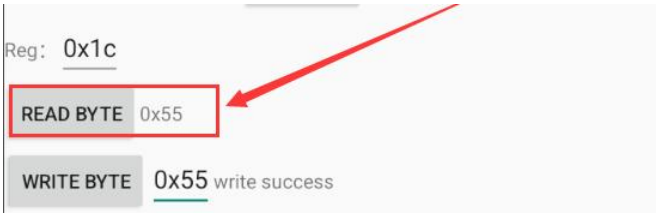
10) Then we test and write a value to the register of the RTC module, for example, write 0x55 to the 0x1c address

a. We first set the address of the register to be written to 0x1c

b. Then set the value to be written to 0x55

c. Then click the **WRITE BYTE** button to execute the write action

11) Then click the **READ BYTE** button to read the value of the 0x1c register. If it shows 0x55, it means the i2c read and write test has passed.

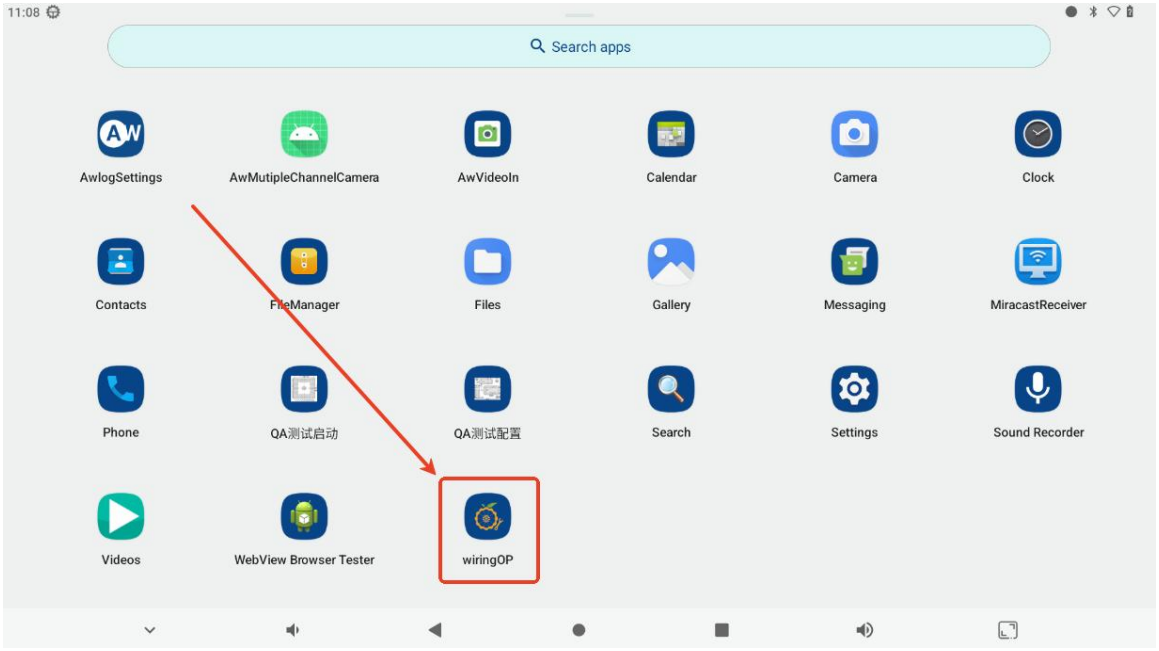


5. 12. 5. 40 Pin PWM Test Method

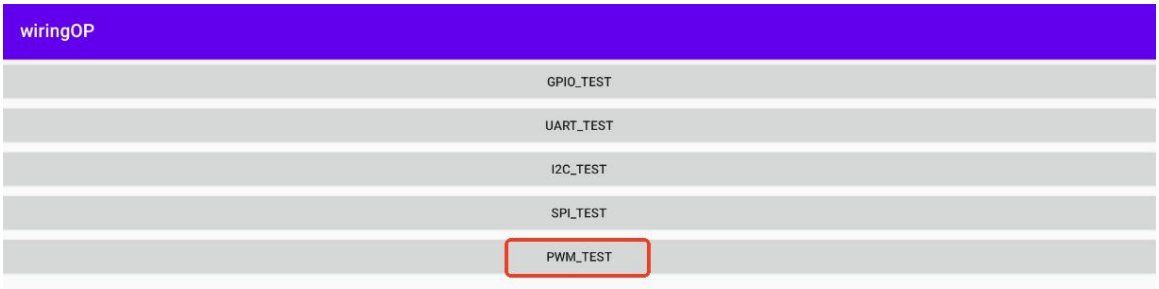
1) Android has **S_PWM0_2**, **PWM0_0**, and **PWM0_2** enabled by default. The corresponding pins are located at 40pin as shown in the figure below.

复用功能	复用功能	GPIO	GPIO序号	引脚序号	引脚序号	GPIO序号	GPIO	复用功能	复用功能
		3.3V		1	2		5V		
	TWI0_SDA	PB3	35	3	4		5V		
	TWI0-SCK	PB2	34	5	6		GND		
	S-PWM0-2	PL4	356	7	8	358	PL6	UART7-TX	
		GND		9	10	359	PL7	UART7-RX	
	UART8-RX	PL9	361	11	12	360	PL8	UART8-TX	
		PL12	364	13	14		GND		
		PK24	344	15	16	357	PL5		
		3.3V		17	18	345	PK25		
TWI2-SDA	SPI3-MOSI	PE2	130	19	20		GND		
TWI3-SCK	SPI3-MISO	PE3	131	21	22	119	PD23		
TWI2-SCK	SPI3-CLK	PE1	129	23	24	140	PE12		
		GND		25	26	132	PE4	SPI3-CS1	TWI3-SDA
	TWI1-SDA	PB5	37	27	28	36	PB4	TWI1-SCK	
	PWM0-0	PD0	96	29	30		GND		
		PD1	97	31	32	101	PD5	PWM0-5	
	PWM0-2	PD2	98	33	34		GND		
	PWM0-3	PD3	99	35	36	102	PD6	PWM0-6	
	PWM0-4	PD4	100	37	38	103	PD7	PWM0-7	
		GND		39	40	365	PL13		

2) First, click the wiringOP icon to open the wiringOP APP



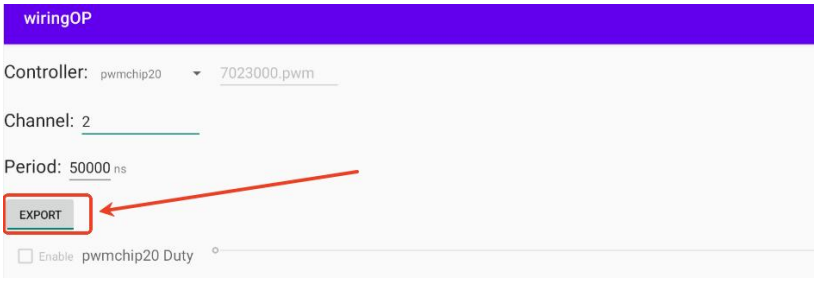
3) Then click the **PWM_TEST** button on the main interface of wiringOP to enter the PWM test interface



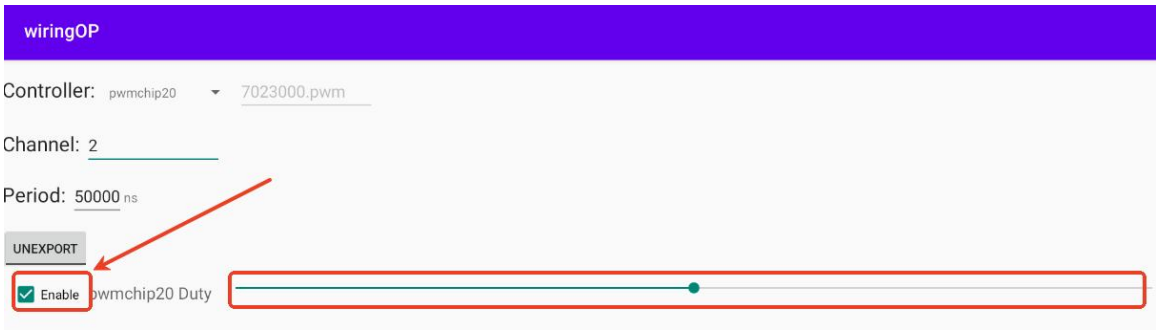
4) The base address of **S_PWM0_2** is **7023000**, and the base address of **PWM0_0** and **PWM0_2** is **2527000**. Taking **S_PWM0_2** as an example, the base address of **S_PWM0_2** is **7023000**. When you select pwmchip20 in the drop-down box,

7023000.pwm will be displayed on the right. If the displayed base address is incorrect, please click the drop-down option to select another pwmchip until **7023000** is displayed on the right.

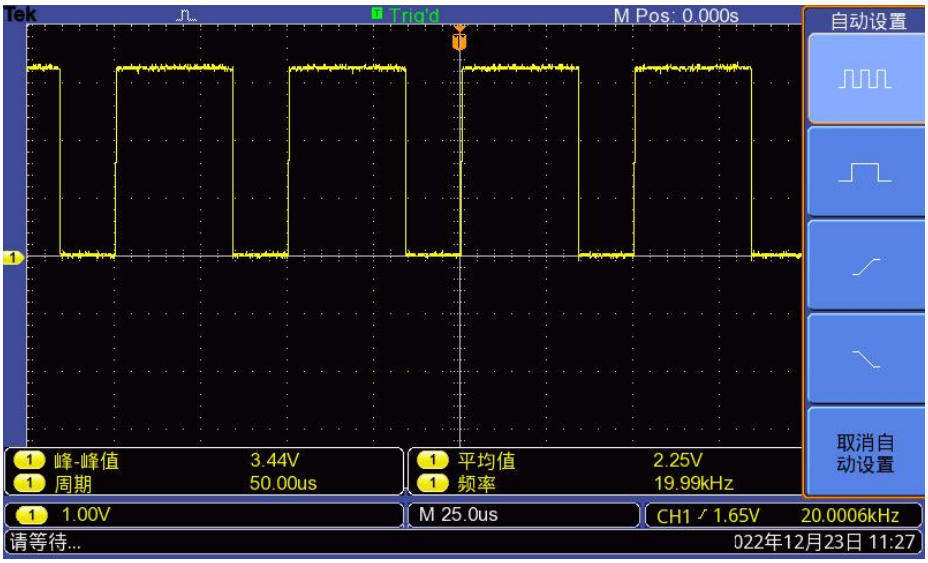
5) Since **S_PWM0_2** corresponds to the second channel of S_PWM0, Channel should be set to 2 and the PWM period should be confirmed. The default configuration is **50000ns**, which converts to a PWM frequency of **20KHz**. You can modify it yourself. Click the **EXPORT** button to export **S_PWM0_2**



6) Then drag the slider below to change the PWM duty cycle, and then check Enable to output the PWM waveform.



7) Then use an oscilloscope to measure pin 7 of the 40-pin development board and you can see the following waveform


Tek
Trig'd
M Pos: 0.000s
自动设置
取消自动设置
峰-峰值 3.44V
周期 50.00us
平均值 2.25V
频率 19.99kHz
1.00V
M 25.0us
CH1 ⟋ 1.65V
20.0006kHz
请等待...
022年12月23日 11:27

6. How to compile Android 13 source code

6.1. Download the source code of Android 13

1) First download the Android 13 source code volume compressed package from Baidu or Google Drive

2) After downloading the compressed package of Android 13 source code, please check whether the MD5 checksum is correct. If it is not correct, please download the source code again. The method of checking the MD5 checksum is as follows：

```
test@test:~$ md5sum -c md5sum
a733_android13.tar.gz00: OK
a733_android13.tar.gz01: OK
.....
```

3) Then execute the following command to decompress the Android source code

```
test@test:~$ cat a733_android13.tar.gz0* | tar -xvzf -
```

6.2. Compile the source code of Android 13

Android 13 is compiled on an x86_64 computer with Ubuntu 22.04 installed. The system package dependencies of other versions of Ubuntu may be slightly different. The image download address of the Ubuntu 22.04 amd64 version is as follows:
https://repo.huaweicloud.com/ubuntu-releases/22.04/ubuntu-22.04.2-desktop-amd64.iso

The recommended hardware configuration for an x86_64 computer for compiling the Android 13 source code is 16GB or more of memory, and 200GB or more of hard disk space is recommended. The more CPU cores, the better.

1) First install the software package required to compile the Android 13 source code

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

```
test@test:~$ sudo apt-get install -y git gnupg flex bison gperf build-essential \
zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 \
lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z1-dev ccache \
libgl1-mesa-dev libxml2-utils xsltproc unzip u-boot-tools python-is-python3 \
libssl-dev libncurses5 clang gawk
```

2) Then compile the code in the **longan** folder, which mainly contains u-boot and linux kernel

a. First run **./build.sh config** to set the compilation options

```
test@test:~$ cd a733_android13/longan
test@test:~/a733_android13/longan$ ./build.sh config
========ACTION List: mk_config ;========
options :
All available platform:
    0. android
    1. linux
Choice [android]: 0
All available ic:
    0. a523
    1. a527
    2. a733
    3. t527
    4. t736
Choice [a733]: 2
All available board:
    0. QA
    1. demo_aiot
    2. demo_car
    3. evb1
    4. fpga
    5. perf1
    6. pro2
    7. pro3
Choice [demo_aiot]: 1
All available flash:
    0. default
```

```
    1. nor
Choice [default]: 0
Setup BSP files
INFO: Prepare toolchain ...
```

b. Then run the **./build.sh** script to start compiling

```
test@test:~/a733_android13/longan$ ./build.sh
```

c. After the compilation is complete, you will see the following output

```
sun60iw2p1 compile all(Kernel+modules+boot.img) successful



INFO: build dts ...
INFO: Prepare toolchain ...
Setup BSP files
'/home/test/a733_android13/longan/out/a733/kernel/staging/sunxi.dtb' ->
'/home/test/a733_android13/longan/out/a733/demo/android/sunxi.dtb'
INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: ----------------------------------------
INFO: build Tina OK.
INFO: ----------------------------------------
```

3) Then use the following command to compile the Android source code and generate the final Android image

```
test@test:~$ cd a733_android13
test@test:~/a733_android13$ source build/envsetup.sh
test@test:~/a733_android13$ lunch a733_demo_aiot_arm64-userdebug
test@test:~/a733_android13$ make -j8
test@test:~/a733_android13$ pack
```

4) The storage path of the compiled Android image is:

```
longan/out/a733_android13_demo_aiot_uart0.img
```

7. Appendix

7.1. User Manual Update History

Version	date	Update Notes
v1.0	2025-10-14	Initial version
v1.1	2025-10-22	1. Linux：Debian bullseye System Hard Solution Test Description 2. Linux：Debian bullseye System GPU Test Description 3. Linux：NPU Usage Instructions
v1.2	2025-10-24	1. Linux：The method of creating a WIFI hotspot through create_ap 2. Linux：Usage of ADB 3. Linux：How to install Docker
v1.3	2025-10-28	1. How to add orangepi-build to synchronize code from the code cloud
v1.4	2025-11-06	1. Instructions for using OV13850 and IMX219 MIPI cameras

7.2. Image Update History

date	Update Notes
2025-10-14	OrangePi4Pro_A733_Android13_v1.0.0.tar.gz OrangePi4Pro_A733_Android13_lcd_v1.0.0.tar.gz Orangepi4pro_1.0.0_ubuntu_jammy_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.0_debian_bookworm_desktop_xfce_linux5.15.147.7z * Initial version
2025-10-22	Orangepi4pro_1.0.0_debian_bullseye_desktop_xfce_linux5.15.147.7z

	* Desktop supports GPU acceleration * Support Gstreamer hardware encoding and decoding
2025-10-24	Orangepi4pro_1.0.2_ubuntu_jammy_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.2_debian_bullseye_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.2_debian_bookworm_desktop_xfce_linux5.15.147.7z * Add support for adb * Add docker support
2025-10-28	Orangepi4pro_1.0.4_ubuntu_jammy_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.4_debian_bullseye_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.4_debian_bookworm_desktop_xfce_linux5.15.147.7z * Optimize SPI Flash+NVME startup stability
2025-11-11	Orangepi4pro_1.0.6_ubuntu_jammy_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.6_debian_bullseye_desktop_xfce_linux5.15.147.7z Orangepi4pro_1.0.6_debian_bookworm_desktop_xfce_linux5.15.147.7z * Support opening OV13850 and IMX219 MIPI cameras through OpenCV * Support displaying startup logo * Resolve some Kingston NVME SSD startup failure issues * Resolve the issue of system startup freezing after enabling LCD configuration Orangepi4pro_1.0.6_ubuntu_jammy_server_linux5.15.147.7z Orangepi4pro_1.0.6_debian_bullseye_server_linux5.15.147.7z Orangepi4pro_1.0.6_debian_bookworm_server_linux5.15.147.7z * Initial version