Orange Pi 4

User Manual

Content

1. Basic features of Orange Pi 4

1. 1. What's Orange Pi 4?

Orange Pi is an open source single-board computer, a new generation of arm64 development board, which can run operating systems such as Android 8.1, Ubuntu and Debian. Orange Pi development board (Orange Pi 4) uses Rockchip RK3399 system-on-chip, and has dual-channel 4GB LPDDR4 memory

1. 2. What can I do with Orange Pi 4?

We can bulid：

- A computer
- A wireless server
- Games
- Music and Sounds
- HD video
- A speaker
- Android

Pretty much anything else,because Orange pi 4 is open source

1. 3. Who's it for?

The Orange Pi development board is not only a consumer product, but also designed for anyone who wants to use technology to create and innovate. It is a very simple, interesting and practical tool, you can use it to create the world around you

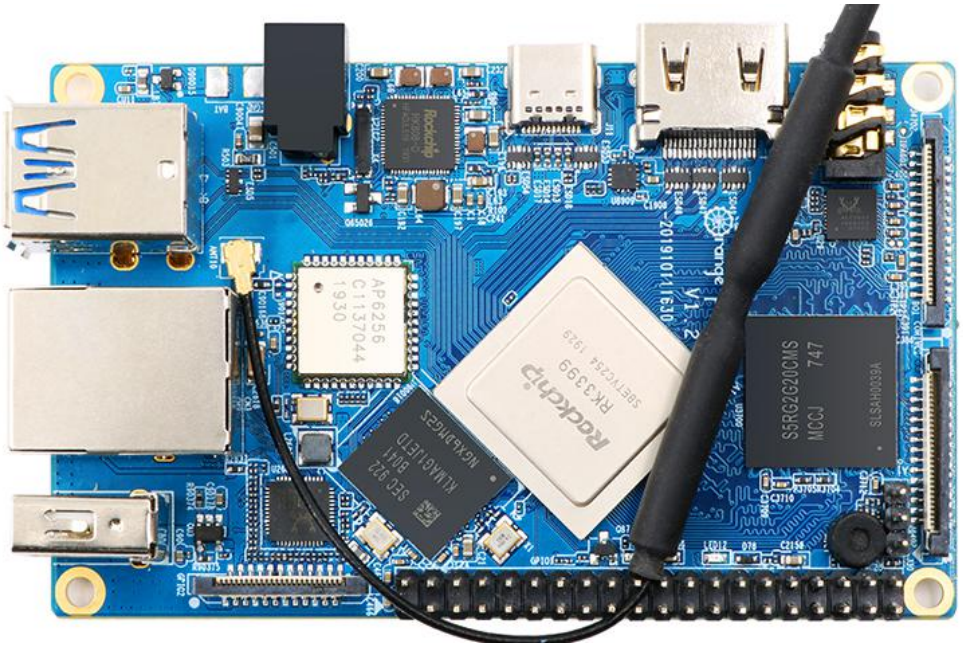
1. 4. Hardware Specification

Hardware Specification Introduction	
CPU	Rockchip RK3399 (28nm HKMG process) 6-core ARM® 64-bit processor ,main frequency speeds up to 2.0GHz Based on the large and small size core architecture of big.LITTLE: Dual-core Cortex-A72 (large core) + Quad-core Cortex-A53 (small core)
GPU	• High-performance multi-core GPU Mali T864 • OpenGL ES1.1/2.0/3.0/3.1 • OpenVG1.1 OpenCL, DX11, AFBC
Memory	Dual 4GB LPDDR4
Onboard Storage	• 16GB EMMC （Default Empty） • TF card slot
Network	10/100/1000Mbps Ethernet(Realtek RTL8211E)
WIFI+Bluetooth	• AP6256, IEEE 802.11 a/b/g/n/ac • BT5.0
Video Output	1 x HDMI 2.0 (Type-A), Supports 4K@60fps output 1 x DP 1.2 (Display Port) , Supports 4K@60fps output Supports Dual MIPI-DSI （4 lines per channel）
Video Input	MIPI-CSIx2 Camera connector (MIPI_RX0 、MIPI_TX1/RX1)
Audio Output	• 3.5mm Jack • HDMI2.0a
Audio Input	• Onboard MIC • Headphone recording
Power Source	• DC 5V/3A or DC 5V/4A • TYPE-C 5V/3A or TYPE-C 5V/4A
USB Port	• USB2.0 HOST x 2 • USB3.0 HOST x 1 • USB3.0 Type-C x 1

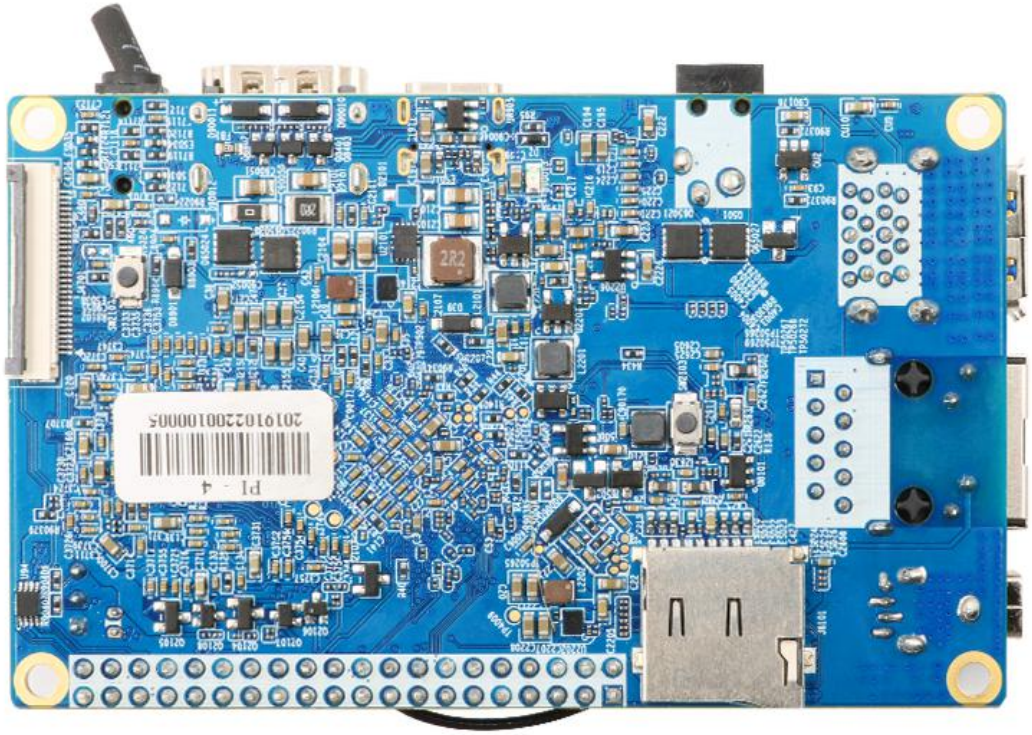
Low-level peripherals	GPIO1 40 pins (with I2Cx2、SPIx1/UART and GPIOx8)
Mini-PCIE	24pin mini-PCIE interface
RTC	Support RTC，on-board battery backup interface
Debug Serial Port	UART-TX、UART-RX and GND
LED	Power led & Status led
Button	Reset x1、Upgrade x1
Support OS	Android8.1、Ubuntu、Debian
Appearance specification introduction	
Dimension	91mm×55.7mm
 range Pi™ is a trademark of the Shenzhen Xunlong Software CO., Limited	

1.5. Top View and Bottom View of Orange Pi 4

Top View：

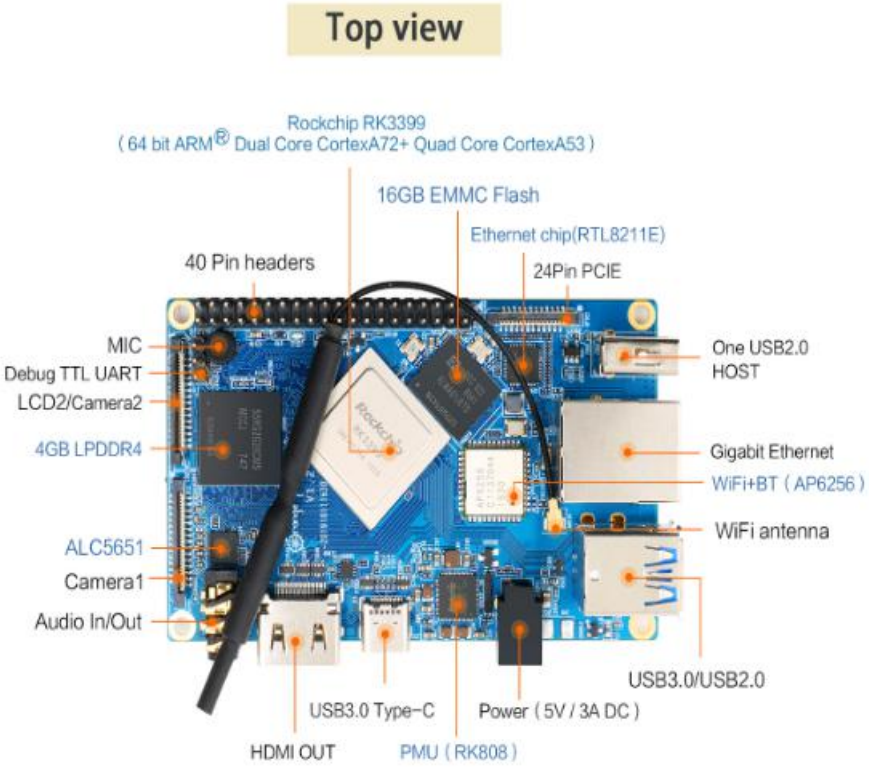


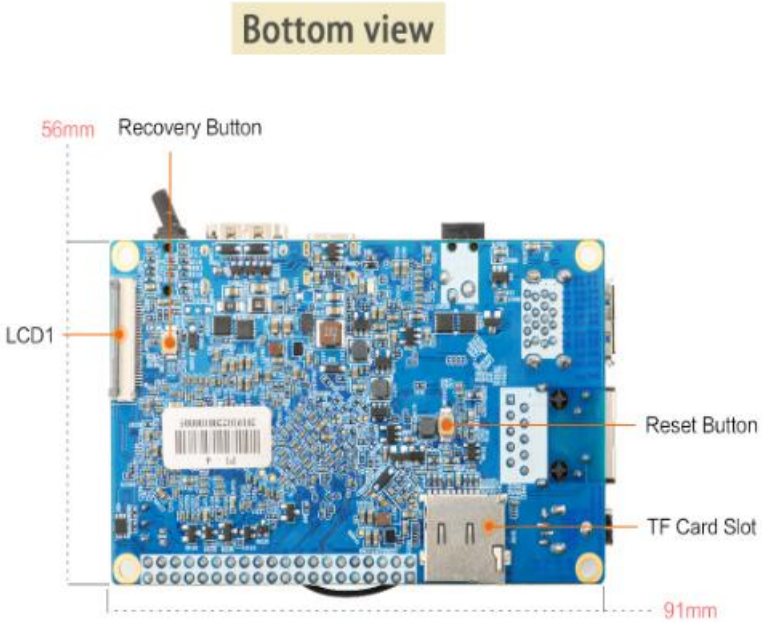
Bottom View：



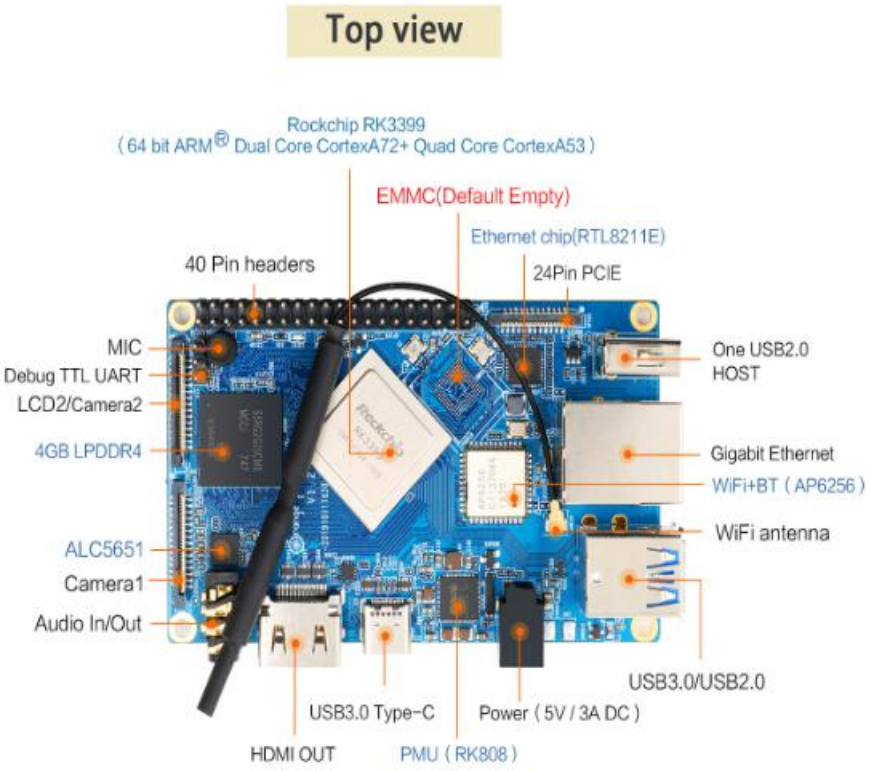
1. 6. Orange Pi 4 interface details

1. 6. 1. With 16GB EMMC Version





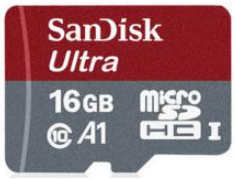
1. 6. 2. Without 16GB EMMC Version



2. Introduction to the use of the development board

2. 1. Prepare the necessary accessories

1) TF card, a high-speed card of class 10 or higher with a minimum capacity of 8GB, it is recommended to use SanDisk TF card, Orange Pi test is to use SanDisk TF card, other brands of TF card may have the problem of system failure



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

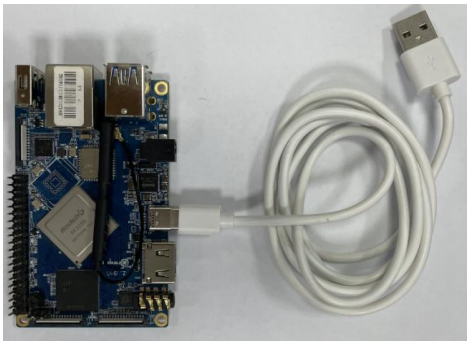
3) HDMI to HDMI cable, used to connect the development board to an HDMI monitor or TV for display

4) Type-C to HDMI cable, connect the development board to the HDMI monitor or TV through the Type-C interface for display

5) Type-C to USB3.0 adapter, used to connect USB3.0 storage devices or mouse keyboards and other devices

6) 10.1 inch MIPI screen, used to display the system interface of the development board

7) Power adapter, 5V/3A or 5V/4A high-quality power adapter, the Type-C USB3.0 interface of the development board can also be used for power input



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

9) 100M or Gigabit network cable, used to connect the development board to the Internet

10) OV13850 13 million camera, Orange Pi 4 dedicated camera, compatible with MIPI interface

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

12) A personal computer with Ubuntu and Windows operating systems

1	Ubuntu14.04 PC	Optional, used to compile Android source code

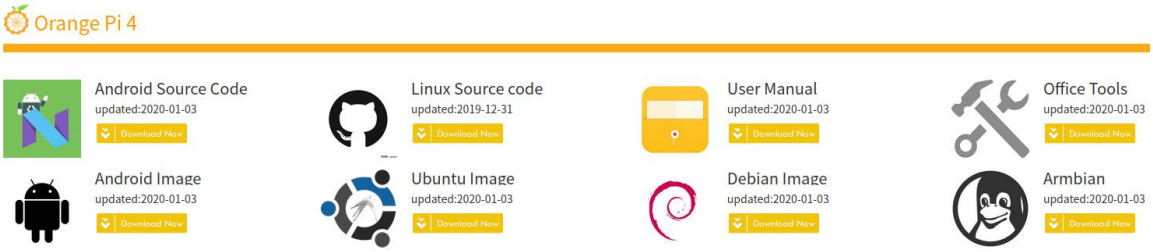
2	Ubuntu18.04 PC	Optional, used to compile Linux source code
3	Windows PC	Used to burn Android and Linux images

2. 2. Download the image and related information of the development board

1) The download URL of the Chinese version is

2) The download URL of the English version is

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



3) The information mainly contains

a. Android source code: saved on Baidu Cloud Disk and Google Cloud Disk
b. Linux source code: saved on github, the link address is

https://github.com/orangepi-xunlong/orangepi-build

c. User manuals and schematic diagrams: chip-related data manuals will also be placed here
d. Official tools: mainly include the software that needs to be used during the use of the development board
e. Android image: saved on Baidu Cloud Disk and Google Cloud Disk
f. Ubuntu image: saved on Baidu Cloud Disk and Google Cloud Disk
g. Debian image: saved on Baidu Cloud Disk and Google Cloud Disk

h. Armbian image link, a image developed and maintained by the Armbian community. If you encounter any problems during use, please go to the armbian forum to give feedback. The maintainer of the Armbian image and other people who use the Armbian image will assist in solving various problems. Is the fastest way to solve the problem. Orange Pi is not responsible for maintaining this image

2.3. Use the Android image pre-installed in eMMC to test the functions of the development board

If you bought the Orange Pi 4 development board with a 16GB eMMC version, after you get the development board, you can use the Android image pre-installed in the eMMC to test the functions of the development board first, and make sure that all the hardware functions of the development board are okay. Burn the system you want to use

2.4. Method of flashing Linux image to TF card based on Windows PC

1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

3) Then format the TF card

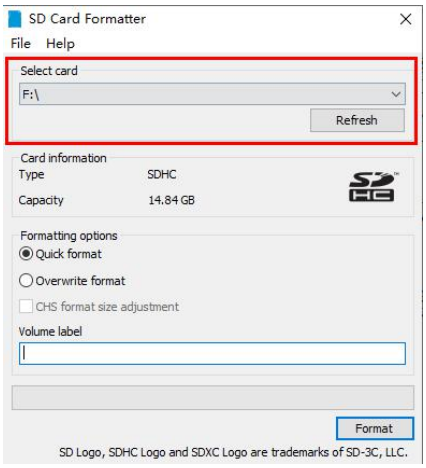
a. You can use the SD Card Formatter software to format the TF card, the download address is

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

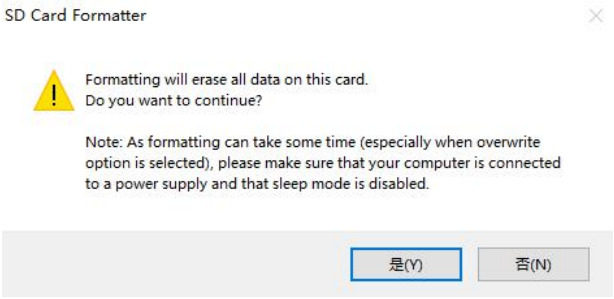
b. After downloading, you can directly unzip and install, and then open the software

c. If the computer only has a TF card inserted, the “Select card” column will display the drive letter of the TF card. If multiple USB storage devices are inserted into the computer, you can select the drive letter corresponding to the

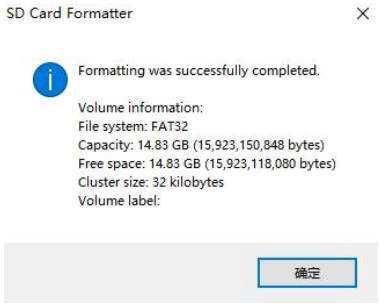
TF card through the drop-down box



d. Then click "Format", a warning box will pop up before formatting, and formatting will start after selecting "Yes (Y)"



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



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

5) Use Win32Diskimager to burn Linux image to TF card

a. The download page of Win32Diskimager is

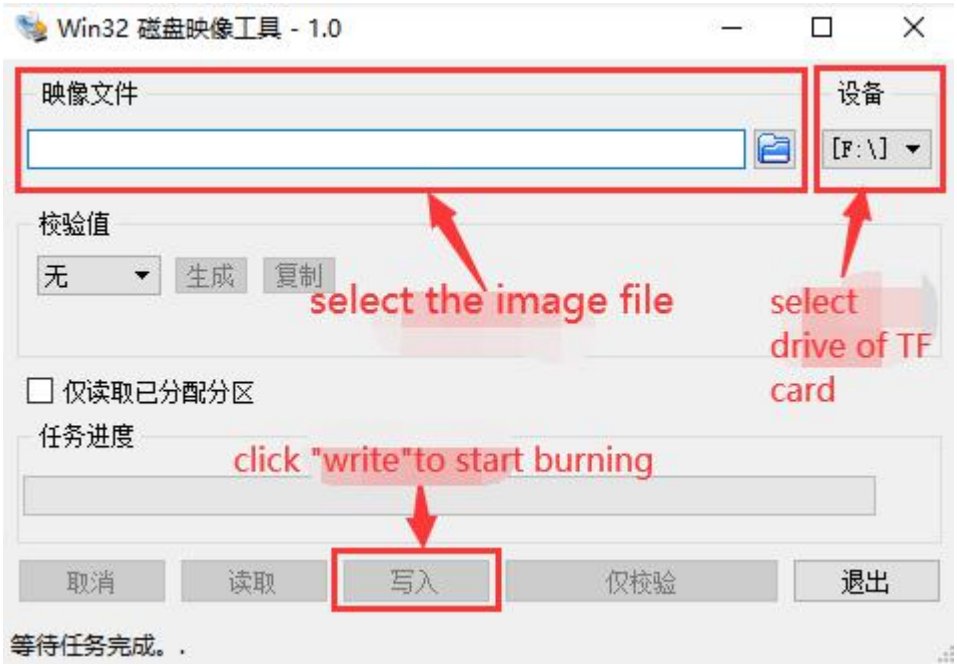
http://sourceforge.net/projects/win32diskimager/files/Archive/

b. Install directly after downloading, the interface of Win32Diskimager is shown below

a) First select the path of the image file

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

c) Finally click "write" to start burning



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

2. 5. Method of flashing Linux image to TF card based on Ubuntu PC

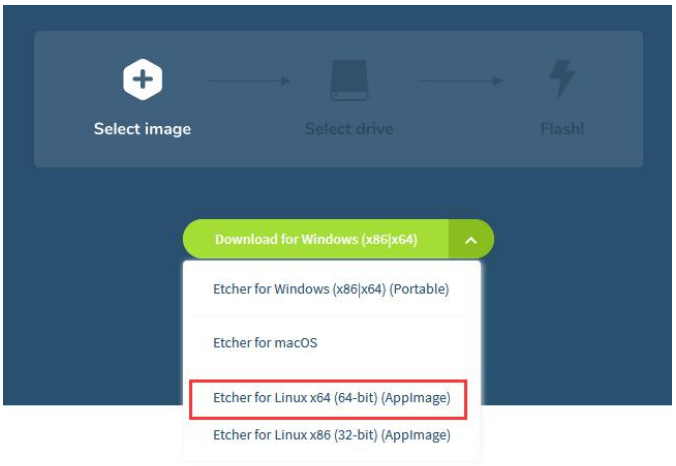
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

3) Download balenaEtcher software, the download address is

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

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



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

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

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

```
test@test:~$ ls Orangepi4_2.1.0_ubuntu_xenial_desktop_linux4.4.179
Orangepi4_2.1.0_ubuntu_xenial_desktop_linux4.4.179.img
Orangepi4_2.1.0_ubuntu_xenial_desktop_linux4.4.179.img.sha      # Checksum file
Orangepi4_2.1.0_ubuntu_xenial_desktop_linux4.4.179.img       # Image file
```

7) After decompressing the image, you can first use the sha256sum -c *.sha command to

calculate whether the checksum is correct. If the prompt indicates that the downloaded image is correct, you can safely burn it to the TF card. If the checksum does not match, it indicates There is a problem with the downloaded image, please try to download again

```
test@test:~$ sha256sum -c *.sha
Orangepi4_2.1.0_ubuntu_xenial_desktop_linux4.4.179.img: success
```

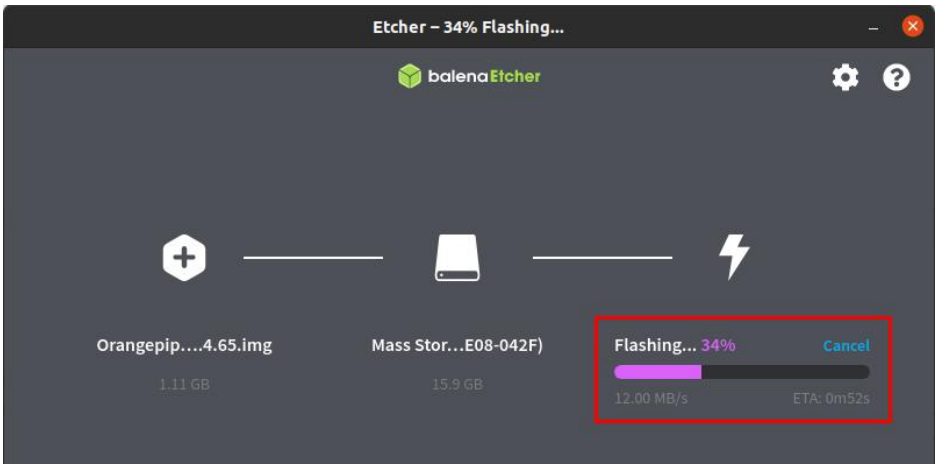
8) Then double-click balenaEtcher-1.5.109-x64.AppImage on the graphical interface of Ubuntu PC to open balenaEtcher (no need to install), the opened interface is as shown in the figure below

a. First select the path of the linux image file

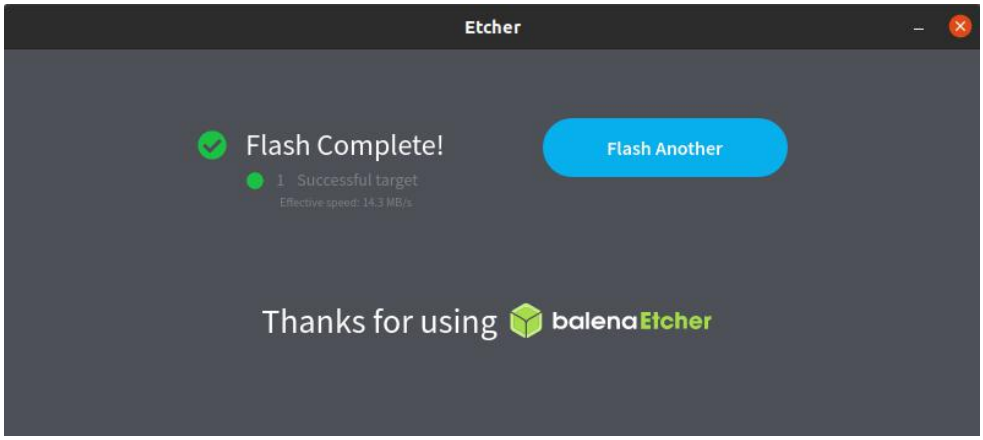
b. Then select the device number of the TF card

c. Finally click Flash to start burning

9) The writing speed and remaining time will be prompted during the burning process



10) After burning, the following interface will be displayed. At this time, you can unplug the TF card from the computer and insert it into the development board to start.



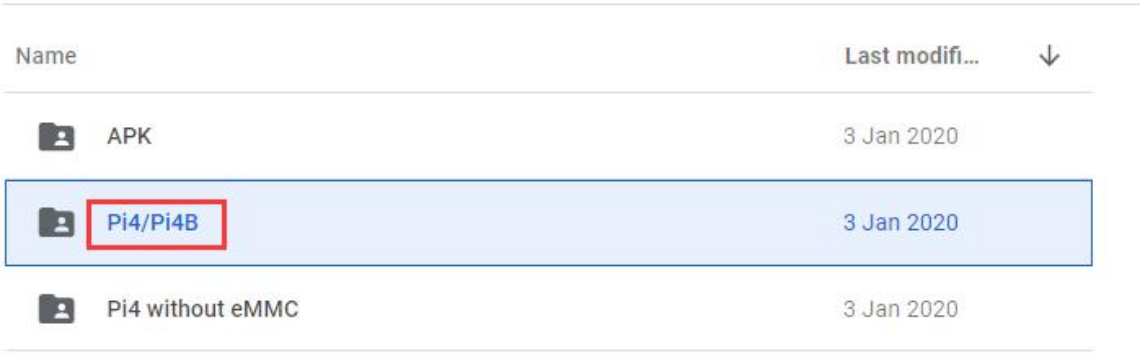
2. 6. Method of flashing Linux image to eMMC

See the method of flashing linux image to EMMC

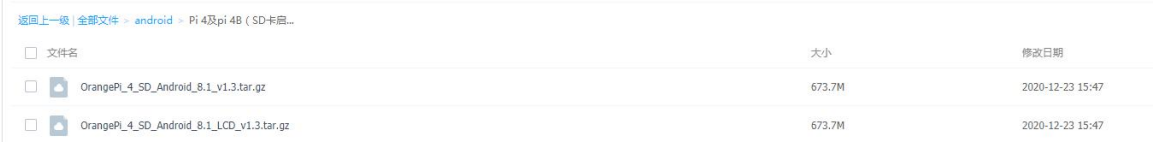
2. 7. How to burn Android firmware to TF card

2. 7. 1. Android image download that supports TF card startup

1) First download the firmware of Android 8.1 from Orange Pi's data download page, pay attention to select the "Pi4 and Pi4B (SD card boot image)" directory, the image in this directory is the image that supports TF card startup



2) There are two Android images in the "**Pi4 and Pi4B**" directory. The one with LCD is the firmware that supports 10.1-inch MIPI screen. The image without LCD only supports HDMI display, not LCD screen display, please choose according to actual needs



2. 7. 2. Make Android image TF card boot card

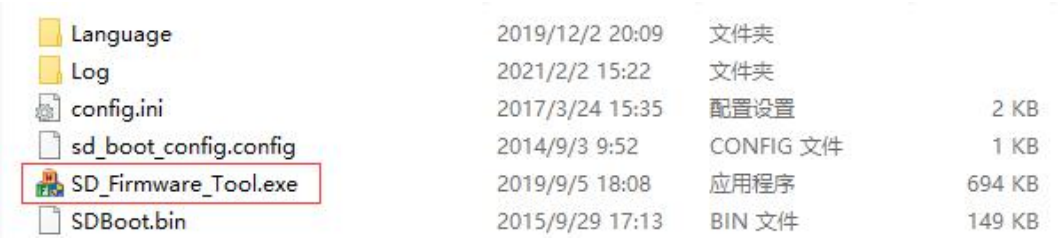
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

3) Then download **SDDiskTool_v1.59.zip** burning tool from Orange Pi's data download page

4) Then use the decompression software to decompress the compressed package of the downloaded Android firmware. In the decompressed file, the file ending with ".img" is the Android firmware

5) Then use the decompression software to decompress **SDDiskTool_v1.59.zip**, this software does not need to be installed, you can find **SD_Firmware_Tool.exe** in the decompressed folder and open it



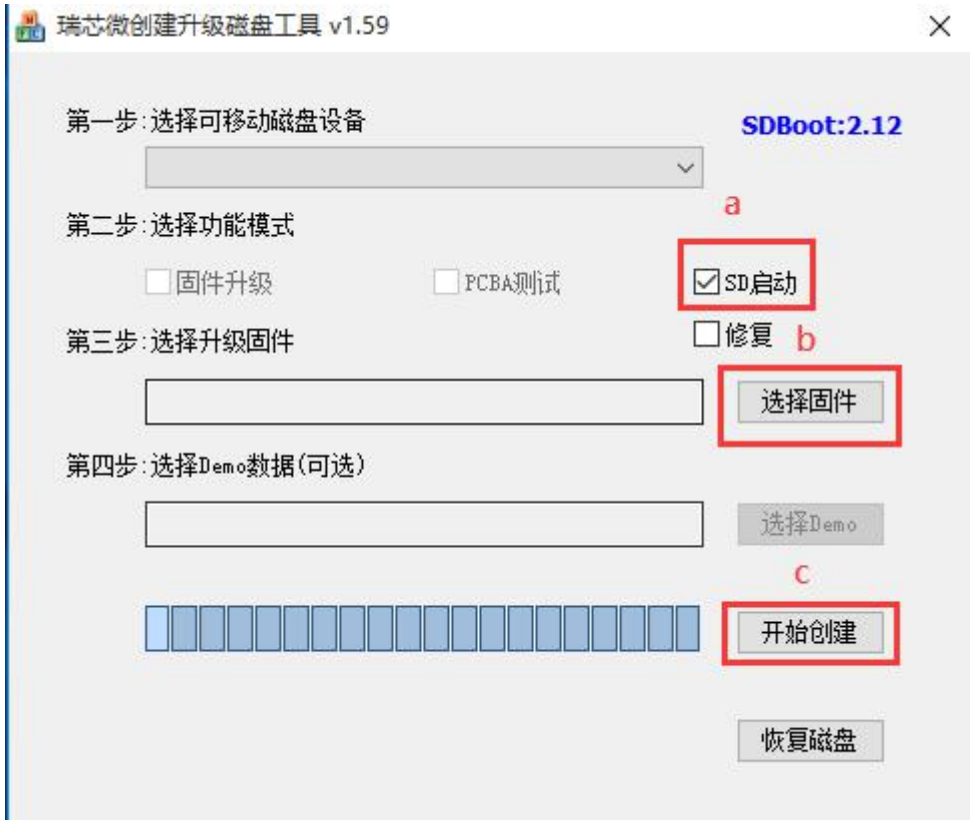
6) After opening **SDDiskTool**, if the TF card is recognized normally, the inserted disk device will be displayed in "**Select Removable Disk Device**". **Please make sure that the displayed disk device is consistent with the drive letter of the TF card you want to burn**. If there is no display, you can try to unplug the TF card

7) Then start to write the Android firmware to the TF card

a. First select "**SD boot**" in "**Select function mode**"

b. Then select the path of Android firmware in the "**Select Firmware Upgrade**" column

c. Finally, click the "**Start to create**" button to start burning the Android image to the TF card

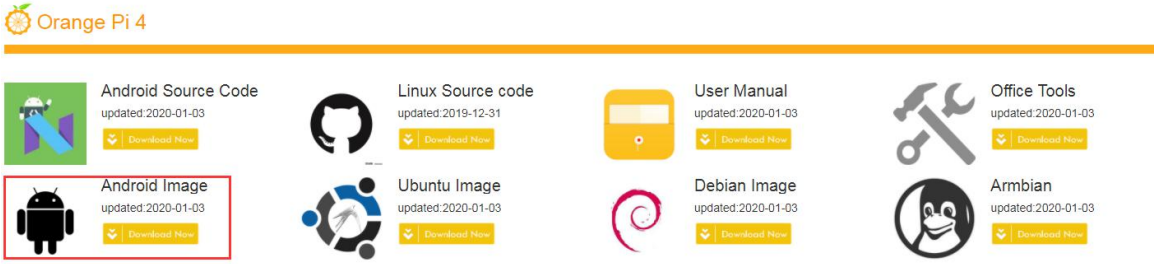


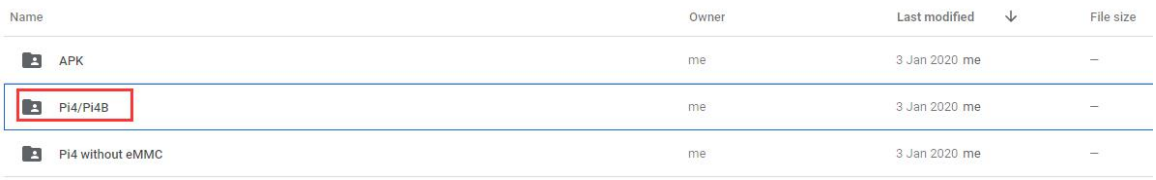
8) After burning, you can exit SDDiskTool, and then you can unplug the TF card from the computer and insert it into the development board to start.

2. 8. Method to Flash Android image to EMMC based on Windows PC

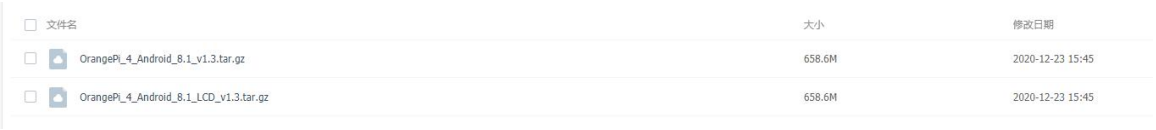
2. 8. 1. Support the download of EMMC boot image

1) First download the firmware of Android 8.1 from Orange Pi's download page, pay attention to select the "**Pi4 and Pi 4B (EMMC boot image)**" directory, the image in this directory is the image that supports EMMC boot





2) There are two Android images in the "**Pi4 and Pi4B (emmc boot image)**" directory. The one with LCD is the firmware that supports 10.1-inch MIPI screen. The image without LCD only supports HDMI display and does not support LCD screen display. Please follow Choose according to actual needs

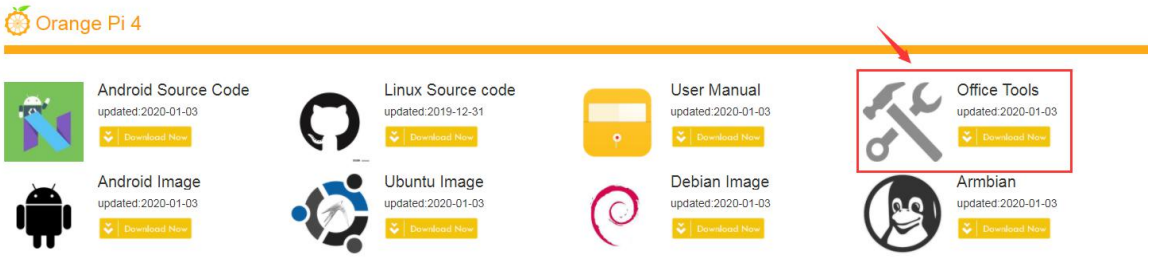


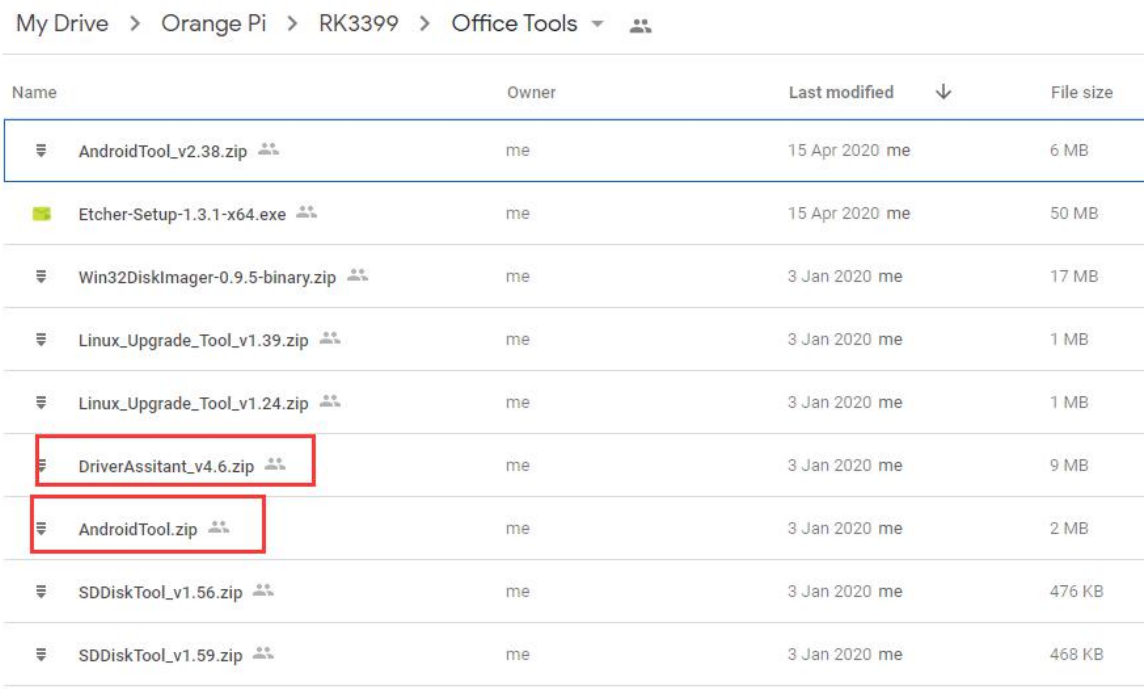
3) Then use the decompression software to decompress the compressed package of the downloaded Android firmware. In the decompressed file, the file ending with ".img" is the Android image file that needs to be burned.

2. 8. 2. Burn Android image directly to EMMC through Type C interface

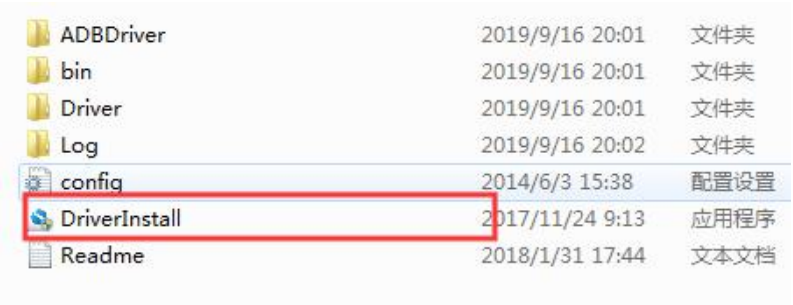
1) First prepare a good quality Type C data cable

2) Then download Rockchip Microdrive **DriverAssitant_v4.6** and the **Android** burning tool **AndroidTool.zip** under **Windows** system

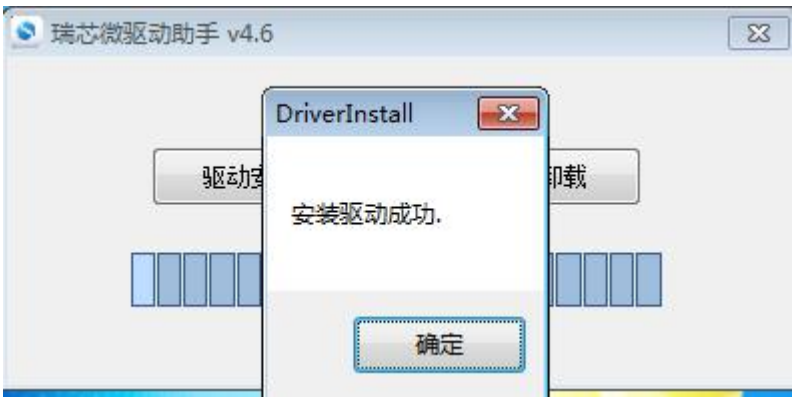




3) After decompressing the **DriverAssitant_v4.6** installation package, click the **DriverInstall.exe** program to start installing Rockchip Microdrive



4) The steps to install Rockchip Microdrive are as follows



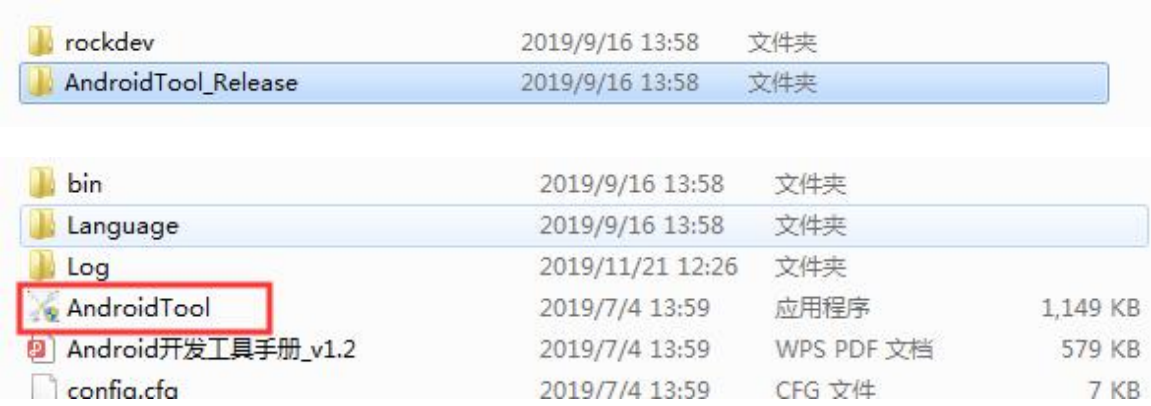
5) Then connect the DC power adapter to the OrangePi 4, and then connect the OrangePi 4 to the Windows PC through the Type-C data cable

6) First press and hold the upgrade button, and then lightly press the reset button. The position of the buttons is shown in the figure below

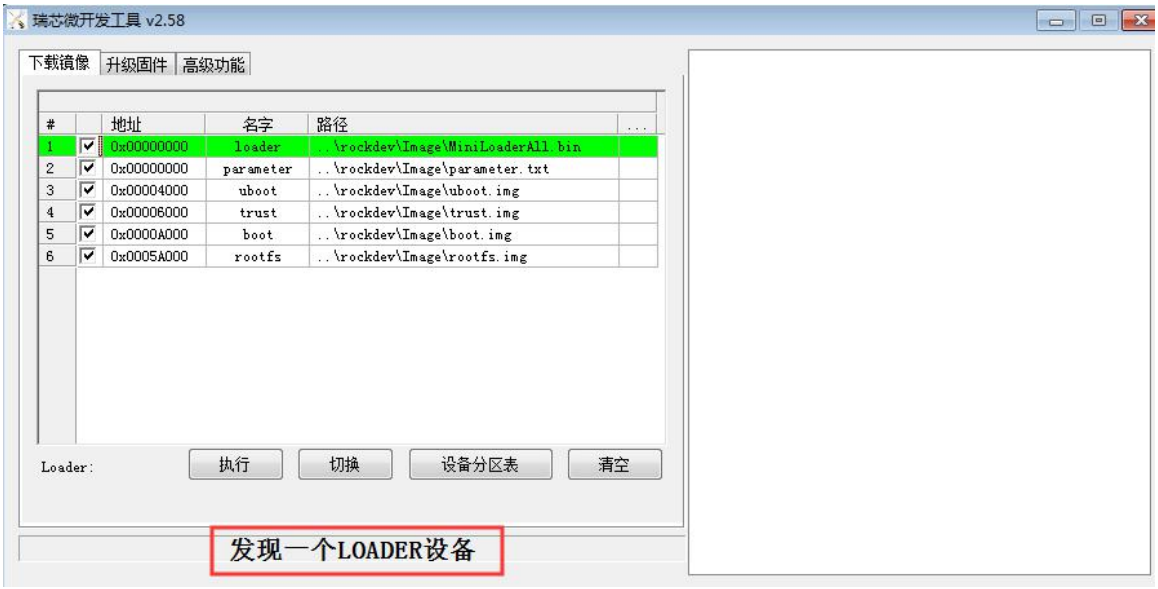
7) If everything is normal at this time, OrangePi 4 will enter loader mode. If the development board is connected to the serial port, you will see the following output information in the serial terminal

```
#Boot ver: 0000-00-00#0.00
empty serial no.
normal boot.
checkKey
vbus = 1
rockusb key pressed.
```

8) Then unzip and open AndroidTool

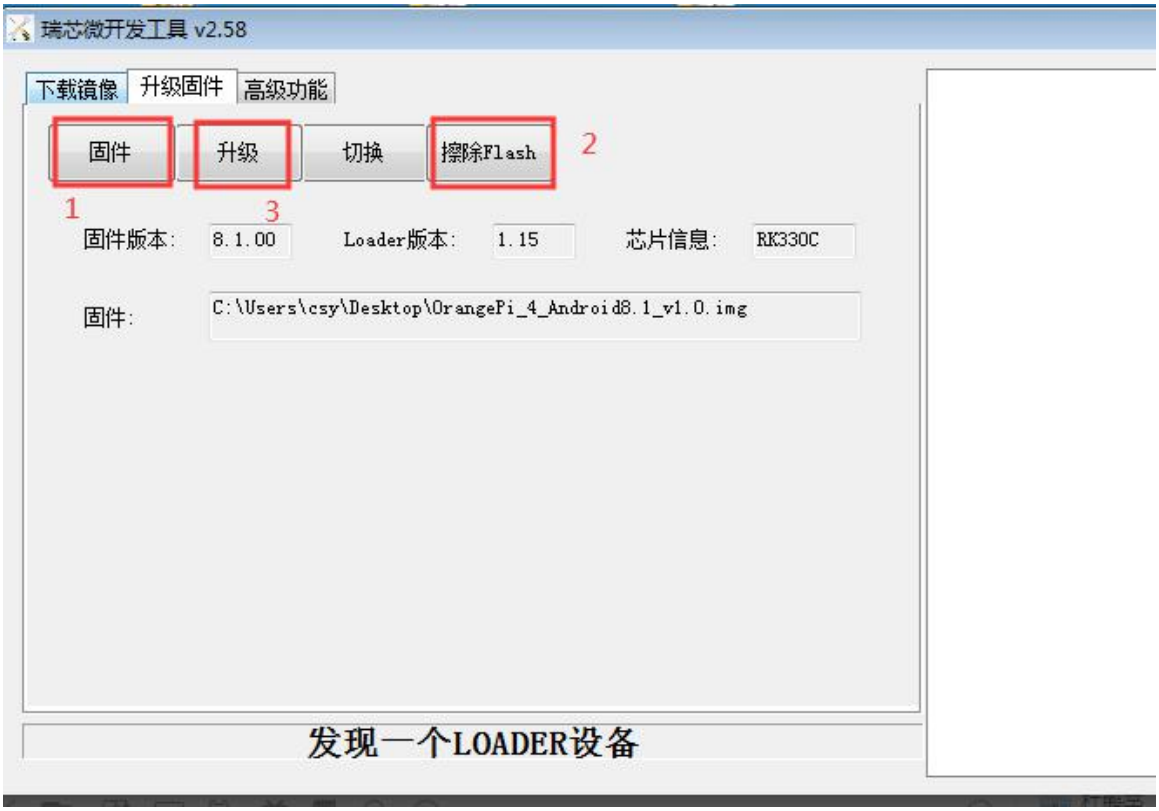


9) At this time, the Loader device recognized will be displayed under the AndroidTool tool, as shown below



10) Then click the "**Upgrade Firmware**" column, and then click the "**Firmware**" button to select the Android firmware path. At this time, you need to wait for a while. After the

firmware is loaded, click "**Erase Flash**" to erase the EMMC system, and finally click the "**Upgrade**" button to burn. The sequence is shown in the figure below.



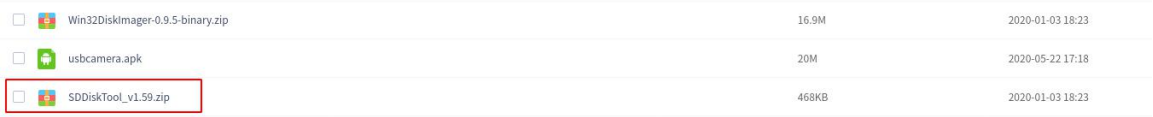
11) After the burning is completed, the AndroidTool display is as shown in the figure below, and the system will automatically start after the burning is completed

2. 8. 3. Burn Android image to EMMC via TF card

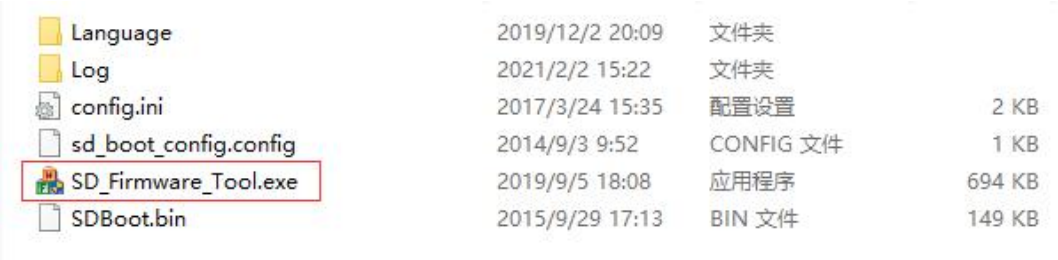
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

3) Download SDDiskTool from Orange Pi's data download page



4) Use decompression software to decompress **SDDiskTool_v1.59.zip**, this software does not need to be installed, you can find **SD_Firmware_Tool.exe** in the decompressed folder and open it



5) After opening **SDDiskTool**, if the TF card is recognized normally, the inserted disk device will be displayed in "Select Removable Disk Device". **Please make sure that the displayed disk device is consistent with the drive letter of the TF card you want to burn**. If there is no display, you can try to unplug and insert the TF card

6) Then start to write the Android firmware to the TF card

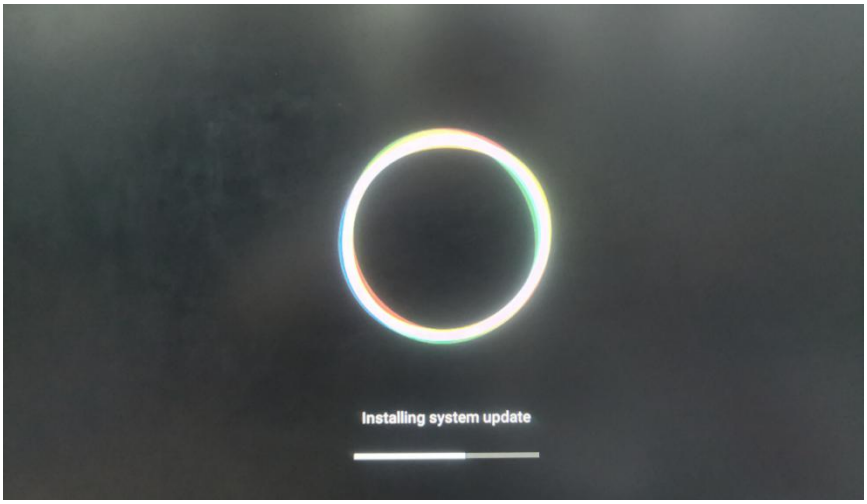
a. First select the path of Android firmware in the "Select Firmware Upgrade" column

b. Then select "Firmware Upgrade" in "Select Function Mode"

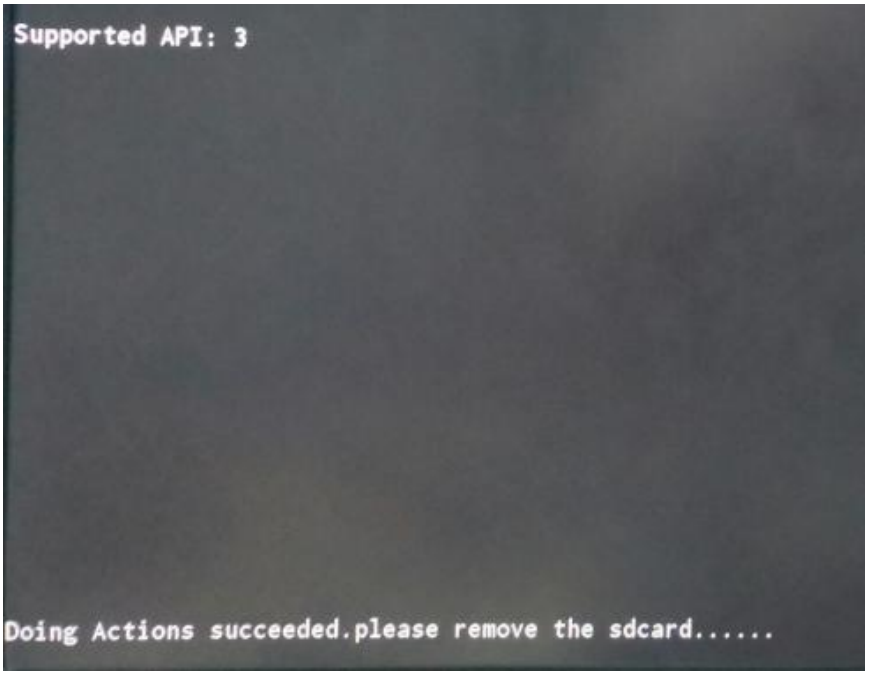
c. Finally, click the "Start Create" button to start burning

7) After burning, you can exit SDDiskTool, and then you can unplug the TF card from the computer and insert it into the development board. The boot will start to burn the Android image in TF to EMMC.

8) If the development board is connected to an HDMI display, you can also see the progress bar of burning Android image to EMMC from the HDMI display



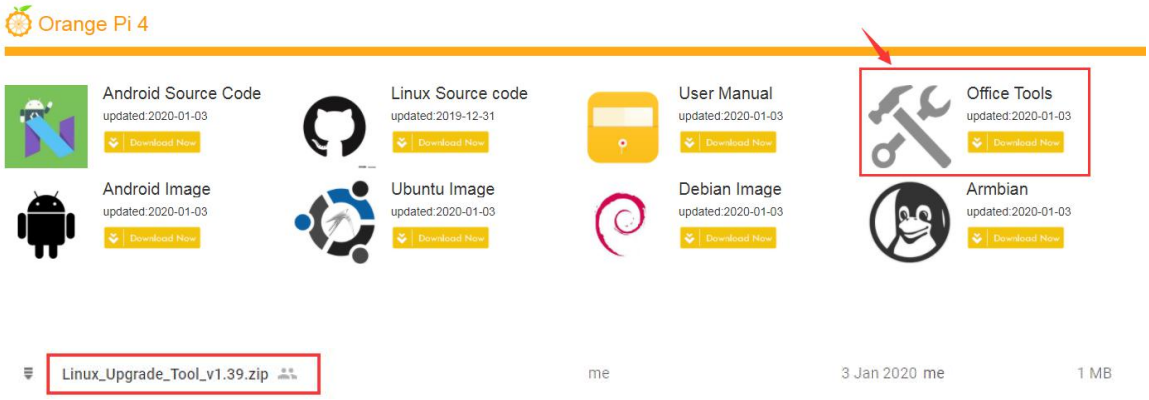
9) When the following message is displayed, it means that burning the Android image to EMMC is complete, and you can pull out the TF card at this time, and then the system will start to boot



2. 9. Method to burn Android image to EMMC based on Ubuntu PC

1) For the image download method, please refer to Support EMMC to start the image download

2) Download the **upgrade_tool** tool, this tool is suitable for Ubuntu PC system. Used to burn Android image to EMMC



3) Then execute the command in the terminal to decompress upgrade_tool and add executable permissions

```
test@test:~$ unzip Linux_Upgrade_Tool_v1.39.zip
test@test:~$ cd Linux_Upgrade_Tool_v1.39
test@test:~/Linux_Upgrade_Tool_v1.39$ sudo chmod +x ./upgrade_tool
```

4) Then connect the DC power adapter to the OrangePi 4 development board, and then use the Type-C data cable to connect the OrangePi 4 development board to the Ubuntu PC

5) Then press and hold the upgrade button of the Orange Pi 4 development board, and then lightly press the reset button. The position of the button is shown in the figure below

6) If everything is normal, OrangePi 4 will enter the loader mode at this time, and then execute the **./upgrade_tool LD** command to check whether the loader device is recognized

```
test@test:~/Linux_Upgrade_Tool_v1.39$ ./upgrade_tool LD
Program Data in /home/csy/.config/upgrade_tool
List of rockusb connected(1)
DevNo=1     Vid=0x2207,Pid=0x330c,LocationID=2010201 Mode=Loader
```

7) Then copy the downloaded image to the Linux_Upgrade_Tool_v1.39 directory

8) The command to erase EMMC is as follows

```
test@test:~$ sudo ./upgrade_tool ef OrangePi_4_Android8.1_v1.0.img
```

9) Execute the following command to start burning Android firmware to EMMC

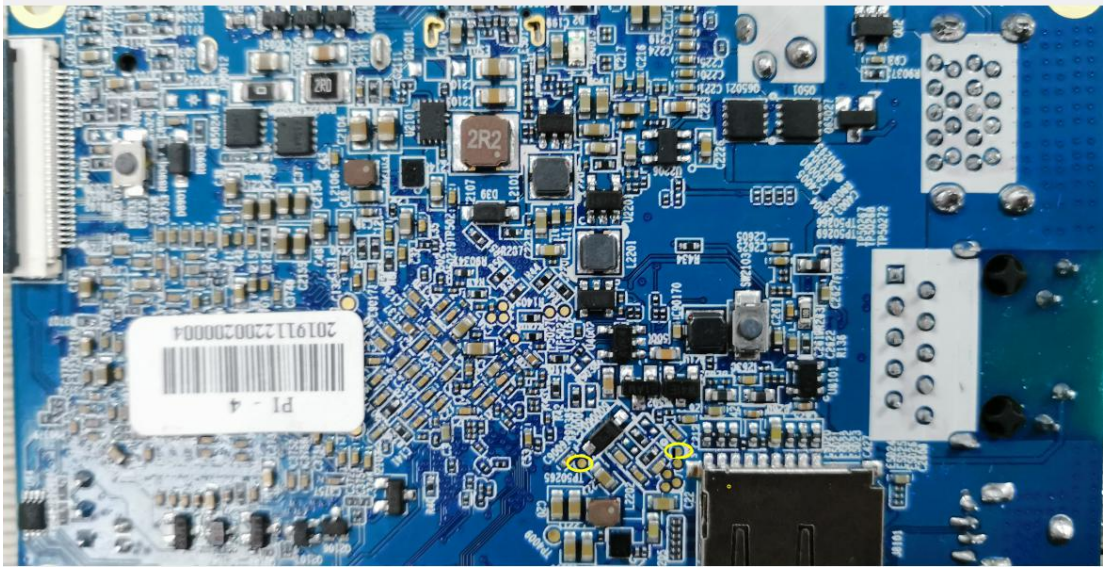
```
test@test:~$ sudo ./upgrade_tool uf OrangePi_4_Android8.1_v1.0.img
```

2. 10. Enter maskrom mode to burn the system to EMMC

Under normal circumstances, there is no need to enter the MaskRom mode. Only when the bootloader is damaged and the system cannot be started, you need to enter the maskrom mode to burn.

1) First make sure that the development board is disconnected from all power sources, and the SD card is pulled out

2) Then use metal tweezers to connect the solder joints reserved for OrangePi 4 (note the yellow circle in the picture below), and keep



3) Then plug the Orange Pi 4 into the DC power supply, wait for 2~3 seconds, and then loosen the metal tweezers. At this time, the OrangePi 4 development board will enter the maskrom mode

4) Then use the Type C cable to connect the OrangePi 4 development board and the Windows PC, and then open the AndroidTool tool, if everything is normal, you can see that AndroidTool has found a maskrom device

At this time, you can burn the Android image, please refer to the steps to burn the Android image directly to EMMC through the Type C interface

2. 11. 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 (if you use the image in EMMC, you don't need to insert the TF card)
2) The development board has an HDMI interface, you can connect the development board to a TV or HDMI display through an HDMI to HDMI cable
3) Connect the 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 for Internet access
5) Connect a 5V/3A (5V/4A is also available) high-quality power adapter

a. Remember not to plug in the 12V power adapter, if you plug in the 12V power adapter, the development board will be burned out

b. Many unstable phenomena during system power-on and startup are basically caused by power supply problems, so a reliable power adapter is very important

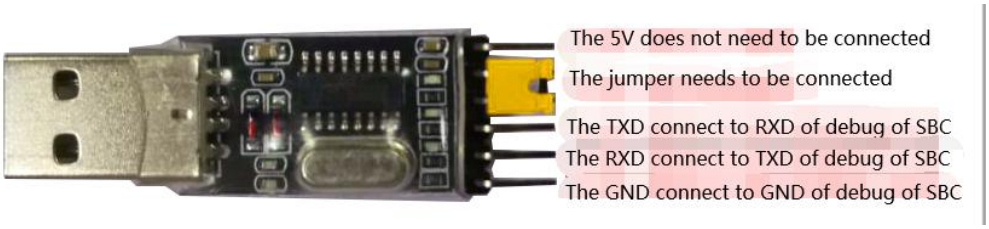
6) Then turn on the switch of the power adapter, if everything is normal, the HDMI display will be able to see the startup screen of the system at this time
7) If you want to view the output information of the system through the debug serial port, please use the USB to TTL module and DuPont cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section

on the use of the debug serial port

2. 12. How to use the debug serial port

2. 12. 1. Debug serial port connection instructions

1) First, you need to prepare a USB to TTL module. For better platform compatibility, CH340 USB to TTL module is recommended. Then insert one end of the USB interface of the USB to TTL module into the USB interface of the computer



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

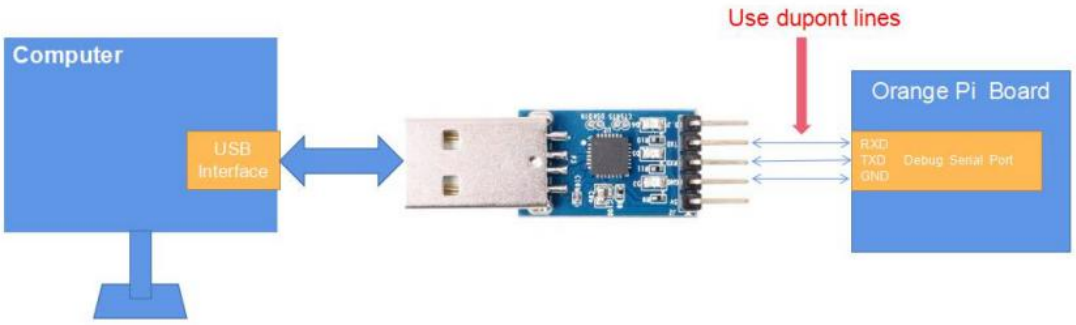
3) The GND, TX and RX pins of the USB to TTL module need to be connected to the debug serial port of the development board through a Dupont cable

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

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

c. Connect the TX of the USB to TTL module to the RX of the development board

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



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

5) If you are using a CP2102 USB to TTL module, under the condition of a baud rate of 1.500000, some systems may encounter garbled or unusable problems. The specific test situation is as follows

USB to TTL module model	Host system	Support situation
CH340	win7	ok
	win10	ok
	ubuntu14.04	ok
	ubuntu18.04	ok
	ubuntu20.04	ok
CP2102	win7	ok
	win10	Not available
	ubuntu14.04	ok
	ubuntu18.04	Partially not supported
	ubuntu20.04	Not available

2.12.2. How to use the debug serial port on the Ubuntu platform

1) If the USB to TTL module is connected normally, you can see the corresponding device node name under /dev of Ubuntu PC, remember this node name, you will use it when setting up the serial port software later

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

2) There are many serial debugging software that can be used under linux, such as putty, minicom, etc. The following shows how to use putty

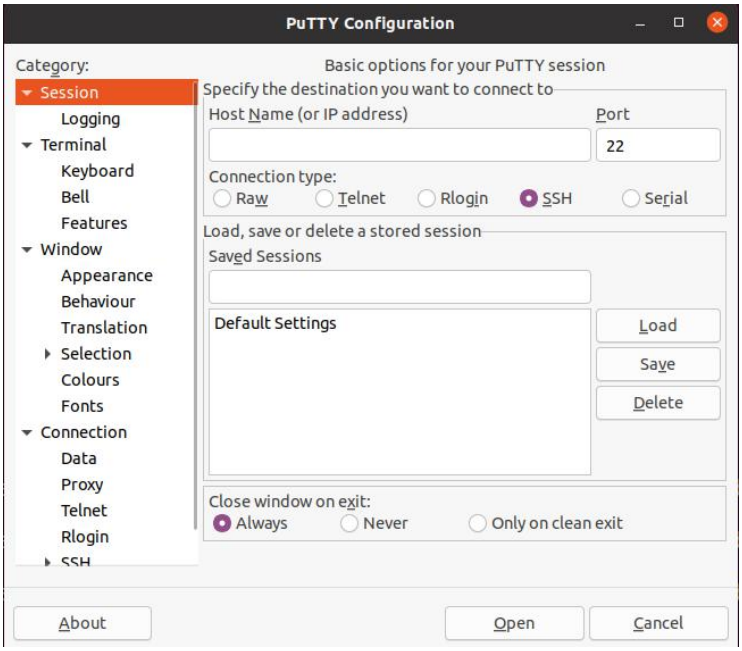
3) First install putty on the Ubuntu PC

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

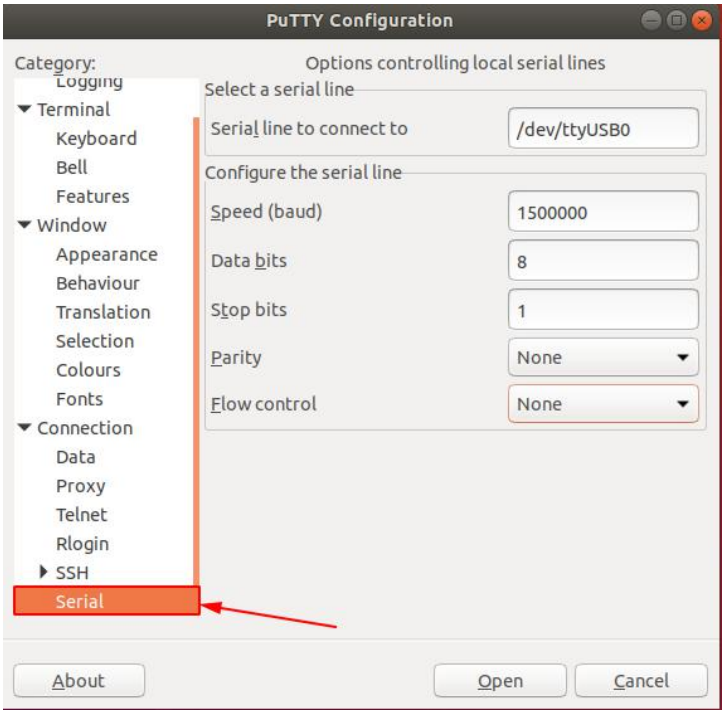
4) Then run putty, remember to add sudo permissions

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

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



6) First select the setting interface of the serial port

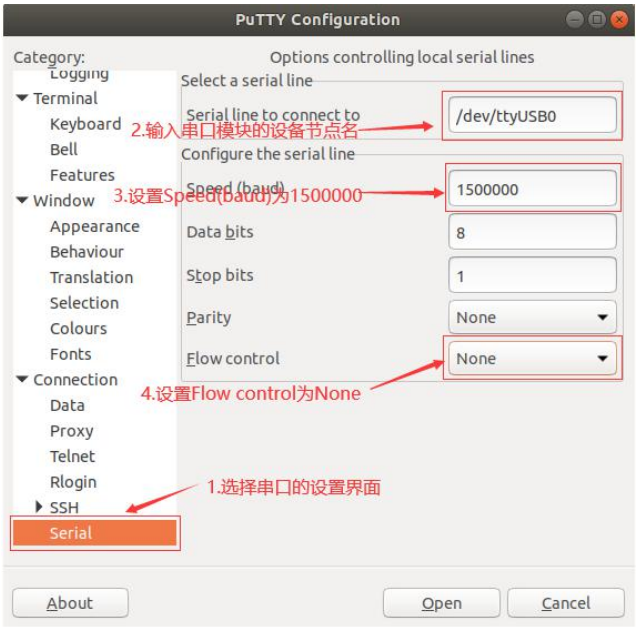


7) Then set the parameters of the serial port

a. Set Serial line to connect to to /dev/ttyUSB0 (modify to the corresponding node name, generally /dev/ttyUSB0)

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

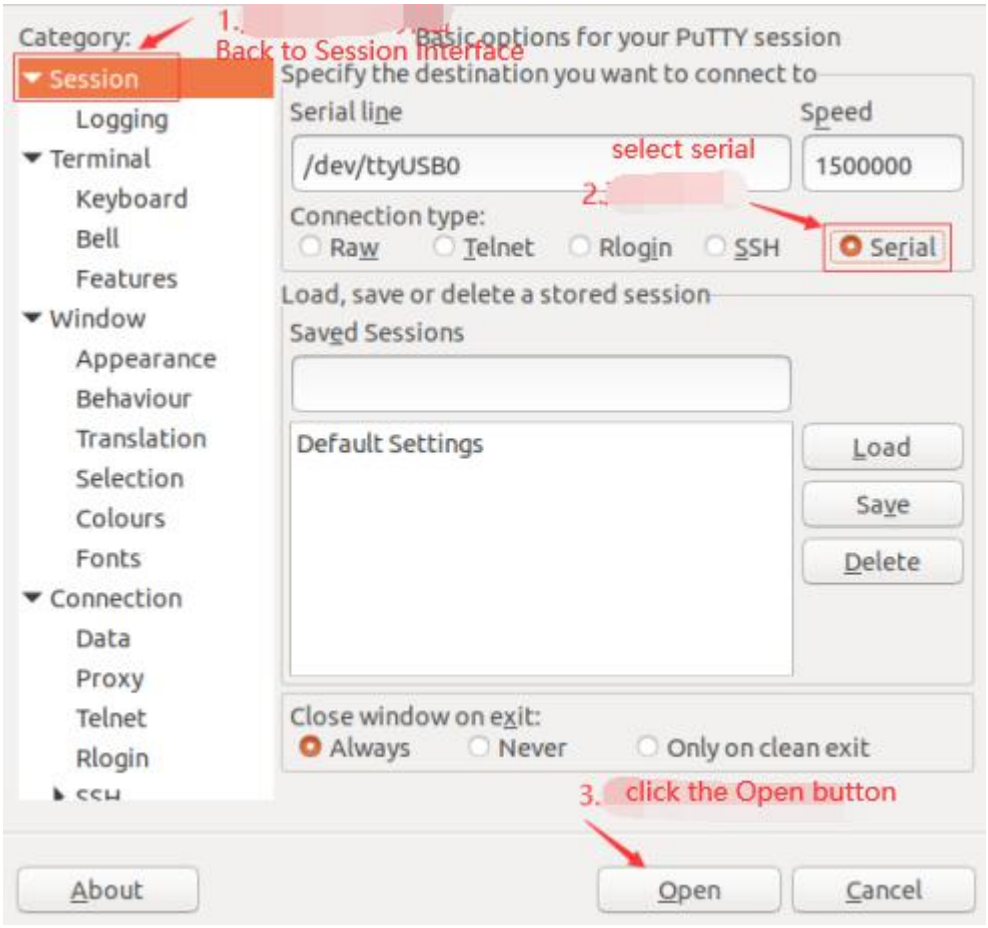
c. Set Flow control to None



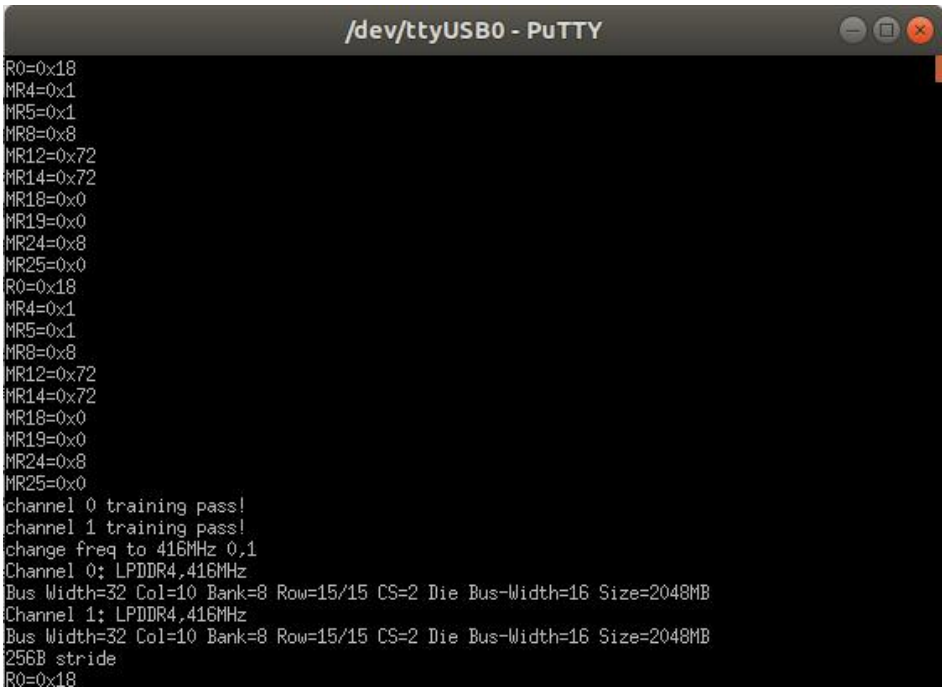
8) After setting the serial port setting interface, return to the Session interface

a. First select the Connection type as Serial

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



9) After starting the development board, you can see the Log information output by the system from the opened serial port terminal



2. 12. 3. How to use the debug serial port on Windows platform

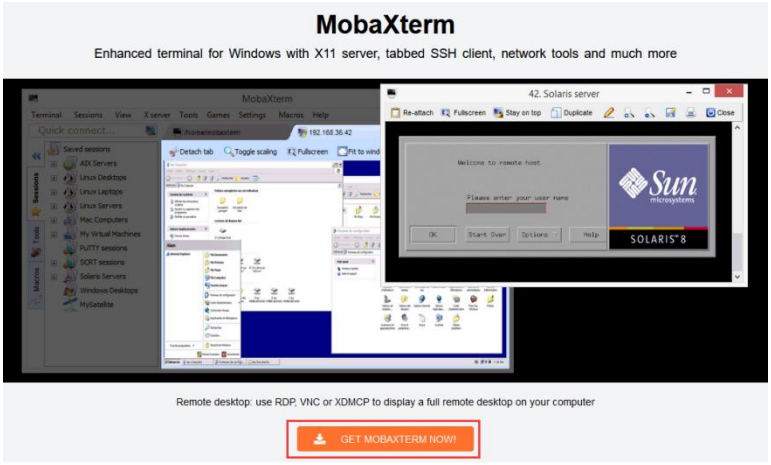
1) There are many serial debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following shows how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

2) Download MobaXterm

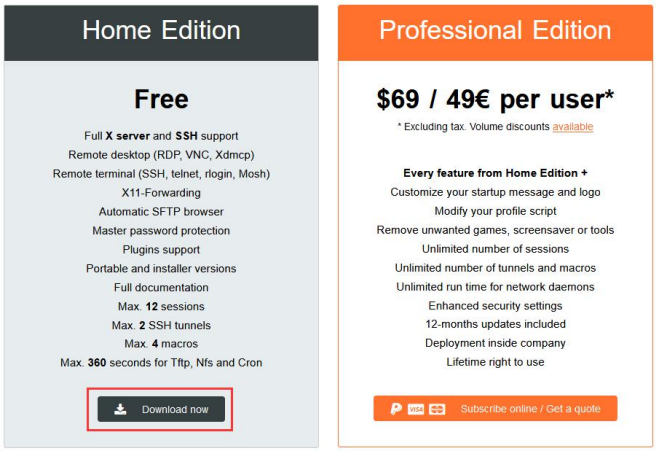
a. Download MobaXterm URL as follows

https://mobaxterm.mobatek.net/

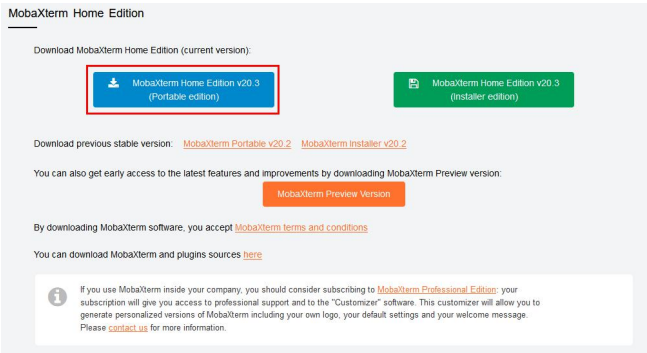
b. **After entering the MobaXterm download page, click GET XOBATERM NOW!**



c. Then choose to download the Home version



d. Then select the Portable version, after downloading, you don't need to install it, you can open it directly



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

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

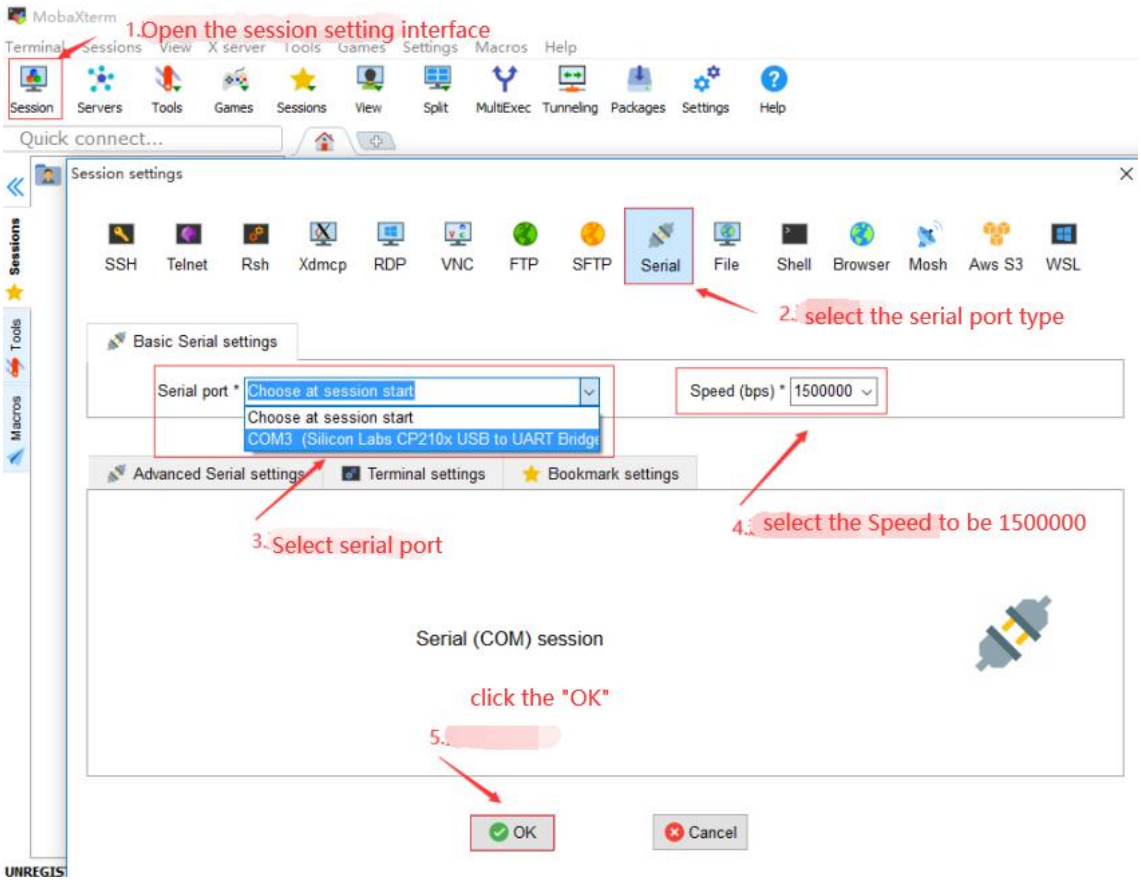
a. Open the session setting interface

b. Select the serial port type

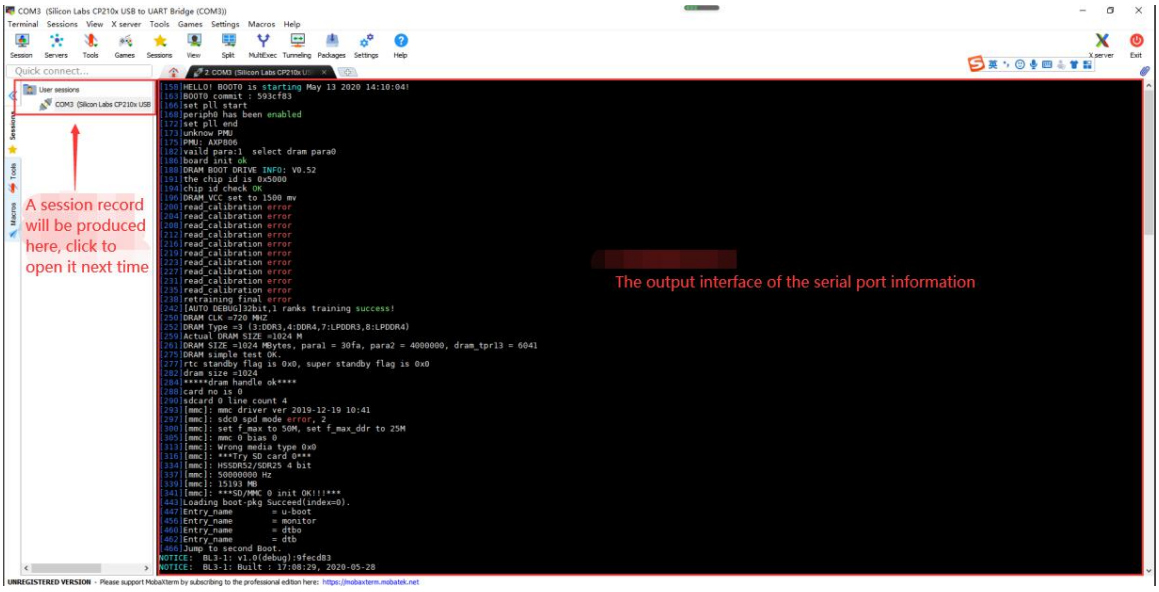
c. Select the port number of the serial port (choose the corresponding port number according to the actual situation), if you can't see the port number, please use the 360 driver master to scan and install the USB to TTL serial chip driver

d. Select the baud rate of the serial port to be 1500000

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



5) After clicking the "OK" button, you will enter the following interface, and you can see the output information of the serial port when you start the development board.



3. Linux system instructions

3. 1. Supported Linux distribution types and kernel versions

Release version	Kernel version	Server version	Desktop version
Ubuntu 16.04	linux4.4	Support	Support
Ubuntu 18.04	linux4.4	Support	Support
Debian 10	Linux4.4	Support	Support

3. 2. linux4.4 kernel driver adaptation situation

Function	Status
USB2.0x2	ok
USB3.0x1	ok
USB Type-C 3.0	ok
Type-C（Power Source）	ok
MiniPCIE	ok
gpio（40pin）	ok
spi/uart4（40pin）	ok
i2c2（40pin）	ok
i2c3（40pin）	ok
pwm（40pin）	ok
Debug serial port	ok
EMMC	ok
TF card boot	ok
HDMI video	ok
HDMI audio	ok
MIPI camera 1	ok
MIPI camera 2/Lcd2	ok
Lcd1	ok

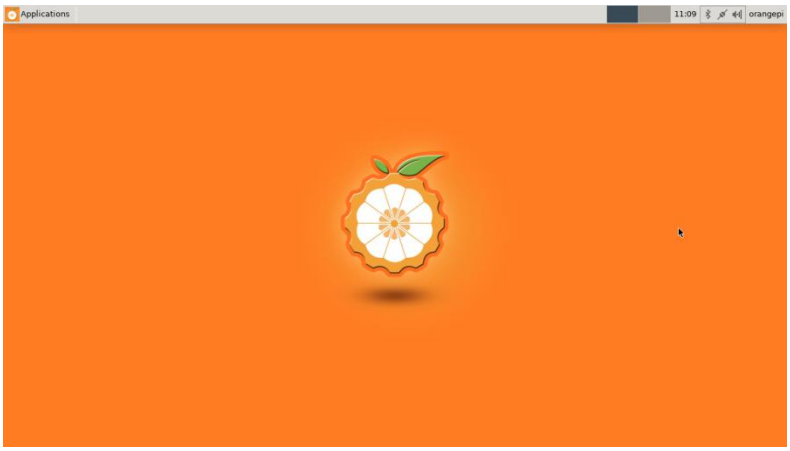
Gigabit Ethernet port RTL8211E	ok
Network port status light	ok
MIC	ok
Headphone playback	ok
Headphone recording	ok
WIFI（AP6256）	ok
Bluetooth（AP6256）	ok
LED	ok
Type-C to HDMI display	ok
GPU	ok
Hardware codec	ok
Reset button	ok
webGL hardware acceleration	ok
Simultaneous display of dual cameras	ok
MIPI screen dual screen display at the same time	ok
MIPI screen dual screen different display	ok
Watchdog test	ok

3.3. Linux system default login account and password

account number	password
root	orangepi
orangepi	orangepi

3.4. Instructions for automatic login of Linux desktop version system

1) The desktop version of the system will automatically log in to the desktop after the system is started by default, no need to enter a password



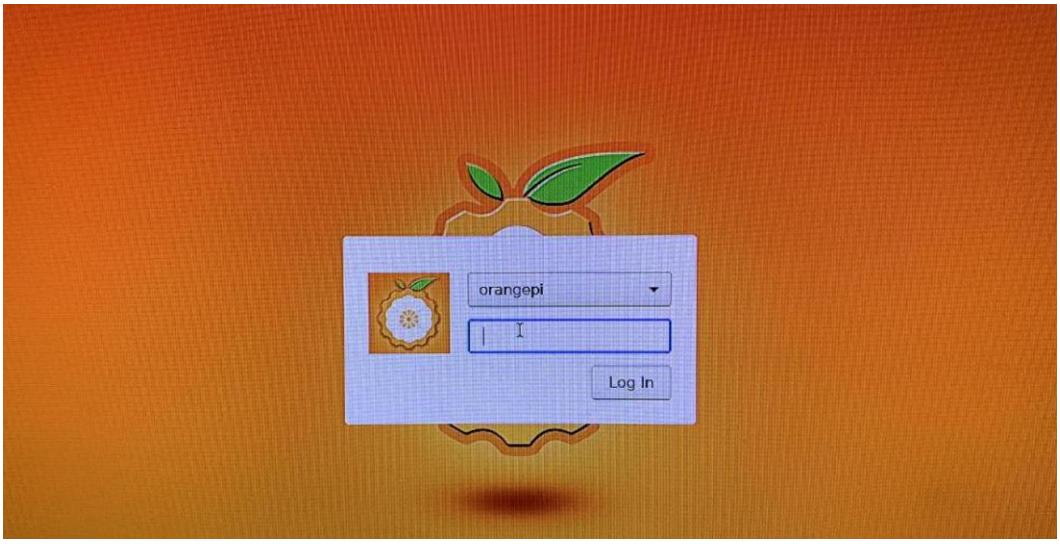
2) Modify the configuration in /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf to prevent the desktop version system from automatically logging in to the desktop. The modification command is as follows, or you can open the configuration file to modify it directly

```
root@orangepi4:~# sed -i "s/autologin-user=orangepi/#autologin-user=orangepi/"
/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
```

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

```
root@orangepi4:~# cat /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
[Seat:*]
#autologin-user=orangepi
autologin-user-timeout=0
user-session=xfce
```

4) Then restart the system, a login dialog box will appear, and you need to enter a password to enter the system



3. 5. Start the rootfs in the auto-expanding TF card for the first time

1) **When the TF card starts the linux system for the first time, it will use the orangepi-resize-filesystem.service systemd service to call the orangepi-resize-filesystem script to automatically expand the rootfs, so manually expand is not necessary.**

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

```
root@orangepi4:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G     0  1.9G   0% /dev
tmpfs           382M  5.6M  376M   2% /run
/dev/mmcblk0p1   15G  2.5G   12G  18% /
tmpfs           1.9G  140K  1.9G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
tmpfs           1.9G   12K  1.9G   1% /tmp
/dev/zram0       49M  3.3M   42M   8% /var/log
cgmfs           100K     0  100K   0% /run/cgmanager/fs
```

```
tmpfs           382M  8.0K  382M   1% /run/user/1000
tmpfs           382M     0  382M   0% /run/user/0
```

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

4) In addition, if you do not need to automatically expand rootfs, you can use the following method to prohibit

a. First burn the linux image to the TF card

b. Then insert the TF card into the Ubuntu PC (Windows does not work), the Ubuntu PC will usually automatically mount the TF card partition. If the automatic mounting is normal, use the ls command to see the following output, the TF card partition name and the following command The names shown are not necessarily the same, please modify according to the actual situation

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

a. Then switch the current user to root user in Ubuntu PC

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

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

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

c. Then you can unmount the TF card, then unplug the TF and plug it into the development board to start. When the linux system starts, when the file .no_rootfs_resize in the /root directory is detected, the rootfs will no longer be automatically expanded

d. After disabling automatic expansion of rootfs, you can see that the available capacity of the TF card is only about 200M

```
root@orangepi4:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G     0  1.9G   0% /dev
tmpfs           382M  5.6M  376M   2% /run
/dev/mmcblk0p1  2.9G  2.3G  487M  83% /
tmpfs           1.9G  140K  1.9G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
tmpfs           1.9G   12K  1.9G   1% /tmp
/dev/zram0       49M  2.6M   43M   6% /var/log
cgmfs           100K     0  100K   0% /run/cgmanager/fs
tmpfs           382M     0  382M   0% /run/user/0
tmpfs           382M   12K  382M   1% /run/user/1000
```

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

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

```
Starting kernel ...

Uncompressing Linux... done, booting the kernel.

Orange Pi 2.1.0 Xenial ttyFIQ0

orangepi login:
```

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

```
root@orangepi4:~# sed -i "s/verbosity=1/verbosity=7/" /boot/orangepiEnv.txt
root@orangepi4:~# sed -i "s/console=both/console=serial/" /boot/orangepiEnv.txt
```

3) The above commands are actually setting variables in /boot/orangepiEnv.txt, after setting, you can open /boot/orangepiEnv.txt to check

```
root@orangepi4:~# cat /boot/orangepiEnv.txt
verbosity=7
console=serial
```

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

3. 7. Ethernet port test

1) First, insert the network cable into the Ethernet interface of the development board, and ensure that the network is unblocked

2) After the system starts, it will automatically assign an IP address to the Ethernet card through DHCP

3) The command to view the IP address is as follows

```
root@orangepi4:~# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.122  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::4443:cee2:9b38:ff21  prefixlen 64  scopeid 0x20<link>
        ether 0a:b4:bd:10:02:90  txqueuelen 1000  (Ethernet)
        RX packets 14616  bytes 14141207 (14.1 MB)
        RX errors 0  dropped 1  overruns 0  frame 0
        TX packets 6845  bytes 725742 (725.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 24
```

4) The command to test network connectivity is as follows

```
root@orangepi4:~# ping www.baidu.com -I eth0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.122 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=1 ttl=56 time=8.56 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=2 ttl=56 time=7.66 ms
```

```
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=9.28 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=56 time=8.29 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 7.666/8.454/9.286/0.584 ms
```

3.8. SSH remote login to the development board

Linux systems have SSH remote login enabled by default, and allow root users to log in to the system. Before ssh login, you need to make sure that the Ethernet or wifi network is connected, and then use the ifconfig command or check the router to obtain the IP address of the development board

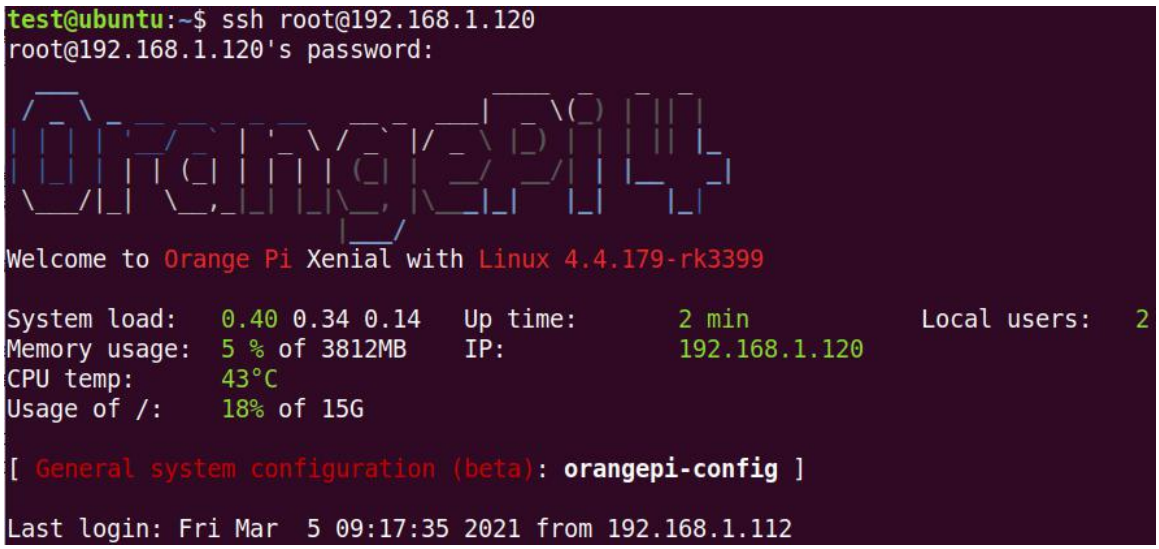
3.8.1. SSH remote login development board under Ubuntu

1) Get the IP address of the development board

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

```
test@test:~$ ssh root@192.168.1.120      (Need to be replaced with the IP address of the development board)
root@192.168.1.120's password:      （Enter the password here, the default password is orangepi）
```

3) The display after successfully logging in to the system is as shown in the figure below



4) If the following error is prompted during ssh login

```
test@test:~$ ssh root@192.168.1.120
Connection reset by 192.168.1.149 port 22
lost connection
```

You can enter the following command on the development board and try to connect

```
root@orangepi4:~# rm /etc/ssh/ssh_host_*
root@orangepi4:~# dpkg-reconfigure openssh-server
```

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

1) First get the IP address of the development board

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

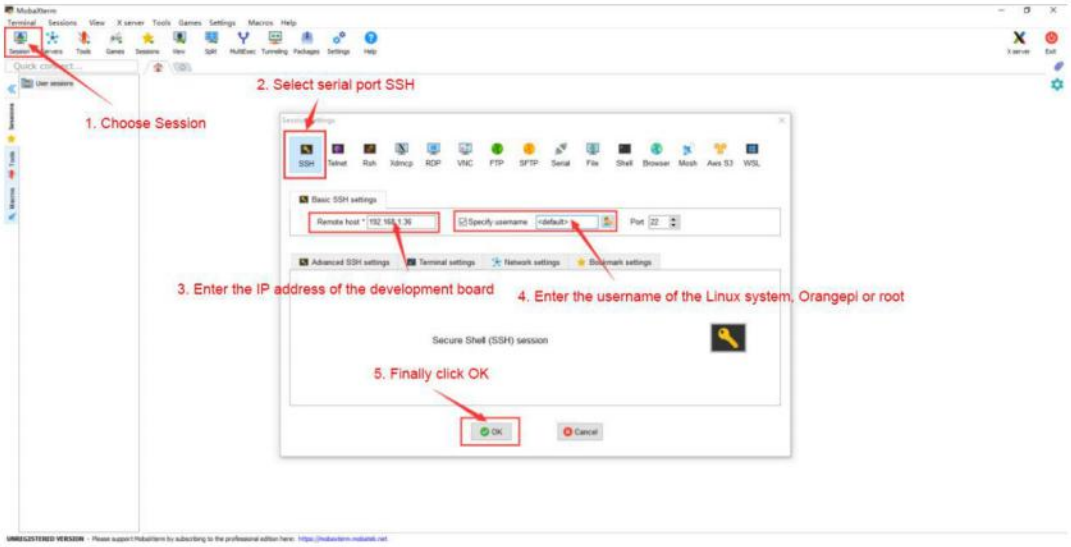
a. Open Session

b. Then select SSH in Session Setting

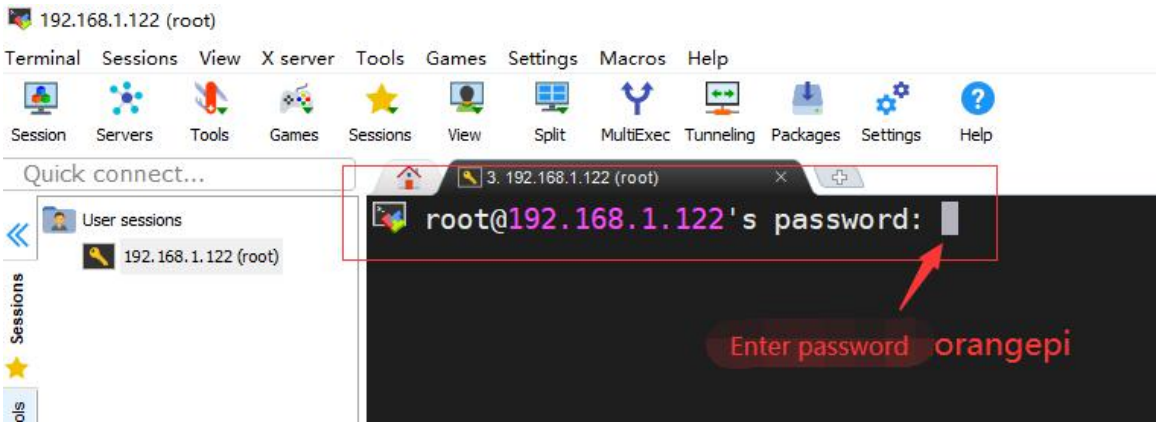
c. Then enter the IP address of the development board in Remote host

d. Then enter the username root or orangepi of the Linux system in Specify username

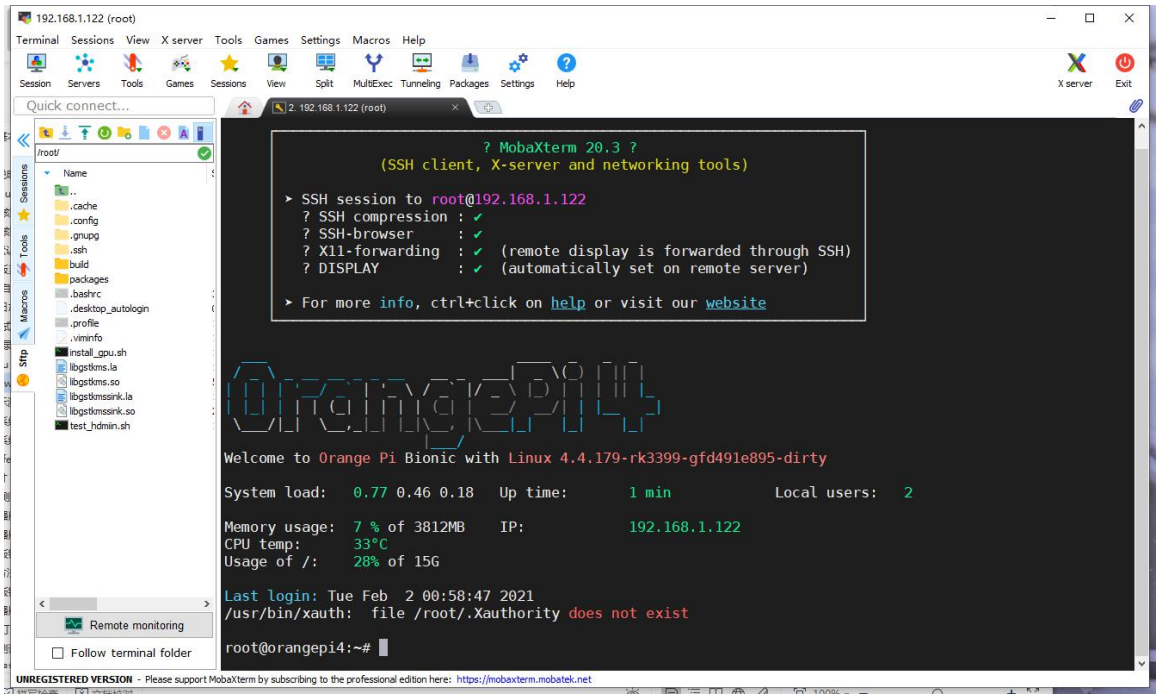
e. Finally click OK



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



4) The display after successfully logging in to the system is as shown in the figure below



3. 9. HDMI display test

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

2) If the HDMI display has image output after starting the linux system, it means that the HDMI interface is in normal use

3. 10. Type C to HDMI display test

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

2) If the HDMI display has image output after starting the linux system, it means that the Type C to HDMI display function is normal

3. 11. WIFI connection test

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

3. 11. 1. Connect to WIFI by command

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

also use the commands demonstrated in this section to connect to the WIFI network.

1) Log in to the linux system first, there are three ways

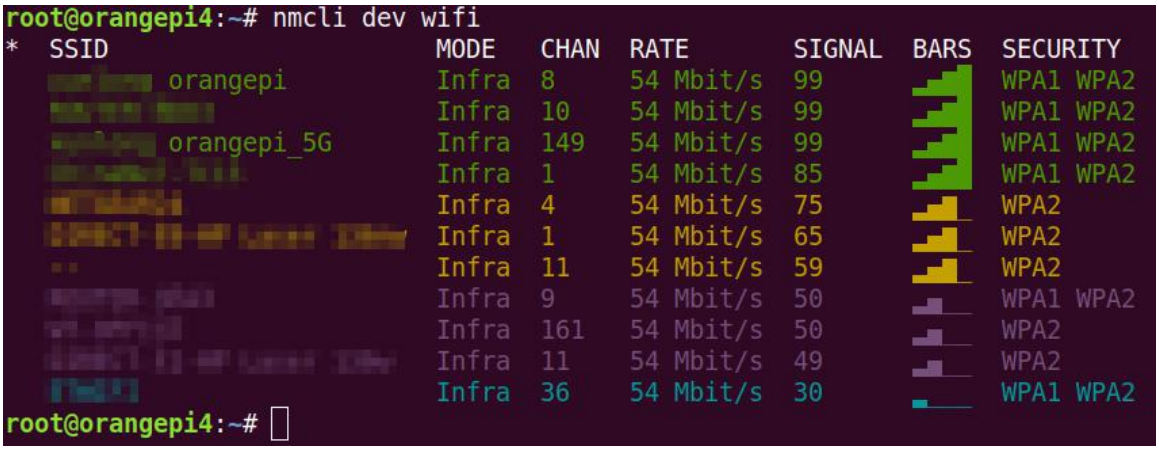
a. If the development board is connected to the network cable, you can log in to the Linux system remotely via SSH

a. If the debug serial port is connected to the development board, you can use the serial terminal to log in to the Linux system

b. If you connect the development board to the HDMI display, you can log in to the Linux system through the HDMI display terminal

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

```
root@orangepi4:~# nmcli dev wifi
```



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

a. wifi_name needs to be replaced with the name of the WIFI hotspot you want to connect to

b. wifi_passwd needs to be replaced with the password of the WIFI hotspot you want to connect to

```
root@orangepi4:~# nmcli dev wifi connect wifi_name password wifi_passwd
Device 'wlan0' successfully activated with '36387224-f4ff-4021-85a1-eda7825ce09e'.
```

3) The wifi IP address can be viewed through the ifconfig command

```
root@orangepi4:~# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr b0:02:47:cf:28:77
          inet addr:192.168.1.187  Bcast:192.168.1.255  Mask:255.255.255.0
```

```
          inet6 addr: fe80::8008:703d:8f92:41c7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17 errors:0 dropped:0 overruns:0 frame:0
          TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1837 (1.8 KB)  TX bytes:8320 (8.3 KB)
```

4) Use the ping command to test the connectivity of the wifi network

```
root@orangepi4:~# ping www.baidu.com -I wlan0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.187 wlan0: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=15.0 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=56 time=12.8 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=5 ttl=56 time=13.7 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=6 ttl=56 time=14.9 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=7 ttl=56 time=13.6 ms
^C
```

3.11.2. Connect to WIFI graphically in the command line

1) Log in to the linux system first, there are three ways

a. If the development board is connected to the network cable, you can log in to the Linux system remotely via SSH

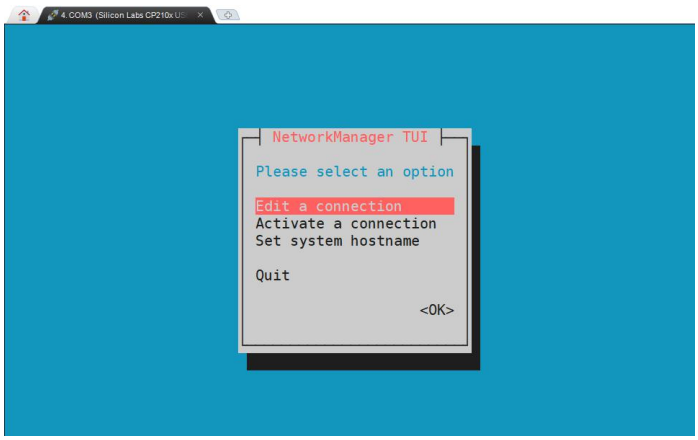
b. If the debugging serial port is connected to the development board, you can use the serial terminal to log in to the Linux system (for serial port software, please use MobaXterm, and minicom cannot display the graphical interface)

c. If the development board is connected to the HDMI display, you can log in to the linux system through the HDMI display terminal

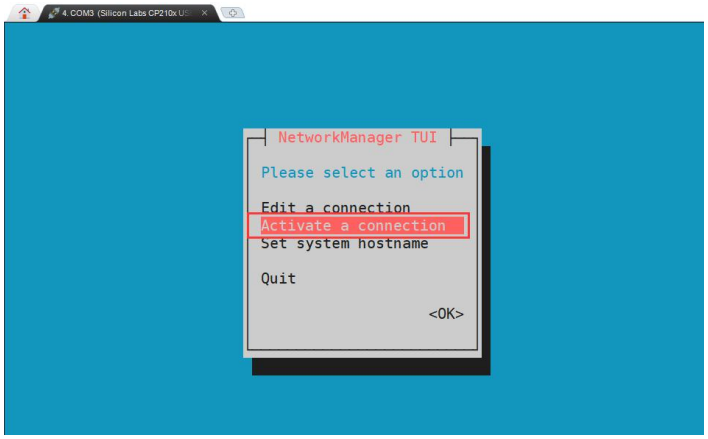
2) Then enter the nmtui command in the command line to open the wifi connection interface

```
root@orangepi4:~# nmtui
```

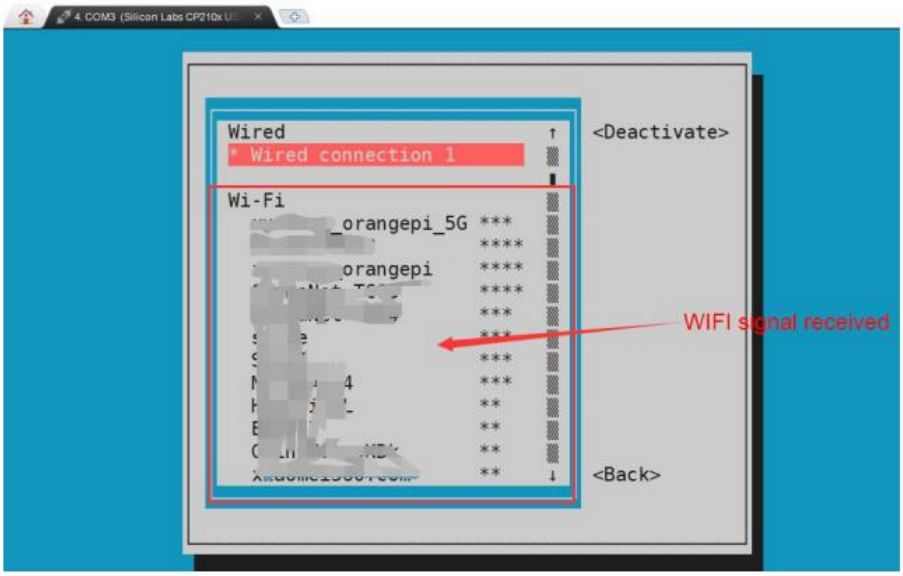
3) Enter the nmtui command to open the interface as shown below



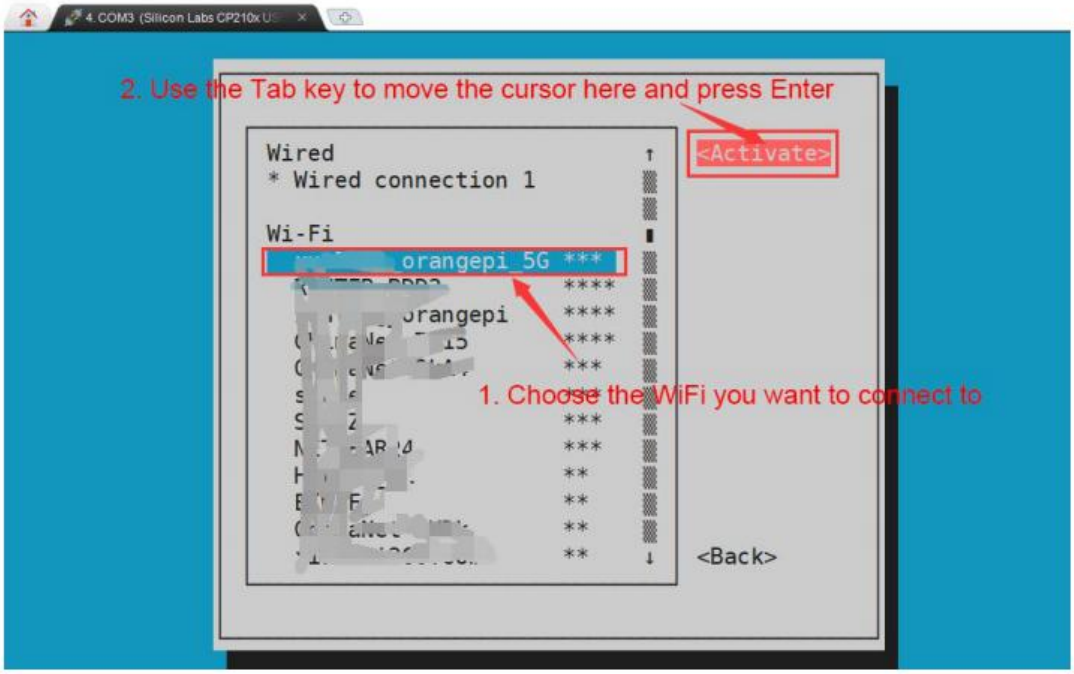
4) Select Activate a connect and press Enter



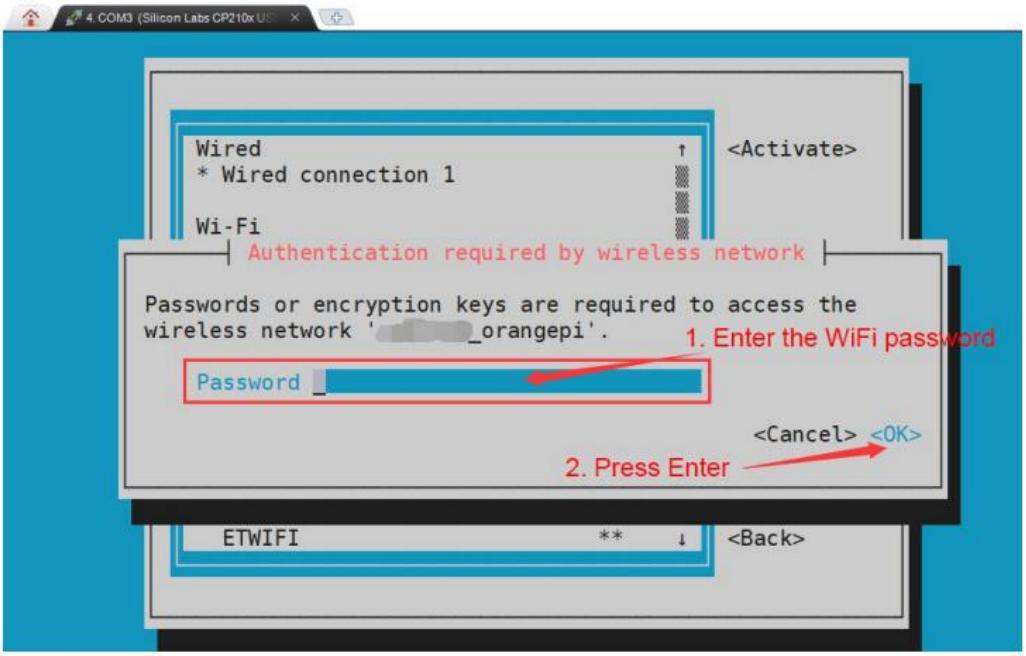
5) Then you can see all the searched WIFI hotspots



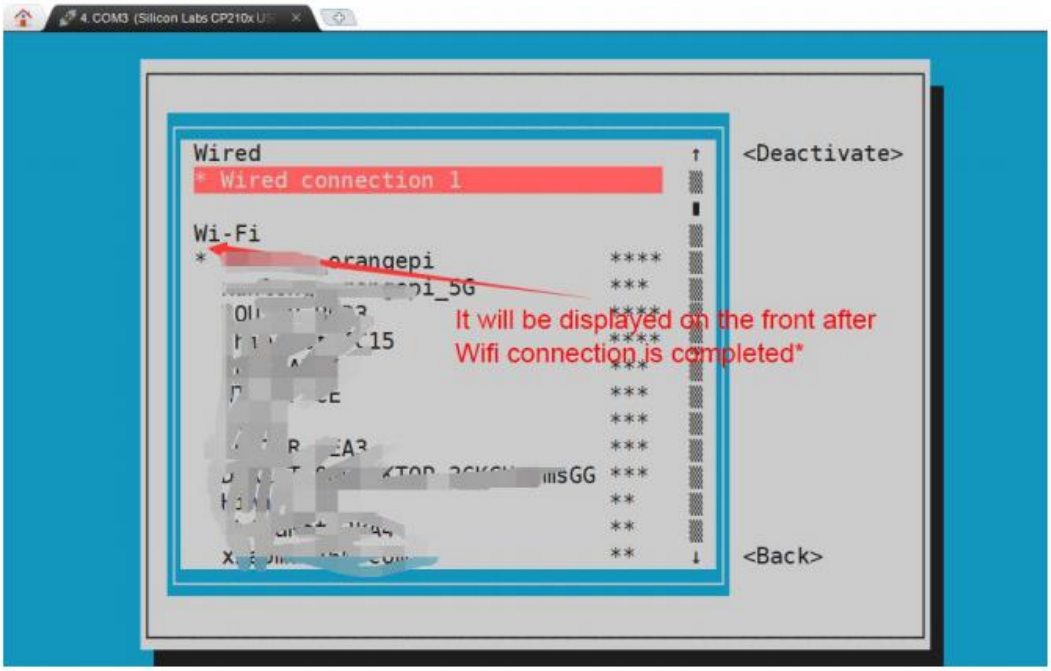
6) Select the WIFI hotspot you want to connect to, then use the Tab key to position the cursor on Activate and press Enter



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



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



9) The wifi IP address can be viewed through the ifconfig command

```
root@orangepi4:~# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.187  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::76bb:f67d:ef98:2f9a  prefixlen 64  scopeid 0x20<link>
        ether 12:81:3e:a8:58:d8  txqueuelen 1000  (Ethernet)
        RX packets 185  bytes 109447 (109.4 KB)
        RX errors 0  dropped 61  overruns 0  frame 0
        TX packets 27  bytes 14783 (14.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

10) Use the ping command to test the connectivity of the wifi network

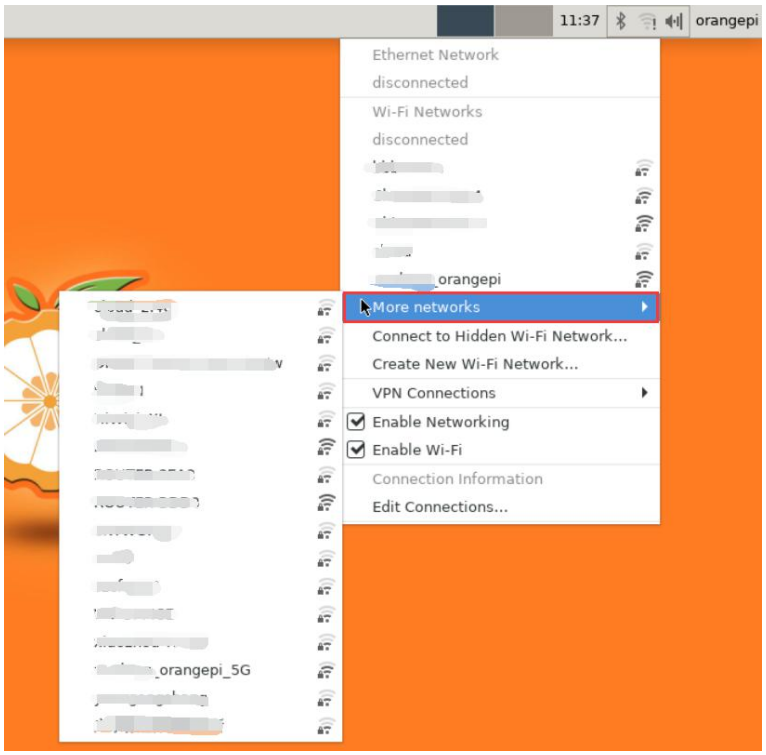
```
root@orangepi4:~# ping www.baidu.com -I wlan0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.187 wlan0: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=15.0 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=56 time=12.8 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=5 ttl=56 time=13.7 ms
```

```
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=6 ttl=56 time=14.9 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=7 ttl=56 time=13.6 ms
^C
```

3. 11. 3. How to use Linux desktop to connect to WIFI

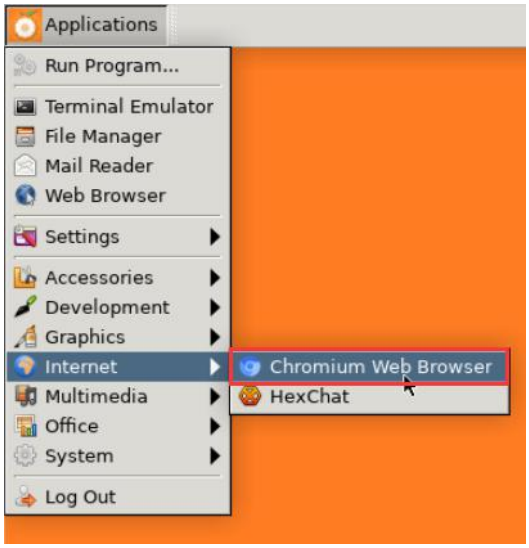
1) Click the network configuration icon in the upper right corner of the desktop (please do not connect the network cable when testing WIFI)

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

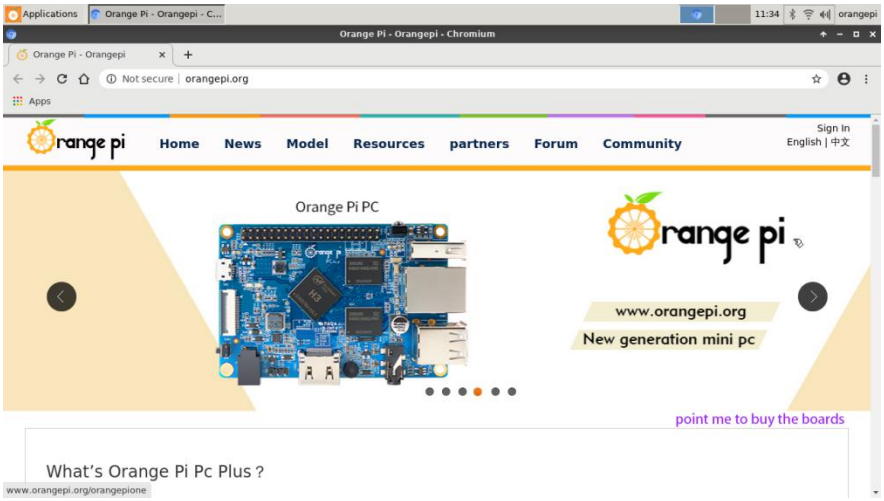


3) Then enter the password of the WIFI hotspot, and then click Connect to start connecting to the WIFI

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



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

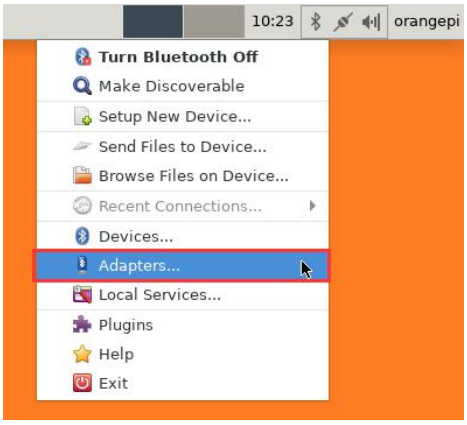


3. 12. How to use Bluetooth

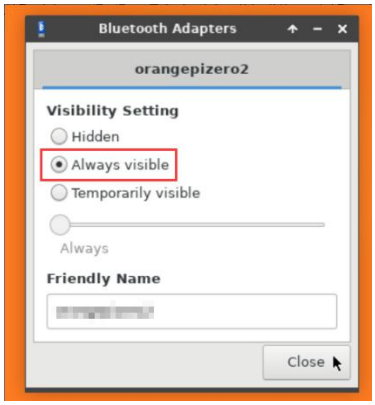
3. 12. 1. How to connect to Bluetooth via desktop

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

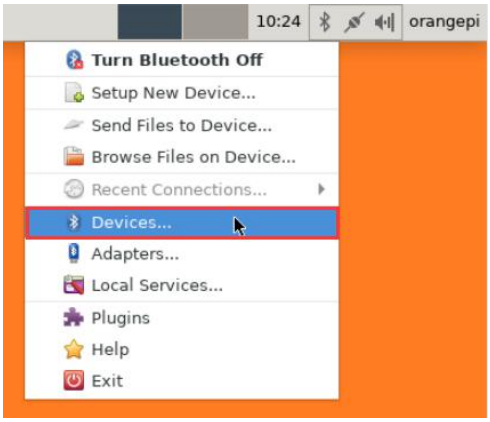
2) Then select the adapter



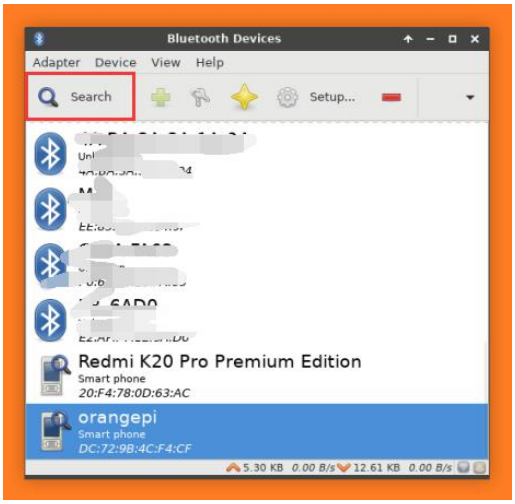
3) Set Visibility Setting to Always visible in the Bluetooth adapter setting interface, and then click close to close



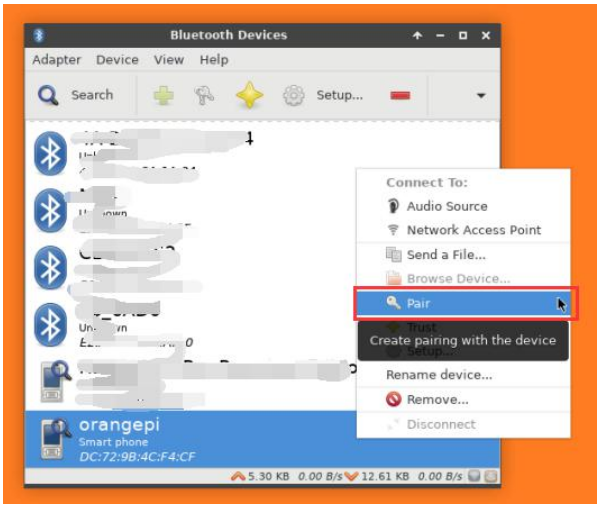
4) Then open the configuration interface of the Bluetooth device



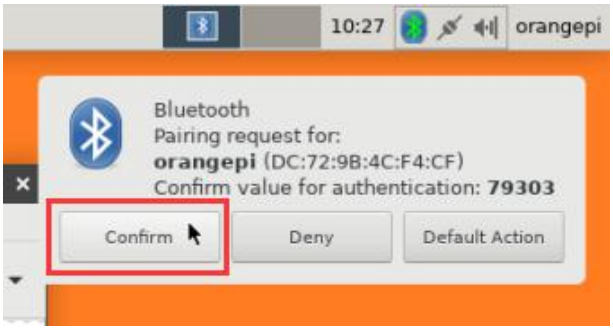
5) Click Search to start scanning the surrounding Bluetooth devices



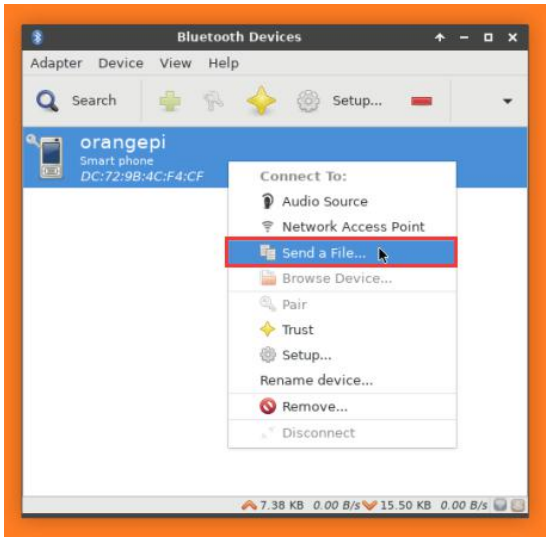
6) Select the Bluetooth device you want to connect to, and then click the right mouse button to pop up the operation interface of the Bluetooth device. Select Pair to start pairing. The demonstration here is pairing with an Android phone



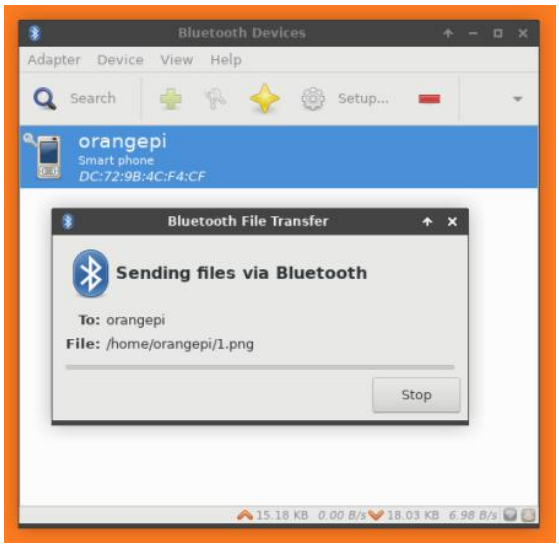
7) When pairing, a pairing confirmation box will pop up in the upper right corner of the desktop, select Confirm to confirm, and the phone also needs to be confirmed at this time



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



9) The interface for sending pictures is shown below



3. 12. 2. How to connect to Bluetooth using commands

1) When the Linux system starts, it will run the rk3399-bluetooth.service service to initialize the Bluetooth device. After entering the system, you can use the hciconfig command to check whether there is a Bluetooth device node. If it exists, the Bluetooth initialization is normal

```
root@orangepi4:~# hciconfig -a
hci0:   Type: BR/EDR   Bus: UART
```

```
    BD Address: 43:45:C5:00:1F:AC    ACL MTU: 1021:8    SCO MTU: 64:1
    UP RUNNING PSCAN ISCAN
    RX bytes:897 acl:0 sco:0 events:65 errors:0
    TX bytes:4355 acl:0 sco:0 commands:65 errors:0
    Features: 0xbf 0xfe 0xcf 0xfe 0xdb 0xff 0x7b 0x87
    Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
    Link policy: RSWITCH SNIFF
    Link mode: SLAVE ACCEPT
    Name: 'orangepi4'
    Class: 0x1c0000
    Service Classes: Rendering, Capturing, Object Transfer
    Device Class: Miscellaneous,
    HCI Version:    (0x9)    Revision: 0x26
    LMP Version:    (0x9)    Subversion: 0x6606
    Manufacturer: Broadcom Corporation (15)
```

2) Use bluetoothctl to scan for Bluetooth devices

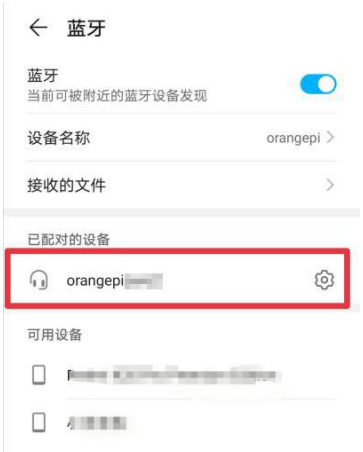
```
root@orangepi4:~# bluetoothctl
[NEW] Controller 10:11:12:13:14:15 orangepi3 [default]
Agent registered
[bluetooth]# power on            # Enable controller
Changing power on succeeded
[bluetooth]# discoverable on       # Set the controller to be discoverable
Changing discoverable on succeeded
[CHG] 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 Xiaomi mobile phone
[NEW] Device DC:72:9B:4C:F4:CF orangepi
[bluetooth]# scan off           #After scanning the Bluetooth device you want to connect,
you can turn off the scan, and then write down the MAC address of the Bluetooth
device. The Bluetooth device tested here is an Android phone, the name of the
```

```
Bluetooth is orangepi, and the corresponding MAC address is DC:72:9B:4C :F4:CF
Discovery stopped
[CHG] Controller 10:11:12:13:14:15 Discovering: no
[CHG] Device DC:72:9B:4C:F4:CF RSSI is nil
```

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

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

4) After the pairing is successful, the bluetooth interface of the mobile phone will be displayed as shown below



3. 13. On-board LED light display description

	Green light	red light
u-boot startup phase	Turn off	Bright
Kernel boot to enter the system	Flashing	Bright

3. 14. USB interface test

3. 14. 1. Connect mouse or keyboard test

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

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

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

3. 14. 2. Connect USB storage device test

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

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

```
root@orangepi4:~# 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

```
root@orangepi4:~# mount /dev/sda1 /mnt/
root@orangepi4:~# ls /mnt/
test.txt
```

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

```
root@orangepi4:~# df -h | grep "sd"
/dev/sda1               29G  208K   29G   1% /mnt
```

3.15. USB camera test

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

2) Through the v4l2-ctl (note that the l in v4l2 is a lowercase letter l, not a number 1) command, you can see that the device node information of the USB camera is /dev/video0

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install v4l-utils
root@orangepi4:~# v4l2-ctl    --list-devices
USB2.0 Camera RGB (usb-xhci-hcd.11.auto-1):
    /dev/video10
```

3) Use fswebcam to test the USB camera

a. Install fswebcam

```
root@orangepi4:~# apt update
root@orangepi4:~# apt-get install fswebcam
```

b. After installing fswebcam, you can use the following command to take pictures
   a) The -d option is used to specify the device node of the USB camera
   b) --no-banner is used to remove the watermark of the photo
   c) -r option is used to specify the resolution of the photo
   d) -S option is used to skip the previous frame number
   e) ./image.jpg is used to set the name and path of the generated photo

```
root@orangepi4:~# fswebcam -d /dev/video10 --no-banner -r 1280x720 -S 5 ./image.jpg
```

c. If there is no HDMI monitor or LCD screen connected, you can use the scp command to transfer the taken pictures to the Ubuntu PC for image after taking the picture.

```
root@orangepi4:~# scp image.jpg test@192.168.1.55:/home/test （Modify the IP address and path according to the actual situation）
```

d. If an HDMI display or LCD screen is connected, you can directly view the captured pictures through the HDMI display or LCD screen

4) Use motion to test the USB camera

a. Install the camera test software motion

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install motion
```

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

```
root@orangepi4:~# sed -i "s/start_motion_daemon=no/start_motion_daemon=yes/" \
/etc/default/motion       (This is a command)
```

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

```
root@orangepi4:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf       (This is a command)
```

d. Modify the configuration of /etc/motion/motion.conf, modify videodevice /dev/video0 to videodevice /dev/video10

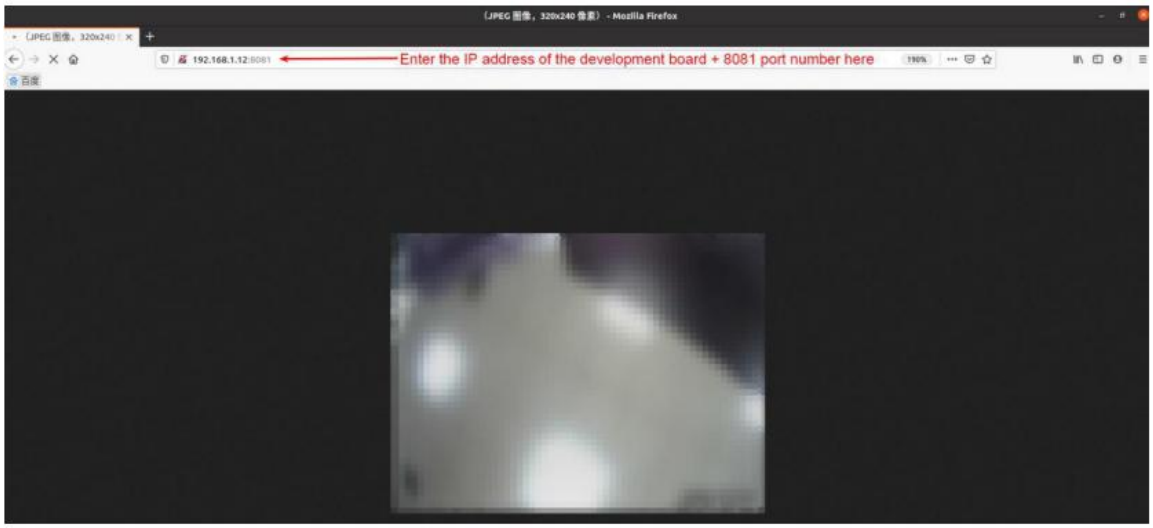
```
root@orangepi4:~# sed -i "s/videodevice \/dev\/video0/videodevice \/dev\/video10/" \
/etc/motion/motion.conf
```

e. Then restart the motion service

```
root@orangepi4:~# /etc/init.d/motion restart
[ ok ] Restarting motion (via systemctl): motion.service.
```

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

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



5) Use mjpg-streamer to test the USB camera

a. Download mjpg-streamer

```
root@orangepi4:~# git clone https://github.com/jacksonliam/mjpg-streamer
```

b. Install dependent packages (under debian10, you need to replace libjpeg8-dev with libjpeg62-turbo-dev)

```
root@orangepi4:~# apt-get install cmake libjpeg8-dev
```

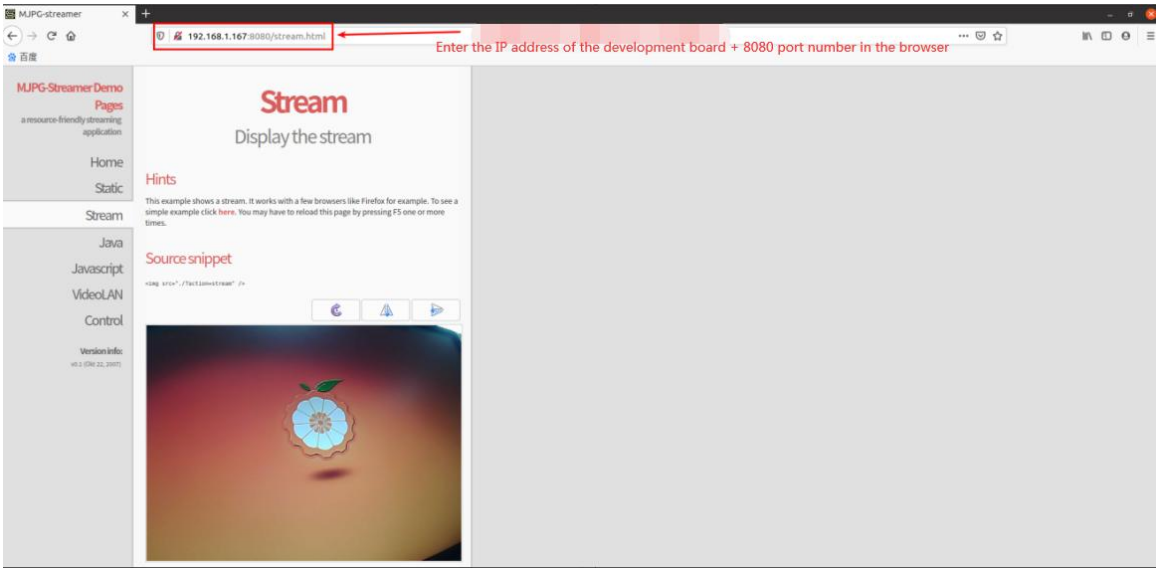
c. Compile and install mjpg-streamer

```
root@orangepi4:~# cd mjpg-streamer/mjpg-streamer-experimental
root@orangepi4:~/mjpg-streamer/mjpg-streamer-experimental# make
root@orangepi4:~/mjpg-streamer/mjpg-streamer-experimental# make install
```

d. Then enter the following command to start mjpg_streamer

```
root@orangepi4:~/mjpg-streamer/mjpg-streamer-experimental# export
LD_LIBRARY_PATH=.          （This is a command）
root@orangepi4:~/mjpg-streamer/mjpg-streamer-experimental# ./mjpg_streamer -i
"./input_uvc.so -d /dev/video0 -u -f 30" -o "./output_http.so -w ./www"
```

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



f. It is recommended to use mjpg-streamer to test the USB camera, it is much smoother than motion, and you can't feel any lag when using mjpg-streamer

3. 16. Audio test

3. 16. 1. Headphone jack play audio test

1) First insert the headset into the audio interface

2) Through the aplay -l command, you can view the sound card devices supported by the linux system, among which realtekrt5651co is the sound card device required for headset playback

```
root@orangepi4:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: realtekrt5651co [realtek,rt5651-codec], device 0: ff890000.i2s-rt5651-aif1 rt5651-aif1-0 []
  Subdevices: 0/1
  Subdevice #0: subdevice #0
card 1: rkhdmidpsound [rk-hdmi-dp-sound], device 0: HDMI-DP multicodec-0 []
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

3) Upload the audio files that need to be played to the /root folder of the Linux system. You can use the scp command to upload in the Ubuntu PC (the IP address in the

command is the IP address of the Orange Pi development board), or copy it with a USB flash drive

```
test@test:~/AudioTest$ scp audio.wav root@192.168.1.xx:/root （Modify the IP address and path according to the actual situation）
```

4) Then use the aplay command to play the audio, the earphone can hear the sound

```
root@orangepi4:~# aplay -D hw:0,0 audio.wav
Playing WAVE 'audio.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

3. 16. 2. Onboard MIC recording test

1) First make sure that the system has opened the recording channel below

```
root@orangepi4:~# amixer cset name="RECMIXL INL1 Switch" on
root@orangepi4:~# mixer cset name="RECMIXL BST2 Switch" on
root@orangepi4:~# amixer cset name="RECMIXR INR1 Switch" on
root@orangepi4:~# amixer cset name="RECMIXR BST2 Switch" on
root@orangepi4:~# amixer cset name="RECMIXL BST3 Switch" off
root@orangepi4:~# amixer cset name="RECMIXR BST3 Switch" off
```

2) The recording commands are as follows

```
root@orangepi4:~# arecord -D hw:0,0 -d 5 -f cd -t wav test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

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

```
root@orangepi4:~# ls -lh
total 862K
-rw-r--r-- 1 root root 862K Feb  5 04:24 test.wav
root@orangepi4:~# aplay -D hw:0,0 test.wav
```

3. 16. 3. Headphone recording

1) First insert the earphone with recording function

2) Then use amixer to open the recording channel below

```
root@orangepi4:~# amixer cset name="RECMIXL INL1 Switch" off
```

```
root@orangepi4:~# mixer cset name="RECMIXL BST2 Switch" off
root@orangepi4:~# amixer cset name="RECMIXR INR1 Switch" off
root@orangepi4:~# amixer cset name="RECMIXR BST2 Switch" off
root@orangepi4:~# amixer cset name="RECMIXL BST3 Switch" on
root@orangepi4:~# amixer cset name="RECMIXR BST3 Switch" on
```

3) The recording commands are as follows

```
root@orangepi4:~# arecord -D hw:0,0 -d 5 -f cd -t wav test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

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

```
root@orangepi4:~# ls -lh
total 862K
-rw-r--r-- 1 root root 862K Feb  5 04:24 test.wav
root@orangepi4:~# aplay -D hw:0,0 test.wav
```

3.16.4. HDMI audio playback test

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

2) Upload the audio files that need to be played to the /root folder of the Linux system. You can use the scp command to upload in the Ubuntu PC (the IP address in the command is the IP address of the Orange Pi development board), or copy it with a USB flash drive

```
test@test:~/AudioTest$ scp audio.wav root@192.168.1.xx:/root （Modify the IP address and path according to the actual situation）
```

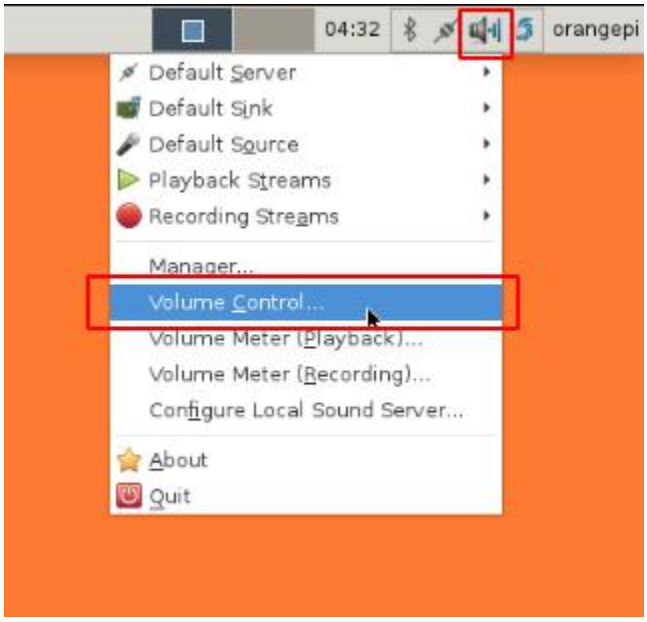
3) HDMI audio playback does not require other settings, just use the aplay command to play directly

```
root@orangepi4:~# aplay -D hw:1,0 audio.wav
```

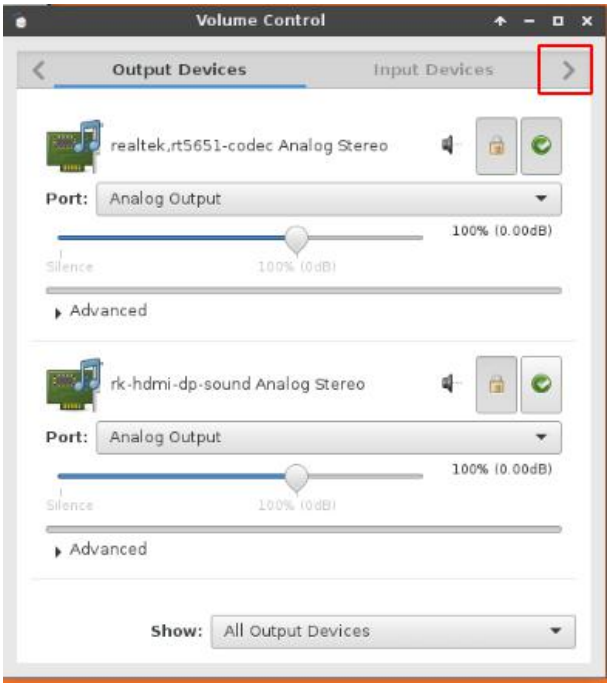
3.16.5. How to replace the audio playback source with desktop image

1) Use the player to play an audio or video on the desktop, click the speaker icon in the

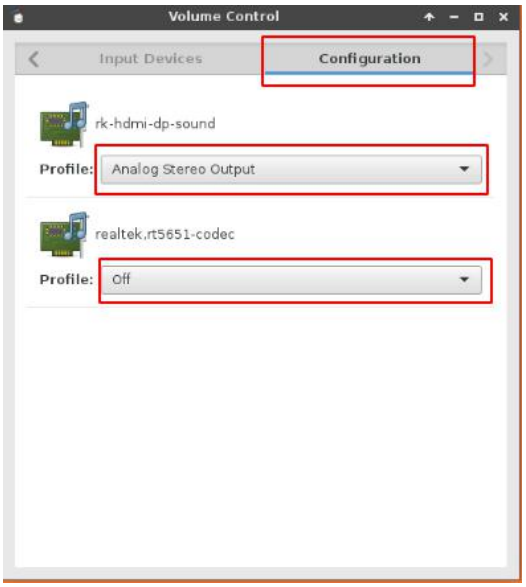
upper right corner to select "Volume Control"



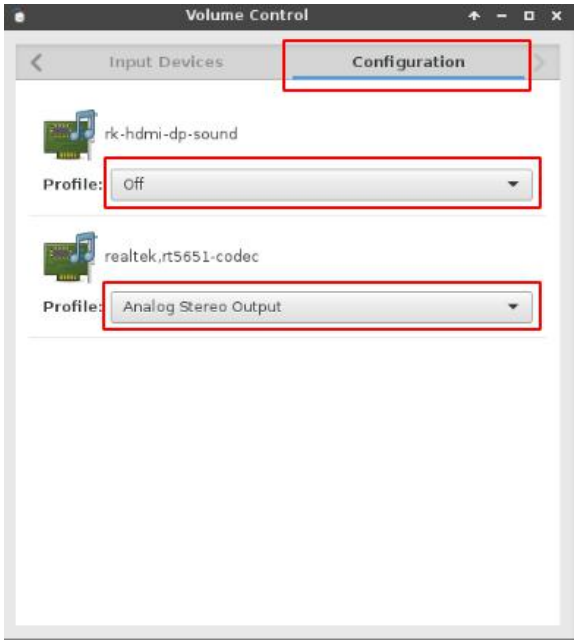
5) Click the arrow in the figure below to switch to the Configuration column



6) Modify the configuration as shown below, and the sound is output from HDMI



7) Modify the configuration as shown in the figure below, and the sound is output from the earphone



3. 17. Temperature sensor

1) RK3399 has a total of 2 temperature sensors, the command to check the temperature is as follows

a. sensor0：CPU

```
root@orangepi4:~# cat /sys/class/thermal/thermal_zone0/type
soc-thermal
root@orangepi4:~# cat /sys/class/thermal/thermal_zone0/temp
48125
```

b. sensor1：GPU

```
root@orangepi4:~# cat /sys/class/thermal/thermal_zone1/type
gpu-thermal
root@orangepi4:~# cat /sys/class/thermal/thermal_zone1/temp
49375
```

3.18. How to install Docker

1) Uninstall the old version of docker that may exist first

```
root@orangepi4:~# apt remove docker docker-engine docker-ce docker.io
```

2) Then install the following packages

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install -y apt-transport-https ca-certificates curl
  software-properties-common          (This is a command)
```

3) Add the key of Alibaba Cloud docker

```
root@orangepi4:~# curl -fsSL http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg
  | sudo apt-key add    -                (This is a command)
```

4) Add the corresponding docker source in the system source of ubuntu

```
root@orangepi4:~# add-apt-repository "deb [arch=arm64]
    https://mirrors.aliyun.com/docker-ce/linux/ubuntu $(lsb_release -cs) stable"
  (This is a command)
```

5) Install the latest version of docker-ce

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install docker-ce
```

6) Verify the status of docker

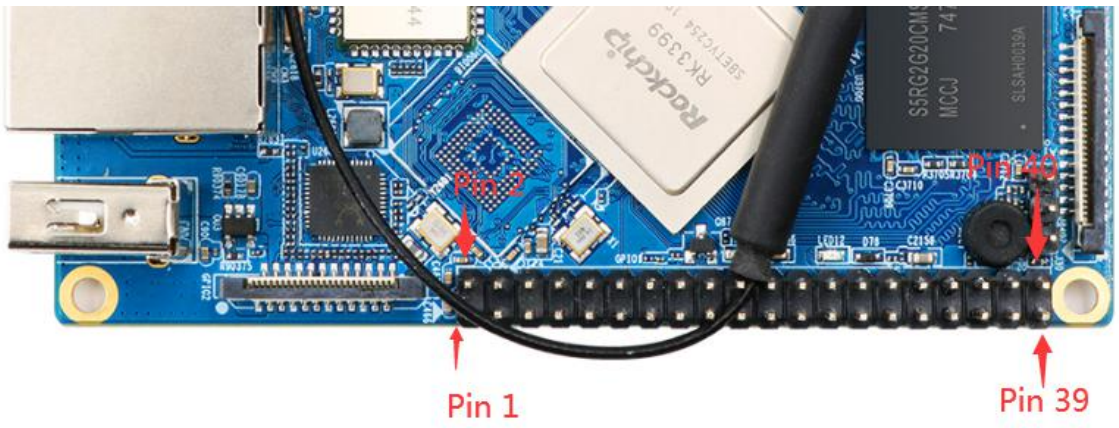
```
root@orangepi4:~# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2021-02-04 08:25:14 UTC; 1min 12s ago
       Docs: https://docs.docker.com
 Main PID: 11155 (dockerd)
      Tasks: 14
     Memory: 39.1M
        CPU: 1.339s
     CGroup: /system.slice/docker.service
             └─11155 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

7) Test docker

```
root@orangepi4:~# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
256ab8fe8778: Pull complete
Digest: sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c2ff1ca09b96391d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

3. 19. 40pin interface pin description

1) Please refer to the figure below for the sequence of the 40 pin interface pins on the Orange Pi 4 development board



2) The functions of the 40 pin interface pins of the Orange Pi 4 development board are shown in the following table

GPIO No.	GPIO	Function	PIN
		3.3V	1
64	GPIO2_A0	I2C2_SDA	3
65	GPIO2_A1	I2C2_SCL	5
150	GPIO4_C6	PWM1	7
		GND	9
33	GPIO1_A1	GPIO1_A1	11
35	GPIO1_A3	GPIO1_A3	13
92	GPIO2_D4	GPIO2_D4	15
		3.3V	17
40	GPIO1_B0	SPI1_TXD	19
39	GPIO1_A7	SPI1_RXD	21
41	GPIO1_B1	SPI1_CLK	23
		GND	25
64	GPIO2_A0	I2C2_SDA	27
121	GPIO3_D1	I2S0_RX	29
122	GPIO3_D2	I2S0_TX	31
120	GPIO3_D0	I2S0_SCK	33
123	GPIO3_D3	I2S0_SI0	35
124	GPIO3_D4	I2S0_SI1	37
		GND	39

PIN	Function	GPIO	GPIONo.
2	5V		
4	5V		
6	GND		
8	I2C3_SCL	GPIO4_C1	145
10	I2C3_SDA	GPIO4_C0	144
12	GPIO1_C2	GPIO1_C2	50
14	GND		
16	GPIO1_C6	GPIO1_C6	54
18	GPIO1_C7	GPIO1_C7	55
20	GND		
22	GPIO1_D0	GPIO1_D0	56
24	SPI1_CS	GPIO1_B2	42
26	GPIO4_C5	GPIO4_C5	149
28	I2C2_SCL	GPIO2_A1	65
30	GND		
32	I2S_CLK	GPIO4_A0	128
34	GND		
36	I2S0_SO0	GPIO3_D7	127
38	I2S0_SI2	GPIO3_D5	125
40	I2S0_SI3	GPIO3_D6	126

3. 20. How to install wiringOP

1) Download the code of wiringOP

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install git
root@orangepi4:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

2) Compile wiringOP

```
root@orangepi4:~# cd wiringOP
root@orangepi4:~/wiringOP# ./build clean
root@orangepi4:~/wiringOP# ./build
```

3) The output of the test gpio readall command is as follows, where the physical pins from 1 to 40 correspond to the 40Pin pins on the development board one-to-one

```
root@orangepi4:~# gpio readall
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   64 |   0 | I2C2_SDA | ALT3 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   65 |   1 | I2C2_SCL | ALT3 | 0 |  5 || 6  |   |      | GND      |     |      |
 |  150 |   2 |     PWM1 |   IN | 0 |  7 || 8  | 1 | ALT2 | I2C3_SCL | 3   | 145  |
 |      |     |      GND |      |   |  9 || 10 | 1 | ALT2 | I2C3_SDA | 4   | 144  |
 |   33 |   5 | GPIO1_A1 |   IN | 0 | 11 || 12 | 1 | IN   | GPIO1_C2 | 6   | 50   |
 |   35 |   7 | GPIO1_A3 |  OUT | 1 | 13 || 14 |   |      | GND      |     |      |
 |   92 |   8 | GPIO2_D4 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO1_C6 | 9   | 54   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | IN   | GPIO1_C7 | 10  | 55   |
 |   40 |  11 | SPI1_TXD | ALT3 | 0 | 19 || 20 |   |      | GND      |     |      |
 |   39 |  12 | SPI1_RXD | ALT3 | 1 | 21 || 22 | 0 | IN   | GPIO1_D0 | 13  | 56   |
 |   41 |  14 | SPI1_CLK | ALT3 | 1 | 23 || 24 | 1 | ALT3 | SPI1_CS  | 15  | 42   |
 |      |     |      GND |      |   | 25 || 26 | 0 | IN   | GPIO4_C5 | 16  | 149  |
 |   64 |  17 | I2C2_SDA | ALT3 | 0 | 27 || 28 | 0 | ALT3 | I2C2_SCL | 18  | 65   |
 |      |     |  I2S0_RX |      |   | 29 || 30 |   |      | GND      |     |      |
 |      |     |  I2S0_TX |      |   | 31 || 32 |   |      | I2S_CLK  |     |      |
 |      |     | I2S0_SCK |      |   | 33 || 34 |   |      | GND      |     |      |
 |      |     | I2S0_SI0 |      |   | 35 || 36 |   |      | I2S0_SO0 |     |      |
 |      |     | I2S0_SI1 |      |   | 37 || 38 |   |      | I2S0_SI2 |     |      |
 |      |     |      GND |      |   | 39 || 40 |   |      | I2S0_SI3 |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
```

3. 21. 40pin interface GPIO, I2C, UART, SPI, PWM test

wiringOP has been adapted to the Orange Pi 4 development board, using wiringOP can test the functions of GPIO, I2C, UART, and SPI

3. 21. 1. 40pin GPIO port test

1) Below, take pin 11-corresponding to GPIO as GPIO1_A1-corresponding to wPi serial number as 5-as an example to demonstrate how to set the high and low levels of the GPIO port

```
root@orangepi4:~# gpio readall
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   64 |   0 | I2C2_SDA | ALT3 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   65 |   1 | I2C2_SCL | ALT3 | 0 |  5 || 6  |   |      | GND      |     |      |
 |  150 |   2 |     PWM1 |   IN | 0 |  7 || 8  | 1 | ALT2 | I2C3_SCL | 3   | 145  |
 |      |     |      GND |      |   |  9 || 10 | 1 | ALT2 | I2C3_SDA | 4   | 144  |
 |   33 |   5 | GPIO1_A1 |   IN | 0 | 11 || 12 | 1 | IN   | GPIO1_C2 | 6   | 50   |
 |   35 |   7 | GPIO1_A3 |  OUT | 1 | 13 || 14 |   |      | GND      |     |      |
 |   92 |   8 | GPIO2_D4 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO1_C6 | 9   | 54   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | IN   | GPIO1_C7 | 10  | 55   |
```

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

```
root@orangepi4:~/wiringOP# gpio mode 5 out
```

Use gpio readall to see that the mode of pin 11 has changed to out

```
root@orangepi4:~# gpio readall
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   64 |   0 | I2C2_SDA | ALT3 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   65 |   1 | I2C2_SCL | ALT3 | 0 |  5 || 6  |   |      | GND      |     |      |
 |  150 |   2 |     PWM1 |   IN | 0 |  7 || 8  | 1 | ALT2 | I2C3_SCL | 3   | 145  |
 |      |     |      GND |      |   |  9 || 10 | 1 | ALT2 | I2C3_SDA | 4   | 144  |
 |   33 |   5 | GPIO1_A1 |  OUT | 0 | 11 || 12 | 1 | IN   | GPIO1_C2 | 6   | 50   |
 |   35 |   7 | GPIO1_A3 |  OUT | 1 | 13 || 14 |   |      | GND      |     |      |
 |   92 |   8 | GPIO2_D4 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO1_C6 | 9   | 54   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | IN   | GPIO1_C7 | 10  | 55   |
```

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

```
root@orangepi4:~/wiringOP# gpio write 5 0
```

Use gpio readall to see that the value (V) of pin 11 has become 0

```
root@orangepi4:~# gpio readall
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   64 |   0 | I2C2_SDA | ALT3 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   65 |   1 | I2C2_SCL | ALT3 | 0 |  5 || 6  |   |      | GND      |     |      |
 |  150 |   2 |     PWM1 |   IN | 0 |  7 || 8  | 1 | ALT2 | I2C3_SCL | 3   | 145  |
 |      |     |      GND |      |   |  9 || 10 | 1 | ALT2 | I2C3_SDA | 4   | 144  |
 |   33 |   5 | GPIO1_A1 |  OUT | 0 | 11 || 12 | 1 | IN   | GPIO1_C2 | 6   | 50   |
 |   35 |   7 | GPIO1_A3 |  OUT | 1 | 13 || 14 |   |      | GND      |     |      |
 |   92 |   8 | GPIO2_D4 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO1_C6 | 9   | 54   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | IN   | GPIO1_C7 | 10  | 55   |
```

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

```
root@orangepi4:~/wiringOP# gpio write 5 1
```

Using gpio readall, you can see that the value (V) of pin 11 has become 1

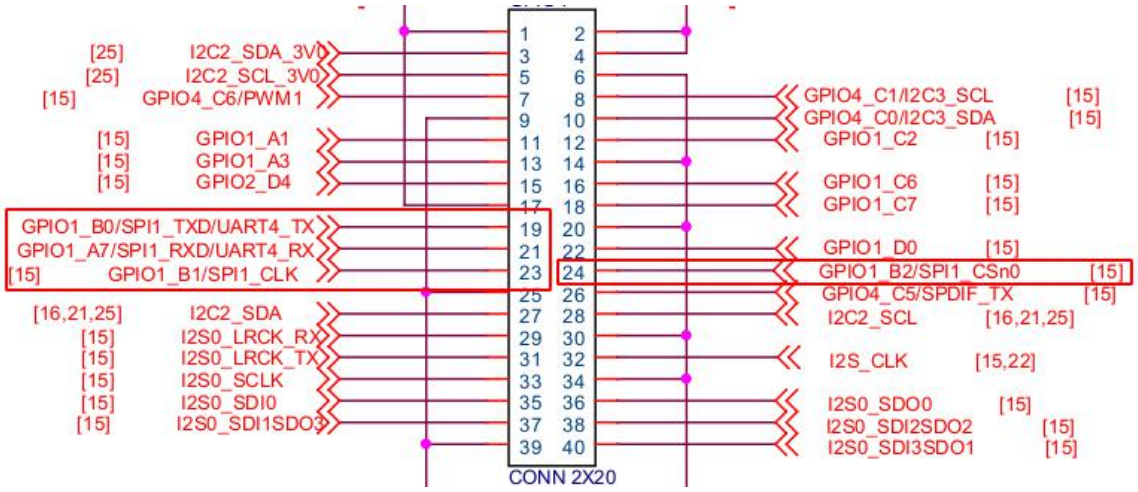
```
root@orangepi4:~# gpio readall
 +------+-----+----------+------+---+OrangePi 4+---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   64 |   0 | I2C2_SDA | ALT3 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   65 |   1 | I2C2_SCL | ALT3 | 0 |  5 || 6  |   |      | GND      |     |      |
 |  150 |   2 |     PWM1 |   IN | 0 |  7 || 8  | 1 | ALT2 | I2C3_SCL | 3   | 145  |
 |      |     |      GND |      |   |  9 || 10 | 1 | ALT2 | I2C3_SDA | 4   | 144  |
 |   33 |   5 | GPIO1_A1 |  OUT | 1 | 11 || 12 | 1 | IN   | GPIO1_C2 | 6   | 50   |
 |   35 |   7 | GPIO1_A3 |  OUT | 1 | 13 || 14 |   |      | GND      |     |      |
 |   92 |   8 | GPIO2_D4 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO1_C6 | 9   | 54   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | IN   | GPIO1_C7 | 10  | 55   |
```

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

3.21.2. 40pin SPI test

1) In the Linux4.4 system, the spi controller in 40pin is turned on by default in dts, and no additional configuration is required

a. According to the schematic diagram of 40pin, the available spi of the development board is spi1



b. After the system starts, you can see the SPI device node under /dev

```
root@orangepi4:~# ls /dev/spi*
/dev/spidev1.0
```

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

```
root@orangepi4:~/wiringOP/examples# make spidev_test
[CC] spidev_test.c
[link]
```

3) Do not short-circuit the txd and rxd pins of SPI1 first, and the output result of running spidev_test is as follows, you can see that the data of TX and RX are inconsistent

```
root@orangepi4:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF 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    | ................................
```

4) Then short-circuit the txd (pin 19 in the 40pin interface) and rxd (pin 21 in the 40pin interface) of SPI1 and run the output of spidev_test as follows, you can see the sending and receiving Same data

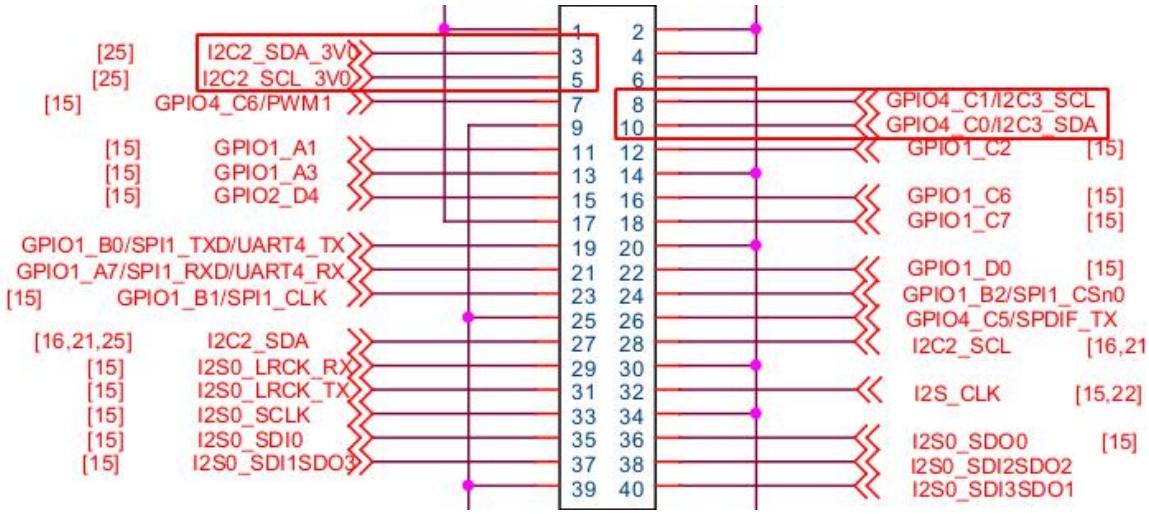
```
root@orangepi4:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
```

```
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF 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. 21. 3. 40pin I2C test

1) The Linux4.4 system turns on the i2c controller in 40pin by default in dts, and the i2c controller can be used without additional configuration

a. According to the 40pin schematic diagram, the available i2c on the development board are i2c2 and i2c3. The data and clock pins of i2c2 can be connected to pins 3 and 5, or pins 27 and 28, but 3 and the output voltage of pin 5 is 3v, the output voltage of pin 27 and 28 is 1.8v



b. After the system starts, you can see the following multiple i2c device nodes under /dev

```
root@orangepi4:~# ls /dev/i2c*
/dev/i2c-0  /dev/i2c-1  /dev/i2c-2  /dev/i2c-3  /dev/i2c-4  /dev/i2c-7  /dev/i2c-9
```

c. The corresponding relationship of different i2c device nodes is shown below, where

a) i2c2 in 40pin corresponds to /dev/i2c-2

b) i2c3 in 40pin corresponds to /dev/i2c-3

```
root@orangepi4:~# ls /sys/class/i2c-adapter/i2c-* -lh
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-0 -> ../../devices/platform/ff3c0000.i2c/i2c-0
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-1 -> ../../devices/platform/ff110000.i2c/i2c-1
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-2 -> ../../devices/platform/ff120000.i2c/i2c-2
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-3 -> ../../devices/platform/ff130000.i2c/i2c-3
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-4 -> ../../devices/platform/ff3d0000.i2c/i2c-4
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-7 -> ../../devices/platform/ff160000.i2c/i2c-7
lrwxrwxrwx 1 root root 0 Jan 18  2013 /sys/class/i2c-adapter/i2c-9 -> ../../devices/platform/fec00000.dp/i2c-9
```

2) Then start to test i2c, first install i2c-tools

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install i2c-tools
```

3) Then connect an i2c device to the i2c2 pin of the 40pin connector (i2c3 test is the same as i2c2, just connect the device to the pin of i2c3, take i2c2 as an example below)

	i2c2 (3v voltage)	i2c2 (1.8v voltage)	i2c3
Sda Pin	Corresponding to pin 3	Corresponding to pin 27	Corresponding to pin 10
Sck Pin	Corresponding to pin 5	Corresponding to pin 28	Corresponding to pin 8
Vcc Pin	Corresponding to pin 1	Corresponding to pin 1	Corresponding to pin 1
Gnd Pin	Corresponding to pin 6	Corresponding to pin 6	Corresponding to pin 6

4) Then use the i2cdetect -y 2 command if the address of the connected i2c device can be detected, it means that i2c can be used normally

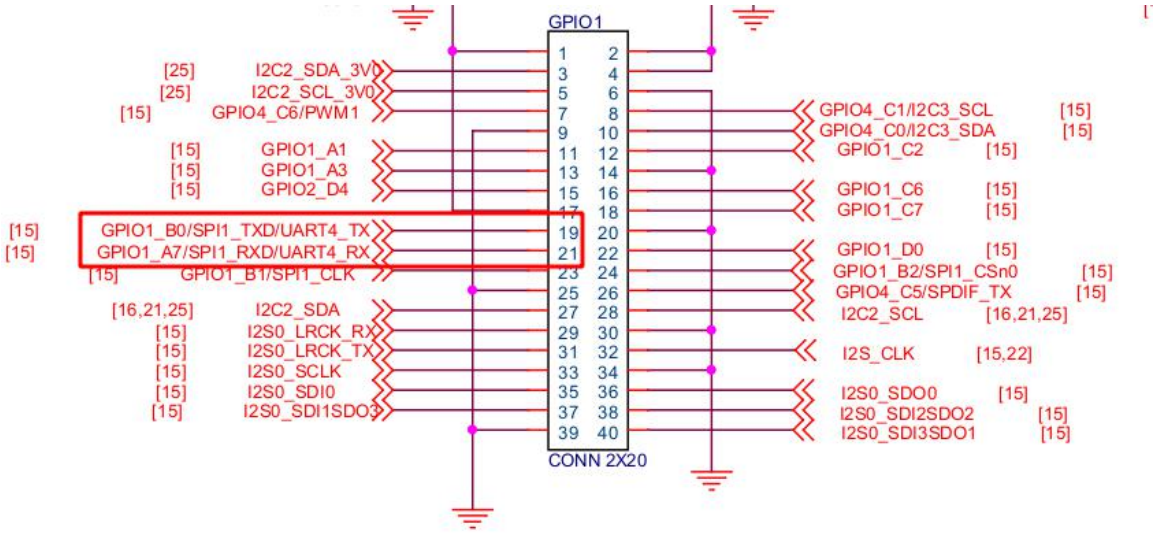
```
root@orangepi4:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- -- -- 0c -- -- --
10: UU -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

3. 21. 4. 40pin UART test

1) OragnePi 4's SPI and UART4 are multiplexed with the same pins. SPI1 in 40pin is

turned on by default in dts, and the UART4 controller is turned off. If you need to use UART4, you first need to open and close the UART4 configuration in the dts of the kernel. SPI1 configuration

a. According to the schematic diagram of 40pin, the uart available on the development board is uart4



b. Then modify dts in the kernel source code to enable uart4

```
test@test:~# cd orangepi-build
test@test:~# vim
kernel/orange-pi-4.4-rockchip64/arch/arm64/boot/dts/rockchiprk3399-orangepi-common.dts
```

c. Find the definition of spi1 and change okay to disabled to disable spi1

```
&spi1 {
status = "okay"; //Change okay to disabled
```

d. Find the configuration of uart4, change disabled to okay to open uart4

```
&uart4 {
status = "disabled";  //Change disabled to okay
```

e. After compiling the kernel, transfer the dtb.deb in the output/debs directory of the orangepi-build compiling system to the linux system of the development board for replacement. For detailed operations, refer to the 10th point of 4.3 compiling the linux kernel

```
test@ubuntu:~/orangepi-build/output/debs$ ls linux-dtb*
linux-dtb-legacy-rk3399_2.1.0_arm64.deb
```

2) Then start to test the uart interface, first use the Dupont line to short-circuit the rx and tx of the uart4 interface to be tested

	Uart4
Tx pin	Corresponding to pin 19
Rx pin	Corresponding to pin 21

3) Then modify the serial device node name opened by the serial test program serialTest in wiringOP to **/dev/ttyS4**

```
root@orangepi4:~/wiringOP/examples# vim serialTest.c
```

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

  if ((fd = serialOpen ("/dev/ttyS4", 115200)) < 0)
  {
    fprintf (stderr, "Unable to open serial device: %s\n", strerror (errno)) ;
    return 1 ;
  }
```

4) Recompile the serial test program serialTest in wiringOP

```
root@orangepi4:~/wiringOP/examples# make serialTest
[CC] serialTest.c
[link]
root@orangepi4:~/wiringOP/examples#
```

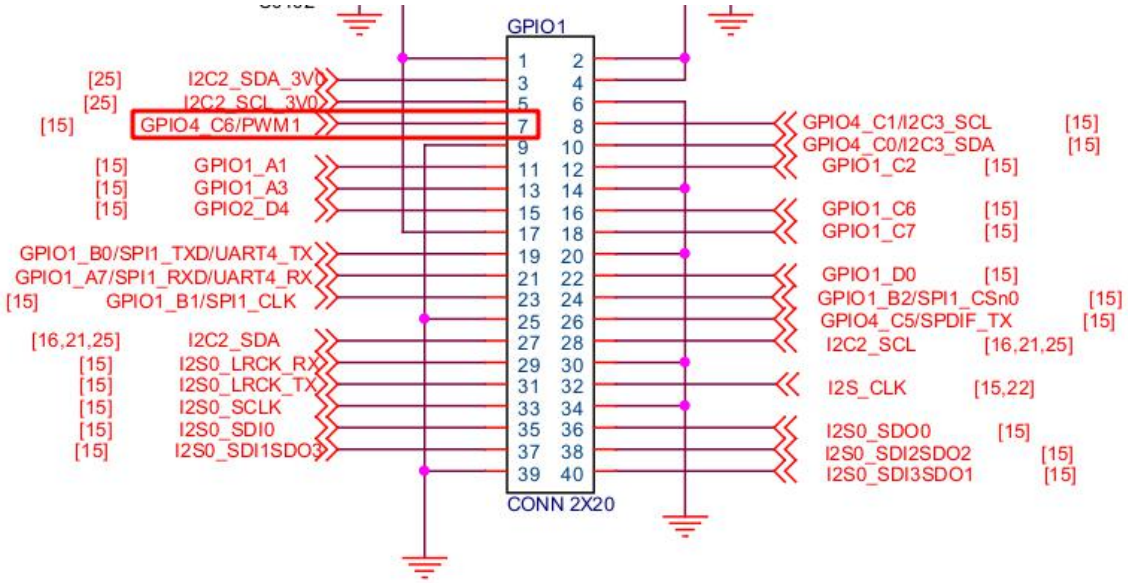
5) Finally run the serialTest, if you can see the following print, it means that the serial communication is normal

```
root@orangepi4:~/wiringOP/examples# ./serialTest
Out:    0:  ->    0
Out:    1:  ->    1
Out:    2:  ->    2
Out:    3:  ->    3
Out:    4:  ->    4
```

```
Out:    5:  ->  5
Out:    6:  ->  6
Out:    7:  ->  7
Out:    8:  ->  8^C
```

3.21.5. 40pin PWM test

1) The 7th pin of 40pin is PWM1. The official image has turned on PWM1 by default. You can use PWM1 without other configuration.



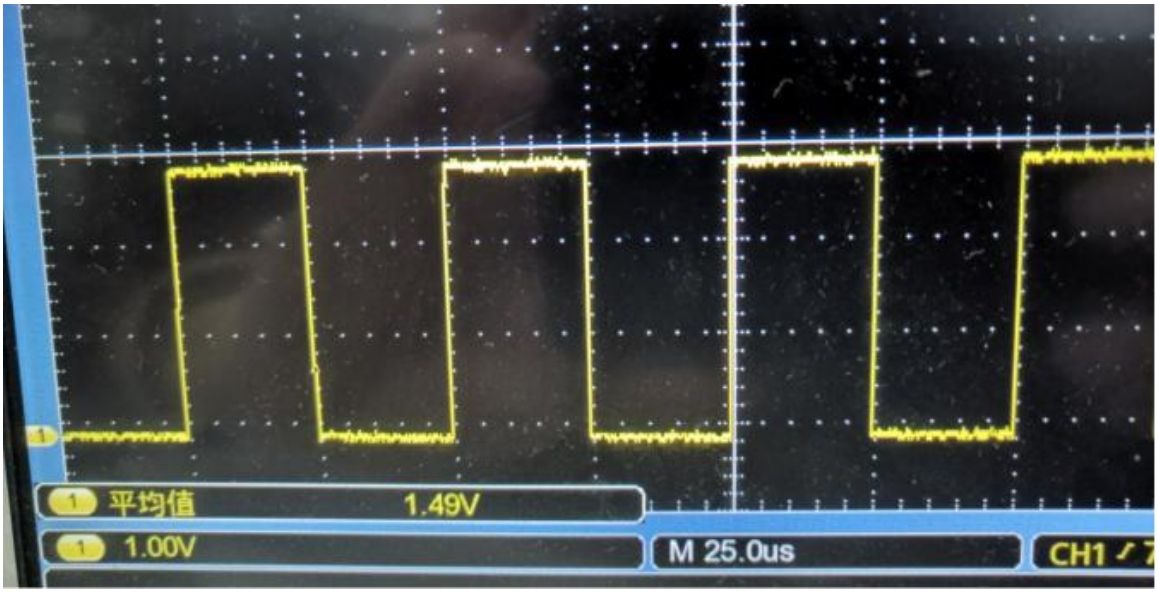
2) After the pwm driver is loaded successfully, the pwmchip1 directory will be generated under /sys/class/pwm/, write 0 to the export file, the pwm timer will be turned on, and a pwm0 directory will be generated. On the contrary, writing 0 to the unexport file will turn off the pwm timer and the pwm0 directory will be deleted. The directory has the following files:

```
enable :        Write 1 to enable pwm, write 0 to close pwm
polarity :      There are two parameter choices of normal and inversed, which means that the output pin level is inverted
duty_cycle : The unit is nanoseconds. In normal mode, it indicates the duration of high level. In inversed mode, it indicates the duration of low level.
period :        The unit is nanosecond, which represents the duration of the pwm wave
```

3) Example of use: Let pwm1 output a square wave with a duty cycle of 50% and a period of 50 microseconds

```
root@orangepi4:~#cd /sys/class/pwm/pwmchip1
root@orangepi4:/sys/class/pwm/pwmchip1#echo 0 > export
root@orangepi4:/sys/class/pwm/pwmchip1#
echo 50000 > pwm0/period
root@orangepi4:/sys/class/pwm/pwmchip1#
echo 25000 > pwm0/duty_cycle
root@orangepi4:/sys/class/pwm/pwmchip1#
echo 1 > pwm0/enable
```

4) On the oscilloscope, you can see the following waveform of pwm1 output



3. 22. How to use 0.96 inch OLED module with I2C interface

1) The 0.96 inch OLED module of Orange Pi is shown in the figure below, and its 7-bit i2c slave address is 0x3c

2) First, connect the 0.96 inch OLED module to the 40pin interface of the Orange Pi development board through the DuPont cable. The wiring method is as follows (the i2c2 is taken as an example below, i2c3 only needs to change scl to pin 8 and sda to change it. (Go to pin 10)

Pins of OLED module	description	Development board 40pin interface i2c2 pin	Development board 40pin interface i2c3 pin
GND	Power ground	6 Pin	6 Pin
VCC	5V	4 Pin	4 Pin
SCL	I2C clock line	5 Pin	8 Pin
SDA	I2C data cable	3 Pin	10 Pin
RST	Connect to 3.3V	1 Pin	1 Pin
DC	Connect to GND	9 Pin	9 Pin
CS	Connect to GND	25 Pin	25 Pin

3) After connecting the OLED module to the development board, first use the i2c-tools tool to check whether the address of the OLED module can be scanned

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install i2c-tools
root@orangepi4:~# i2cdetect -y 2
```

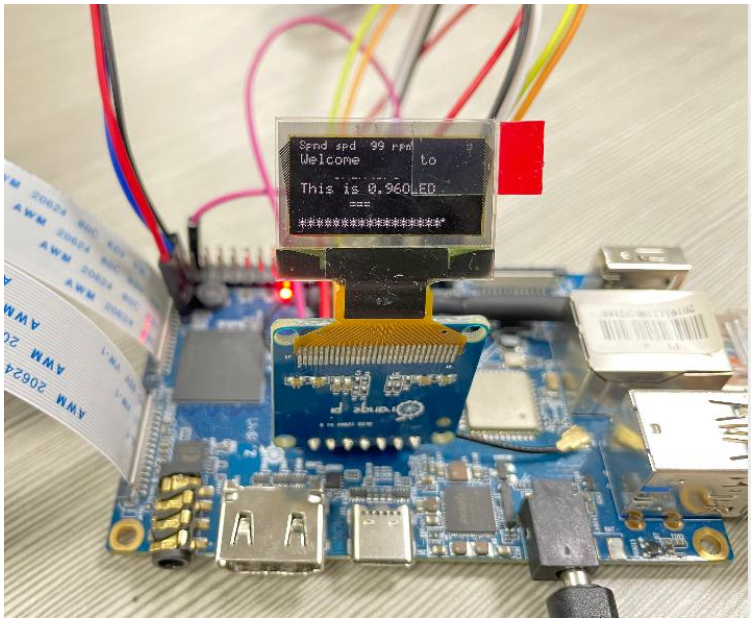
```
root@orangepi4:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- -- 0c -- -- --
10: UU -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@orangepi4:~#
```

4) Then you can use the oled_demo in wiringOP to test the OLED module, the test steps are as follows

```
root@orangepi4:~# git clone https://github.com/orangepi-xunlong/wiringOP
root@orangepi4:~# cd wiringOP
root@orangepi4:~/wiringOP# ./build clean && ./build
root@orangepi4:~/wiringOP# cd examples
root@orangepi4:~/wiringOP/examples# make oled_demo
root@orangepi4:~/wiringOP/examples# ./oled_demo /dev/i2c-2
```

```
---------start--------
----------end---------
```

5) After running oled_demo, you can see the following output on the OLED screen



3. 23. Hardware watchdog test

1) Download the code of wiringOP

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install git
root@orangepi4:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

2) Compile the watchdog test program

```
root@orangepi4:~# cd wiringOP/examples/
root@orangepi4:~/wiringOP/examples# gcc watchdog.c -o watchdog
```

3) Run the watchdog test program

a. The second parameter 10 indicates the counting time of the watchdog. If there is no dog feeding 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 the success of feeding the dog

```
root@orangepi4:~/wiringOP/examples# ./watchdog 10
open success
options is 33152,identity is Synopsys DesignWare Watchdog
put_usr return,if 0,success:0
The old reset time is: 21
return ENOTTY,if -1,success:-1
return ENOTTY,if -1,success:-1
put_user return,if 0,success:0
put_usr return,if 0,success:0
keep alive
keep alive
```

3.24. View the serial number of the rk3399 chip

1) The command to view the serial number of the RK3399 chip is as follows, the serial number of each chip is different and unique, so you can use the serial number to distinguish multiple development boards

```
root@orangepi4:~# cat /proc/cpuinfo
processor       : 5
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x0
CPU part        : 0xd08
CPU revision    : 2

Serial          : a64e6031a34aa990
```

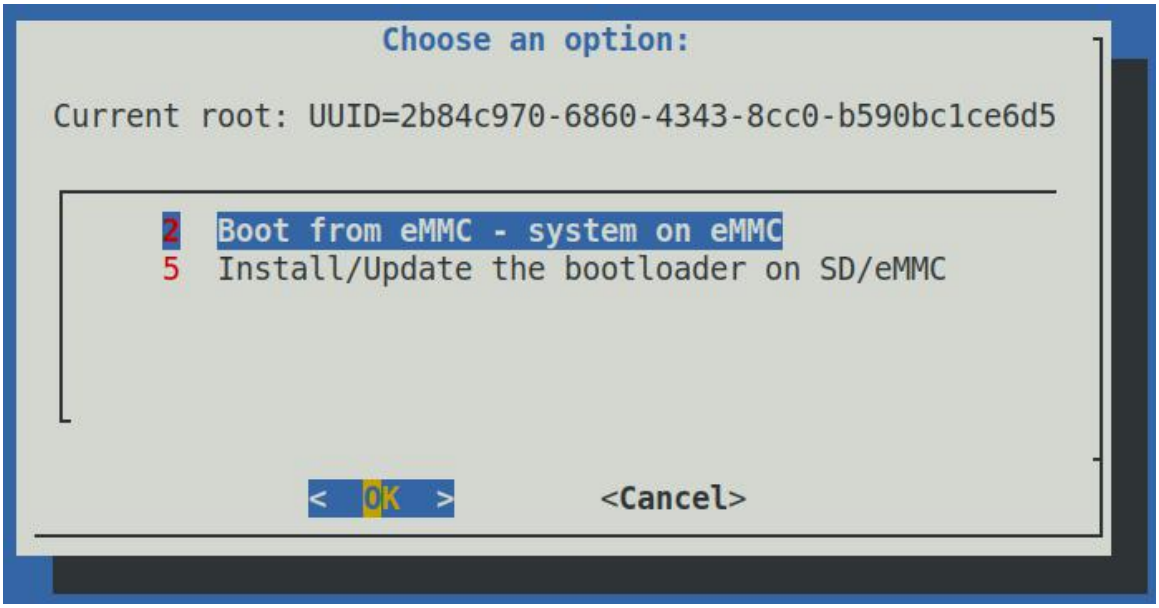
3.25. Method of flashing linux image to eMMC

1) Burning the linux image to eMMC requires the help of a TF card. First, burn the linux image to the TF card, and then start the development board to enter the linux system

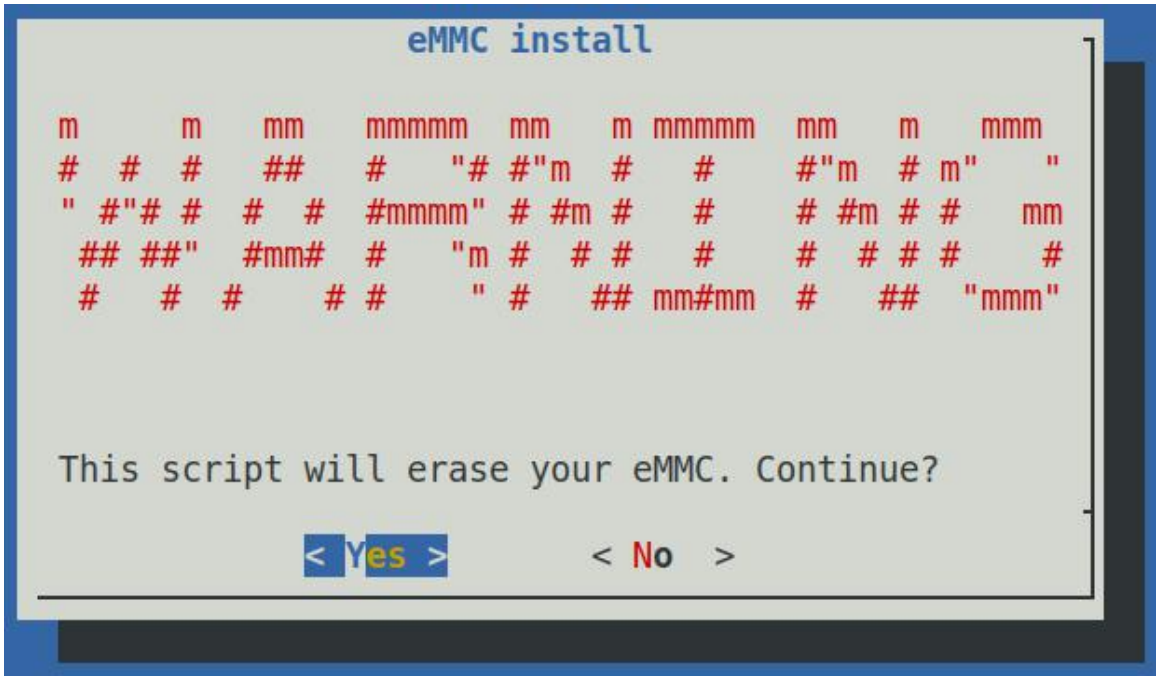
2) Then run the **nand-sata-install** script

```
root@orangepi4:~# nand-sata-install
```

3) Then choose**2 Boot from eMMC - system on eMMC**

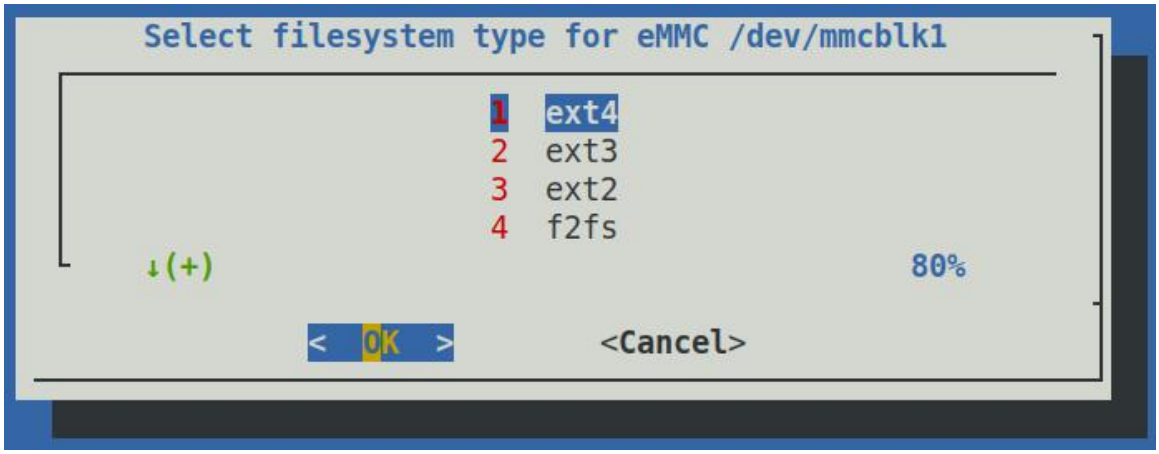


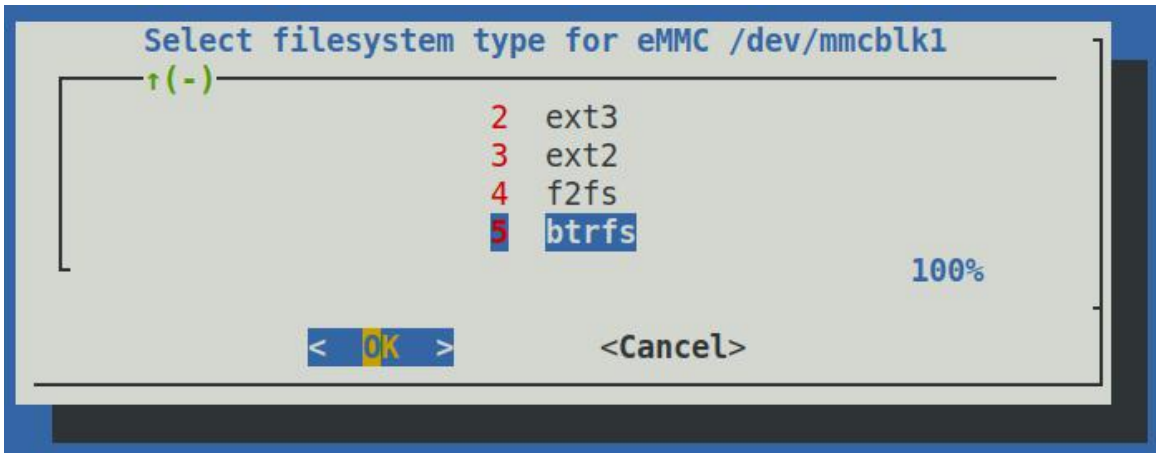
4) Then a warning will pop up, the script will erase all data on the eMMC, select <Yes> to continue



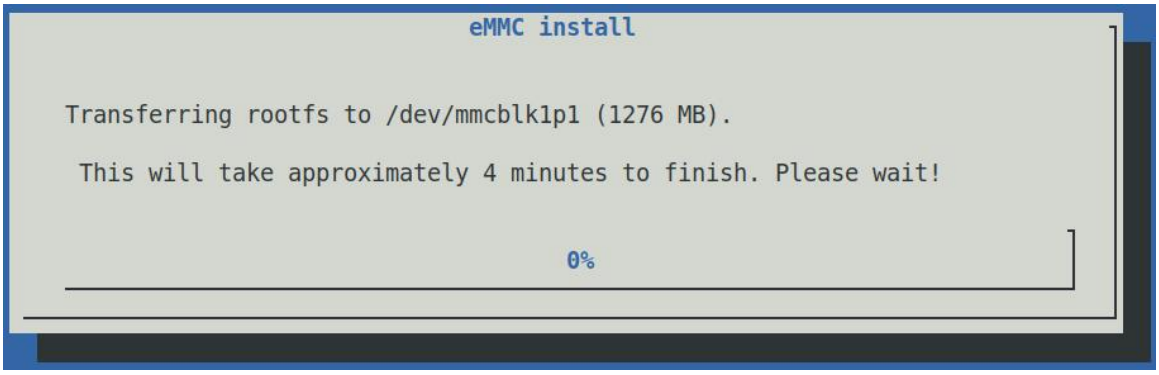
5) Then you will be prompted to select the type of file system, supporting five file

systems ext2/3/4, f2fs and btrfs

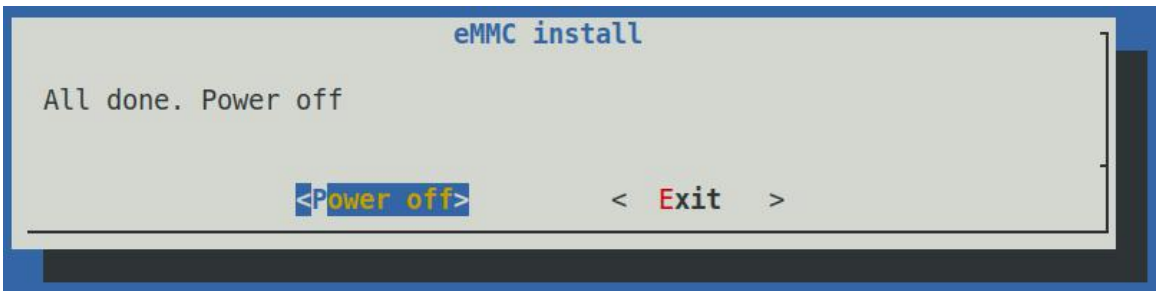




6) Then it will start to format the eMMC, and after the eMMC is formatted, it will start to burn the linux image to the eMMC



7) After burning, the following options will be prompted, you can select <Power off> to shut down directly



8) Then pull out the TF card and power on again, the linux system in eMMC will start

3. 26. How to use the OV13850 camera

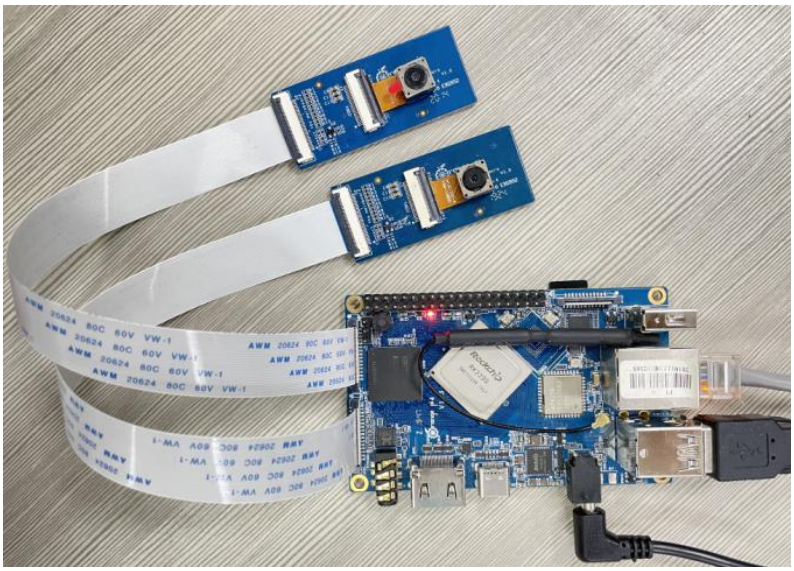
3. 26. 1. Camera connection instructions

1) Orange Pi 4 has two Camera interfaces, both interfaces only support OV13850 camera by default. The two Canera interfaces can use one of them separately, or use two Camera interfaces to connect two cameras at the same time

2) The OV13850 camera kit includes an OV13850 camera, an adapter board and a cable

3) First insert the OV13850 camera into the adapter board, and then insert the cable into another card slot of the adapter board

4) Then insert the other end of the cable into the Camera interface of the development board. The interface can be connected to two cameras at the same time or one camera alone. Start the Linux system after connecting the camera **(don't plug in the camera after power-on)**



5) After starting the system, execute the following command, if the following message appears, the camera is working normally, if there is no such message, please check whether the camera is connected correctly

```
root@orangepi4:~# dmesg | grep Async
[       1.623685] rkisp1: Async subdev notifier completed
```

3. 26. 2. Single camera use

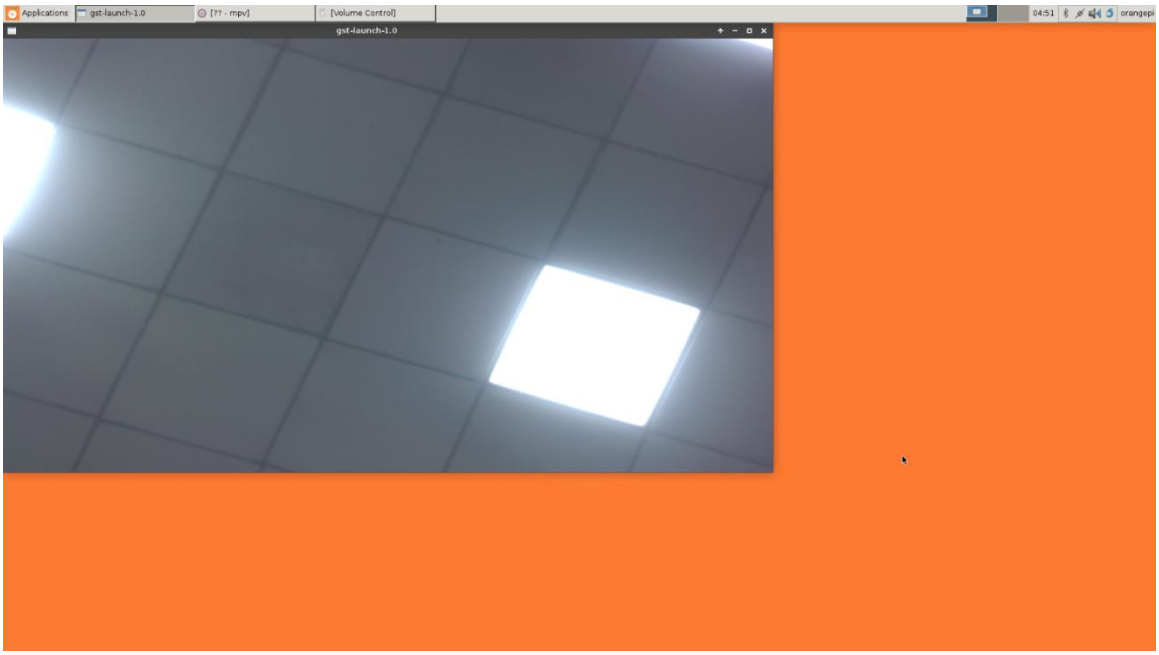
The use script of the camera has been integrated in the system, and the user can execute the corresponding script to use the camera according to the needs

1) Turn on a single camera

a. First run the test_camera-gst.sh script

```
root@orangepi4:~# test_camera-gst.sh
Setting pipeline to PAUSED ...
media get entity by name: lens is null
Pipeline is live and does not need PREROLL ...
```

b. The effect is shown in the figure below, a real-time camera window will be opened on the desktop



c. This command opens camera1 by default, so please connect the camera to the camera1 interface (the camera interface near the earphone holder)

d. If you need to test the camera2 interface (far away from the headset socket camera interface), you can also modify the test_camera-gst.sh script to open camera2 by yourself, just change the video0 in the test_camera-gst.sh script to video5, and test_camera-gst The path where the .sh script is located is

```
/usr/local/bin/test_camera-gst.sh
```

2) Use the camera to take a photo, run **test_camera-capture.sh** to take a photo, and the photo will be saved in the /home/orangepi directory

```
root@orangepi4:~# test_camera-capture.sh
```

```
Setting pipeline to PAUSED ...
media get entity by name: lens is null
Pipeline is live and does not need PREROLL ...

root@orangepi4:~# ls /home/orangepi/
camera_capture_frame0.jpg      camera_capture_frame1.jpg
```

3) Use the camera to record, run the **test_camera-record.sh** script to start recording a 17s video, and the recorded video will be saved in the /home/orangepi directory. This command turns on camera1 by default (the camera interface near the earphone holder), if you need to turn on camera2, please modify the script by yourself

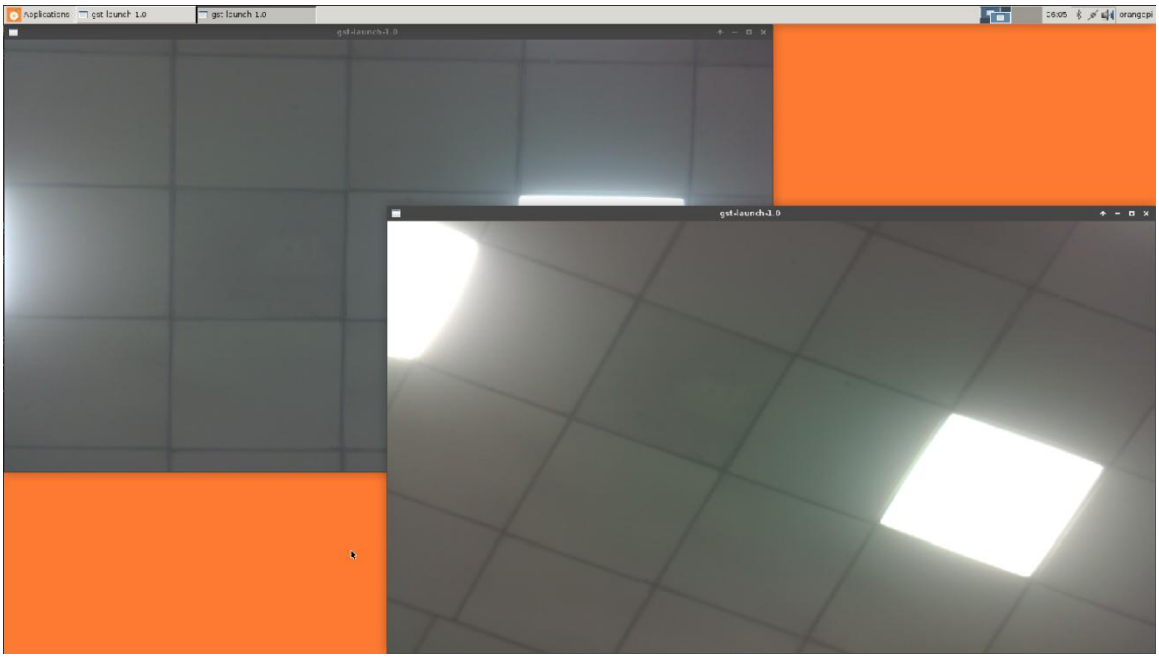
```
root@orangepi4:~# test_camera-record.sh
Setting pipeline to PAUSED ...
mpi: mpp version: Without VCS info
mpp_rt: NOT found ion allocator

root@orangepi4:~# ls /home/orangepi/outpu*
/home/orangepi/output.ts
```

3.26.3. Dual camera use

1) Run **test_camera-dual.sh** to open the dual camera

```
root@orangepi4:~# test_camera-dual.sh
Start MIPI CSI Camera Preview!
Setting pipeline to PAUSED ...
Setting pipeline to PAUSED ...
```

2) The effect is shown in the figure below, and the real-time windows of the two cameras will be opened on the desktop

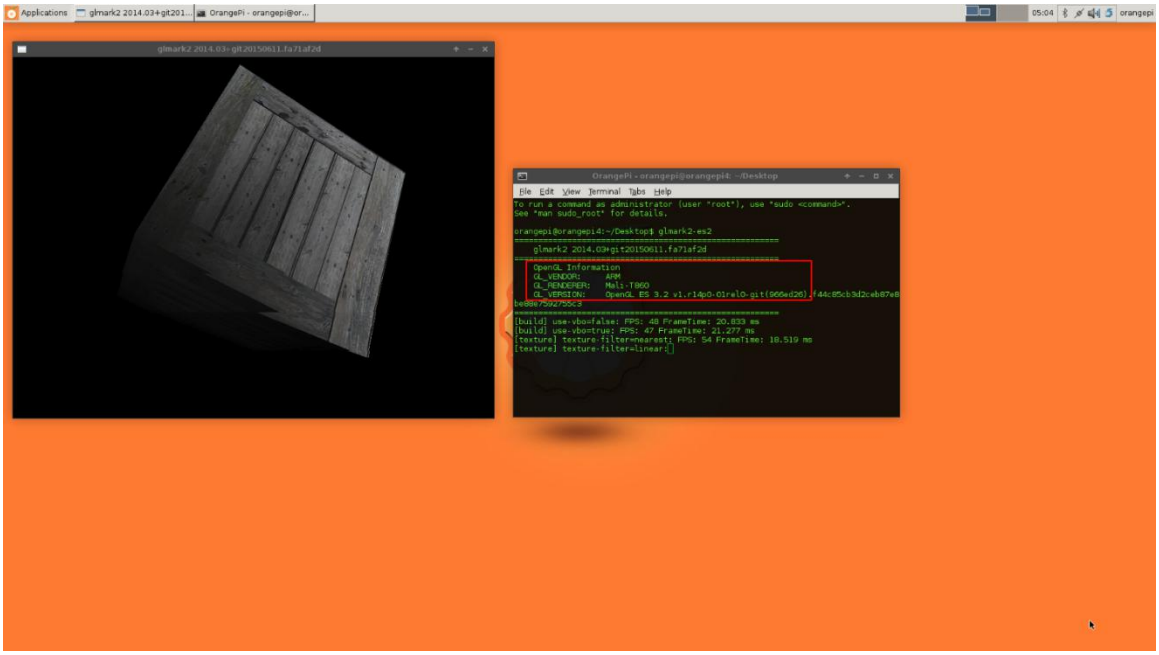


3. 27. GPU test method

1) The Linux system has been adapted to the GPU driver, use the glmark2-es2 tool to verify**(The glmark2-es2 command needs to be run in the terminal on the desktop）**

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install glmark2-es2
root@orangepi4:~# glmark2-es2
```

2) The effect of running glmark2-es2 is shown in the figure below. From the output log, you can see that the GPU used is Mali-T860



3. 28. How to use MIPI screen

3. 28. 1. Instructions for use of LCD interface

1) The system uses HDMI and Type C to HDMI output by default, LCD output is not turned on by default, if you need to use LCD output, you need to modify the configuration of the kernel dts

2) The Linux system has two LCD interfaces, LCD1 is an independent interface, and LCD2 is multiplexed with camera 2. If LCD2 is needed, you need to turn on the dts configuration of LCD2 and turn off the configuration of camera 2

3) Because of the interface characteristics, the LCD output will be recognized as the main screen. If HDMI is connected at this time, the HDMI display will be recognized as an extended screen

4) The system supports dual LCD output, and the dts configuration needs to be modified. Note that the system only supports two outputs. After turning on the dual LCD, you need to turn off the HDMI and Tpye C to HDMI output, otherwise the system may be abnormal

3. 28. 2. Modify dts in the source code to enable the method of lcd interface

1) Open the dts file

```
test@test:~$ cd orangepi-build
root@orangepi4:~/orangepi-build$ vim
kernel/orange-pi-4.4-rockchip64/arch/arm64/boot/dts/rockchiprk3399-orangepi-lcd.
dts        (This is a command)
```

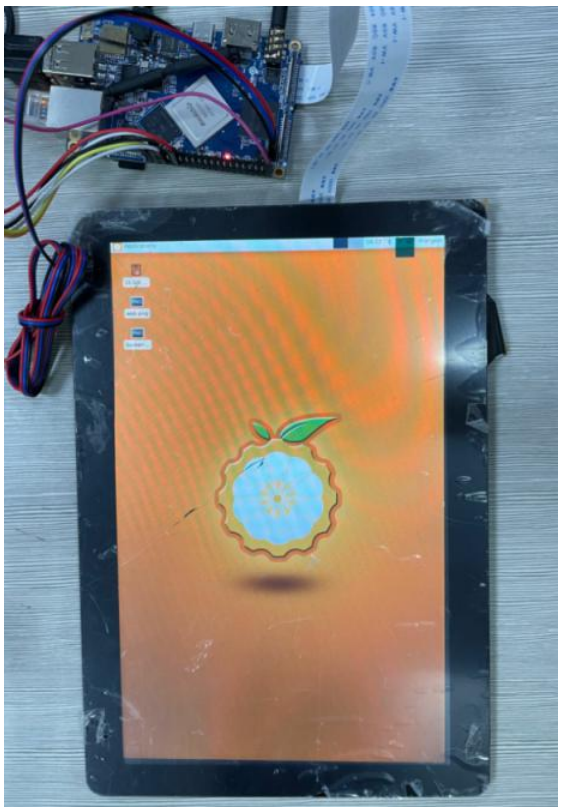
2) Find the configuration of dsi in dts

```
&dsi {
        status = "disabled"; //Change disabled to okay
```

3) After compiling the kernel, transfer the dtb.deb in the output/debs directory to the board for replacement. For detailed operations, refer to the 10th point of 4.3 Compiling Linux Kernel

```
test@ubuntu:~/orangepi-build/output/debs$ ls linux-dtb*
linux-dtb-legacy-rk3399_2.1.0_arm64.deb
```

4) After replacing the dtb of the kernel, restart the linux system to use the lcd1 interface



3. 28. 3. Using dual LCD to output dual screen different display method

No need for other settings for dual-screen display, just open the configuration of two lcds in dts

3. 28. 4. Using dual LCD to output dual screens with the same display method

1) Use the following command to set (Note: orangepi is the user name of the desktop login user, DSI-1, DSI-2 are the names of the display device), **if the system restarts after setting, please confirm whether the power supply is sufficient**

```
root@orangepi:~#
su orangepi -c "DISPLAY=:0 xrandr --output DSI-1 --same-as DSI-2"
```

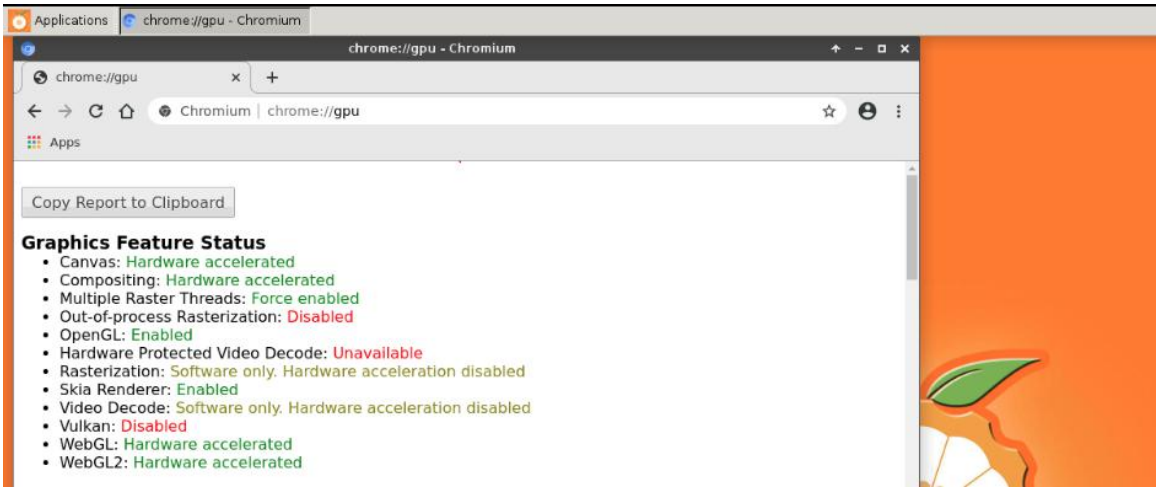
2) When connecting to other outputs such as HDMI or DP, if you want to use dual-screen simultaneous display, you can also refer to the first step for setting

```
root@orangepi:~#
su orangepi -c "DISPLAY=:0 xrandr --output HDMI-1 --same-as DP-1"
```

3) The dual LCD output dual screen simultaneous display effect is shown in the figure below

3. 29. How to use Web hardware acceleration

The Chromium web browser pre-installed in the system has hardware acceleration enabled by default. To support WebGL, you can enter the URL chrome://gpu to learn about hardware acceleration. As shown below



3.30. Install OpenCV

3.30.1. Install libopencv library

Run the following command to install the libopencv library

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install libopencv-dev python-opencv
```

3.30.2. Test OpenCV

1) Write the test program test.py, the content is as follows

```
import cv2 as cv
img = cv.imread("/root/orangepi.jpg")
cv.namedWindow("Image")
cv.imshow("Image", img)
cv.waitKey(0)
cv.destroyAllWindows()
```

2) Run the test program on the desktop terminal, the corresponding picture orangepi.jpg under the directory of the test program will be opened

```
root@orangepi4:~# python test.py
```

3.31. Use VNC to log in remotely

1) Execute the following command to install **tightvncserver**

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install tightvncserver
```

2) After the installation is complete, perform the initial configuration of the VNC server. You can use the vncserver command to set a security password and create an initial configuration file

```
root@orangepi4:~# vncserver

You will require a password to access your desktops.

Password:
```

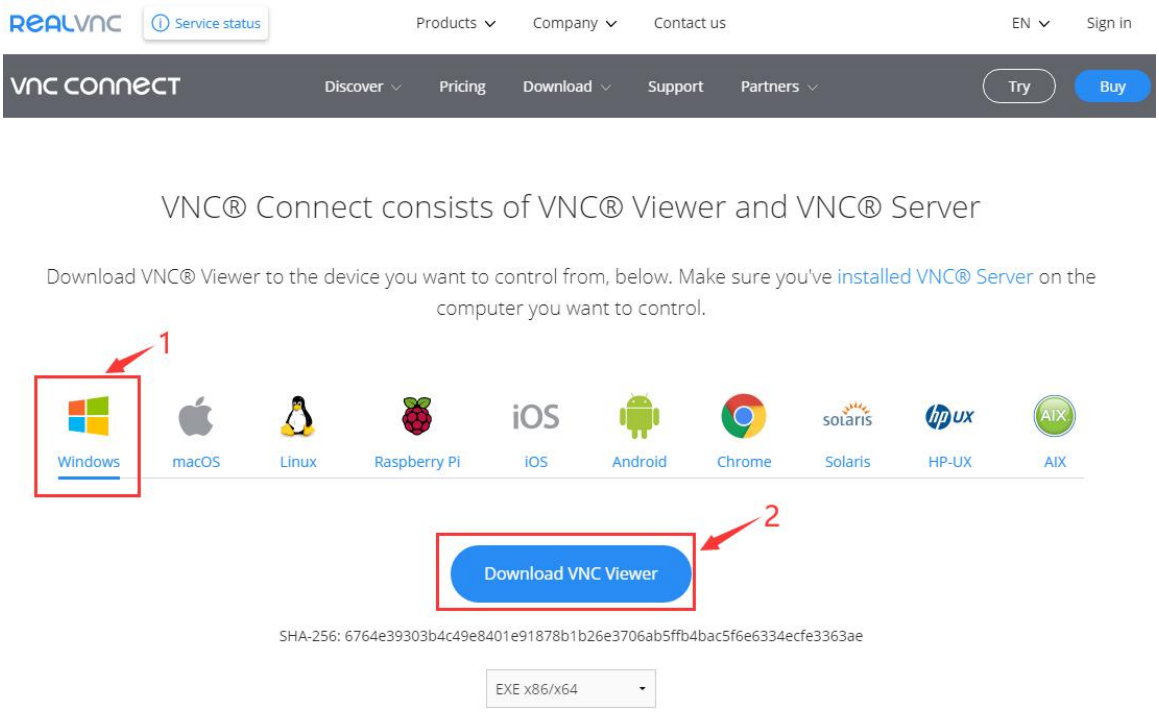
```
Verify:
Would you like to enter a view-only password (y/n)? n
xauth:    file /root/.Xauthority does not exist

New 'X' desktop is orangepi4:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/orangepi4:1.log
```

3) Modify the .vnc/xstartup file and set the vnc server to start the xfce desktop environment

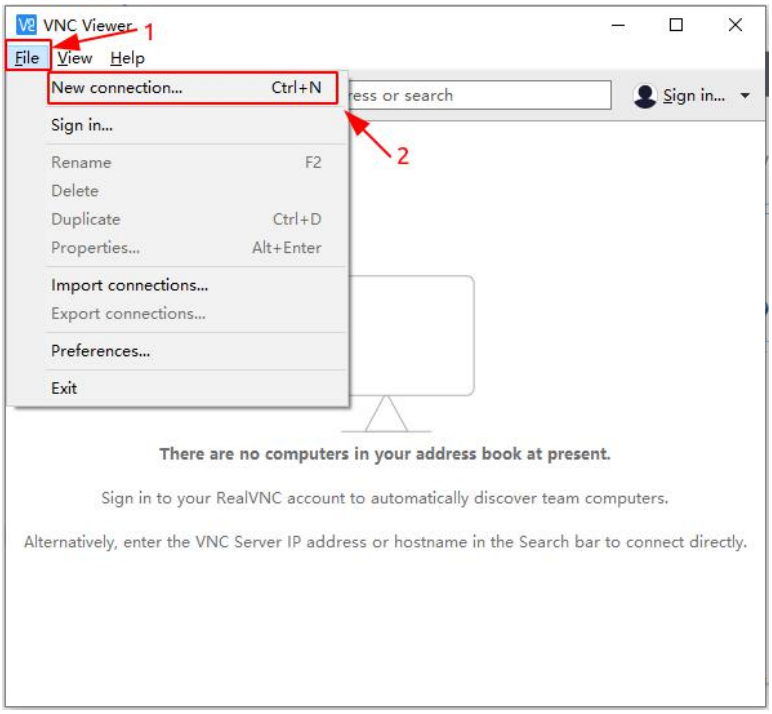
```
root@orangepi4:~# vim .vnc/xstartup
#!/bin/sh

xrdb $HOME/.Xresources
startxfce4 &
#xsetroot -solid grey
#x-terminal-emulator -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
#x-window-manager &
# Fix to make GNOME work
#export XKL_XMODMAP_DISABLE=1
#/etc/X11/Xsession
```

4) Download and install the VNC Viewer client on the windows PC, the download link is as follows

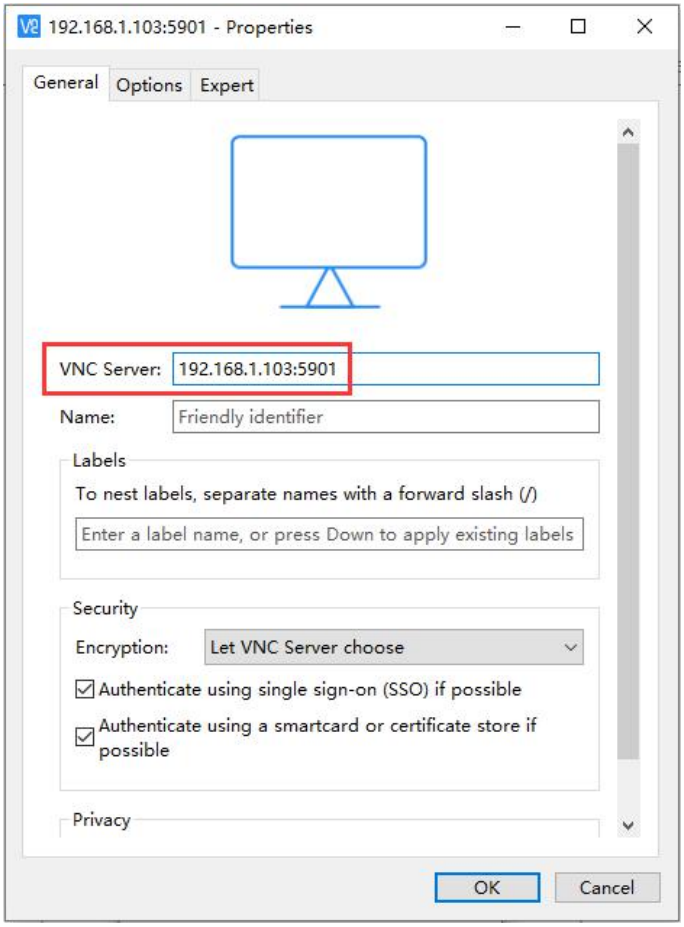
https://www.realvnc.com/en/connect/download/viewer/



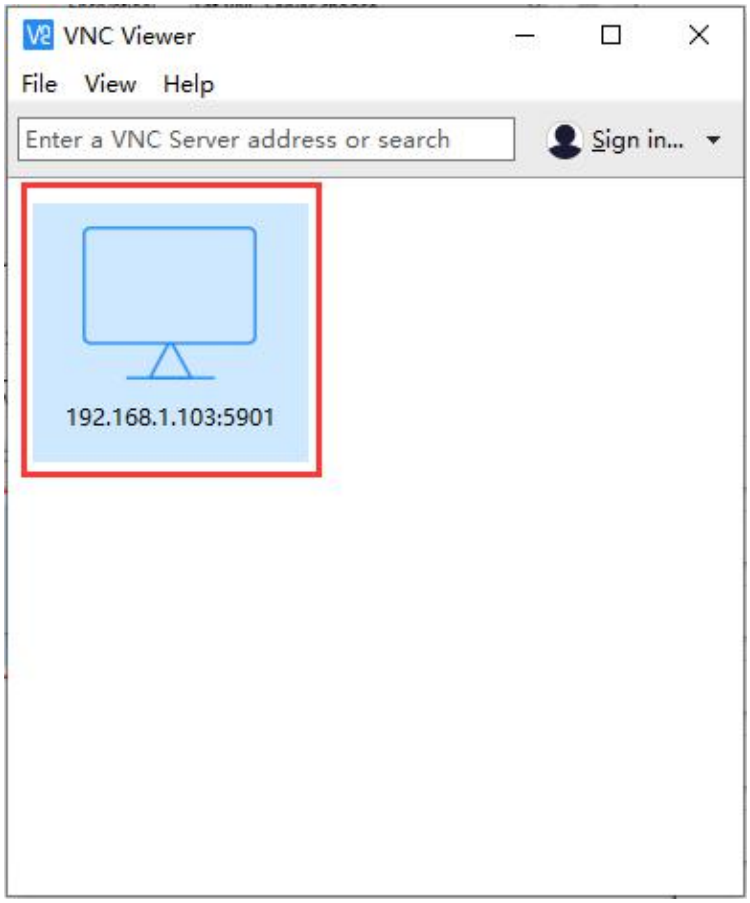
5) After the installation is complete, open VNC Viewer, click "File" -> "New connection" to create a new connection

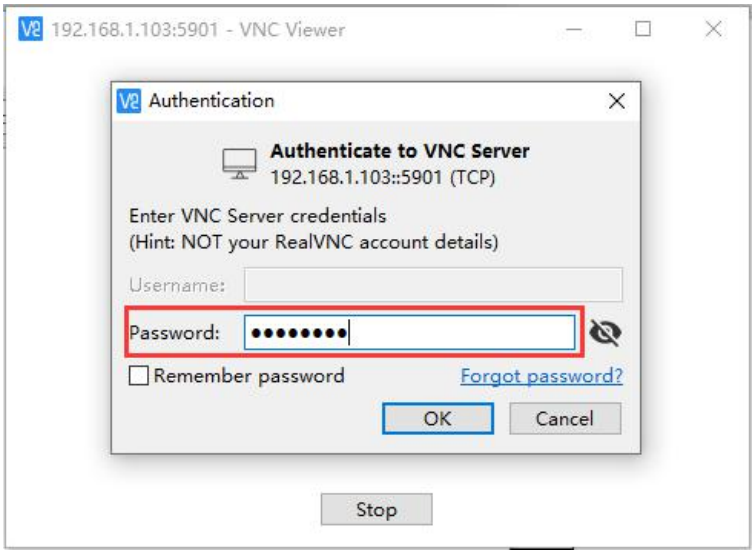


6) Enter the ip address of the development board and port 5901 at the VNC Server

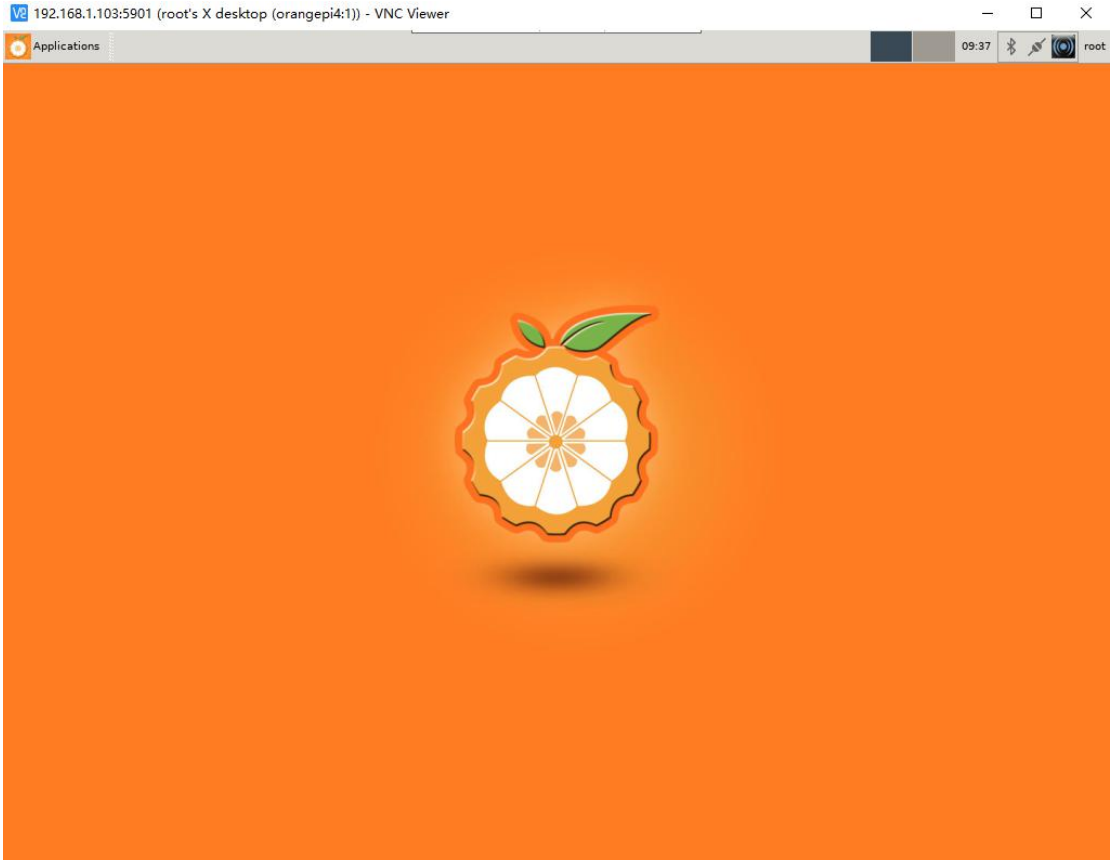


7) Open the newly created connection and enter the password set in the second step to remotely log in to the system desktop





8) After successful login, you can remotely operate on the system desktop



3. 32. Install QT and its usage example

3. 32. 1. Install QT

1) Use the following commands to install QT5 and qtcreator

```
root@orangepi4:~# apt update
root@orangepi4:~# apt install qt5-default qtcreator
```

2) After the installation is complete, you can use the following command to check whether the installation of QT is successful

```
root@orangepi4:~# qmake -v
QMake version 3.1
Using Qt version 5.9.5 in /usr/lib/aarch64-linux-gnu
```

3. 32. 2. QT usage example: Minesweeper game

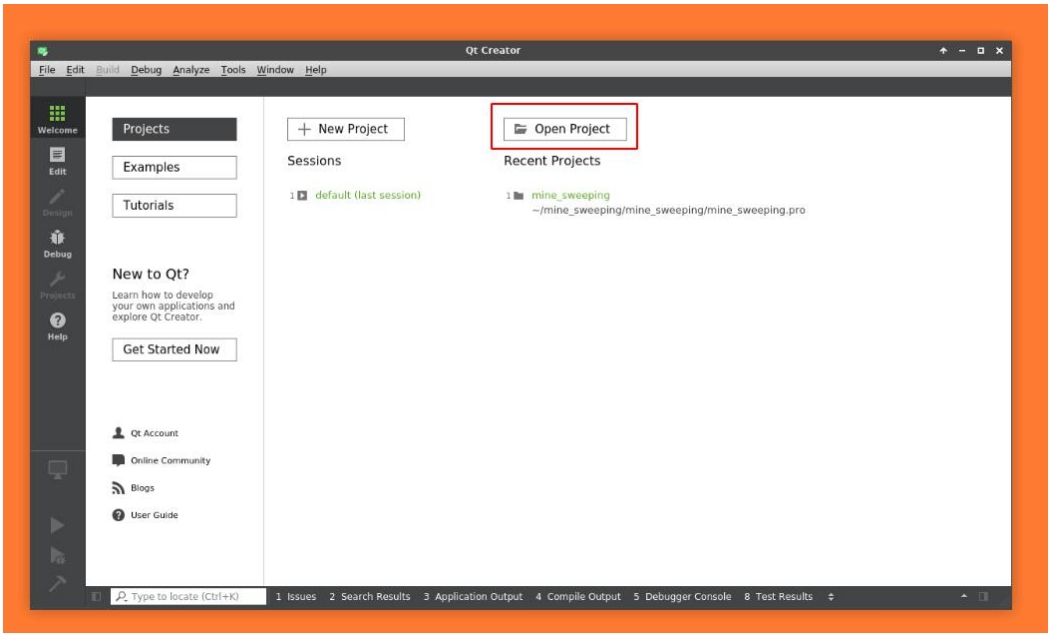
1) Download the code of the minesweeper game

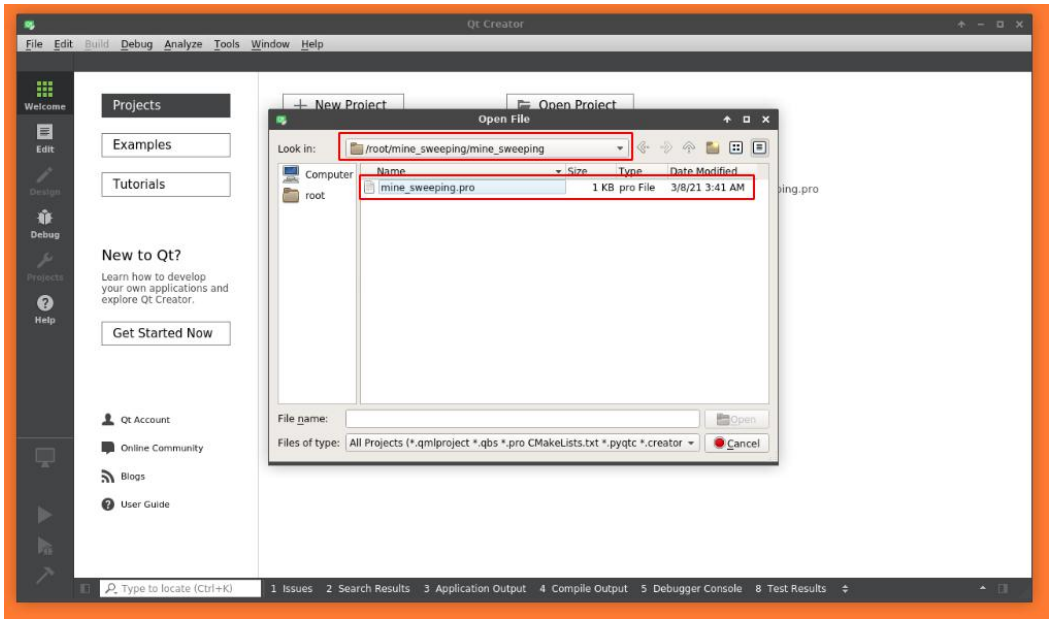
```
root@orangepi4:~# apt update
root@orangepi4:~# apt install git
root@orangepi4:~# git clone https://github.com/qmeng0207/mine_sweeping
```

2) Enter the following command in the terminal of the system desktop to open qtcreator

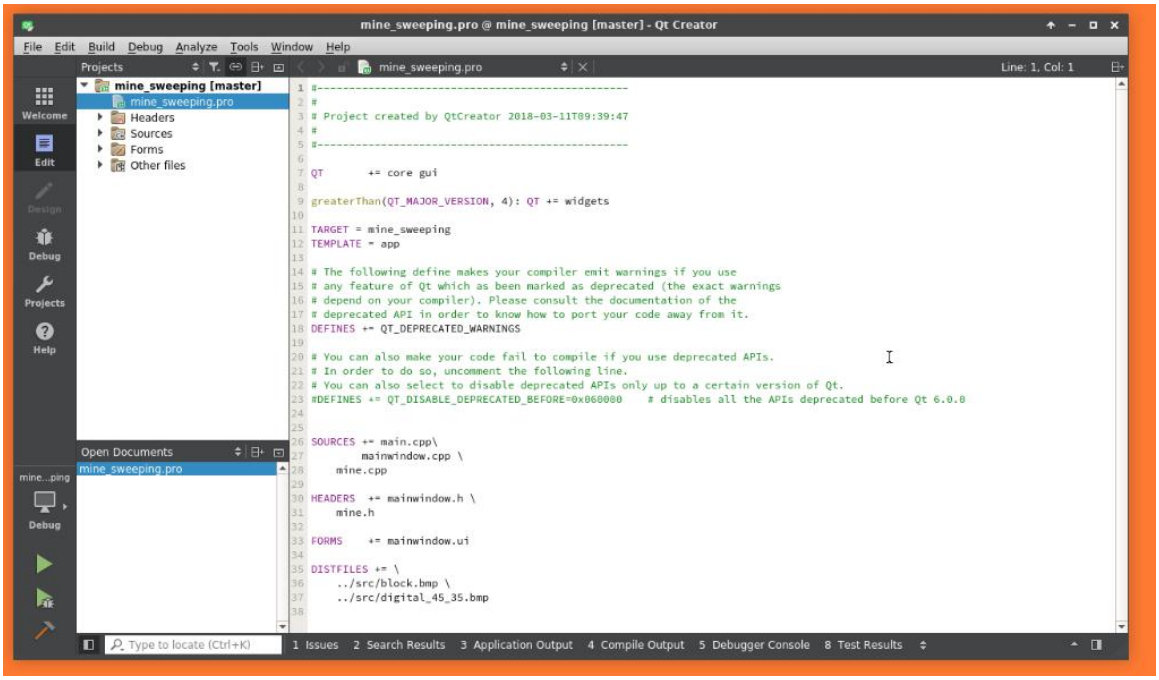
```
root@orangepi4:~# qtcreator
```

3) Select Open Project to open the source code of the minesweeper game just downloaded

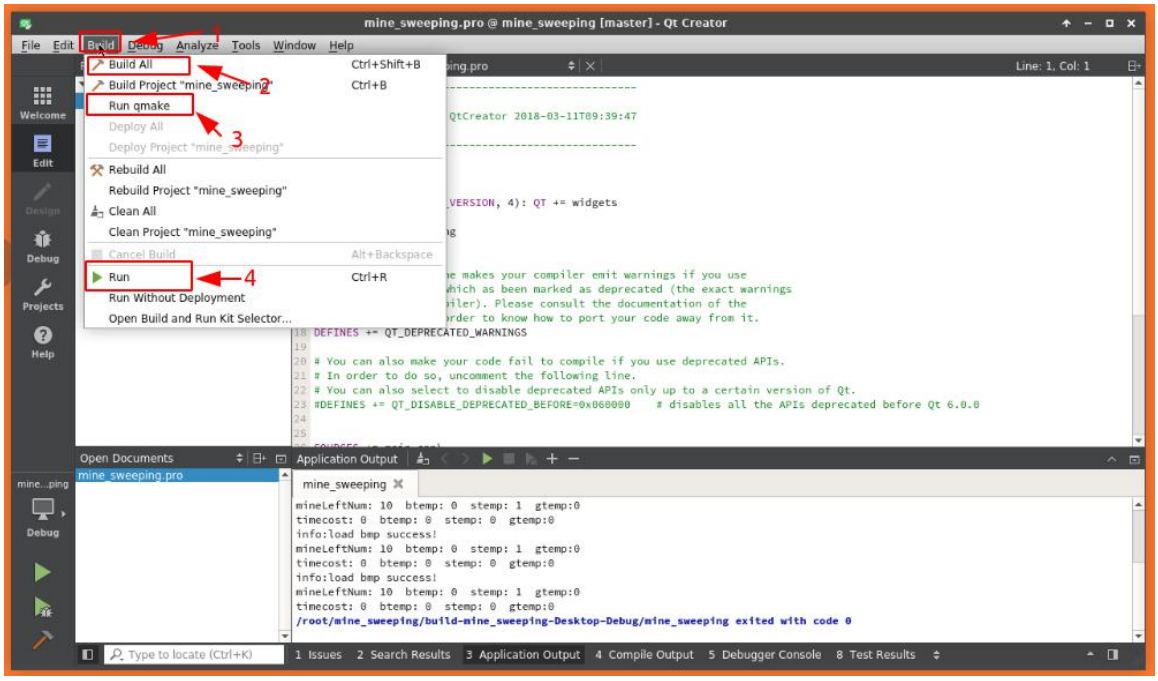




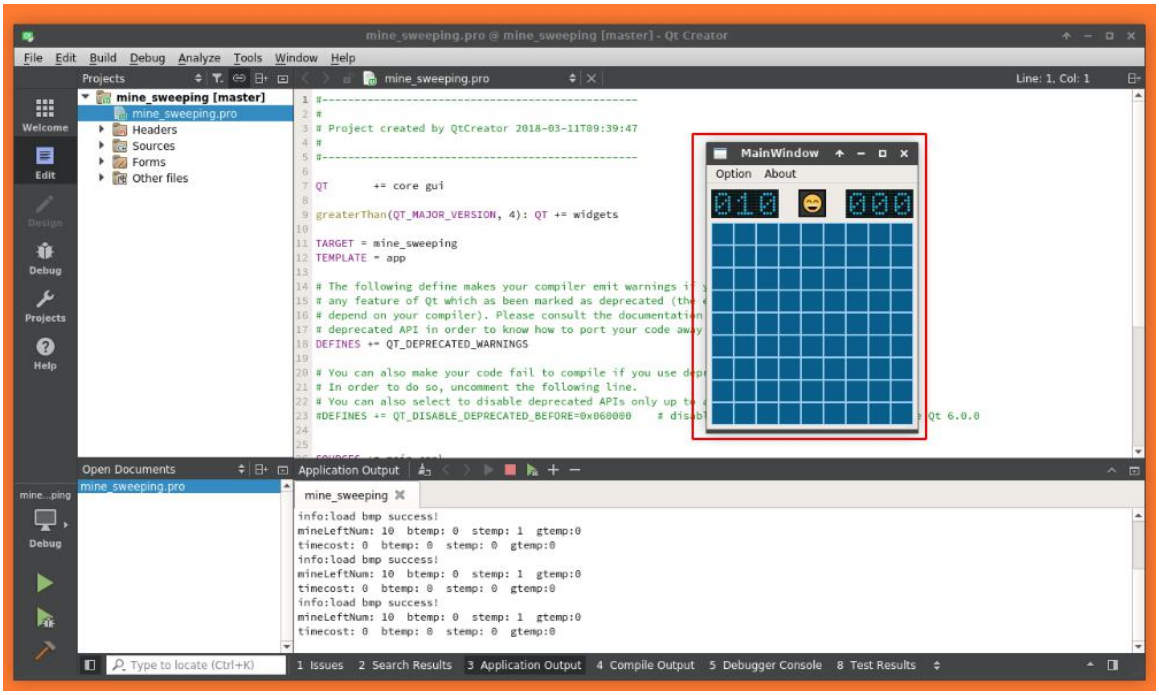
4) Open the source code as shown in the figure below



5) Compile the source code and run



6) The running result of the minesweeper game is shown in the figure below



3. 33. Install Chinese input method

The system does not have an input method installed by default. If you need to input

Chinese, please install the Chinese input method according to the following method. The following method is only available for testing on Ubuntu 18.04, and other systems have not been tested yet

1) First execute the following commands in the terminal to install the ibus input method framework

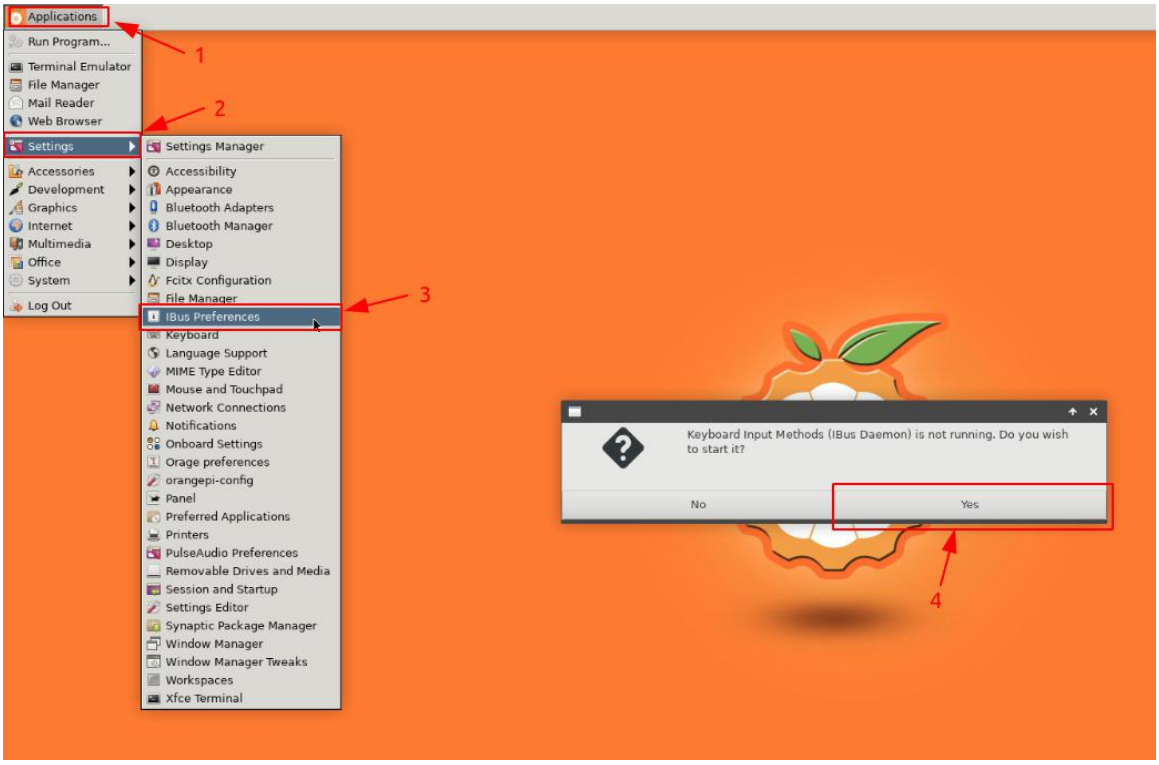
```
root@orangepi4:~# apt update
root@orangepi4:~# apt install ibus ibus-clutter ibus-gtk ibus-gtk3 ibus-qt4
```

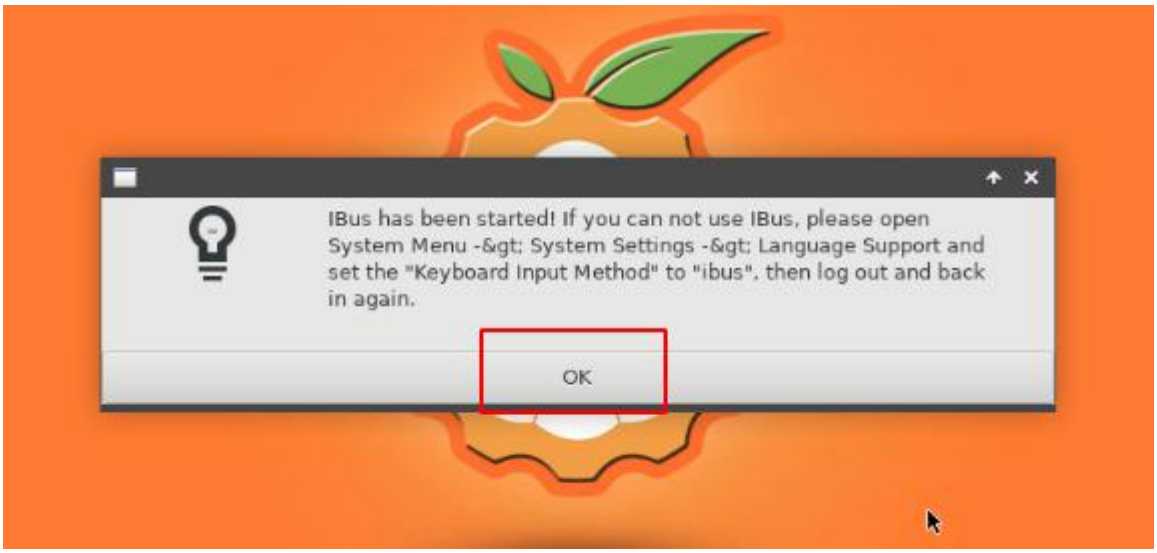
2) Then enter the following command to install ibus-pinyin

```
root@orangepi4:~# apt install ibus-pinyin ibus-sunpinyin
```

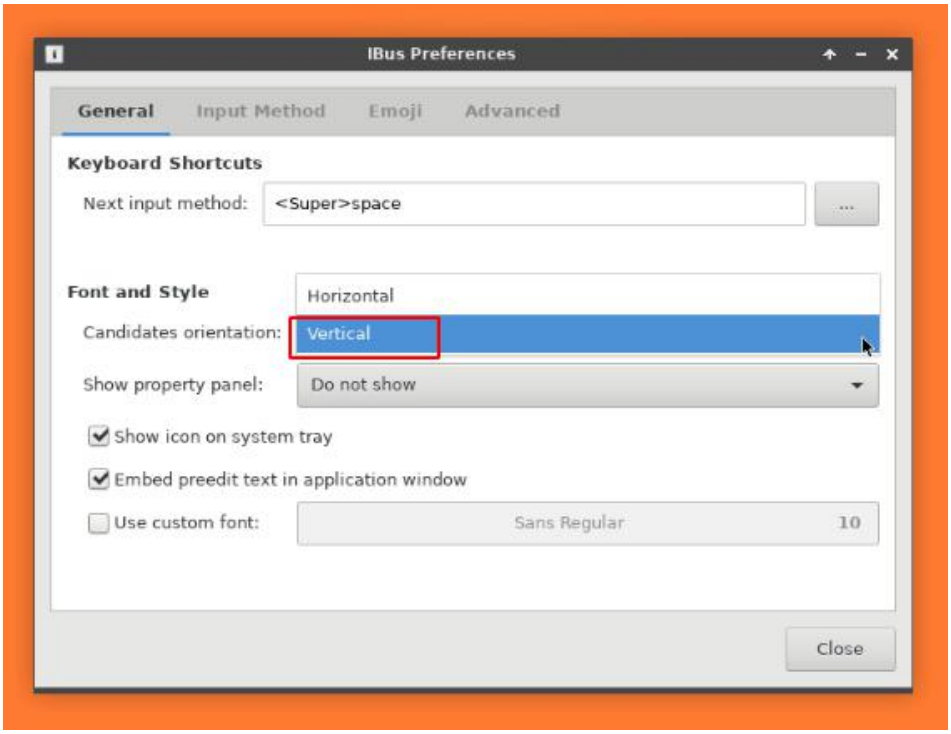
3) Follow the steps below to configure the input method on the desktop

a. Select Applications -> Setting -> iBus Preferencs in the upper left corner to open the ibus configuration interface, select yes in the pop-up window, and then select ok in the pop-up window

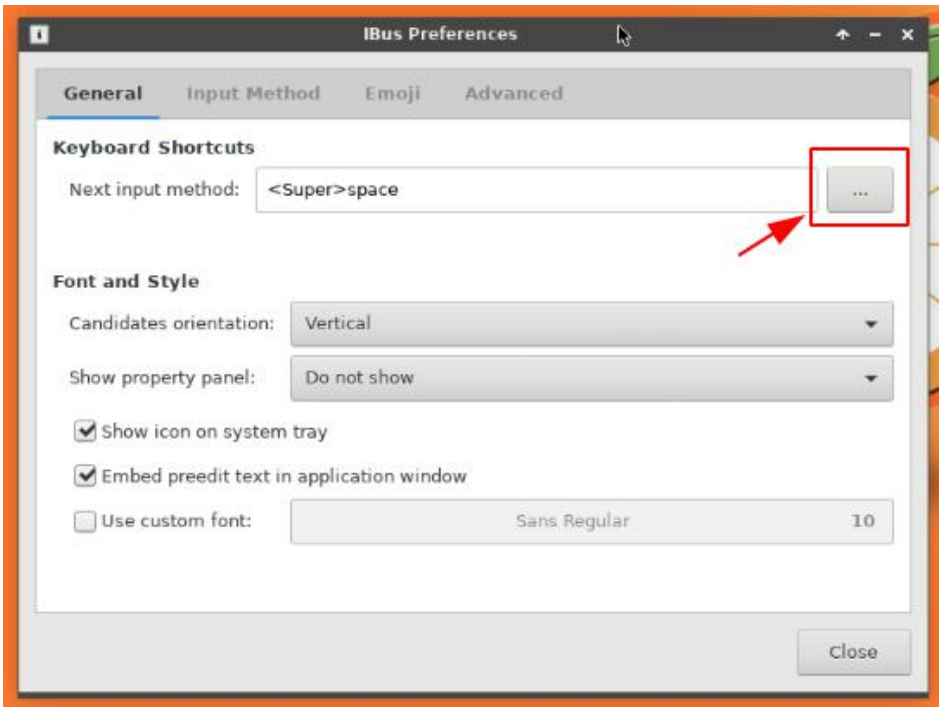




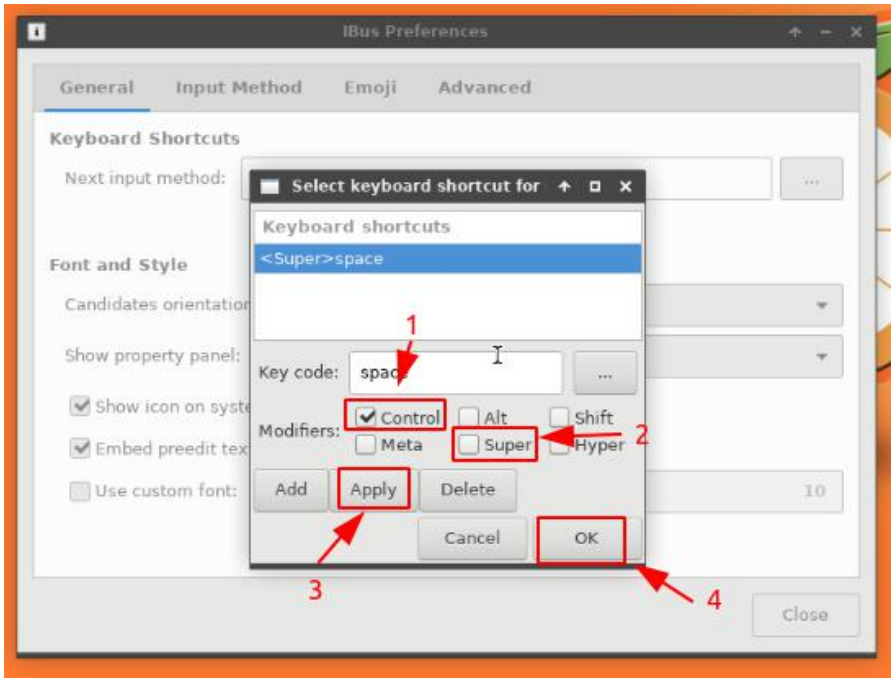
b. After the configuration interface pops up, select Vertical as the input method direction



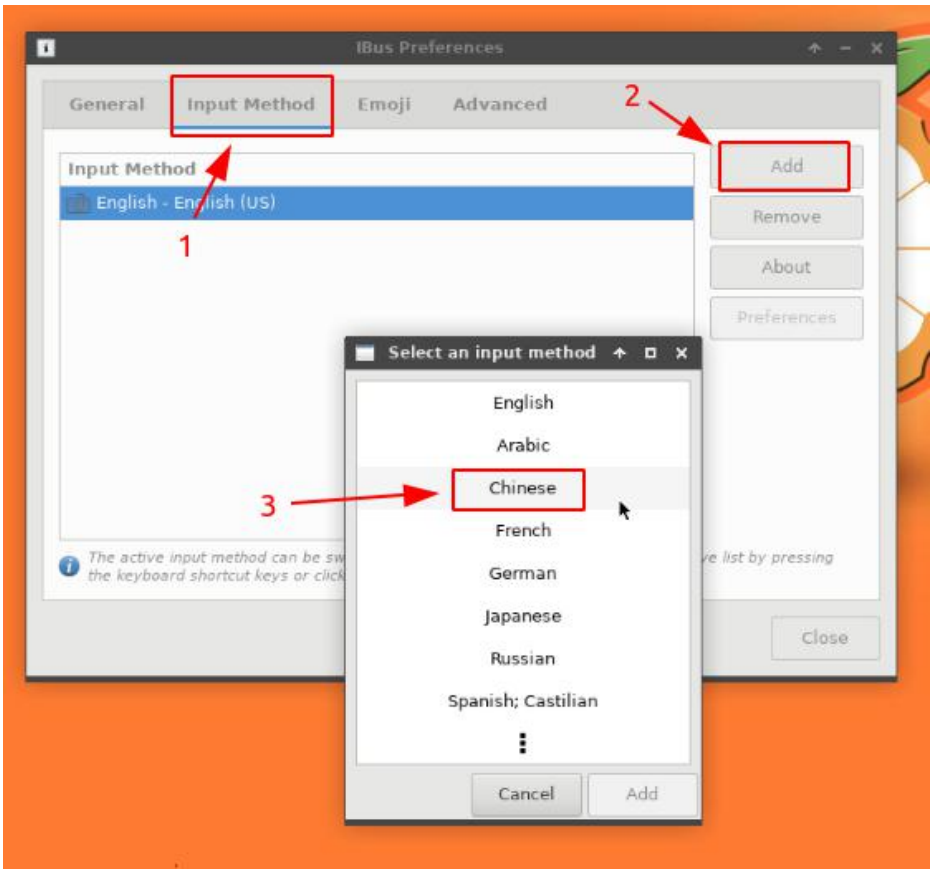
c. Click the "..." button in the figure below to select the shortcut key for switching

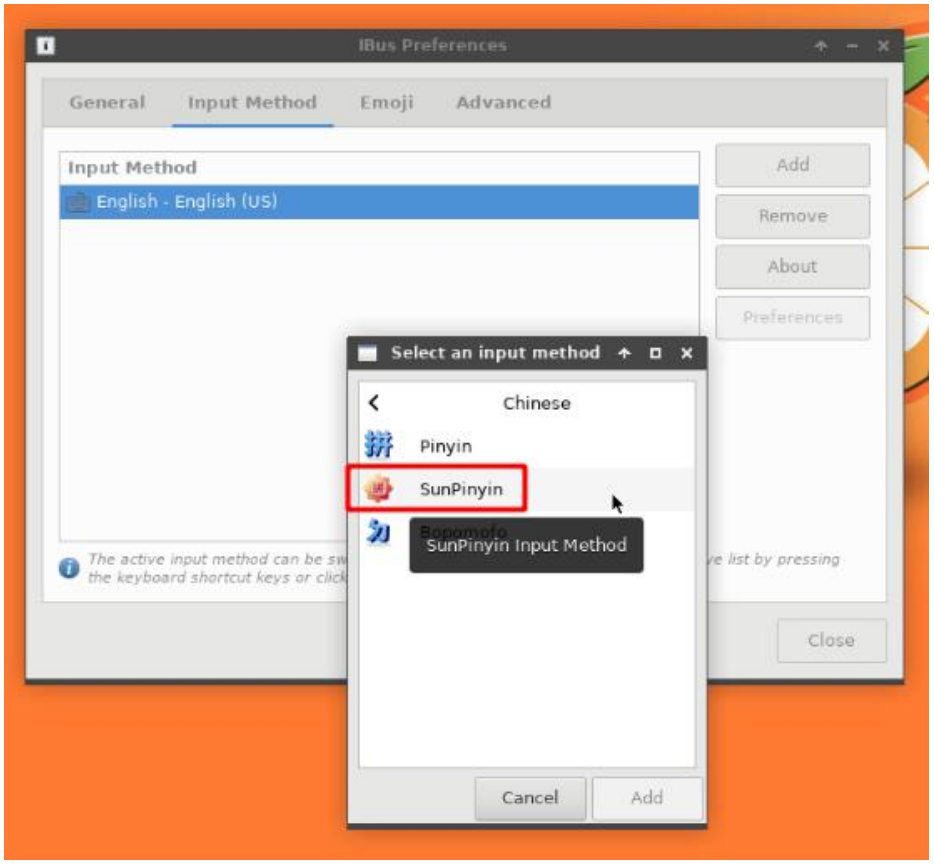


d. In the pop-up window, check Control, remove the check in front of Super, then click Apply, and finally click OK to complete the setting

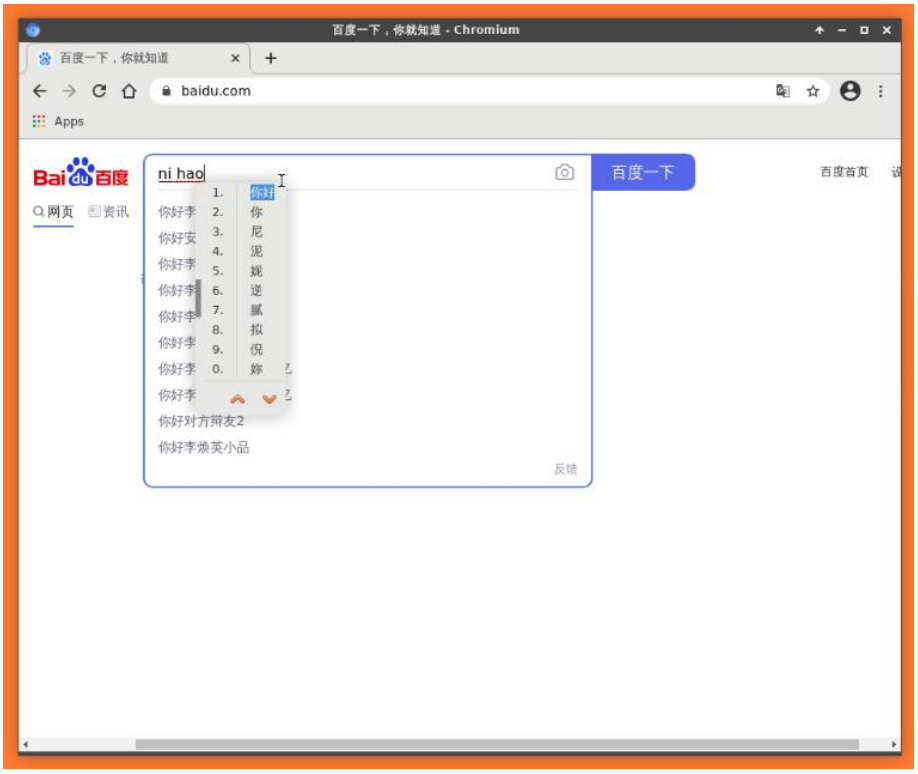


e. Select the Input Method tab, click Add to select the language to add, click Chinese, and then select the input method SunPinyin, click Add to complete the addition


IBus Preferences
General
Input Method
Emoji
Advanced
2
Input Method
English - English (US)
1
Add
Remove
About
Preferences
Select an input method
English
Arabic
Chinese
3
French
German
Japanese
Russian
Spanish; Castilian
Cancel
Add
Close

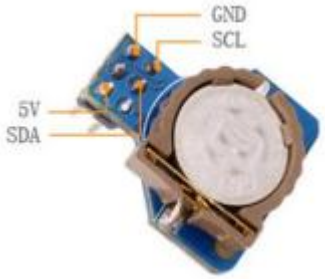

IBus Preferences
General
Input Method
Emoji
Advanced
Input Method
English - English (US)
Add
Remove
About
Preferences
Select an input method
Chinese
Pinyin
SunPinyin
Bopomofo
SunPinyin Input Method
Cancel
Add
Close

f. After completing the above settings, you can use the Chinese input method, hold down the Ctrl key, and then press the space bar to switch the input method, the use effect is as shown in the figure below



3. 34. How to use the orange pi DS1307 RTC clock module

1) The orange pi DS1307 RTC clock module is shown in the figure below. It uses the i2c interface to communicate with the development board, and the i2c device address is 0x68. The RTC module is not equipped with a battery by default, a button battery is required before use



2) First, connect the RTC module to the 40pin of the development board, the wiring

method is as follows

Pins of RTC module	The corresponding pins of the development board 40pin
5V	2 Pin
GND	6 Pin
SDA	3 Pin
SCL	5 Pin

3) After connecting the RTC module, first use the i2cdetect command to check whether the device address of the RTC module can be detected

```
root@orangepi:~# apt update
root@orangepi:~# apt install i2c-tools
root@orangepi:~# i2cdetect -y 2
```

```
root@orangepi4:~# i2cdetect -y 2
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: UU -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

4) Because the kernel has opened the driver of ds1037 by default, you can directly test the function. Execute the following command to add the rtc device, and view the generated rtc device, where rtc0 is the onboard rtc, and rtc1 is the newly added external rtc

```
root@orangepi:~# echo "ds1307 0x68" > /sys/class/i2c-adapter/i2c-2/new_device
root@orangepi:~# ls /dev/rtc*
/dev/rtc   /dev/rtc0   /dev/rtc1
```

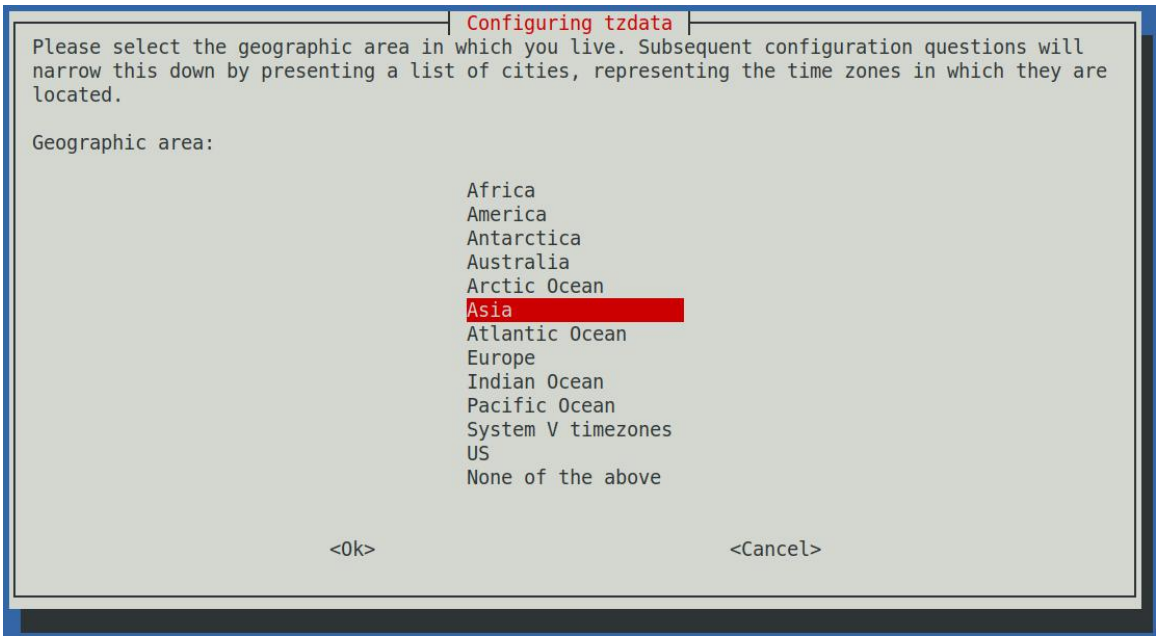
5) When the linux system is started, if the development board is connected to the network, the linux system will automatically synchronize the system time to the correct time through the network. The default time of the linux system is UTC. In China, the time

zone needs to be changed to **Asia/Shanghai**. Use the data command to obtain the correct time, the method is as follows

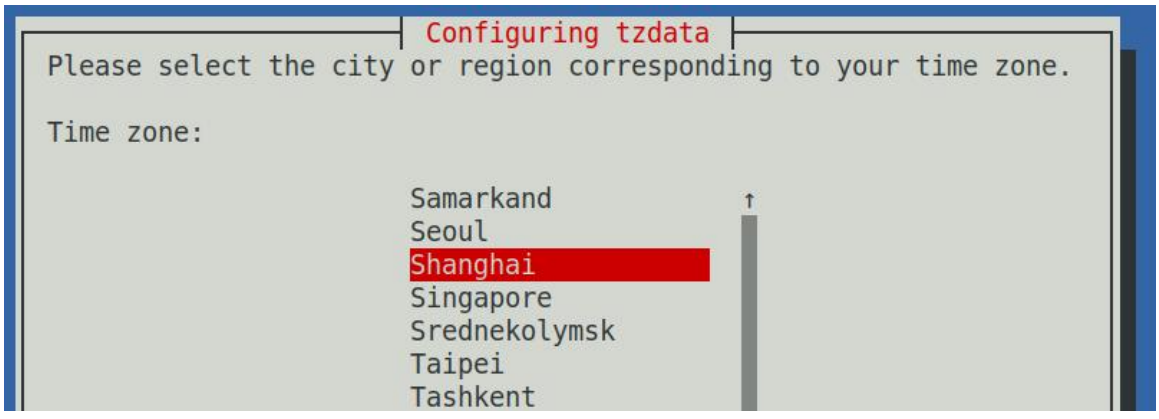
a. Execute the following command

```
root@orangepi:~# dpkg-reconfigure tzdata
```

b. Then select the geographic region as **Asia**



c. Then select the time zone as **Shanghai**



d. After the configuration is completed, use the date command to view the time and it will be normal

```
root@orangepi:~# date
```

6) If the current time of the system is incorrect, please connect to the network first, and then use the following command to synchronize the time. The reason why the system time is set correctly here is to prepare for the time synchronization of the RTC module

later.

```
root@orangepi:~# apt -y update
root@orangepi:~# apt install ntpdate
root@orangepi:~# ntpdate 0.cn.pool.ntp.org
```

7) The command to view the current time of the RTC module is as follows, because the rtc specified by hwclock by default is the rtc0 device, so you need to use the -f option to specify the rtc1 device

```
root@orangepi:~# hwclock -r -f /dev/rtc1
```

8) The time read by the RTC module for the first time is definitely wrong. You can use the following command to synchronize the current time of the system to the RTC module. Before synchronization, ensure that the current time of the system is correct.

```
root@orangepi:~# date                    #First make sure that the current system time is correct
root@orangepi:~# hwclock -w -f /dev/rtc1  #Then write the system time into the RTC module
root@orangepi:~# hwclock -r -f /dev/rtc1  #Finally read the time of the RTC module to confirm that the settings are correct
```

9) If you confirm that the time in the RTC module is correct, you can unplug the power, then power on, and execute the following command to synchronize the time in the RTC module to the system

```
root@orangepi:~#
echo "ds1307 0x68" > /sys/class/i2c-adapter/i2c-2/new_device   #Add rtc device
root@orangepi:~# hwclock -s -f /dev/rtc1   #Then synchronize the RTC module time to the system
root@orangepi:~# date                      #Finally use the date command to check whether the system time is correct
```

10) The above operation is to manually synchronize the time of the RTC module to the system. If you need to automatically synchronize the system time at startup, you need to set the startup script according to the following operations to automatically synchronize the system time

a. Create rc-local.service file

```
root@orangepi:~# sudo vi /etc/systemd/system/rc-local.service
```

Copy the following content into the rc-local.service file

```
[Unit]
Description=/etc/rc.local Compatibility
ConditionPathExists=/etc/rc.local

[Service]
Type=forking
ExecStart=/etc/rc.local start
TimeoutSec=0
StandardOutput=tty
RemainAfterExit=yes
SysVStartPriority=99

[Install]
WantedBy=multi-user.target
```

b. Create rc.local file

```
root@orangepi:~# sudo vi /etc/rc.local
```

c. Copy the following content into the rc-local.service file

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
```

```
echo "ds1307 0x68" > /sys/class/i2c-adapter/i2c-2/new_device
hwclock -s -f /dev/rtc1

exit 0
```

d. Add permissions to rc.local

```
root@orangepi:~# sudo chmod +x /etc/rc.local
```

e. Enable service

```
root@orangepi:~# sudo systemctl enable rc-local
```

f. Start the service and check the status

```
root@orangepi:~# sudo systemctl start rc-local.service
root@orangepi:~# sudo systemctl status rc-local.service
```

11) At this point, you can disconnect all the network connections of the development board, wait a few minutes, restart the system, and then check the system time, and you will find that the system time is correct even if there is no network.

3.35. Reset and shutdown methods

1) Use the poweroff command to shut down

```
root@orangepi:~# poweroff
```

2) You can also short press the power button on the development board to reset

3) The command to restart the linux system is

```
root@orangepi:~# reboot
```

4. Linux SDK instructions

The compilation of the Linux SDK is performed on a PC or virtual machine (VirtualBox or VMware) with Ubuntu 18.04 installed. Please do not use other versions of the Ubuntu system or compile the Linux SDK on WSL

4. 1. Get the source code of linux sdk

4. 1. 1. Download orangepi-build from github

1) First download the code of orangepi-build. The code of orangepi-build is modified based on the armbian build system. At present, the RK3399 series development board only supports the legacy branch.

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

To download the code of orangepi-build through the git clone command, you do not need to enter the username and password of the github account (the other codes in this manual are also the same), if you enter the git clone command, the Ubuntu PC prompts

the user who needs to enter the github account Name and password, usually the address of the orangepi-build warehouse behind git clone is entered incorrectly. First of all, please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account.

2) The u-boot and linux kernel versions currently used by the RK3399 series development board are as follows

Branch	u-boot version	linux kernel version
legacy	u-boot 2020.04	linux4.4
~~current~~	~~In development~~	~~In development~~

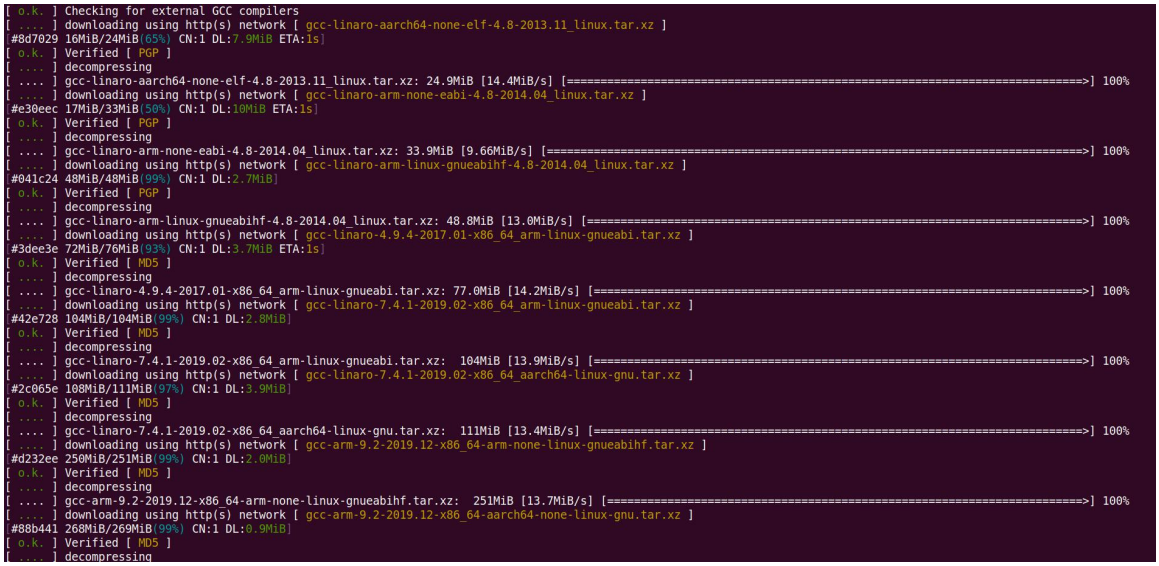
3) After orangepi-build is downloaded, it will contain the following files and folders

a. **build.sh**: Compile the startup script
b. **external**: Contains the configuration files needed to compile the image, specific scripts and the source code of some programs, etc.
c. **LICENSE**: GPL 2 license file
d. **README.md**: orangepi-build description file GPL 2 license file
e. **scripts**: General script for compiling linux image

```
test@test:~/orangepi-build$ ls
build.sh  external  LICENSE  README.md  scripts
```

4.1.2. Download the cross-compilation tool chain

1) When orangepi-build is run for the first time, it will automatically download the cross-compilation toolchain and place it in the toolchains folder. Every time the orangepi-build build.sh script is run, it will check whether the cross-compilation toolchain in toolchains exists. , If it does not exist, it will restart the download, if it exists, it will be used directly, and the download will not be repeated



2) The image URL of the cross-compilation tool chain in China is the open source software image site of Tsinghua University

```
https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/
```

3) After toolchains is downloaded, it will contain multiple versions of cross-compilation toolchains

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

4) The cross-compilation tool chain used to compile the Orange Pi 4 linux4.4 kernel source code is

```
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
```

5) The cross-compilation tool chain used to compile the Orange Pi 4 u-boot v2020.04 source code is

gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu

4. 1. 3. Orangepi-build complete directory structure description

1) After the orangepi-build repository is downloaded, it does not contain the linux kernel, u-boot source code, and cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository **(please do not download and use the kernel and u separately -boot source code to compile, unless you know how to use it)**）

a. The git repository where the linux kernel source code is stored is as follows, pay attention to switch the branch of the linux-orangepi repository to **orange-pi-4.4-rockchip64**

https://github.com/orangepi-xunlong/linux-orangepi/tree/**orange-pi-4.4-rockchip64**

b. The git repository where u-boot source code is stored is as follows, pay attention to switch the branch of u-boot-orangepi repository to **v2020.04-rockchip64**

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/**v2020.04-rockchip64**

2) When orangepi-build runs for the first time, it will download the cross-compilation tool chain, u-boot and linux kernel source code. After successfully compiling a linux image, the files and folders that can be seen in orangepi-build are:

a. **a.build.sh:** Compile the startup script
b. **b.external:** Contains the configuration files needed to compile the image, scripts for specific functions, and the source code of some programs. The rootfs compressed package cached during the compiling of the image is also stored in external
c. **c.kernel:** Stores the source code of the Linux kernel. The folder named orange-pi-4.4-rockchip64 stores the kernel source code of the legacy branch of the RK3399 series development board. Please do not modify the name of the kernel source folder manually. If you modify it , The kernel source code will be re-downloaded when the build system is running
d. **d.LICENSE:** GPL 2 license file
e. **e.README.md:** orangepi-build instruction file
f. **f.output:** store files such as u-boot, linux and other deb packages generated by compilation, compilation logs, and images generated by compilation
g. **g.scripts:** general scripts for compiling linux images

h. **h.toolchains:** store cross-compilation toolchains

i. **iu-boot:** Store the source code of u-boot. The folder named v2020.04-rockchip64 inside stores the u-boot source code of the legacy branch of the RK3399 series development board. Please do not modify the name of the u-boot source code folder manually , If modified, the u-boot source code will be re-downloaded when the compiling system is running

j. **j.userpatches:** store configuration files needed to compile scripts

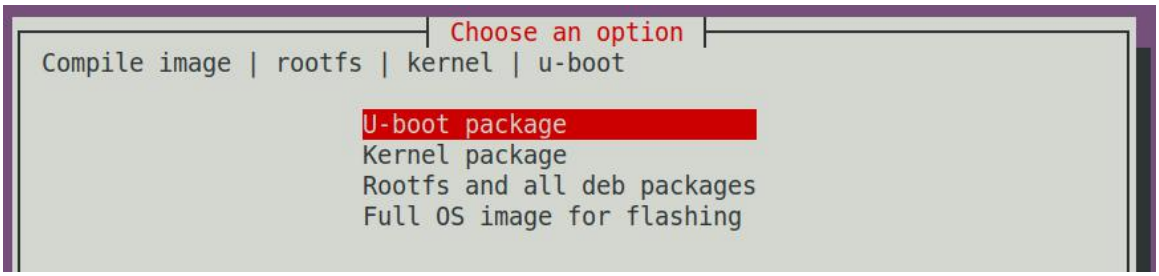
```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts
toolchains  u-boot  userpatches
```

4. 2. Compile u-boot

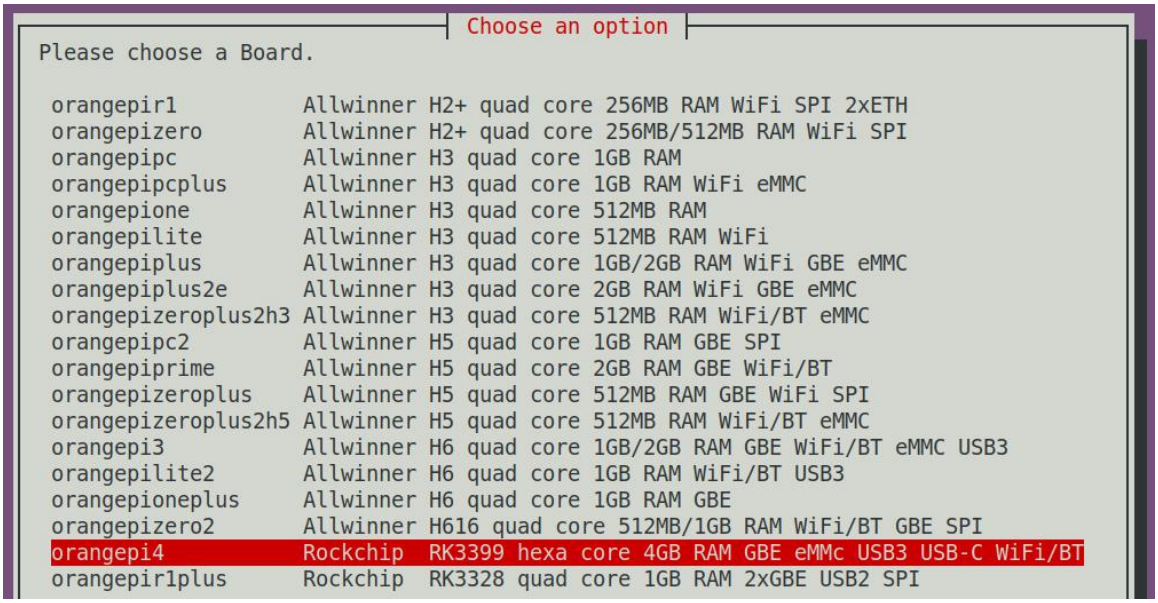
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **U-boot package**, then press Enter



3) Then select the model of the development board



4) Then it will start to compile u-boot, some of the information prompted during compiling are explained as follows

a. u-boot source version

[o.k.] Compiling u-boot [**v2020.04**]

b. The version of the cross-compilation toolchain

[o.k.] Compiler version [**aarch64-none-linux-gnu-gcc 9.2.1**]

c. The path of the u-boot deb package generated by compiling

[o.k.] Target directory [**orangepi-build/output/debs/u-boot**]

d. The package name of the compiled u-boot deb package

[o.k.] File name [**linux-u-boot-legacy-orangepi4_2.1.0_arm64.deb**]

e. Compilation time

[o.k.] Runtime [**1 min**]

f. Repeat the command to compile u-boot, use the following command without selecting through the graphical interface, you can start compiling u-boot directly

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepi4 BRANCH=legacy BUILD_OPT=u-boot KERNEL_CONFIGURE=no**]

5) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-legacy-orangepi4_2.1.0_arm64.deb
```

6) The files contained in the generated u-boot deb package are as follows

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \
linux-u-boot-legacy-orangepi4_2.1.0_arm64.deb .      (Note that there is a "." at the
  end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-legacy-orangepi4_2.1.0_armhf.deb    usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-legacy-orangepi4_2.1.0_arm64
    │   ├── idbloader.bin
    │   ├── trust.bin
    │   └── uboot.img
    └── u-boot
        ├── LICENSE
        ├── orangepi-4-rk3399_defconfig
        └── platform_install.sh

3 directories, 6 files
```

7) When the orangepi-bulid build system compiles the u-boot source code, it will first synchronize the u-boot source code with the u-boot source code of the github server, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code **(You need to compile u-boot once to turn off this function, otherwise you will be prompted that u-boot's source code cannot be found),** otherwise the 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"
```

8) When debugging u-boot code, you can use the following method to update u-boot in the linux image for testing

a. Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-legacy-orangepi4_2.1.0_arm64.deb root@192.168.1.xxx:/root
```

b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-u-boot-orangepi4-legacy
```

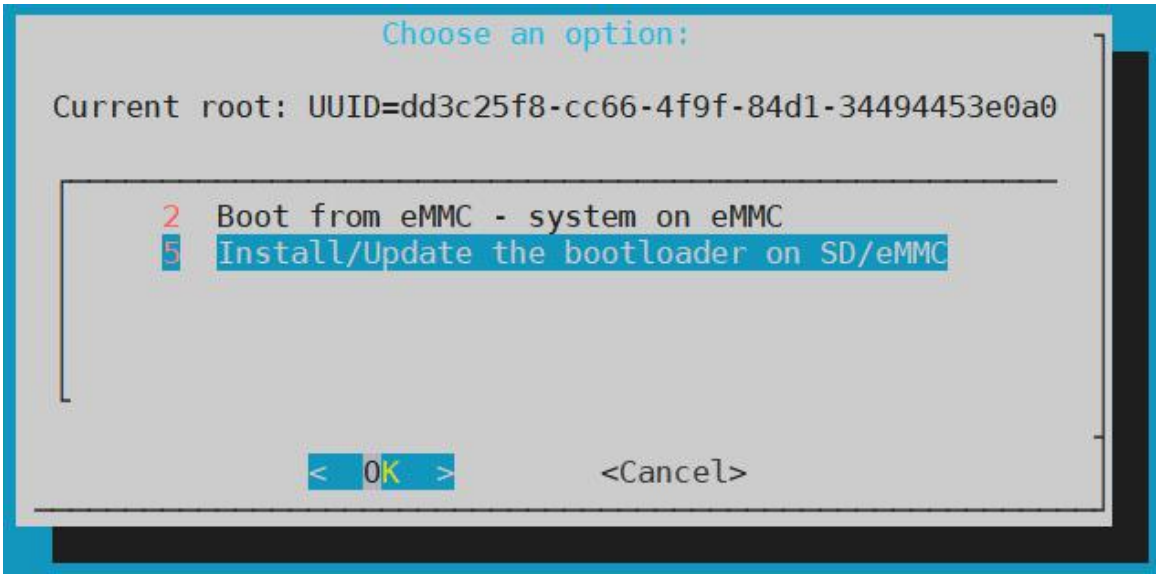
c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-legacy-orangepi4_2.1.0_arm64.deb
```

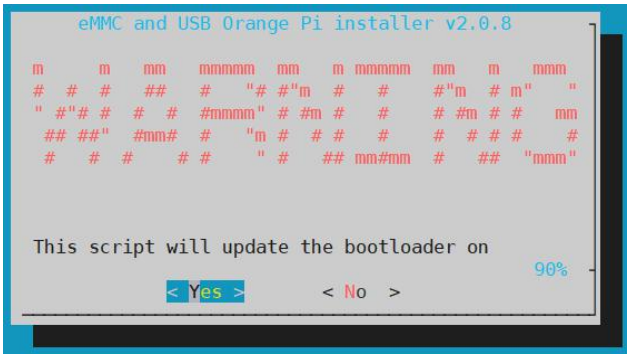
d. Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

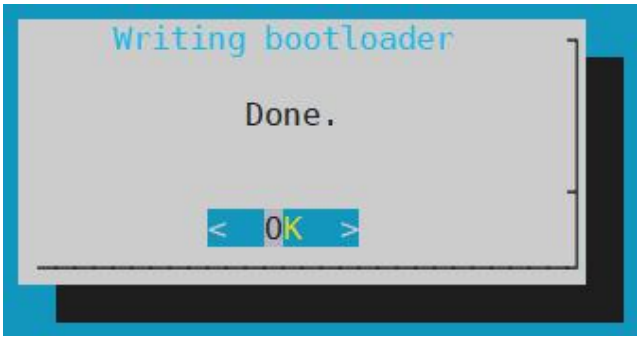
e. Then select **5 Install/Update the bootloader on SD/eMMC**



f. After pressing the Enter key, a Warring will pop up first



g. Press Enter again to start updating u-boot, and the following information will be displayed after the update is complete



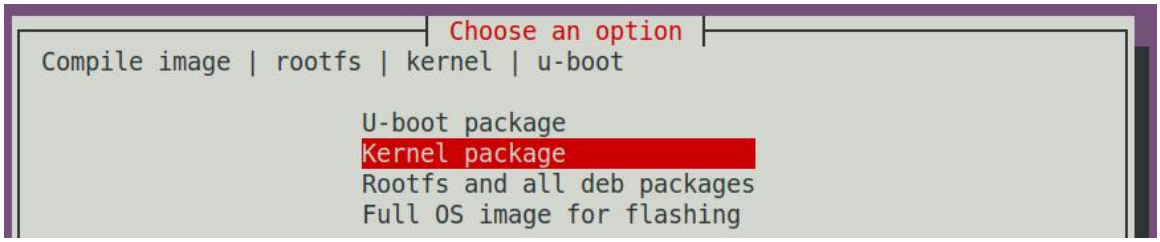
h. Then you can restart the development board to test whether the u-boot modification takes effect

4. 3. Compile the linux kernel

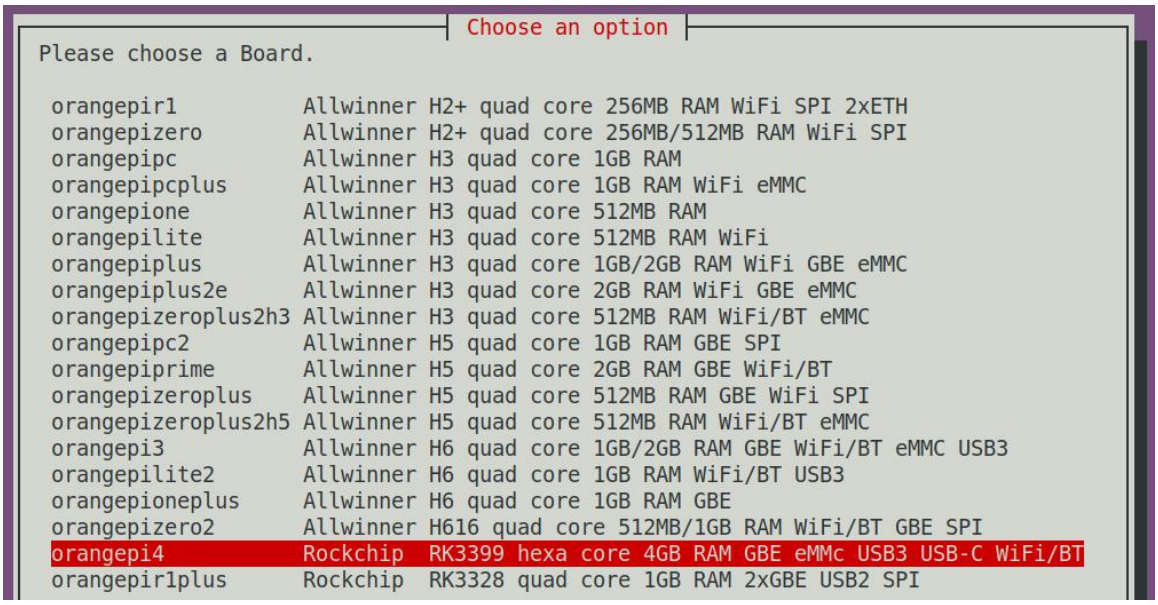
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

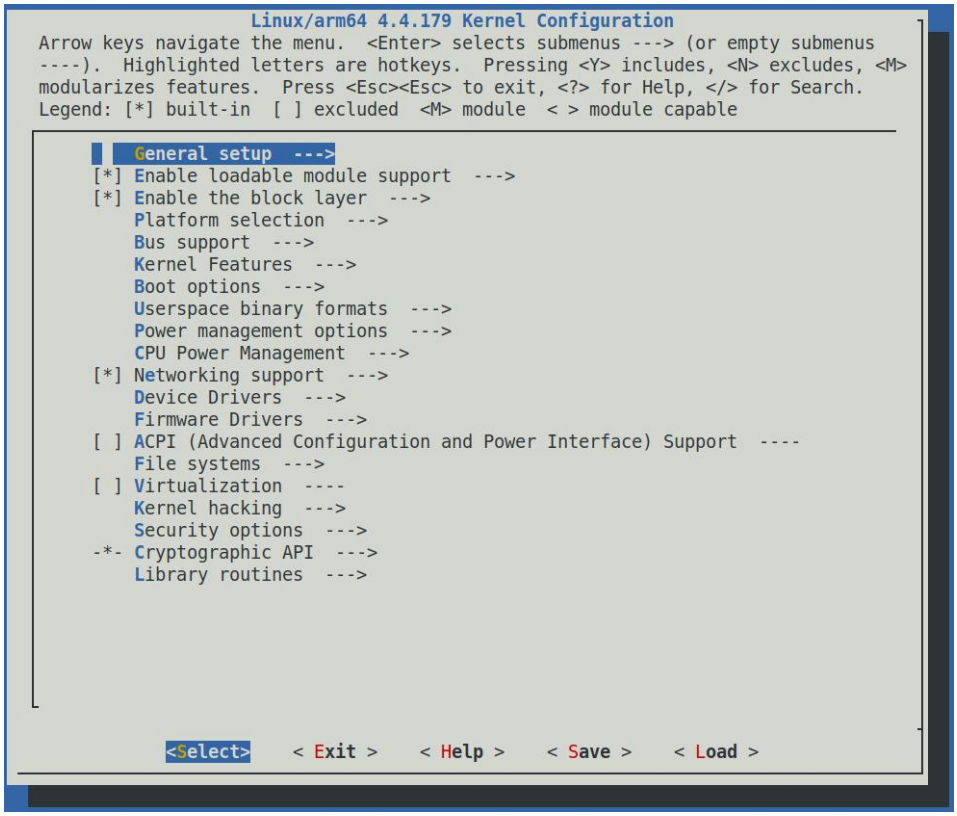
2) Select **Kernel package**, and then press Enter



3) Then select the model of the development board



4) Then the kernel configuration interface opened by make menuconfig will pop up. At this time, you can directly modify the kernel configuration. If you don't need to modify the kernel configuration, just exit it. After exiting, the kernel source code will be compiled.



a. If you do not need to modify the configuration options of the kernel, when you run the build.sh script, pass in KERNEL_CONFIGURE=no to temporarily block the pop-up kernel configuration interface

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

b. You can also set **KERNEL_CONFIGURE=no** in the orangepi-build/userpatches/config-default.conf configuration file to disable this feature permanently

c. If the following error is prompted when compiling the kernel, this is because the terminal interface of the Ubuntu PC is too small and the make menuconfig interface cannot be displayed. Please adjust the terminal of the Ubuntu PC to the maximum, and then re-run the build.sh script

```
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf  Kconfig
Your display is too small to run Menuconfig!
It must be at least 19 lines by 80 columns.
scripts/kconfig/Makefile:28: recipe for target 'menuconfig' failed
make[1]: *** [menuconfig] Error 1
Makefile:560: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
[ error ] ERROR in function compile_kernel [ compilation.sh:376 ]
[ error ] Error kernel menuconfig failed
[ o.k. ] Process terminated
```

5) Part of the information prompted when compiling the kernel source code is explained as follows

a. The version of the linux kernel source code

[o.k.] Compiling legacy kernel [**4.4.179**]

b. The version of the cross-compilation tool chain used

[o.k.] Compiler version [**aarch64-linux-gnu-gcc 7.4.1**]

c. The configuration file used by the kernel by default and the path where it is stored

[o.k.] Using kernel config file [**config/kernel/linux-rk3399-legacy.config**]

d. If KERNEL_CONFIGURE=yes, the final configuration file .config used by the kernel will be copied to **output/config**. If the kernel configuration is not modified, the final configuration file is consistent with the default configuration file.

[o.k.] Exporting new kernel config [**output/config/linux-rk3399-legacy.config**]

e. The path of the deb package related to the kernel generated by the compilation

[o.k.] Target directory [**output/debs/**]

f. The package name of the compiled kernel image deb package

[o.k.] File name [**linux-image-legacy-rk3399_2.1.0_arm64.deb**]

g. Compilation time

```
[ o.k. ] Runtime [ 5 min ]
```

h. At the end, it will display the compiling command to recompile the kernel selected last time, use the following command without selecting through the graphical interface, you can directly start compiling the kernel source code

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh    BOARD=orangepi4 BRANCH=legacy BUILD_OPT=kernel KERNEL_CONFIGURE=yes ]
```

6) View the deb package related to the kernel generated by the compilation

a. **linux-dtb-legacy-rk3399_2.1.0_arm64.deb** Not used yet, don't care about it

b. **linux-headers-legacy-rk3399_2.1.0_arm64.deb** Contains kernel header files

c. **linux-image-legacy-rk3399_2.1.0_arm64.deb** Contains kernel image and kernel module

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-legacy-rk3399_2.1.0_arm64.deb
output/debs/linux-headers-legacy-rk3399_2.1.0_arm64.deb
output/debs/linux-image-legacy-rk3399_2.1.0_arm64.deb
```

7) The files contained in the generated linux-image deb package are as follows

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-legacy-rk3399_2.1.0_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-legacy-rk3399_2.1.0_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-legacy-rk3399_2.1.0_arm64.deb  usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi-build/output/debs/test$ tree -L 2
.
├── boot
```

```
│   ├── config-4.4.179-rk3399          //Configuration file used to compile the
kernel source code
│   ├── System.map-4.4.179-rk3399
│   └── vmlinux-4.4.179-rk3399         //Compile the generated kernel image file
├── etc
│   └── kernel
├── lib
│   └── modules                        //Compile the generated kernel module
├── linux-image-legacy-rk3399_2.1.0_arm64.deb
└── usr
    ├── lib
    └── share

8 directories, 4 files
```

8) When the orangepi-bulid compilation system compiles the linux kernel source code, it first synchronizes the linux kernel source code with the linux kernel source code of the github server, so if you want to modify the source code of the linux kernel, you first need to turn off the update function of the source code (you need to compile the linux kernel once. This function can only be turned off after the source code, otherwise it will be prompted that the source code of the linux kernel cannot be found), otherwise the 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 you modify the kernel, 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-legacy-rk3399_2.1.0_arm64.deb root@192.168.1.207:/root
```

b. Then log in to the development board and uninstall the deb package of the installed linux kernel

```
root@orangepi:~# apt purge -y linux-image-legacy-rk3399
```

c. Install the deb package of the new linux kernel just uploaded

```
root@orangepi:~# dpkg -i linux-image-legacy-rk3399_2.1.0_arm64.deb
```

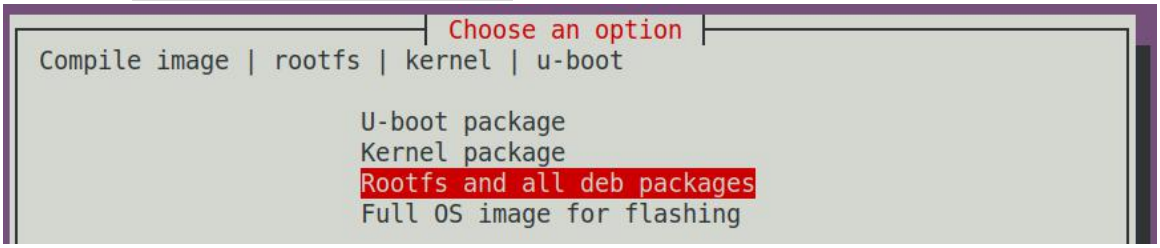
d. Then restart the development board, and then check whether the kernel-related changes have taken effect

4. 4. Compile rootfs

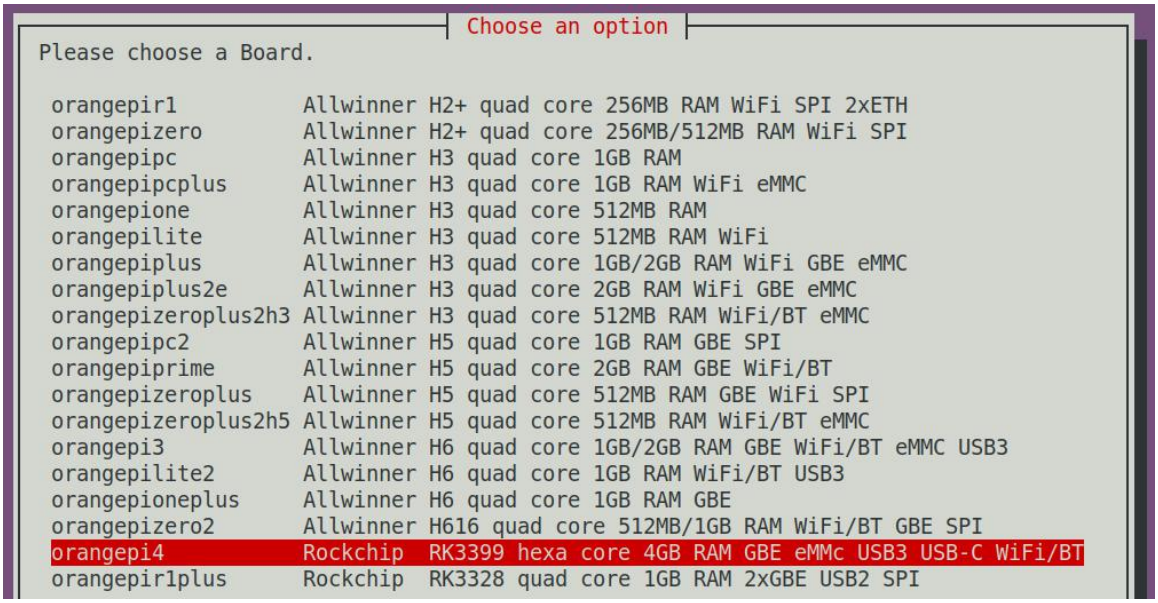
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Rootfs and all deb packages**，and then press Enter



3) Then select the model of the development board

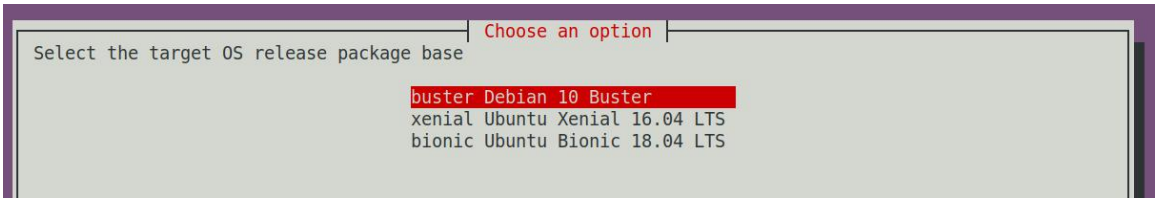


4) Then select the type of rootfs

a. **Buster** means Debian 10

b. **Xenial** means Ubuntu 16.04

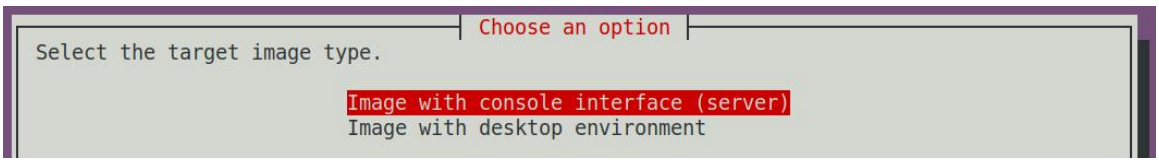
c. **Bionic** means Ubuntu 18.04



5) Then select the type of image

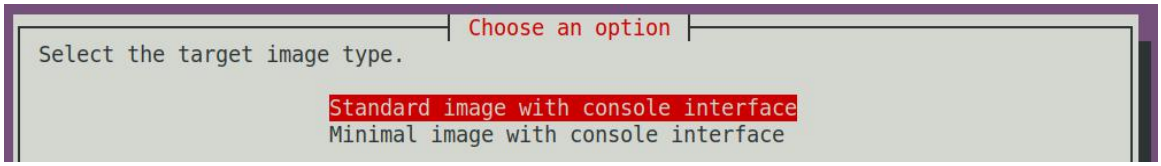
a. **Image with console interface (server)** Represents the image of the server version, which is relatively small

b. **Image with desktop environment** Indicates a image with a desktop, which is relatively large



6) If it is to compile the image of the server version, you can also choose to compile the Standard version or the Minimal version. The pre-installed software of the Minimal

version will be much less than the Standard version.



7) After selecting the type of image, rootfs will start to compile. Part of the information prompted during compilation is explained as follows

a. Type of rootfs

[o.k.] local not found [Creating new rootfs cache for **xenial**]

b. The storage path of the compiled rootfs compressed package

[o.k.] Target directory [**external/cache/rootfs**]

c. The name of the rootfs compressed package generated by the compilation

[o.k.] File name
[**xenial-desktop-arm64.45500b0cedbf73ce2f05b2f61d6832d0.tar.lz4**]

d. Compilation time

[o.k.] Runtime [**13 min**]

e. Repeat the command to compile rootfs, use the following command without selecting through the graphical interface, you can directly start compiling rootfs

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepi4 BRANCH=legacy BUILD_OPT=rootfs RELEASE=xenial BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes**]

8) View the compiled rootfs compressed package

a. xenial-desktop-arm64.45500b0cedbf73ce2f05b2f61d6832d0.tar.lz4 is a compressed package of rootfs, the meaning of each field of the name is

a) **xenial** represents the type of linux distribution of rootfs

b) **desktop** means rootfs is the desktop version type, if it is cli, it means the server version type

c) **arm64** represents the architecture type of rootfs

d) **45500b0cedbf73ce2f05b2f61d6832d0** 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 Need to recompile rootfs

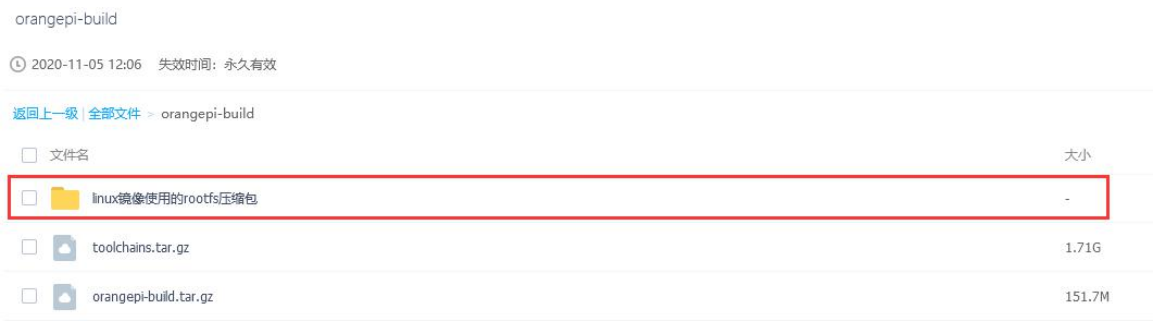
b. xenial-desktop-arm64.45500b0cedbf73ce2f05b2f61d6832d0.tar.lz4.list lists the package names of all packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
xenial-desktop-arm64.45500b0cedbf73ce2f05b2f61d6832d0.tar.lz4
xenial-desktop-arm64.45500b0cedbf73ce2f05b2f61d6832d0.tar.lz4.list
```

9) If the required rootfs already exists under external/cache/rootfs, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to external/cache/rootfs to find out whether it is already Rootfs with cache available, if available, use it directly, which can save a lot of downloading and compiling time

10) Because it takes a long time to compile rootfs, if you don't want to compile rootfs from scratch, or if there is a problem with compiling rootfs, you can directly download the rootfs compressed package cached by Orange Pi. The download link of rootfs compressed package Baidu cloud disk is shown below, download A good rootfs compressed package needs to be placed in the external/cache/rootfs directory of orangepi-build to be used normally by the compiled script

link:：https://pan.baidu.com/s/1vWQmCmSYdH7iCDFyKpJtVw
Extraction code: zero

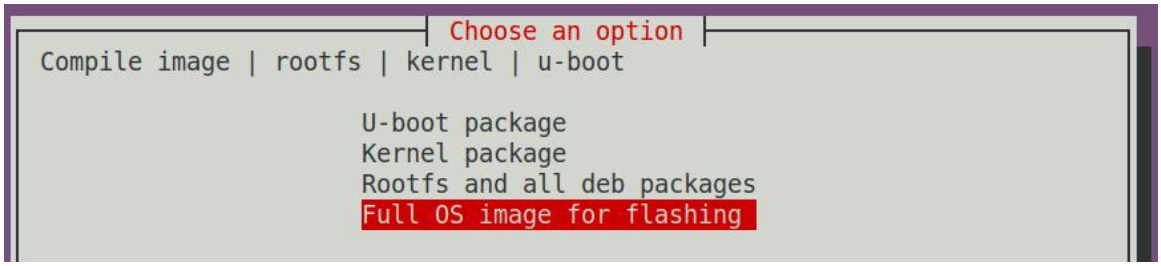


4. 5. Compile linux image

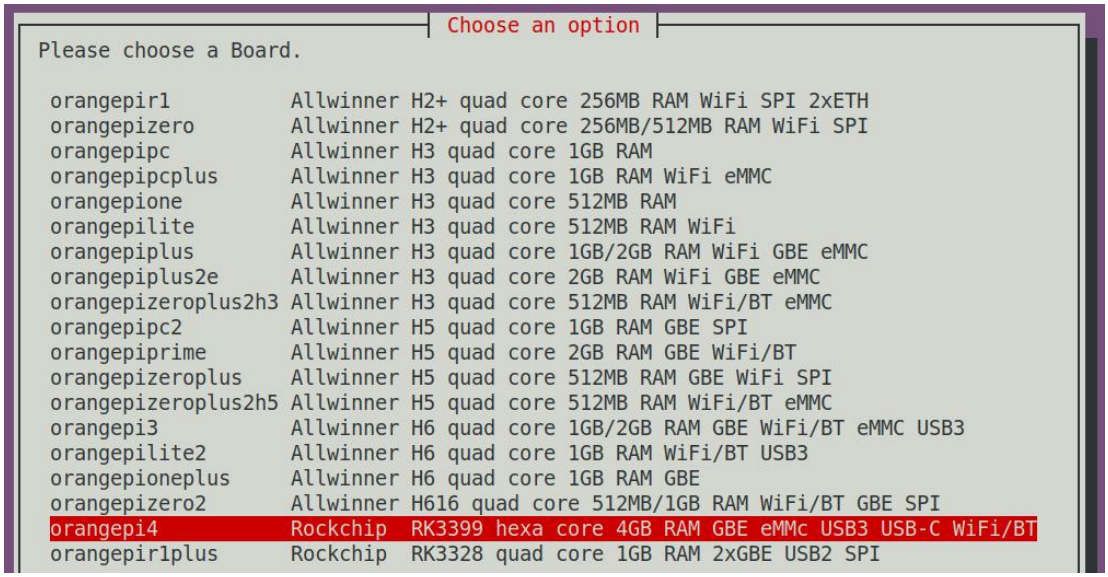
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Full OS image for flashing**, and then press Enter

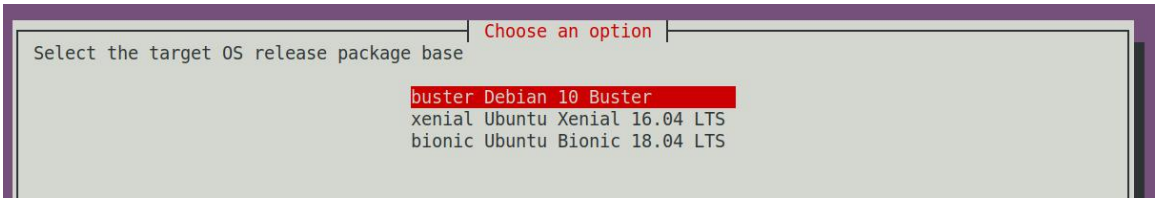


3) Then select the model of the development board



4) Then select the type of rootfs

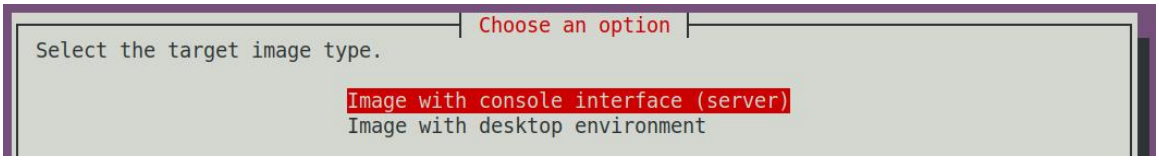
a. **Buster means** Debian 10
b. **Xenial means** Ubuntu 16.04
c. **Bionic means** Ubuntu 18.04



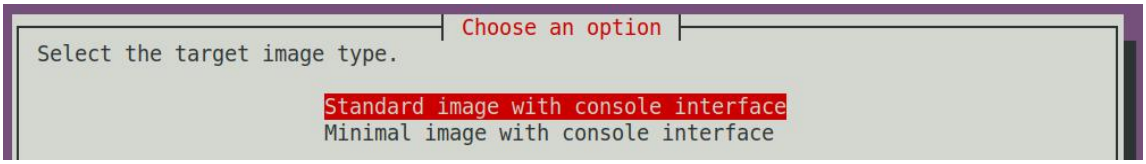
5) Then select the type of image

a. **Image with console interface (server)** indicates the image of the server version, which is relatively small

b. **Image with desktop environment** indicates a image with a desktop, which is relatively large



6) If it is to compile the image of the server version, you can also choose to compile the Standard version or the Minimal version. The pre-installed software of the Minimal version will be much less than the Standard version.



7) After selecting the type of image, it will start to compile the linux image. The general process of compilation is as follows

a. Initialize the compilation environment of Ubuntu PC and install the software packages needed for the compilation process

b. Download the source code of u-boot and linux kernel (if it has been cached, only update the code)

c. Compile u-boot source code and generate u-boot deb package

d. Compile linux source code, generate linux related deb package

e. Make deb package of linux firmware

f. Make deb package of orangepi-config tool

g. Make board-level support deb package

h. If it is to compile the desktop version image, the desktop related deb package

will also be made

i. Check whether the rootfs has been cached, if there is no cache, re-create the rootfs, if it has been cached, just unzip and use
j. Install the previously generated deb package into rootfs
k. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configuration, etc.
l. Then make an image file and format the partition, the default type is ext4
m. Then copy the configured rootfs to the image partition
n. Then update the initramfs
o. Finally, write the bin file of u-boot to the image through the dd command

8) After compiling the image, the following information will be prompted

a. The storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/orangepi4_2.1.0_ubuntu_xenial_server_linux4.4.170/orangepi4_2.1.0_ubuntu_xenial_server_linux4.4.170.img ]
```

b. Compilation time

```
[ o.k. ] Runtime [ 19 min ]
```

c. Repeat the command to compile the image, use the following command without selecting through the graphical interface, you can directly start to compile the image

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepi4 BRANCH=legacy BUILD_OPT=image RELEASE=xenial BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

5. Android system instructions

5.1. Supported Android version

Android version	Kernel version
Android 8.1	linux4.4

5.2. Android 8.1 feature adaptation

Function	status
HDMI video	OK
HDMI audio	OK
USB2.0 x 2	OK
USB3.0 x 1	OK
TypeC USB3.0	OK
TF card boot	OK
EMMC starts	OK
Network card	OK
WIFI	OK
Bluetooth	OK
Bluetooth earphone	OK
Headphone recording	OK
Headphone playback	OK
Microphone recording	OK
LED lights	OK
Temperature Sensor	OK
USB camera	OK
GPU	OK
Video codec	OK
Reset button	OK
Upgrade key	OK
ADB debugging	OK
OV13850 camera	OK

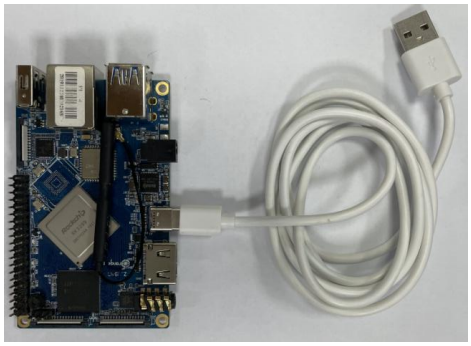
10.1 inch MIPI screen	OK
mini-PCIE	OK
TypeC to HDMI	OK

5.3. Onboard LED light display description

	Green light	Red light
u-boot startup phase	off	on
Kernel boot to enter the system	Flashing	on

5.4. How to use ADB

1) First, you need to use the Type-C USB interface data cable to connect the development board to the USB interface of the computer (please use a DC power supply to power the development board at the same time)



2) Install adb tool on Ubuntu PC

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

3) View the identified ADB device

```
test@test:~$ adb devices
List of devices attached
S63QCF54CJ      device
test@test:~$ lsusb
Bus 003 Device 006: ID 2207:0006
```

4) Then you can log in to the android system through the adb shell on the Ubuntu PC

```
test@test:~$ adb shell
rk3399_mid:/ $
```

5) If you need to change the Android system files, you need to turn off the security verification, execute the following command

```
test@test:~$ adb root
test@test:~$ adb disable-verity
```

6) Then execute the command to restart the system

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

7) Remount the Android system

```
test@test:~$ adb root
test@test:~$ adb remount
```

8) Then you can transfer files to Android system

```
test@test:~$ adb push example.txt /system/
```

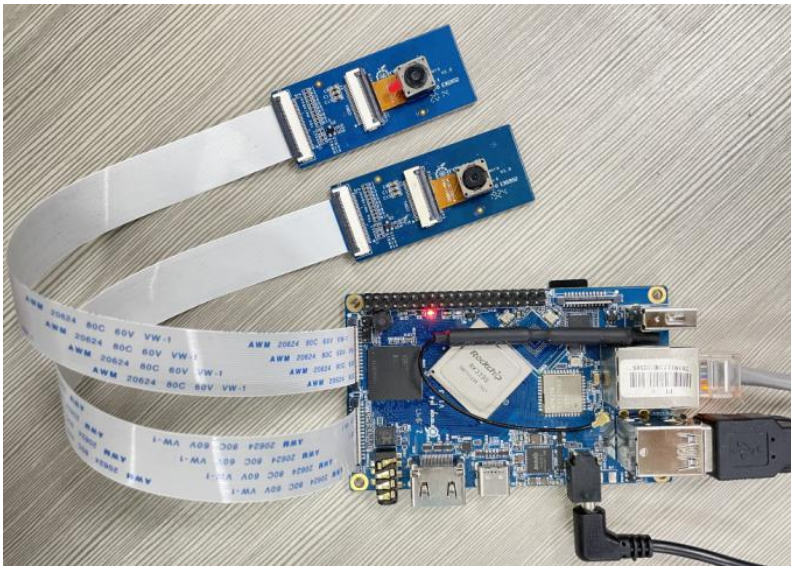
5. 5. How to use the OV13850 camera

1) OrangePi 4 has two Camera interfaces and supports OV13850 camera. The two Canera interfaces can use one of them separately, or use two Camera interfaces to connect two cameras at the same time. After connecting two cameras, one is front and the other is rear

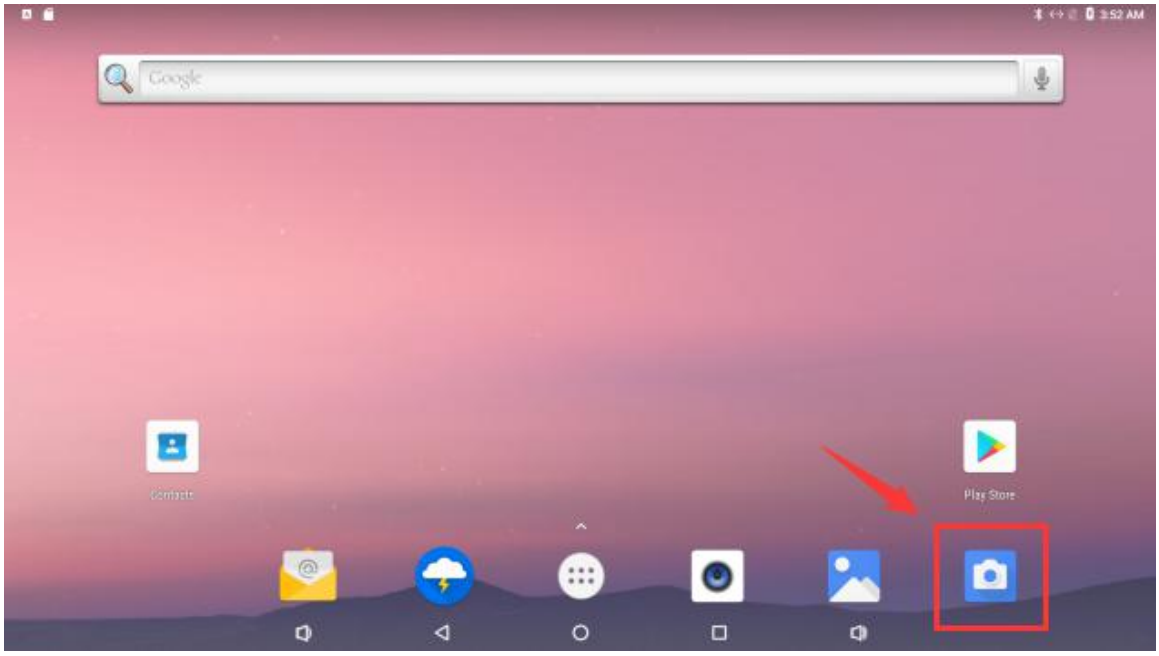
2) The OV13850 camera kit includes an OV13850 camera, an adapter board and a cable

3) First insert the OV13850 camera into the adapter board, and then insert the cable into another card slot of the adapter board

4) Then insert the other end of the cable into the Camera interface of the development board. The interface can be connected to two cameras at the same time or a single camera. Start the Android system after connecting the camera (do not insert the camera after power-on)



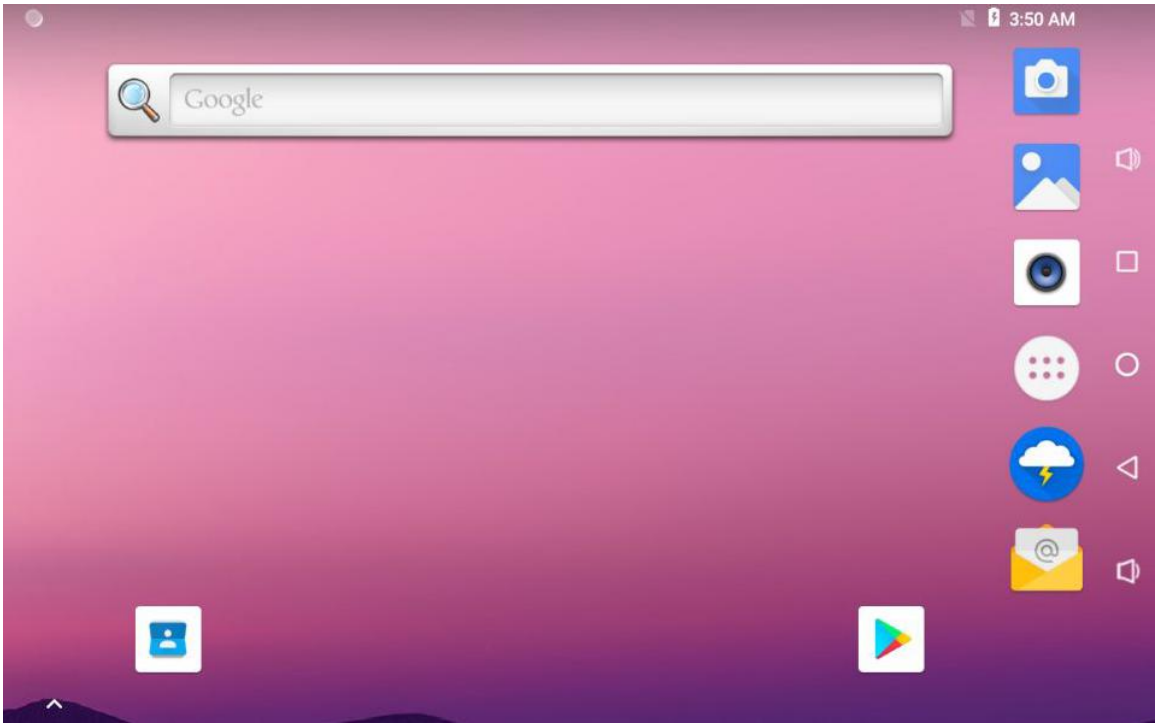
5) After the Android system is started, open the camera APP and you can see the output of the OV13850 camera. The location of the camera APP is shown in the figure below:



5. 6. How to use HDMI interface

The Android pre-installed on the Orange Pi 4 development board supports HDMI display. Since it also supports a 10.1-inch MIPI screen, its UI resolution is only 1280x800. After connecting the development board to the HDMI monitor or TV through the HDMI

to HDMI cable, the interface after system startup is as shown in the figure below

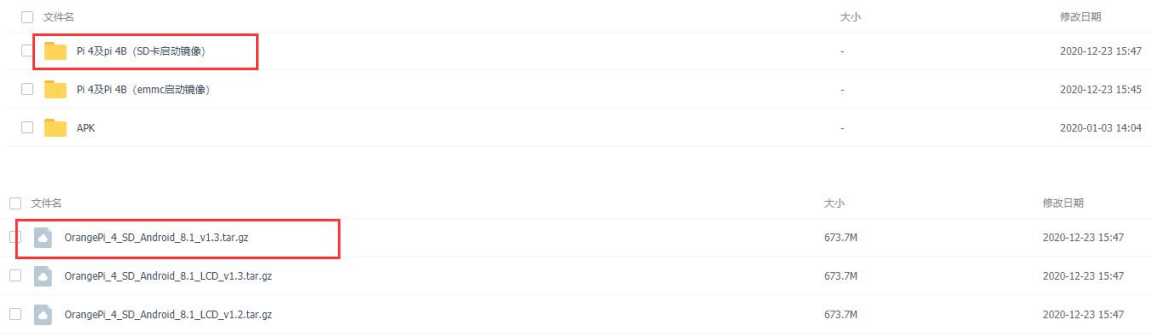


After testing all functions with the pre-installed system and confirming that there is no problem, in order to obtain a better display effect, it is recommended to re-burn the image that supports HDMI display. The steps are as follows:

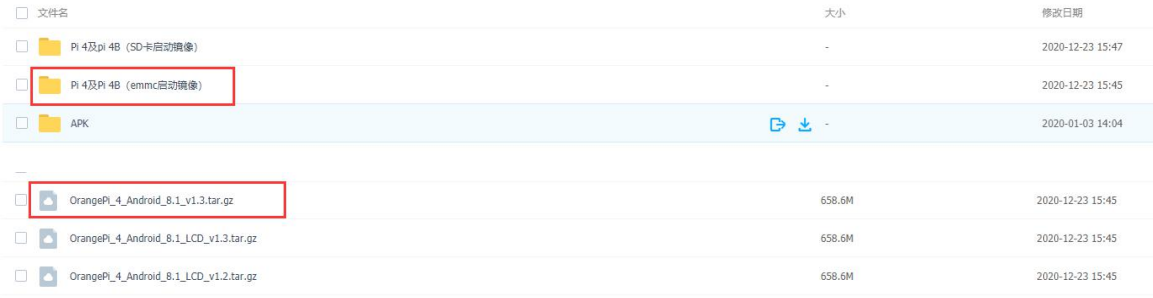
1) Download the image that supports HDMI display. There are Android image that support EMMC startup and those that support TF card startup. These two images cannot be mixed

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

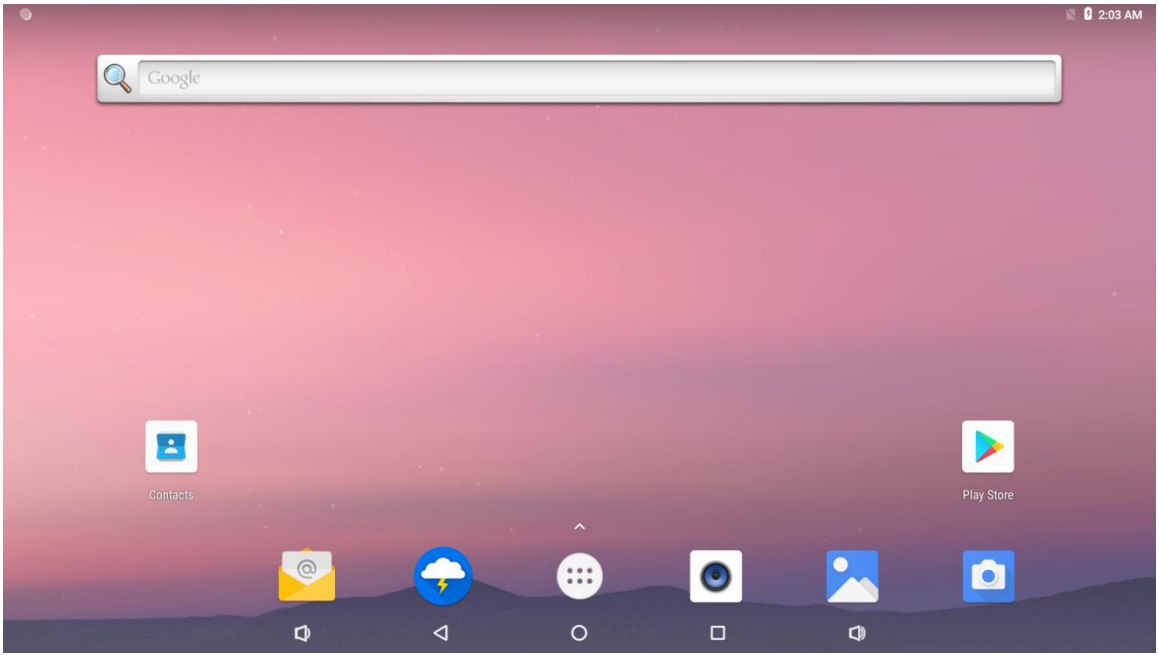
2) If you need to boot from a TF card, please select and download the Android image as shown in the red box as shown in the figure below, pay attention to selecting the one without LCD



3) If you need to start from EMMC, please select and download the Android image according to the content of the red box in the figure below, and pay attention to selecting the one without LCD



4) Use HDMI display image, after the system is started, the display interface is as shown in the figure below, and the UI resolution is 1920*1080



5.7. The method of displaying the system interface through the TypeC interface

1）The Android pre-installed on the Orange Pi 4 development board supports Type-C to HDMI display. You need to prepare a Type-C to HDMI cable, and connect the development board to an HDMI monitor or TV through the Type-C interface for display

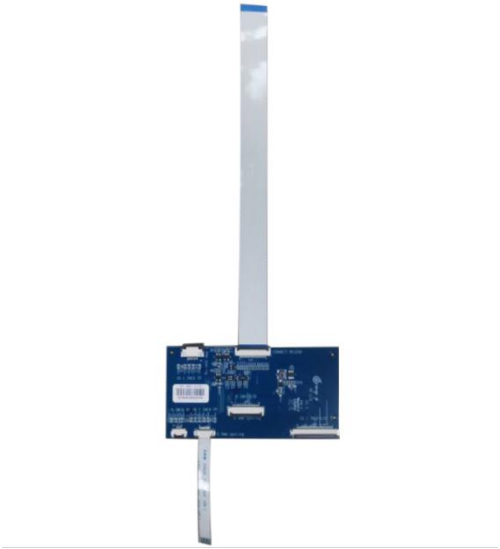
2）Regarding the UI resolution, please refer to how to use the HDMI interface

5.8. How to use the 0.1-inch MIPI screen

The Android system pre-installed on the Orange Pi 4 development board already supports MIPI screens. The connection method of MIPI screens is as follows:

1) 10.1 inch MIPI screen delivery list, including a touch screen, a MIPI LCD screen, a 31pin to 40pin cable, a 12pin touch screen cable, a 30pin MIPI cable, and an adapter board

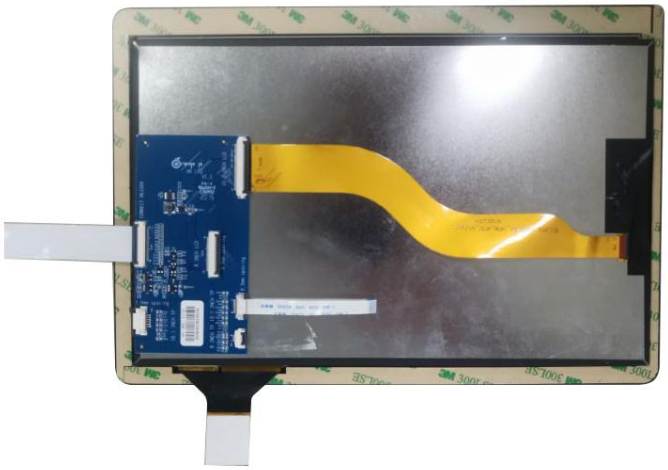
2) Connect the 12pin touch screen cable and 30pin MIPI cable to the adapter board according to the following figure. Note that the direction of the 12pin touch screen cable is the blue bar facing down.



3) With the touch screen facing down, stack the MIPI LCD screen on top of the touch screen as shown in the figure below

4) Place the connected adapter board on the MIPI LCD screen as shown in the figure below

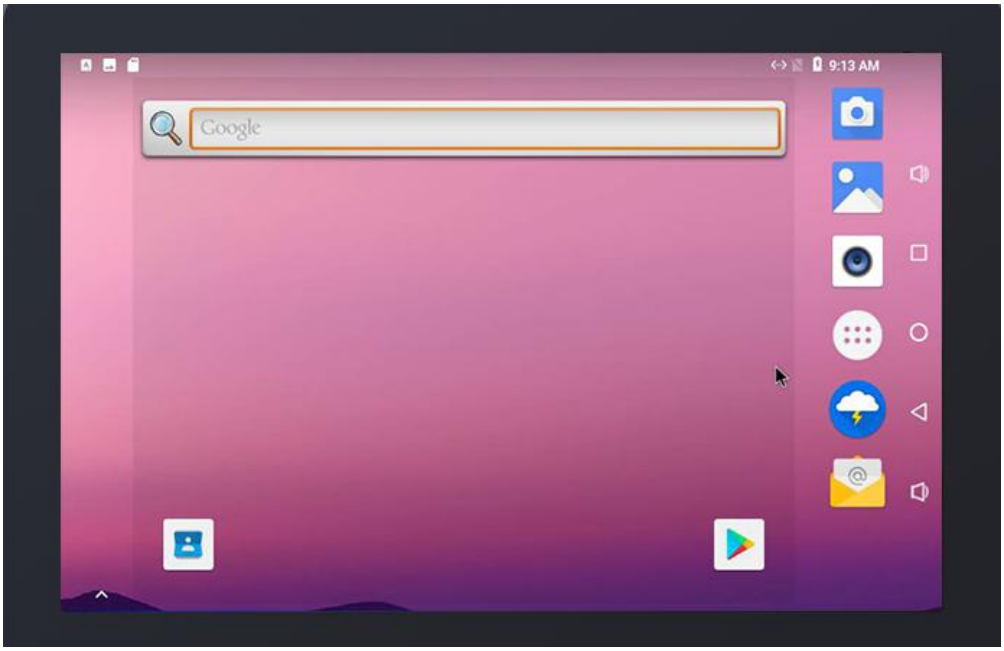
5) Connect MIPI LCD screen and adapter board through 31pin to 40pin cable



6) Connect the touch screen and the adapter board through a 12pin touch screen cable

7) Connect the adapter board to the LCD1 interface of Orange Pi 4 through a 30pin MIPI cable

8) After the development board is powered on, the interface defaults to the vertical screen after the system is started. If an HDMI display is connected at the same time, it will switch to landscape mode, the interface is as shown in the figure below



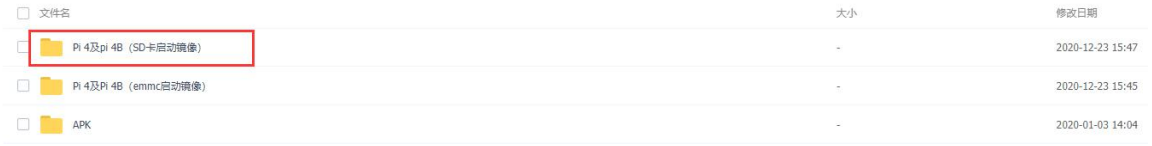
9) If the existing system of the development board is not the pre-installed Android system, please follow the steps below to burn the image that supports the 10.1-inch MIPI screen display

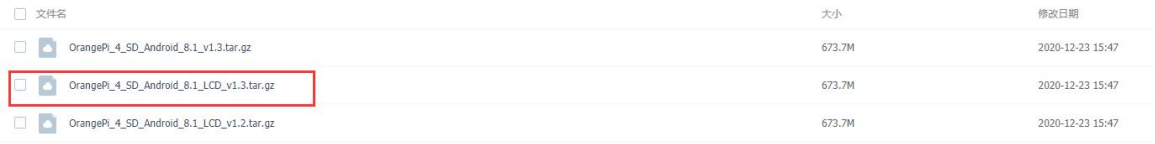
a. Download the image that supports the 10.1-inch MIPI screen display. There are Android image that support EMMC startup and those that support TF card startup. These two image cannot be mixed

b. htt

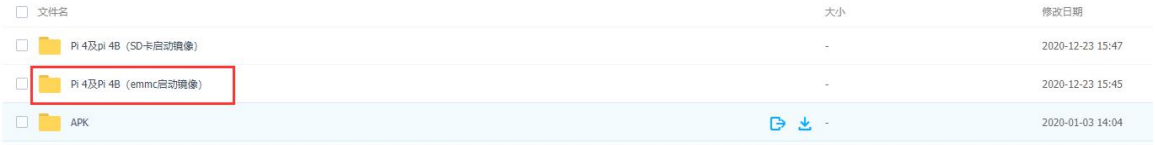
http://www.orangepi.cn/downloadresourcescn/

c. If you need to boot from a TF card, please select and download the Android image as shown in the red box as shown in the figure below, and pay attention to selecting the one with LCD





d. If you need to start from EMMC, please select and download the Android image as shown in the red box as shown below, and pay attention to selecting the one with LCD



e. Burn Android image that supports 10.1-inch MIPI screen display

5. 9. How to use USB camera

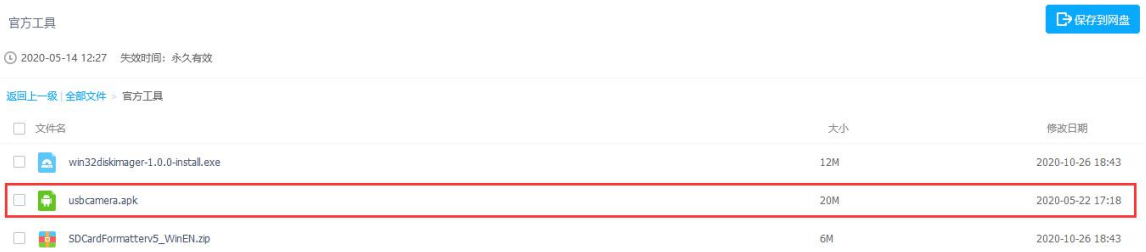
1) First insert the USB camera into the USB interface of the development board. If the USB camera is recognized normally, the corresponding video device node f will be generated under /dev

```
rk3399_mid:/ $ ls /dev/video*
/dev/video0
rk3399_mid:/ $ ls /sys/class/video4linux/ -lh
total 0
lrwxrwxrwx 1 root root 0 2020-09-30 03:29 video0 \
-> ../../devices/platform/usb@fe900000/fe900000.d0
```

2) Then make sure that the adb connection between the Ubuntu PC and the development

board is normal

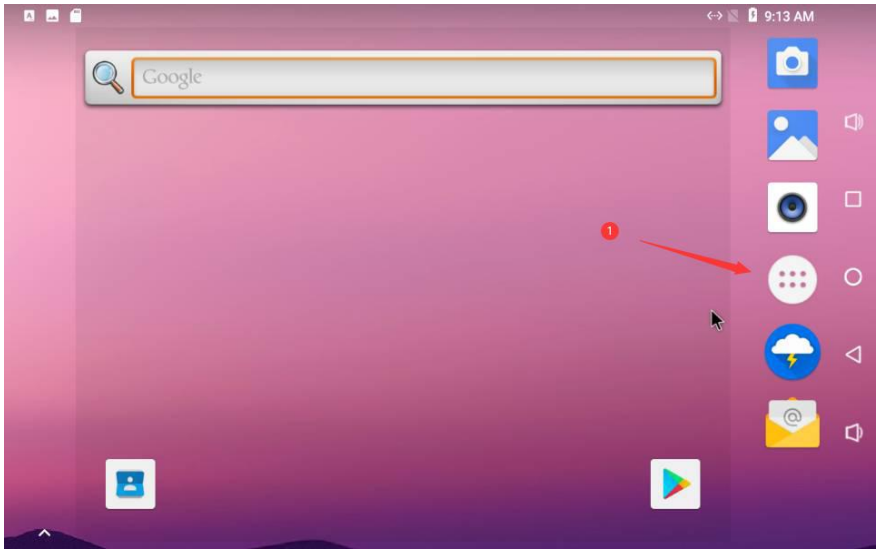
3) Download the USB camera test APP from the official tool on the Orange Pi 4 data download page

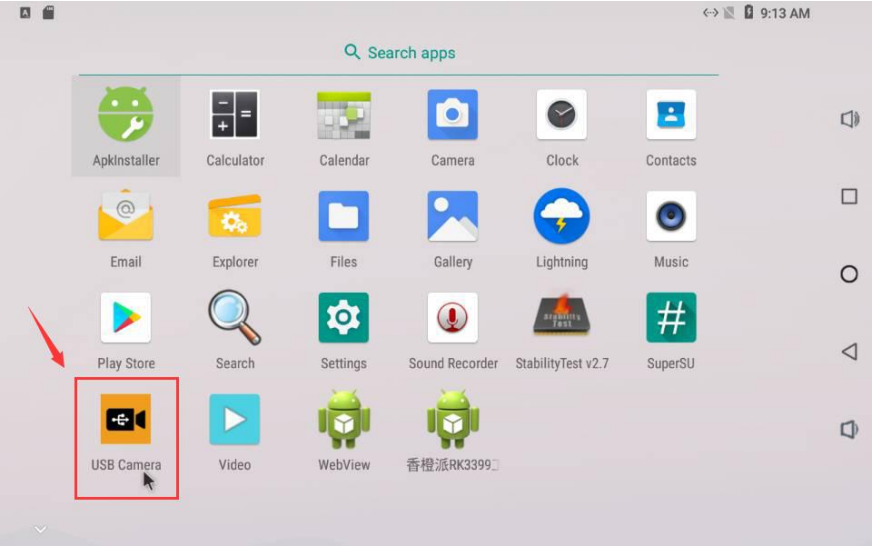


4) Then use the adb command to install the USB camera test APP to the Android system, of course, you can also use the U disk copy method to install

```
test@test:~$ adb install usbcamera.apk
```

5) After installation, you can see the startup icon of the USB camera on the Android desktop



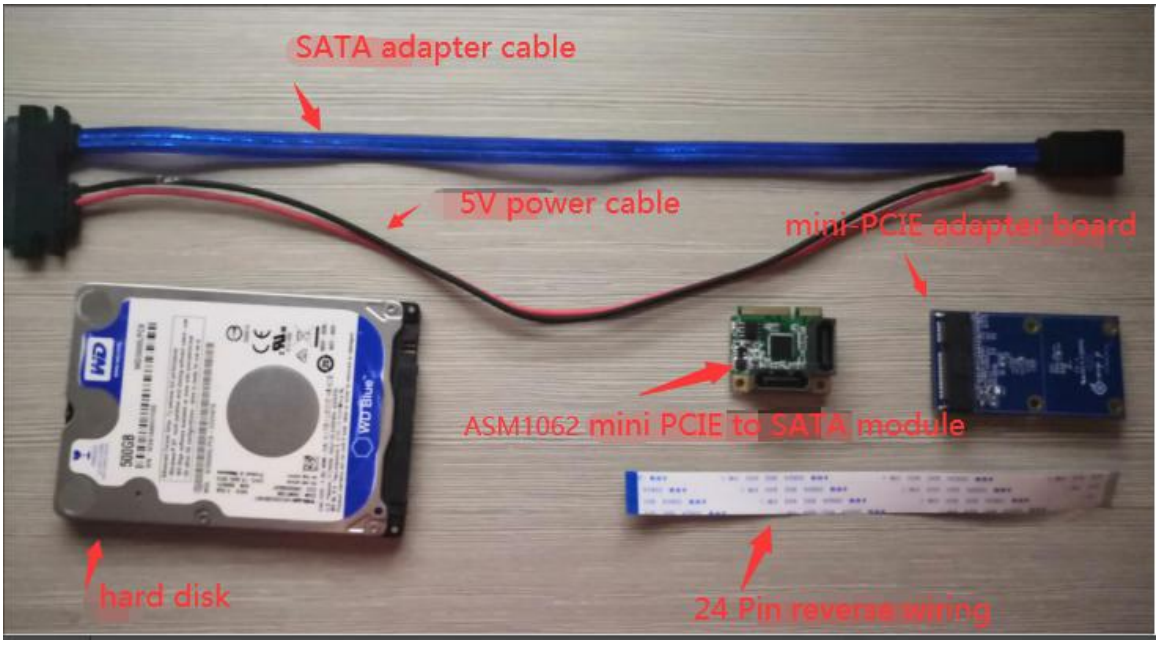


6) Then double-click to open the USB camera APP and you can see the output video of the USB camera

5. 10. How to use Mini-PCIE

5. 10. 1. How to use mini-PCIE to connect hard disk

1) Prepare the required accessories including 24pin reverse row wiring, mini-PCIE adapter board, ASM1062 mini-PCIE to SATA module, hard disk, SATA adapter cable, and 5V power cord. The accessories are as follows

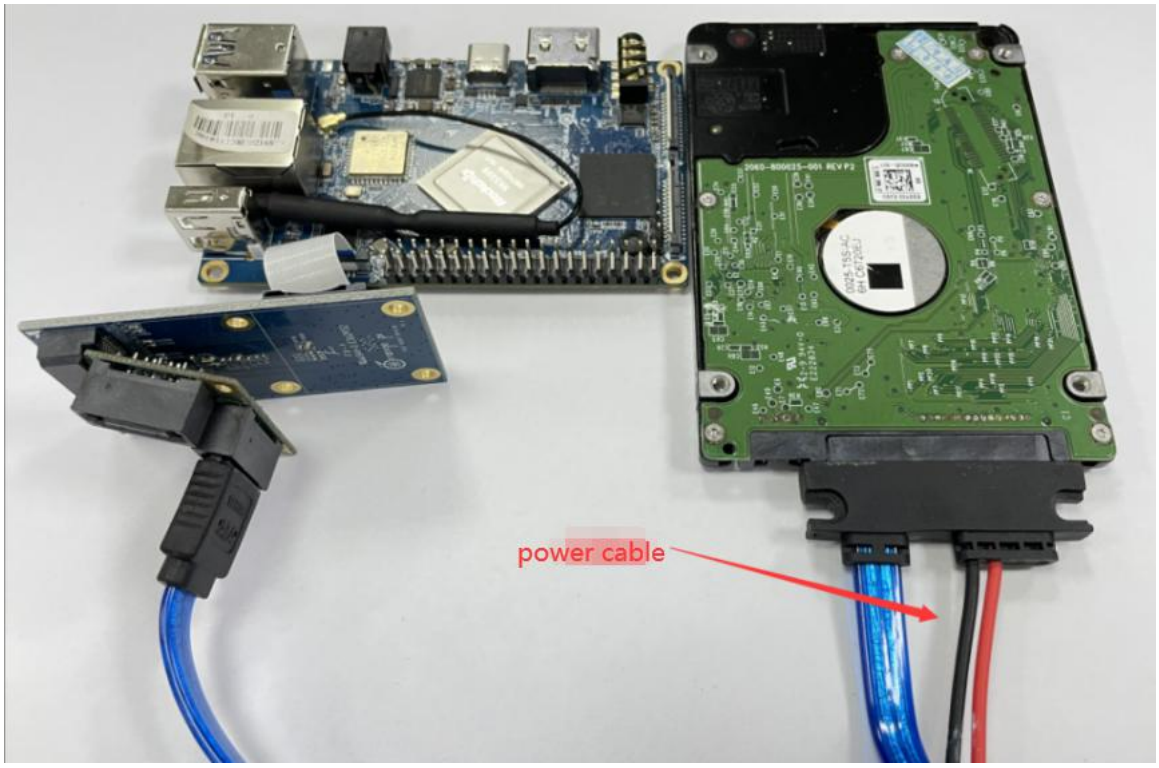


2) Connect the 24pin reverse cable to the mini-PCIE adapter board as shown in the figure below. Note that the blue strip of the cable faces outwards.

3) Connect the mini-PCIE adapter board to the 24pin interface of the Orange Pi 4 development board

4) Connect the ASM1062 mini-PCIE to SATA module to the mini-PCIE adapter board

5) Connect the hard disk to the interface of the mini-PCIE to SATA module through the SATA cable

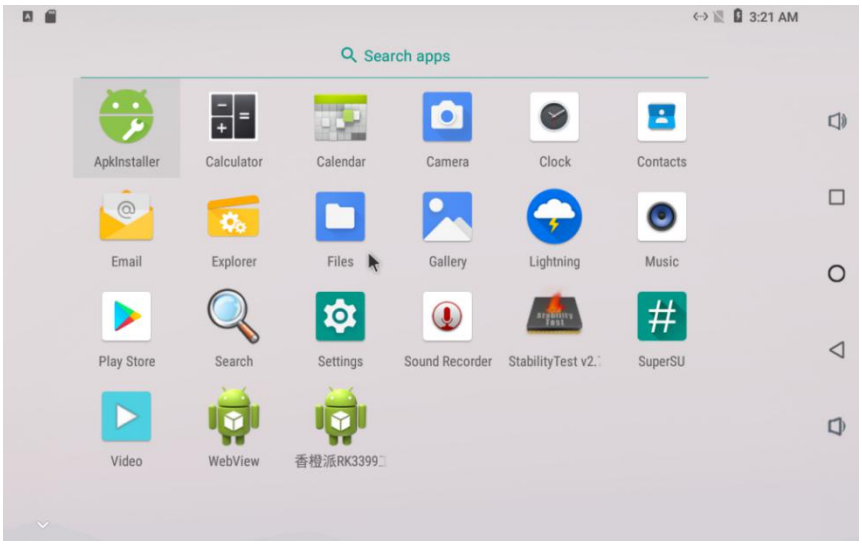


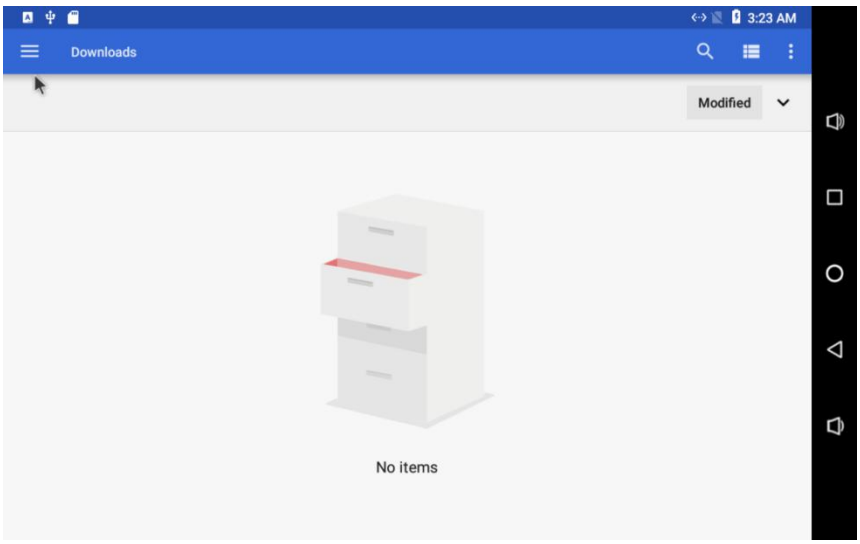
6) Connect the power cord of the SATA adapter cable to the 5V power supply. After the connection is completed, the development board is connected to the DC power supply

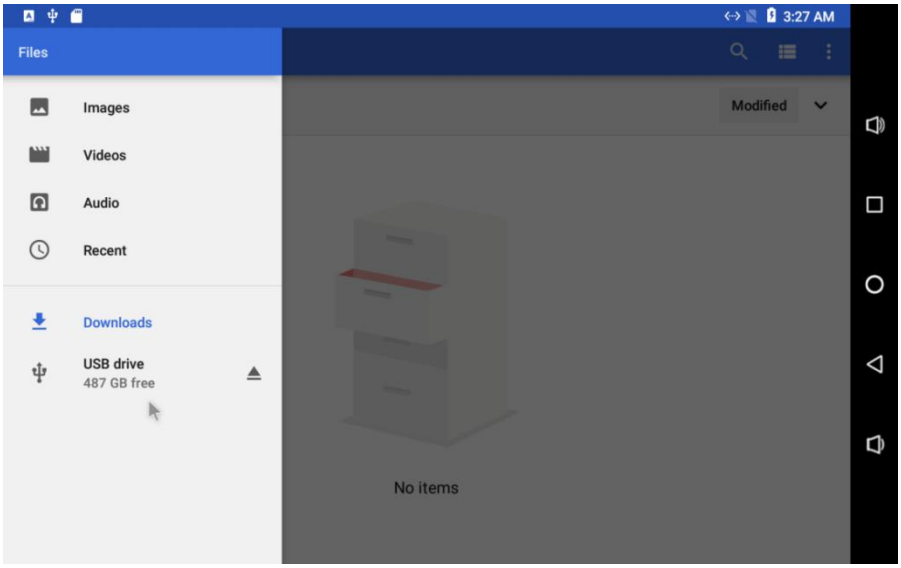
and powered on. The LED of the ASM1062 mini-PCIE to SATA module flashes, indicating that the connection is successful

Note: This function does not support hot-swappable, it can be powered on only after the connection is completed

7) Power on and start the development board, after the system is started, open the file manager app, you can see the hard disk device





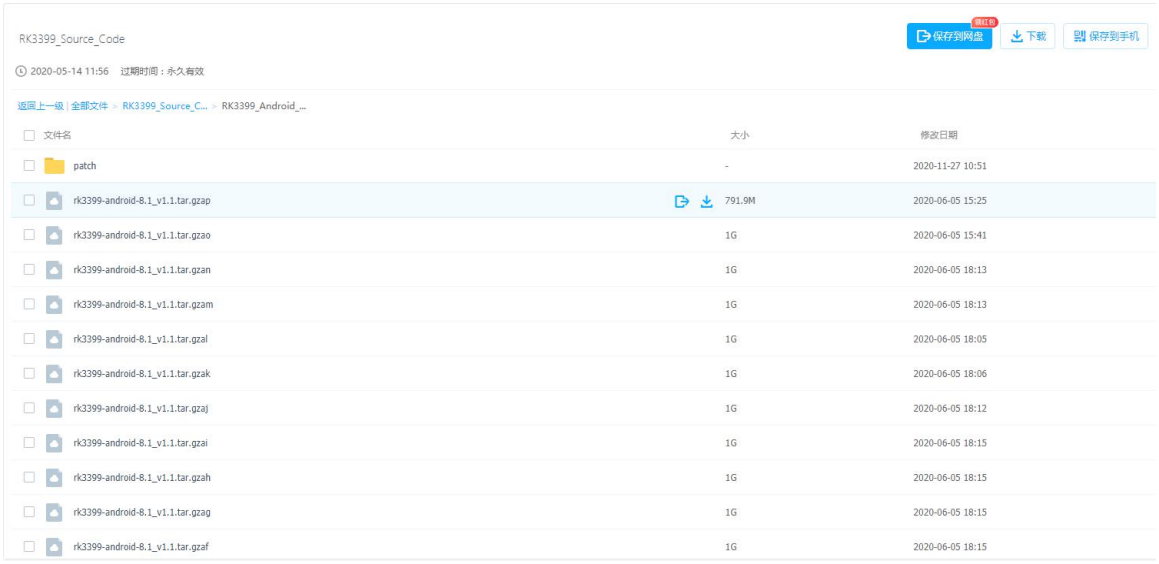


6. Android SDK instructions

It is recommended to compile the Android SDK on a PC with Ubuntu 14.04 installed. There may be some differences in other versions of Ubuntu.

6.1. Download the source code of Android SDK

1) First download the sub-volume compressed package of Android SDK from Baidu Cloud Disk



2) After the Android source code package is downloaded, you first need to merge multiple compressed files into one, and then unzip

```
test@test:~$ mkdir OrangePi_4
test@test:~$ cat rk3399-android-8.1_v1.1.tar.gz* > rk3399-android-8.1_v1.1.tar.gz
test@test:~$ tar xvf rk3399-android-8.1_v1.1.tar.gz -C OrangePi_4
```

6. 2. Build Android compilation environment

1) Install JDK

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

2) Configure JAVA environment variables

a. First determine the installation path of java, generally

```
test@test:~$ ls /usr/lib/jvm/java-8-openjdk-amd64
ASSEMBLY_EXCEPTION  bin  docs  include  jre  lib  man  src.zip
THIRD_PARTY_README
```

b. Then use the following command to export java environment variables

```
test@test:~$ export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
test@test:~$ export PATH=$JAVA_HOME/bin:$PATH
```

```
test@test:~$ export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

3) Install platform support software

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install git gnupg flex bison gperf build-essential \
zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 \
lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z1-dev ccache \
libgl1-mesa-dev libxml2-utils xsltproc unzip

test@test:~$ sudo apt-get install u-boot-tools
```

6. 3. Compile Android image

6. 3. 1. Compile u-boot

1) The compilation method of u-boot is as follows

```
test@test:~$ cd OrangePi_4/rk3399-android-8.1
test@test:~/OrangePi_4/rk3399-android-8.1$ ./make.sh -B
```

2) After the compilation is successful, the output content is as follows

```
out:trust.img
merge success(trust.img)
load addr is 0x200000!
pack input u-boot.bin
pack file size: 682652
crc = 0xc21153c6
pack uboot.img success!
```

6. 3. 2. Compile the kernel

1) The compilation method of the kernel is as follows

```
test@test:~/OrangePi_4/rk3399-android-8.1$ ./make.sh -K
```

2) After the compilation is successful, the output content is as follows

```
scripts/kconfig/conf    --silentoldconfig Kconfig
```

```
  CHK     include/config/kernel.release
  CHK     include/generated/uapi/linux/version.h
......
make[2]: “include/generated/vdso-offsets.h”is the latest.
  Building modules, stage 2.
  MODPOST 13 modules
Pack to resource.img successed!
  Image:  resource.img (with rk3399-orangepi.dtb logo.bmp logo_kernel.bmp) is ready
  Image:  boot.img (with Image resource.img) is ready
```

6. 3. 3. Compile android

1) The compilation method of android is as follows

```
test@test:~/OrangePi_4/rk3399-android-8.1$ ./make.sh -A
```

2) After the compilation is successful, the output content is as follows

```
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
......
[100% 129/129] Install system fs image: out/target/product/rk3399_mid/system.img
out/target/product/rk3399_mid/system.img+out/target/product/rk3399_mid/obj/PACKAG
ING/recovery_patch_intermediates/recovery_from_boot.p          maxsize=2740531200
blocksize=135168 total=1198447603 reserve=27709440

#### build completed successfully (03:53 (mm:ss)) ####
```

6. 3. 4. Package a complete image

1) Android image packaging method is as follows

```
test@test:~/OrangePi_4/rk3399-android-8.1$ ./make.sh -M -u
```

2) After the compilation is successful, the output content is as follows

```
create uboot.img...done.
create trust.img...done.
```

```
create loader...done.
.....
Make firmware OK!
------ OK ------
********RKImageMaker ver 1.63********
Generating new image, please wait...
Writing head info...
Writing boot file...
Writing firmware...
Generating MD5 data...
MD5 data generated successfully!
New image generated successfully!
Making update.img OK.
```

3) After compiling, the generated image file will be placed under rockdev/Image-rk3399_mid/. Among them, update.img is the Android firmware that can be burned and run

```
test@test:~/OrangePi_4/rk3399-android-8.1$ cd rockdev/Image-rk3399_mid/
test@test:~/OrangePi_4/rk3399-android-8.1$ ls update*
update.img
```