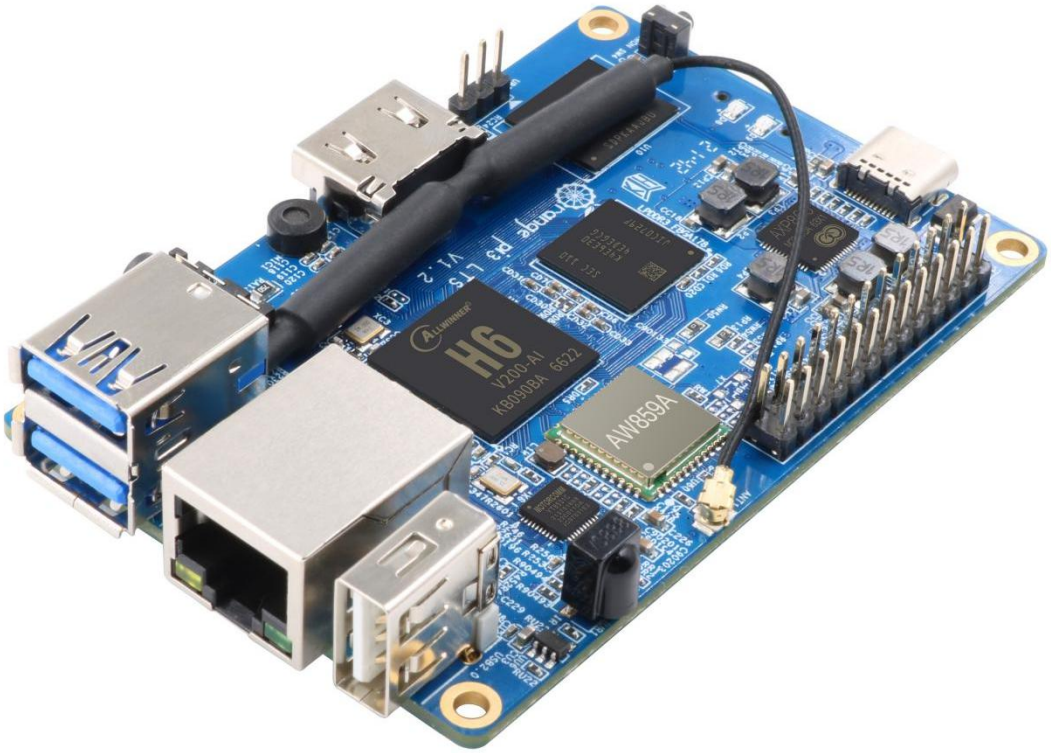
Orange Pi 3 LTS

User Manual



Contents

1. Basic functions of Orange Pi 3 LTS

1. 1. What's Orange Pi 3 LTS ?

Orange Pi 3 LTS is an open source single-board card computer, a new generation of arm64 development board, it can run Android 9.0, Ubuntu and Debian and other operating systems. Orange Pi 3 LTS uses Allwinner H6 system-on-chip and has 2GB LPDDR3 memory

1. 2. What can l do with Orange Pi 3 LTS ?

We can use it to build:

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

Pretty much anything else, because Orange Pi is open source.

1. 3. Who's it for?

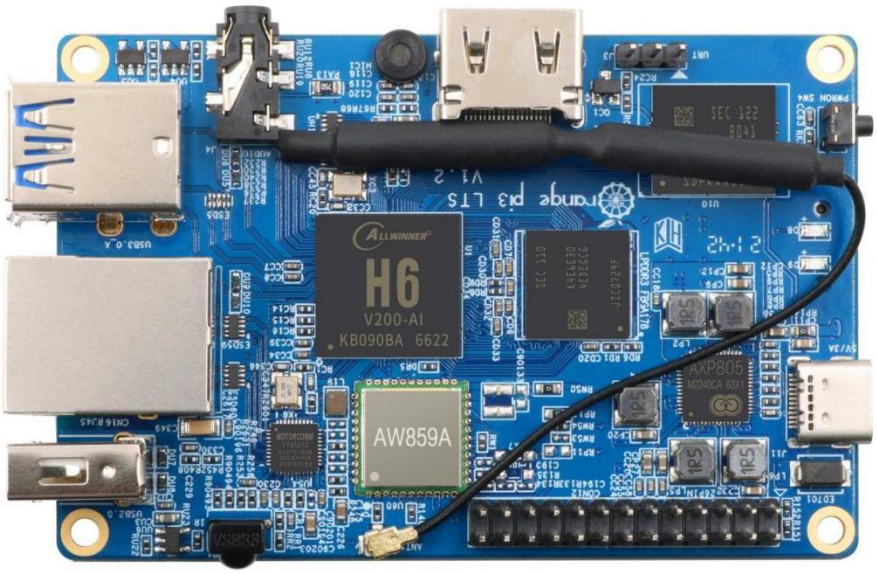
The Orange Pi development board is not only a consumer product, but also designed for anyone who wants to use technology to create and innovate. It is a simple, interesting and practical tool, you can use it to create the world around you

1. 4. Hardware specification

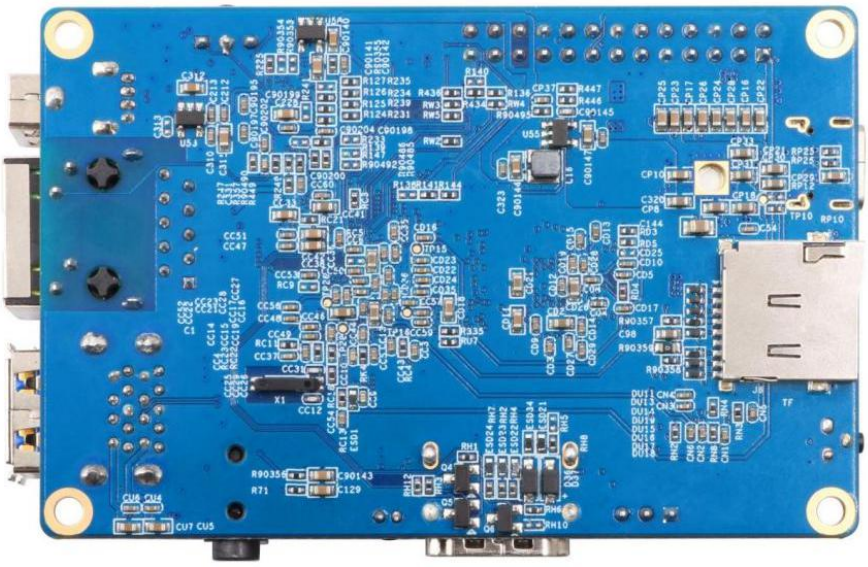
Hardware feature introduction	
CPU	Allwinner H6 Quad-core 64-bit 1.8GHz high-performance Cortex-A53 processor
GPU	• High-performance multi-core GPU Mali T720 • OpenGL ES3.1/3.0/2.0/1.1
RAM	2GB LPDDR3 (shared with GPU)
Onboard Storage	TF card slot、8GB EMMC
Onboard Network	• YT8531C Chip • Support 10/100M/1000M Ethernet
Onboard WIFI+Bluetooth	• AW859A Chip • Support IEEE 802.11 a/b/g/n/ac • Support BT5.0
Video Output	HDMI 2.0a 、TV CVBS output
Audio output	• HDMI Output • 3.5mm Audio Port
Power Source	5V3A Type-C
Power management chip	AXP805
USB port	1*USB 3.0 HOST、1*USB 2.0 HOST、1*USB2.0 OTG
26pin headers	1*I2C、1*SPI、1*UART&Multiple GPIO Ports
Debug serial port	UART-TX、UART-RX &GND
LED	Power LED& Status LED
IR Receiver	Support IR remote control for Orange Pi
Button	Power Button (SW4)
Supported OS	Android9.0、Ubuntu、Debian
Appearance specification introduction	
Dimension	85mm×56mm
Weight	45g

1.5. The top and bottom views of Orange Pi 3 LTS

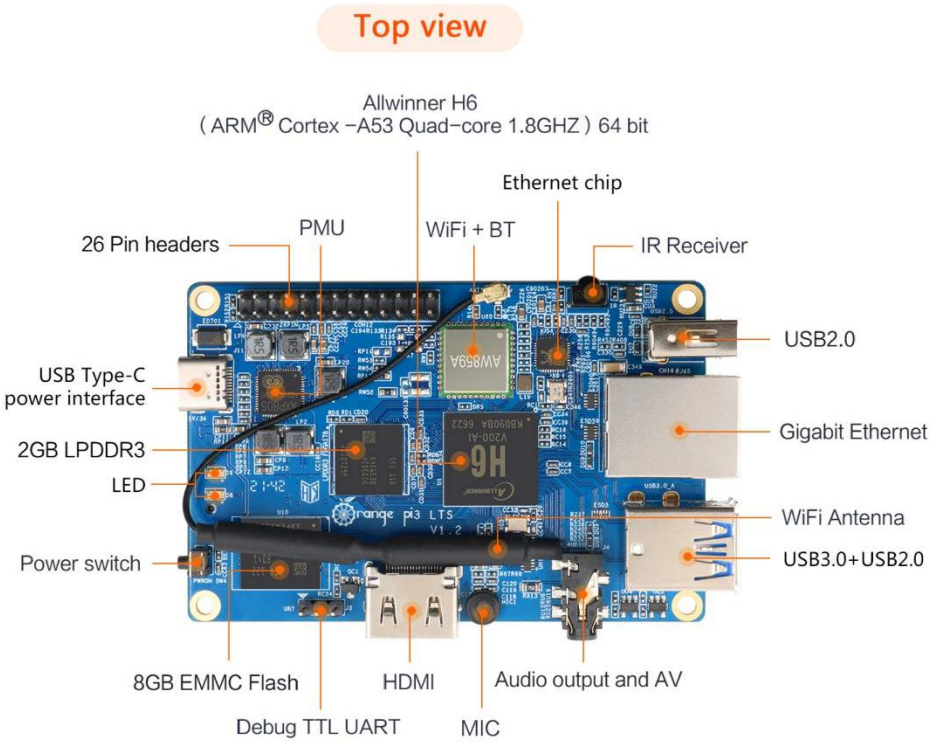
Top View：

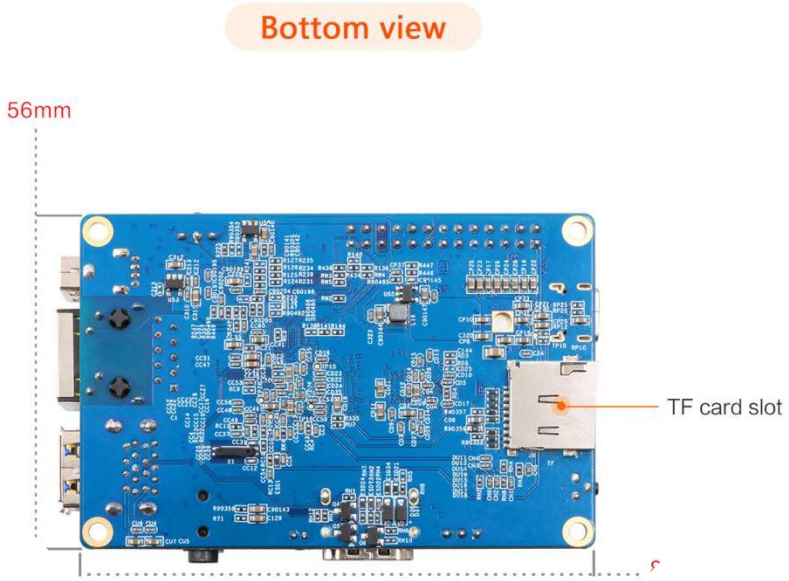


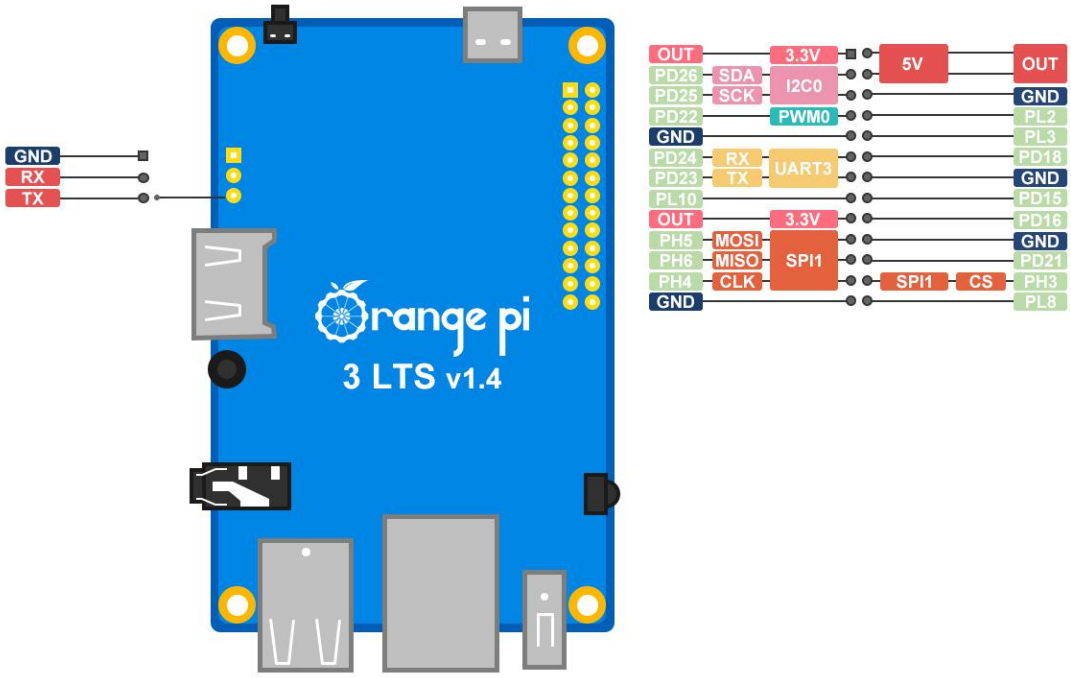
Bottom View：



1. 6. Orange Pi 3 LTS interface details






GND
RX
TX
range pi
3 LTS v1.4
OUT
PD26
PD25
PD22
GND
PD24
PD23
PL10
OUT
PH5
PH6
PH4
GND
SDA
SCK
RX
TX
MOSI
MISO
CLK
3.3V
I2C0
PWM0
UART3
3.3V
SPI1
5V
SPI1
CS
OUT
GND
PL2
PL3
PD18
GND
PD15
PD16
GND
PD21
PH3
PL8

2. Introduction to use the development board

2. 1. Prepare the necessary accessories

1) TF card, a class 10 or higher high-speed card with a minimum capacity of 8GB, it is recommended to use SanDisk's TF card. Orange Pi tests all with SanDisk's TF card. Other brands of TF cards may cause the system to fail to start.

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) The power supply requires a USB Typc C interface data cable and a 5V/3A high-quality power adapter

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

6) Infrared remote control, mainly used to control Android system

7) 100M or Gigabit network cable, used to connect the development board to the Internet

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

9) USB to 3.3v TTL module and DuPont cable. When using the debug serial port, USB to TTL module and DuPont cable are required to connect the development board and the computer

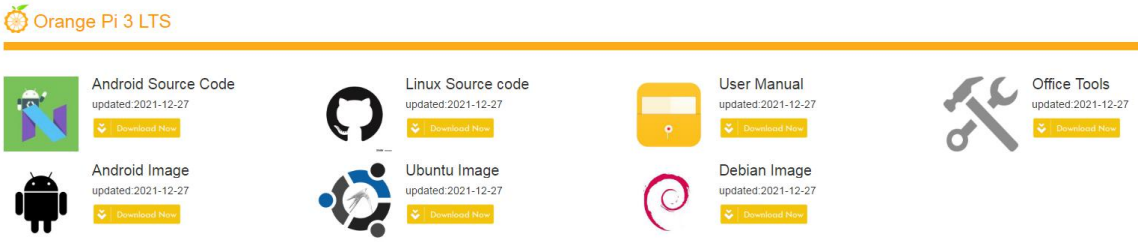
10) A personal computer with Ubuntu and Windows operating systems

1	Ubuntu18.04 PC	Optional, used to compile Linux source code and Android source code
2	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 is

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



2) The information mainly contains

a. Android source code: saved on Google Cloud Disk

b.Linux source code: saved on github

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 Google Cloud Disk

f. Ubuntu image: saved on Google Cloud Disk

g. Debian image: saved on Google Cloud Disk

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

The development board comes with 8GB eMMC. After you get the development board, you can first use the Android9.0 image pre-installed in the eMMC to test the functions of the development board. After confirming that all the hardware functions of the development board can work, then 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

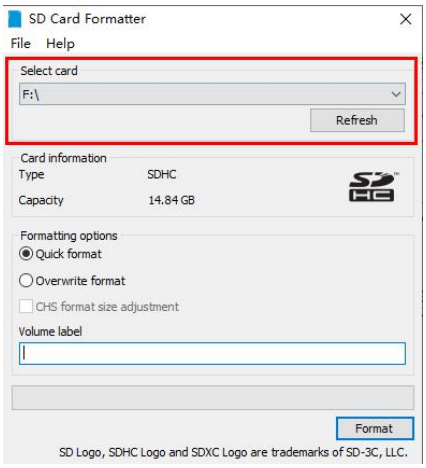
3) Then format the TF card

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

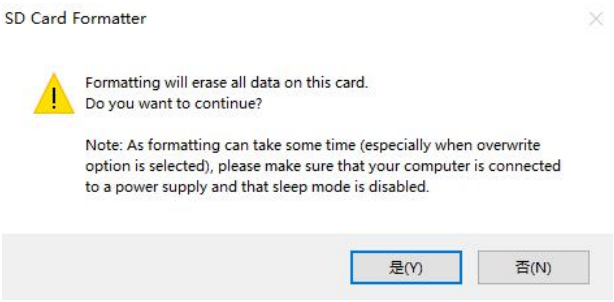
https://www.sdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip

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

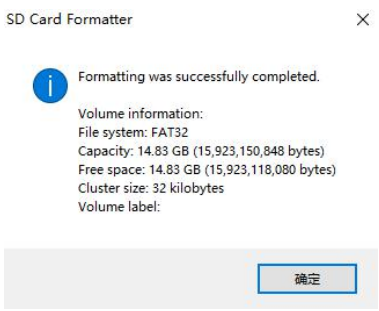
c. If the computer only has a TF card inserted, the “Select card” column will display the drive letter of the TF card. If the computer has multiple USB storage devices inserted, you can select the drive letter corresponding to the TF card through the drop-down box



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

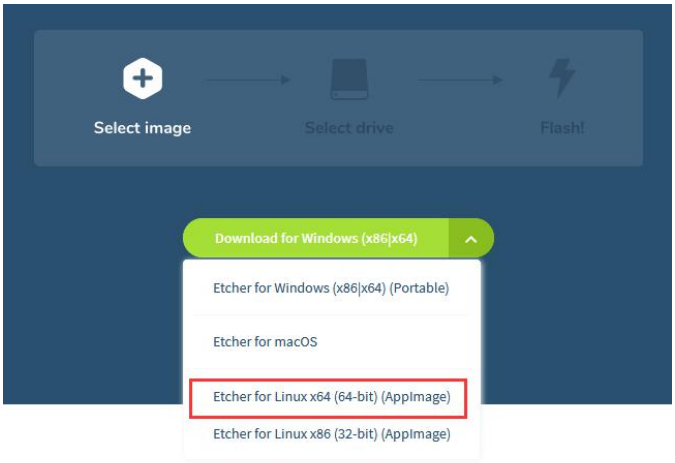
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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 the Linux image

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

6) Download the 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

a. The decompression command of the compressed package at the end of a.7z is as follows

```
test@test:~$ 7z x Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.7z
test@test:~$ ls Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.*
Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.7z
Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.img.sha    #Checksum file
Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.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 is **successful**, it means that the downloaded image is not wrong, and you can safely burn to the TF card. If the **checksum does not match**, it means There is a problem with the downloaded image, please try to download again

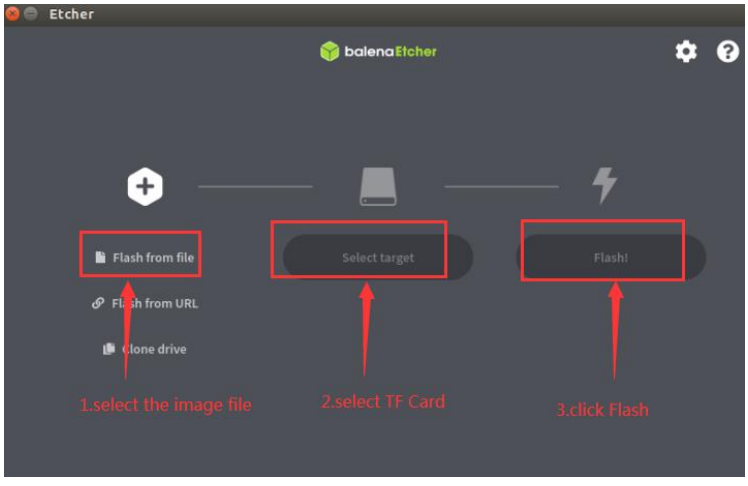
```
test@test:~$ sha256sum -c *.sha
Orangepi3-lts_2.1.6_ubuntu_bionic_server_linux4.9.118.img: success
```

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

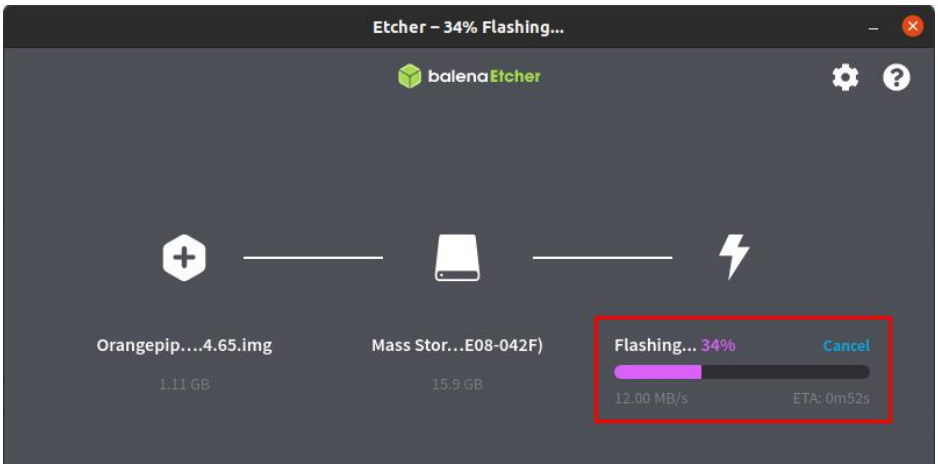
a. First select the path of the linux image file

b. Then select the device number of the TF card

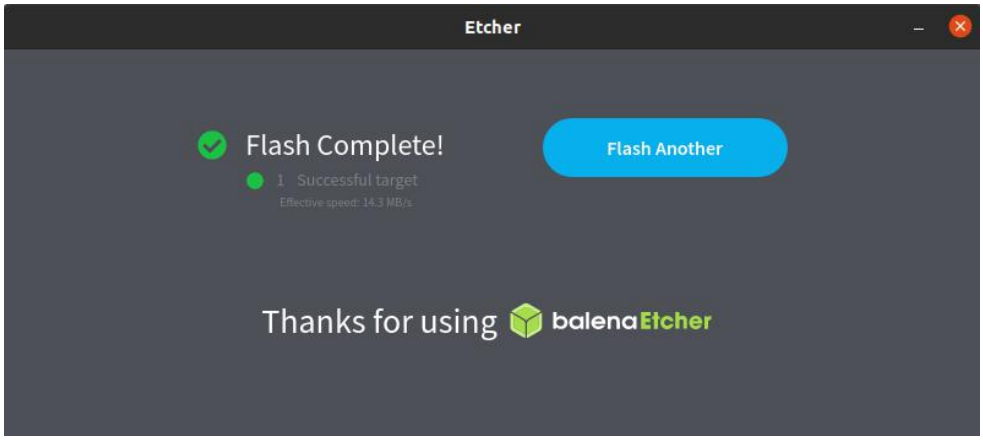
c. Finally click Flash to start burning



9) The writing process will prompt the writing speed and remaining time



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

Android image can only be burned to TF card using PhoenixCard software under Windows platform, but cannot be burned under Linux platform

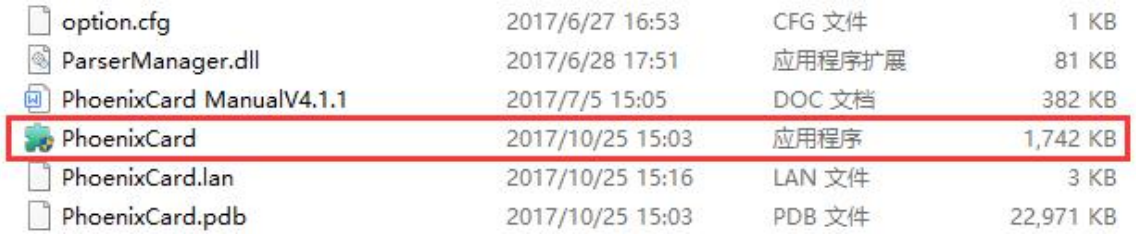
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be **above class10**. It is recommended to use a TF card of SanDisk and other brands

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

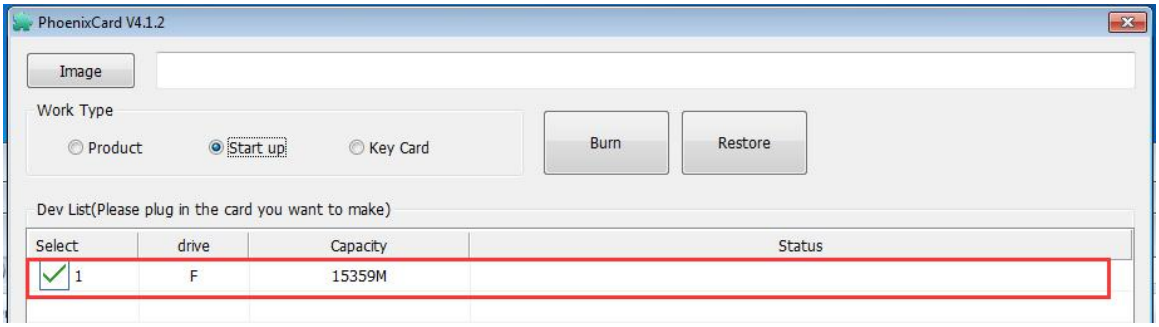
3) Download Android 9.0 firmware and PhoenixCard burning tool from Orange Pi's data download page. Please make sure that the version of PhonenixCrad tool is **PhoenixCard v4.1.2 or PhoenixCard v4.1.2 or higher**

4) 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

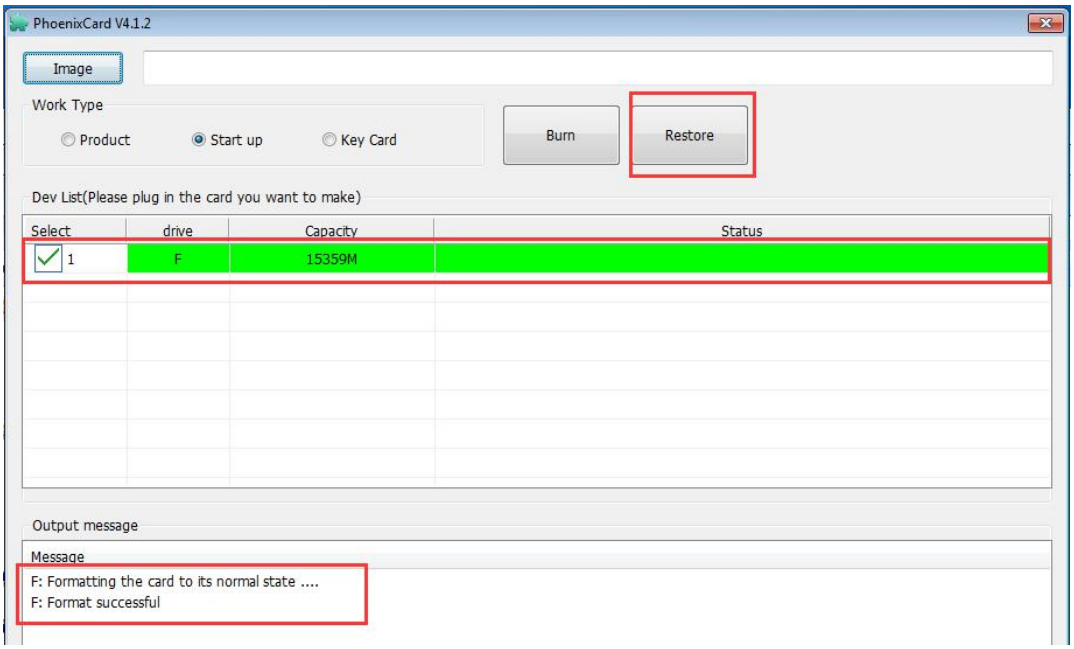
5) Use decompression software to decompress **PhoenixCard v4.1.2.rar**, this software does not need to be installed, you can find **PhoenixCard** in the decompressed folder and open it



6) After opening PhoenixCard, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that the displayed drive letter is consistent with the drive letter of the TF card you want to burn**. There is no display, you can try to unplug and insert the TF card



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

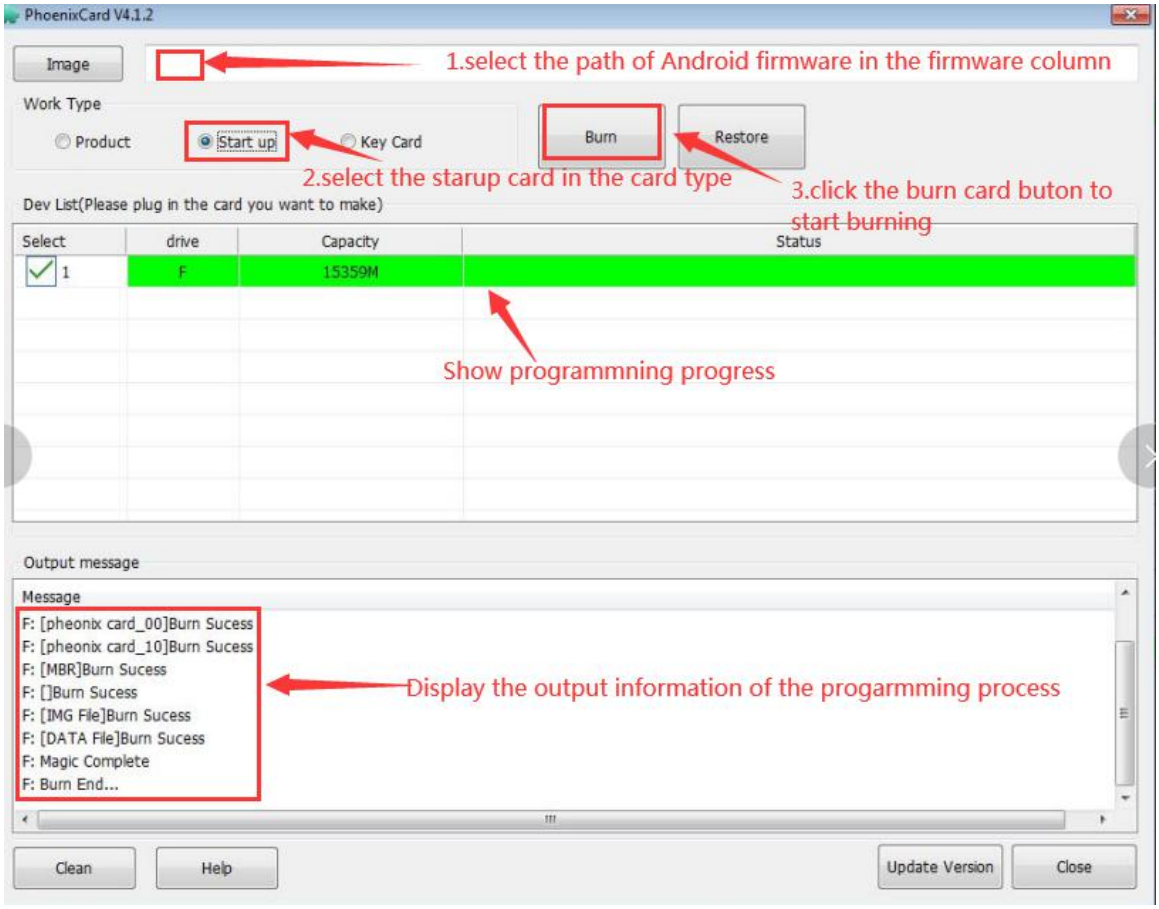


8) Then start to write the Android firmware to the TF card

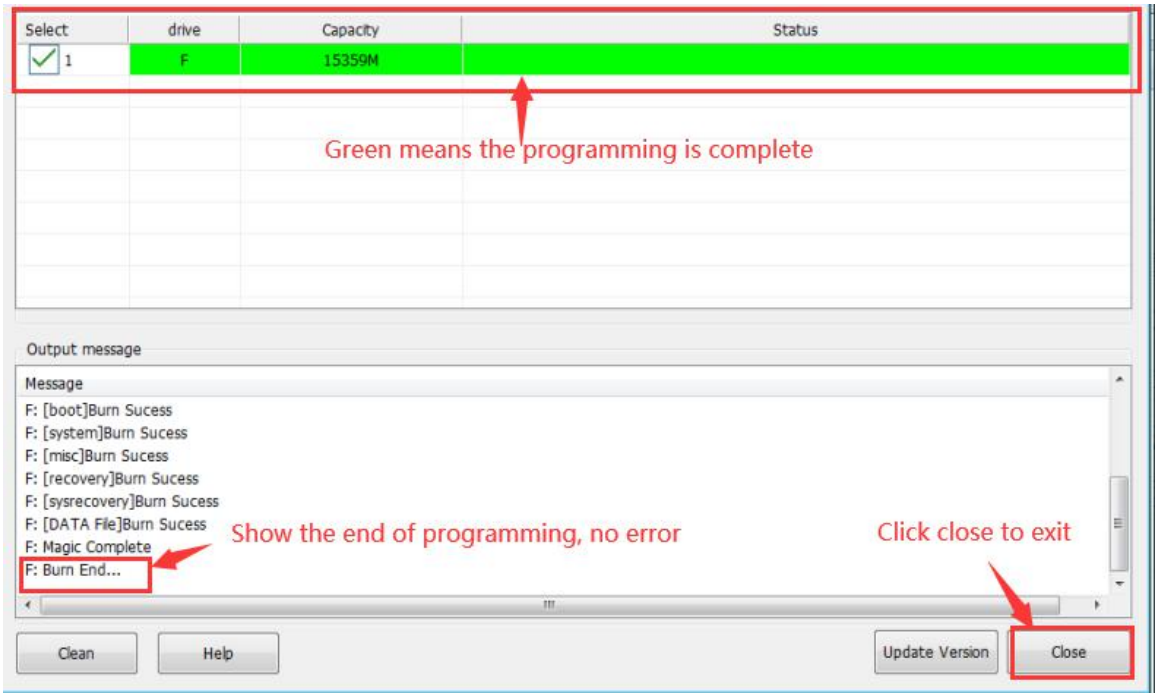
a. First select the path of Android firmware in the **"Firmware"** column

b. Select **"Startup Card"** in the **"Type of Making Card"**

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



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



2. 8. How to burn Android firmware to eMMC

Android image can only be burned to EMMC using PhoenixCard software under Windows platform, but cannot be burned under Linux platform

1) First of all, please note that burning the Android firmware to the eMMC of the development board needs to be completed with the help of a TF card, which is mainly divided into the following two steps

a. First use PhoenixCard to burn the Android firmware to the TF card as **a mass production card**

b. Then use TF card to burn Android firmware into eMMC

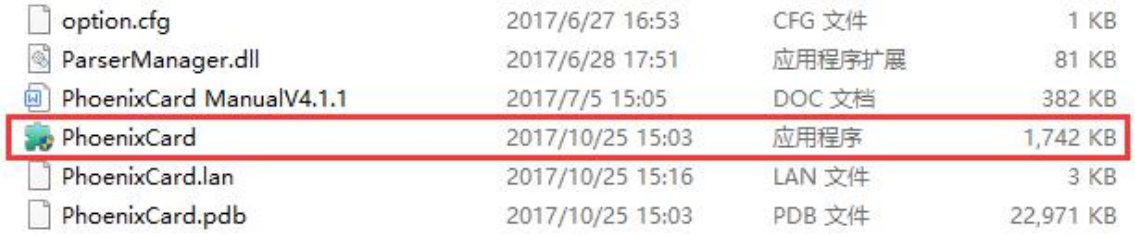
2) First prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be **above class10**. It is recommended to use a TF card of SanDisk and other brands

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

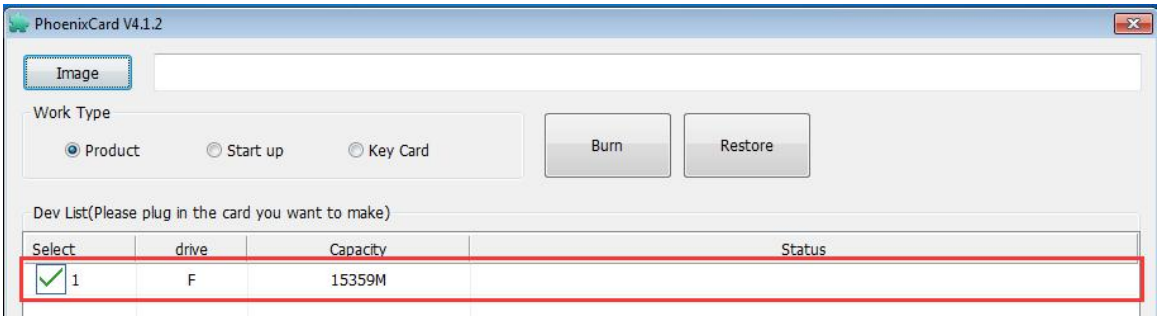
4) Download Android 9.0 firmware and PhoenixCard burning tool from Orange Pi's data download page, please make sure that the version of PhonenixCard tool is **PhoenixCard v4.1.2 or PhoenixCard v4.1.2 or higher**

5) 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

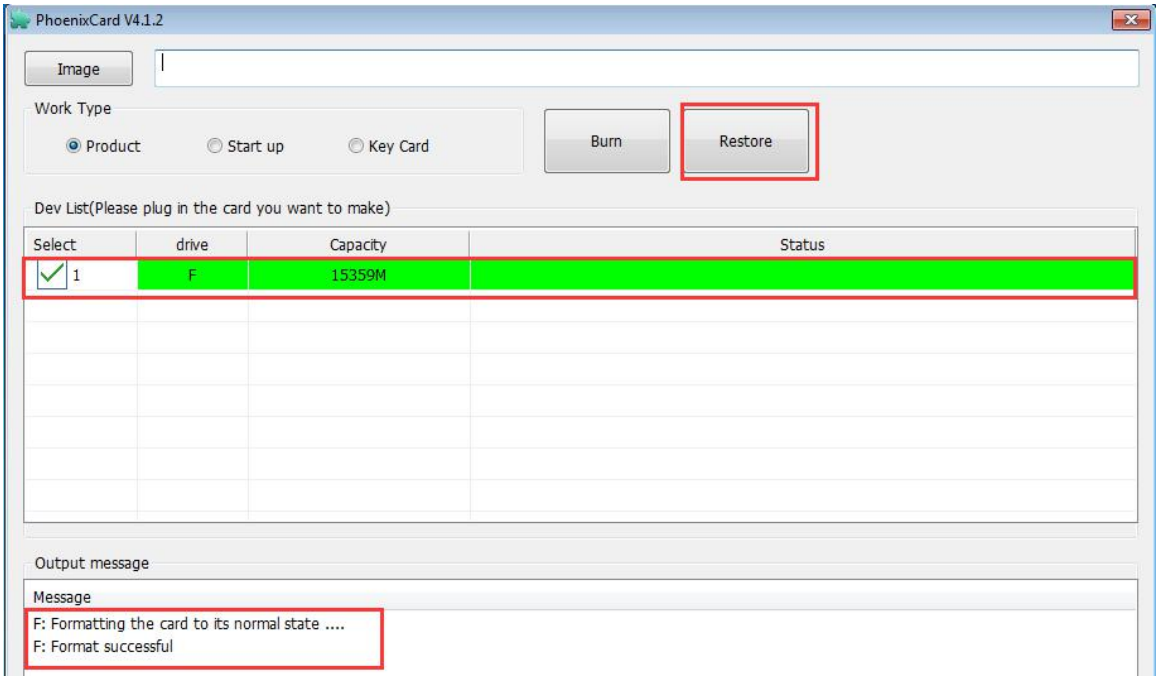
6) Use decompression software to decompress **PhoenixCard v4.1.2.rar**, this software does not need to be installed, you can find **PhoenixCard** in the decompressed folder and open it



7) After opening **PhoenixCard**, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. **Please make sure that the displayed drive letter is consistent with the drive letter of the TF card you want to burn**. There is no display, you can try to unplug and insert the TF card



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

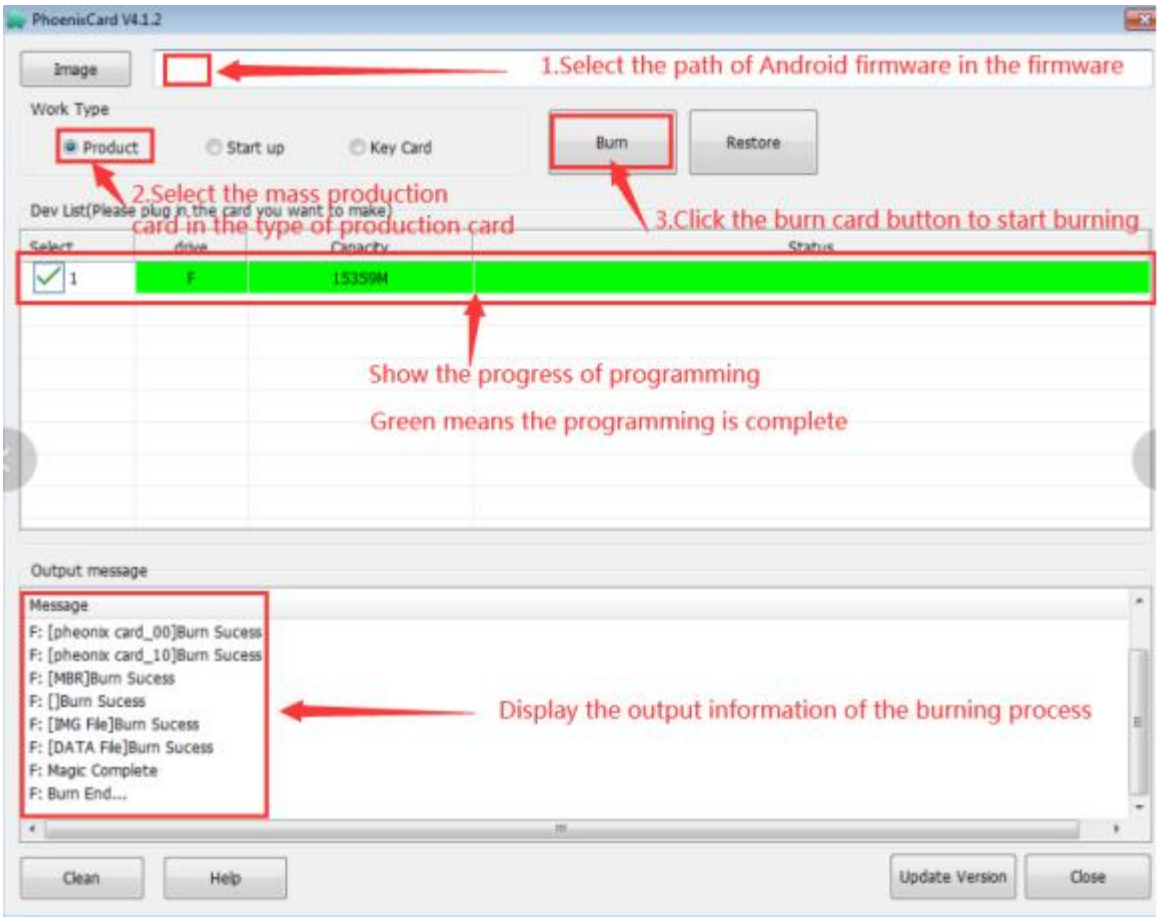


1) Then start to write the Android firmware to the TF card

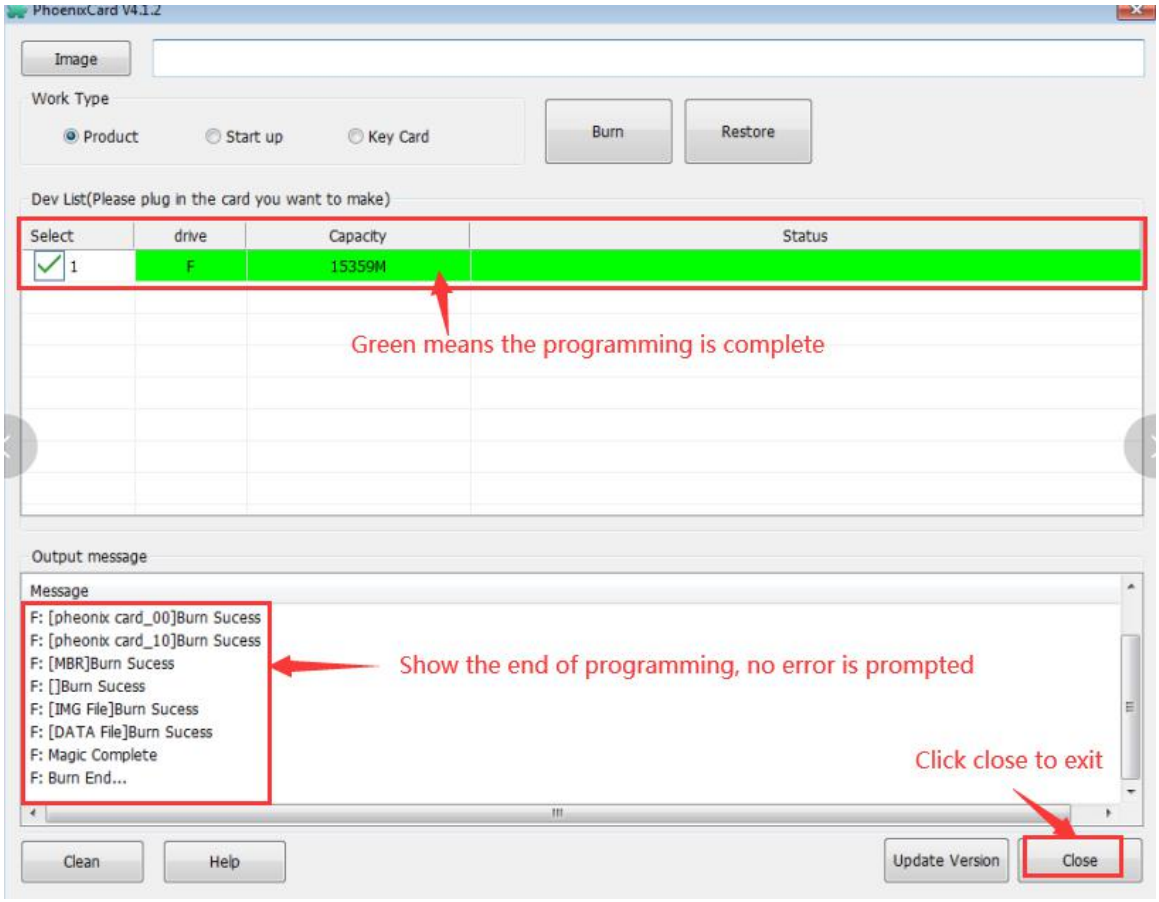
a. First select the path of Android firmware in the **"Firmware"** column

b. Select **"mass production card"** in the **type of "production" card**

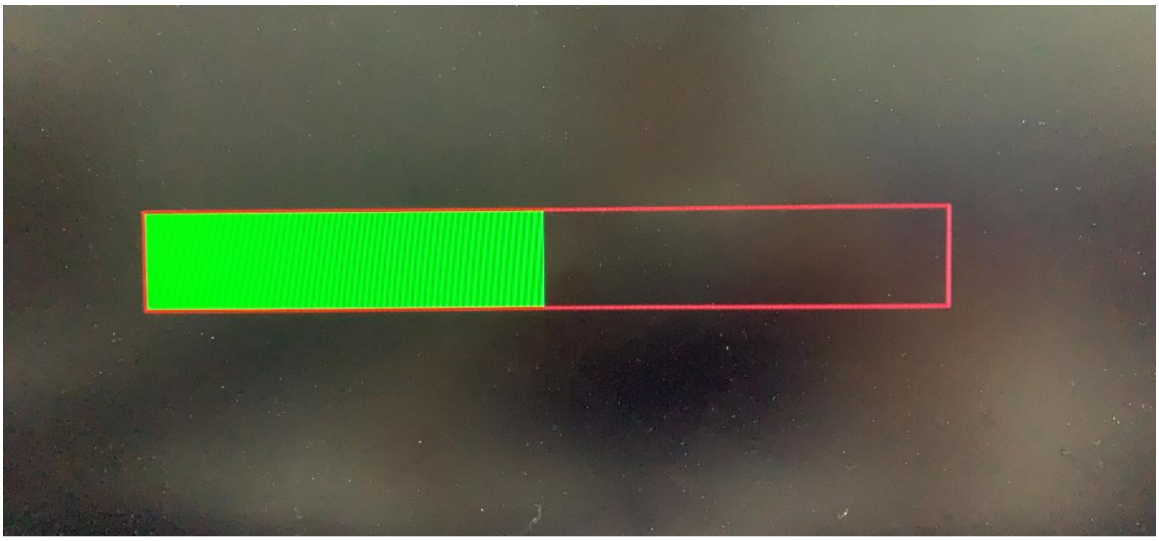
c. Then click the **"Burn Card"** button to start burning



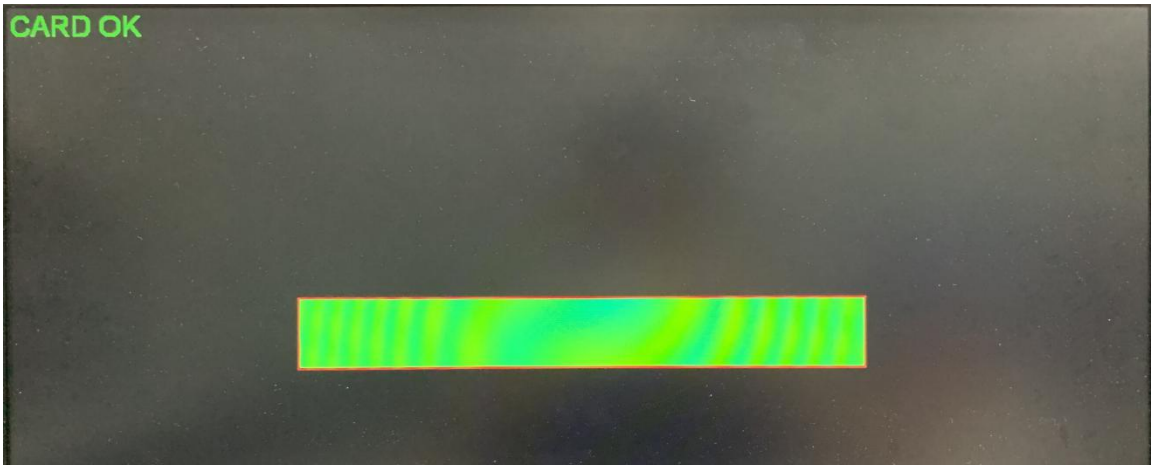
2) After burning, PhoenixCard will display as shown in the figure below. At this time, click the close button to exit PhoenixCard



3) Then insert the TF card into the development board. After powering on the development board, the Android firmware in the TF card will be automatically burned into the eMMC of the development board. During the burning process, the red light on the development board will keep flashing. If the development board is connected to an HDMI display, the progress bar of burning Android firmware to eMMC will be displayed on the HDMI display



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



5) At this time, you can pull out the TF card, then power on again, and the Android system in eMMC will start

2. 9. Start the Orange Pi development board

1) The development board has an on-board eMMC, which has an Android9.0 image burned by default, and you can directly use the image in the eMMC to start and fully test the function when you just get the development board.

2) If you need to use linux image, you can insert the TF card with the burned linux image into the TF card slot of the Orange Pi development board

3) 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

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

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

6) Connect a 5V/3A high-quality power adapter

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

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

7) 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

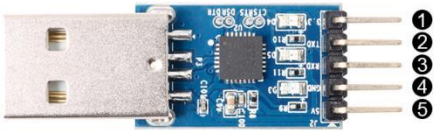
8) If you want to view the output information of the system through the debug serial port, please use the USB to TTL module and DuPont cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section on the use of the debug serial port

2. 10. How to use the debug serial port

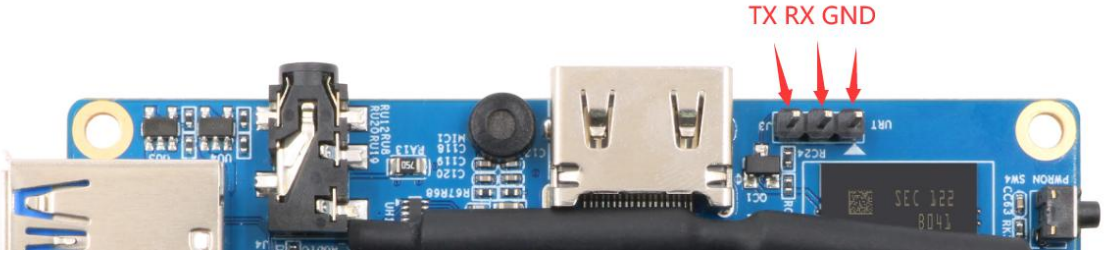
2. 10. 1. Debug serial port connection instructions

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

❶ The 3.3V of the USB to TTL module does not need to be connected
❷ The TXD of the USB to TTL module is connected to the RXD of the debugging serial port of the development board
❸ The RXD of the USB to TTL module is connected to the TXD of the debugging serial port of the development board
❹ The GND of the USB to TTL module is connected to the GND of the debugging serial port of the development board
❺ The 5V of the USB to TTL module does not need to be connected



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



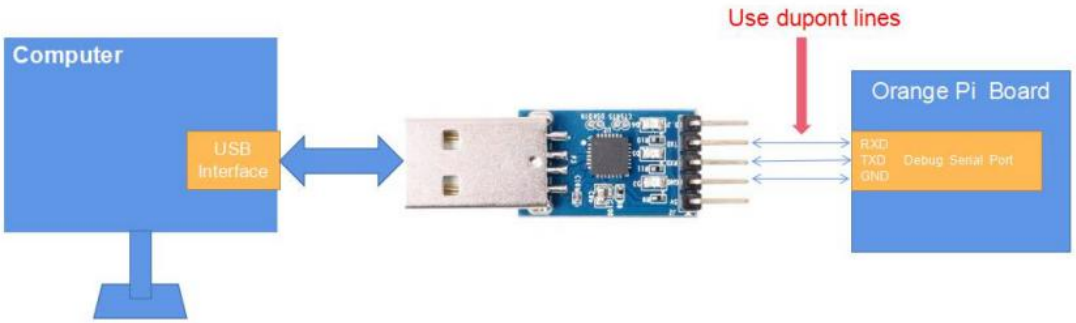
3) The GND, TX and RX pins of the USB to TTL module need to be connected to the debug serial port of the development board through a Dupont cable

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

2. 10. 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

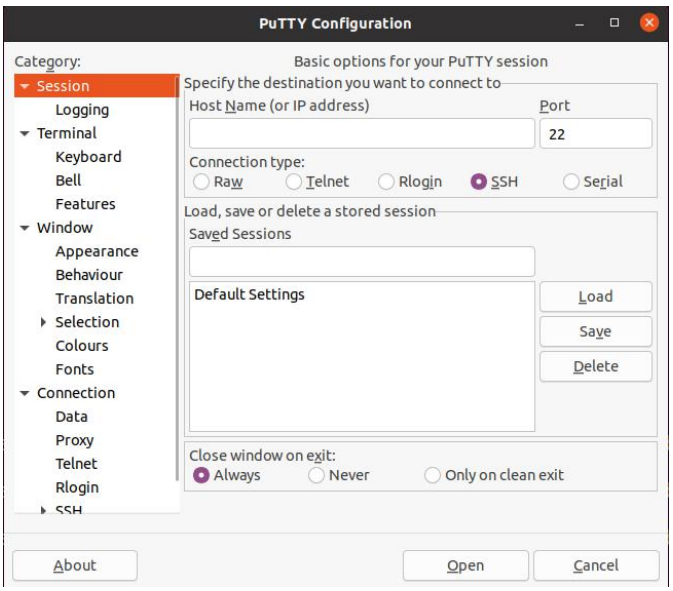
3) First install putty on the Ubuntu PC

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

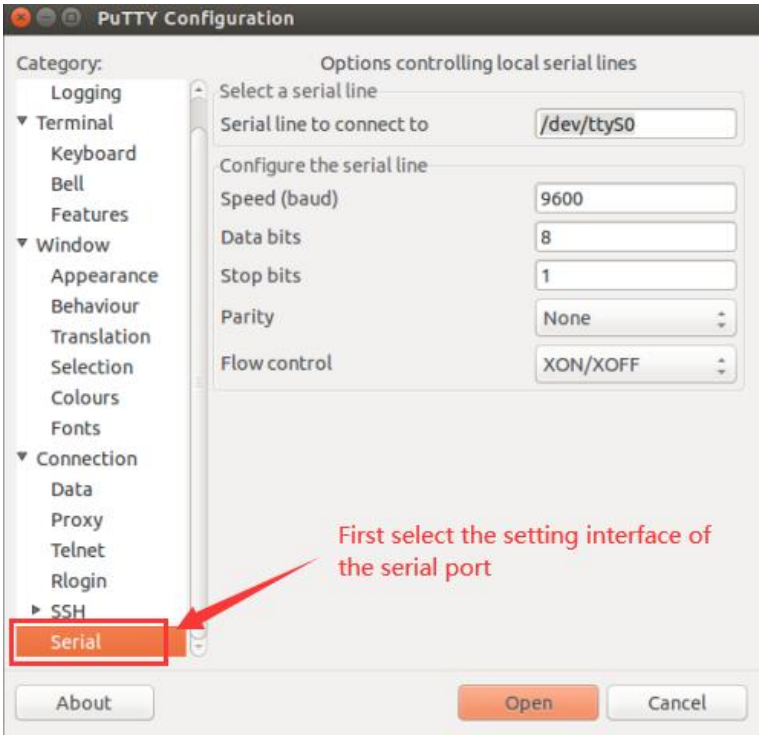
4) Then run putty, remember to add sudo permissions

```
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

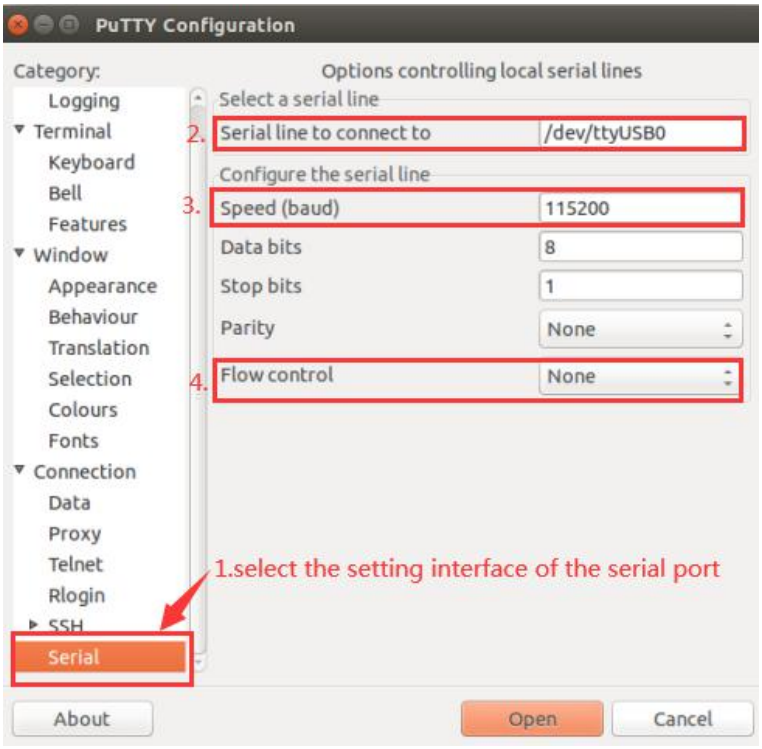


7) Then set the parameters of the serial port

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

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

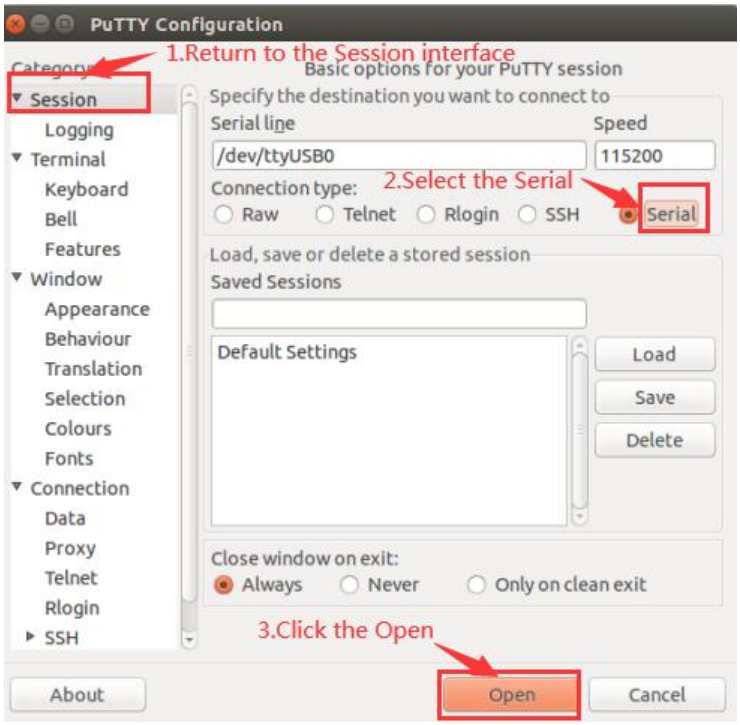
c. Set Flow control to None



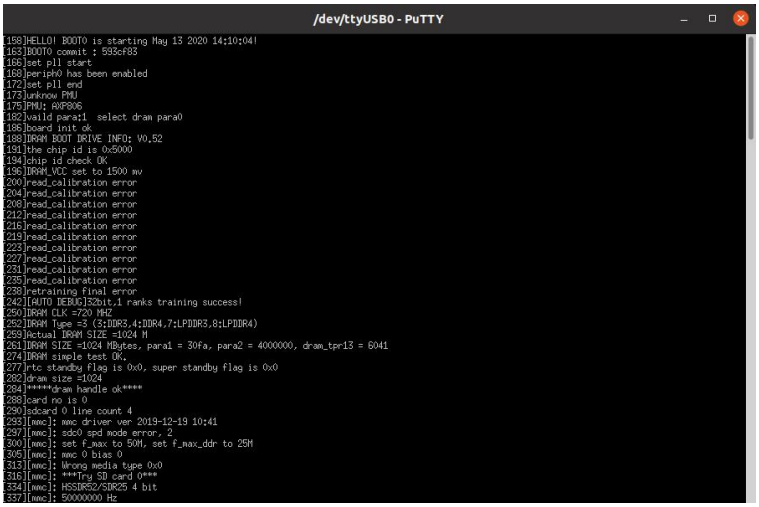
8) After setting the serial port setting interface, return to the Session interface

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

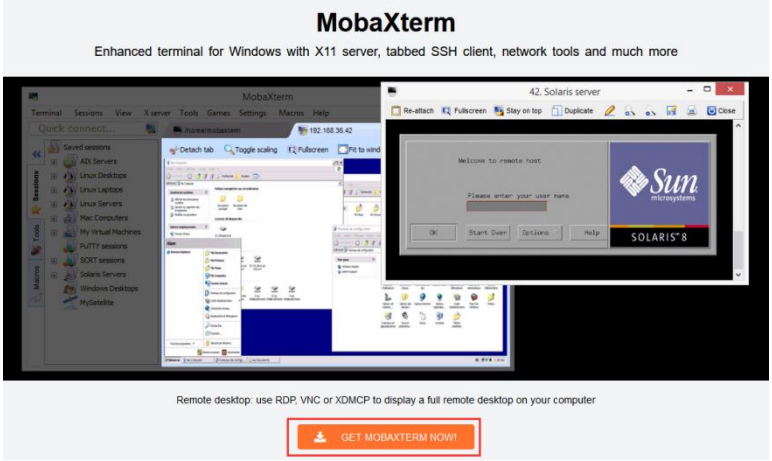
1) There are many serial debugging software that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following shows how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

2) Download MobaXterm

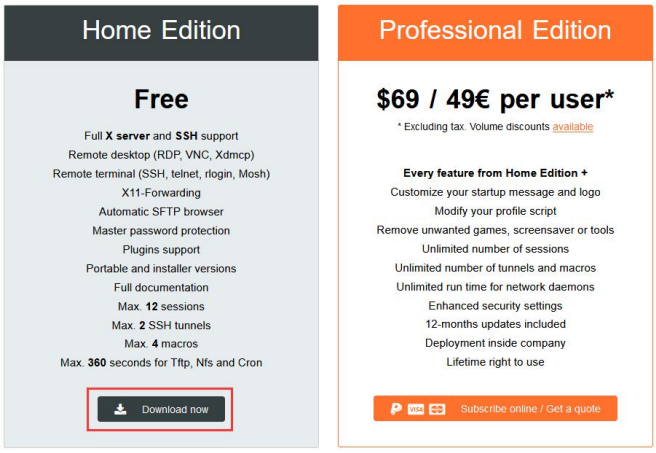
a. Download the MobaXterm URL as follows

https://mobaxterm.mobatek.net/

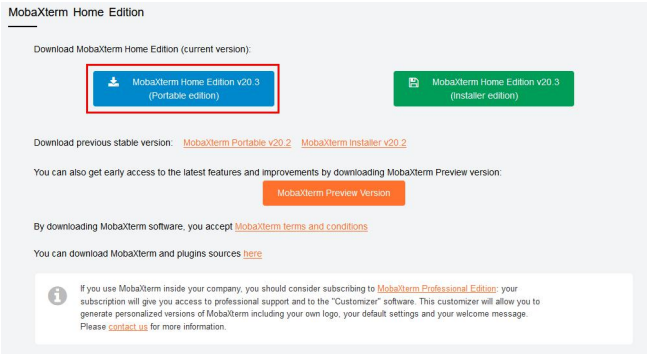
b. After entering the MobaXterm download page, click **GET XOBATERM NOW!**



c. Then choose to download the Home version



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



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 up the serial port connection are as follows

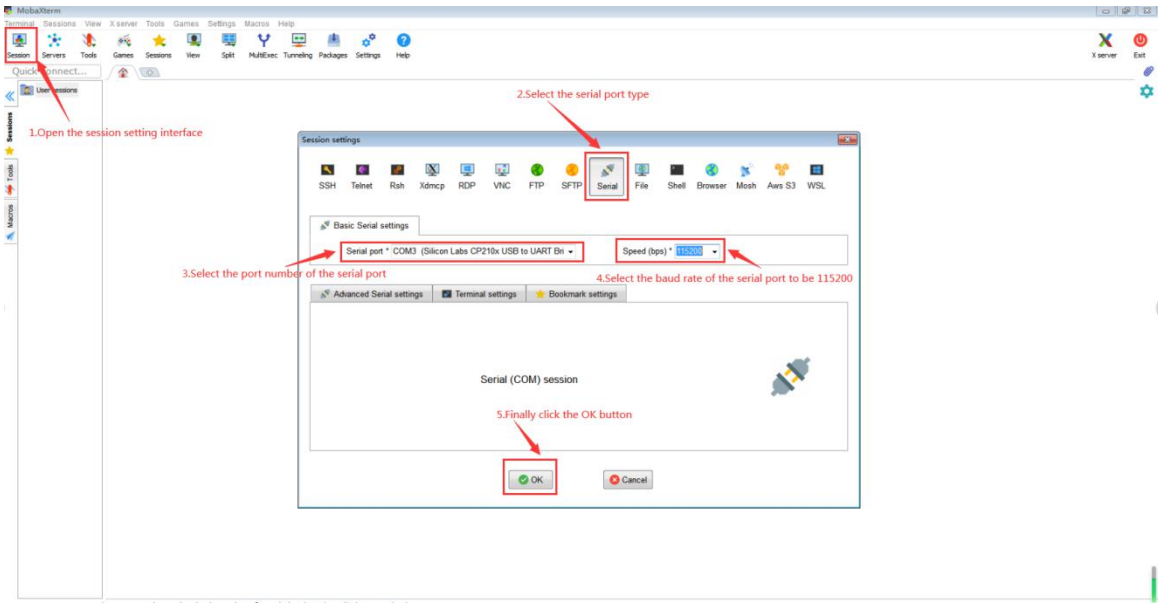
a. Open the session setting interface

b. Select the serial port type

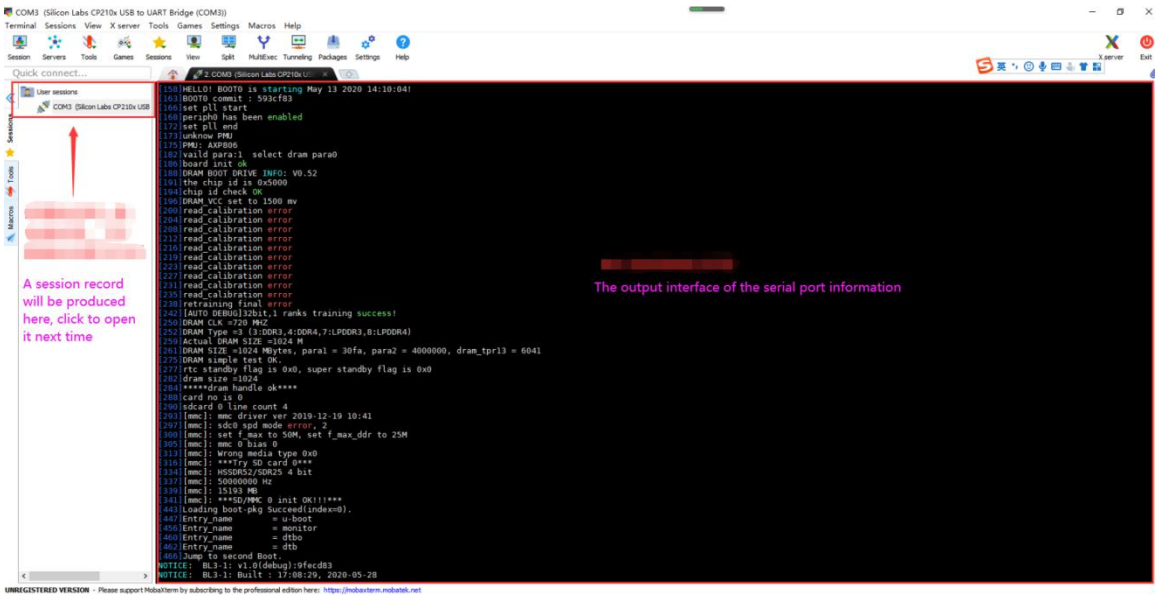
c. Select the port number of the serial port (choose the corresponding port number according to the actual situation), if you can't see the port number, please use the **360 driver master** to scan and install the USB to TTL serial chip driver

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

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 type and kernel version

Release version	Kernel version	Server version	Desktop version
Ubuntu 18.04	linux4.9	Support	Support
Ubuntu 20.04	linux4.9	Support	Support
Debian 10	linux4.9	Support	Support
Ubuntu 20.04	linux5.10	Support	Support
Debian 10	linux5.10	Support	Support

3.2. Linux4.9 kernel driver adaptation situation

Function	Status
HDMI Video	OK
HDMI Audio	OK
USB3.0	OK
USB2.0x2	OK
TF card boot	OK
Network Card	OK
IR Receiver	OK
WIFI	OK
BT	OK
Headphone Audio	OK
MIC recording	OK
USB camera	OK
LED	OK
26pin GPIO	OK
I2C0	OK
SPI1	OK
UART3	OK
Temperature Sensor	OK
Hardware watchdog	OK

Switch button	OK
eMMC boot	OK
TV-OUT	NO
Mali GPU	NO
Video codec	NO

3. 3. linux5.10 kernel driver adaptation situation

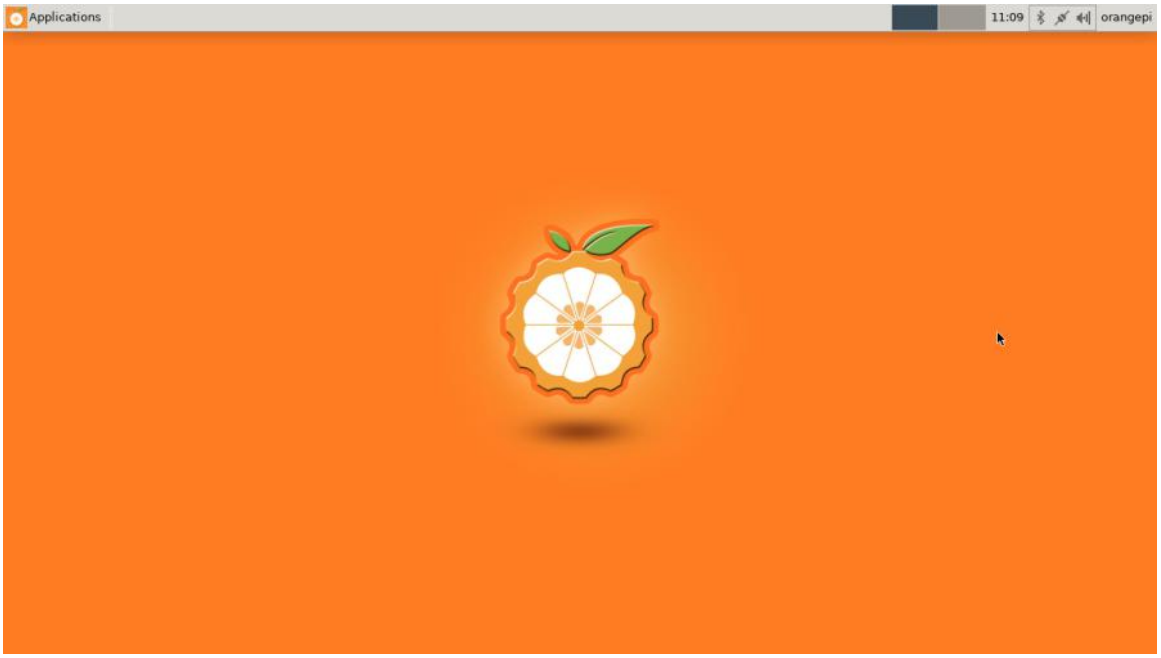
Function	Status
HDMI video	OK
HDMI Audio	OK
USB3.0	OK
USB2.0 x 2	OK
TF card boot	OK
Network card	OK
IR Receiver	OK
WIFI	OK
BT	OK
Headphone Audio	NO
MIC recording	NO
USB camera	OK
LED	OK
26pin GPIO	OK
I2C0	OK
SPI1	OK
UART 3	OK
Temperature Sensor	OK
Hardware watchdog	OK
Switch button	OK
eMMC boot	OK
TV-OUT	NO
Mali GPU	NO
Video codec	NO

3. 4. Linux system default login account and password

Account	Password
root	orangepi
orangepi	orangepi

3. 5. Linux desktop version system automatic login instructions

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



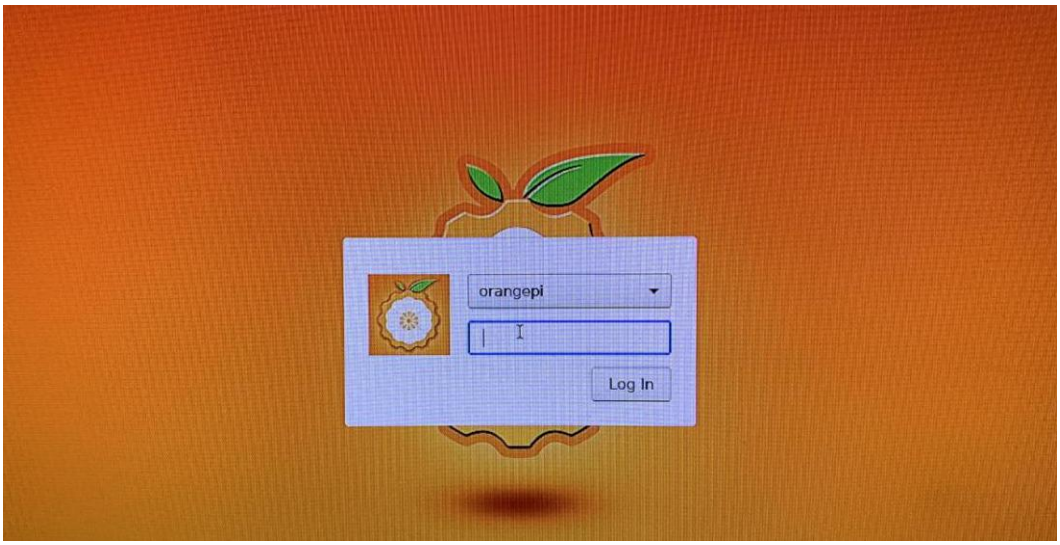
2) Modify the configuration in /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf to prohibit the desktop version system from automatically logging in to the desktop. The modification command is as follows, or you can open the configuration file to modify it directly

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

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



3.6. 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 there is no need to manually expand**

2) After logging in to the system, you can use the df -h command to check 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@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            430M     0  430M   0% /dev
tmpfs           100M  5.6M   95M   6% /run
```

```
/dev/mmcblk0p1     15G   915M     14G     7% /
tmpfs              500M        0   500M     0% /dev/shm
```

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), 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
```

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

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

d.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
```

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

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

```
root@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            927M     0  927M   0% /dev
tmpfs           200M  5.6M  194M   3% /run
/dev/mmcblk0p1  1.5G  1.3G  196M  87% /
tmpfs           997M     0  997M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           997M     0  997M   0% /sys/fs/cgroup
tmpfs           997M  4.0K  997M   1% /tmp
/dev/zram0       49M  1.5M   44M   4% /var/log
tmpfs           200M     0  200M   0% /run/user/0
```

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

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

```
Starting kernel ...

Uncompressing Linux... done, booting the kernel.

Orange Pi 2.1.6 Focal ttyS0

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

3) The above commands are actually to set the variables in **/boot/orangepiEnv.txt**, after

setting, you can open **/boot/orangepiEnv.txt** to check

```
root@orangepi:~# 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. 8. 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@orangepi:~# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.47  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::e56:c34d:62f0:8d6e  prefixlen 64  scopeid 0x20<link>
        ether 02:81:3e:a8:58:d8  txqueuelen 1000  (Ethernet)
        RX packets 2165  bytes 177198 (177.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 312  bytes 40435 (40.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 39
```

4) The command to test network connectivity is as follows

```
root@orangepi:~# ping www.baidu.com -I eth0
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=56 time=6.74 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=56 time=6.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=56 time=6.26 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=56 time=7.27 ms
```

```
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 6.260/6.770/7.275/0.373 ms
```

3.9. 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

3.9.1. 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@192.168.1.36        (Need to be replaced with the IP address of the development board)
root@192.168.1.36's password:        （Enter the password here, the default password is orangepi）
```

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

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

Welcome to Orange Pi Focal with Linux 5.10.46-sunxi64

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

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

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

root@orangepi3-lts:~#
```

4) If the following error is prompted during ssh login

```
test@test:~$ ssh root@192.168.1.36
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@orangepi:~# rm /etc/ssh/ssh_host_*
root@orangepi:~# dpkg-reconfigure openssh-server
```

3.9.2. SSH remote login development board under Windows

1) First obtain 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

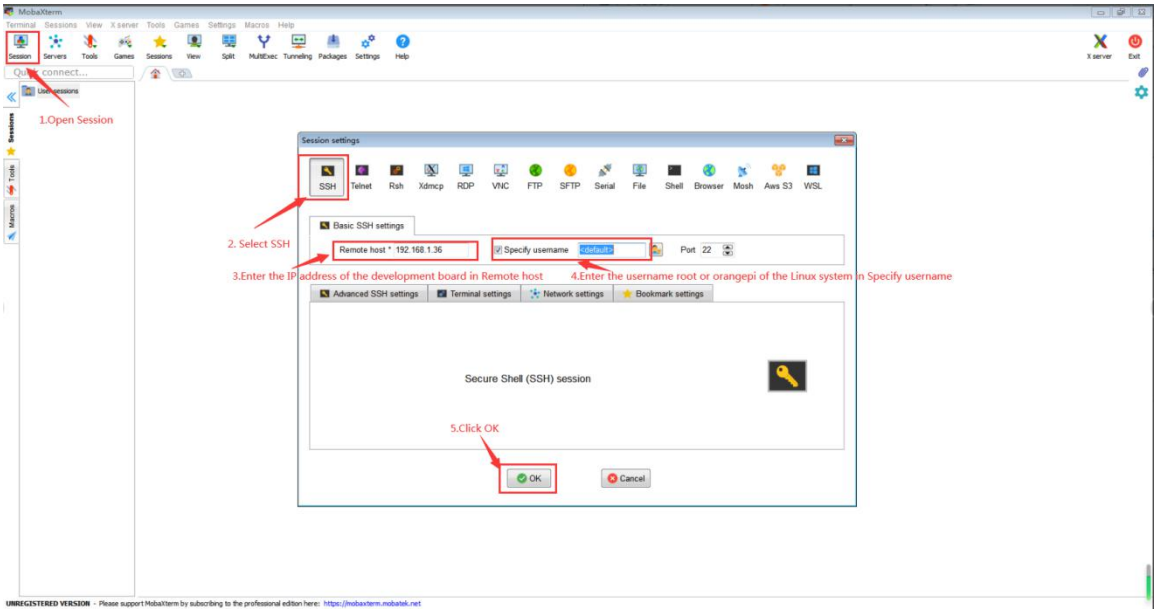
a. Open **Session**

b. Then select **SSH** in **Session Setting**

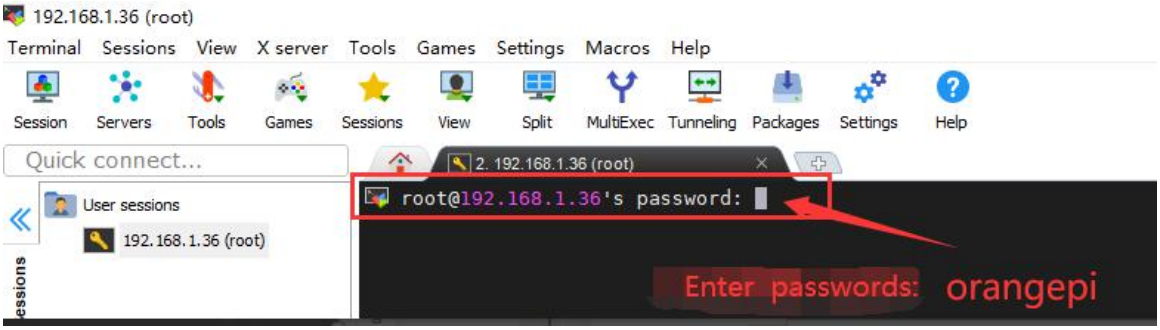
c. Then enter the IP address of the development board in **Remote host**

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

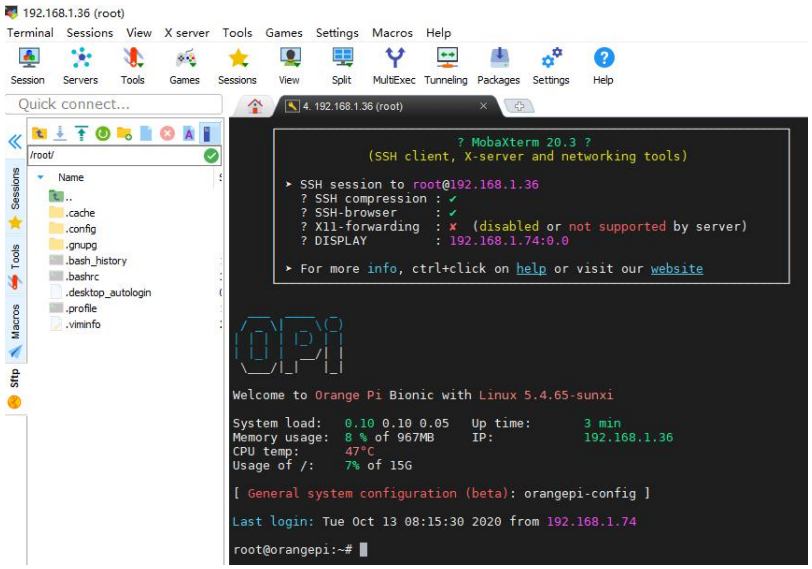
e. Finally click **OK**



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



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



3. 10. 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. 11. Linux4.9 system HDMI resolution setting

Note: This method is only applicable to linux4.9 kernel systems, and linux5.10 kernel systems cannot be used

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

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

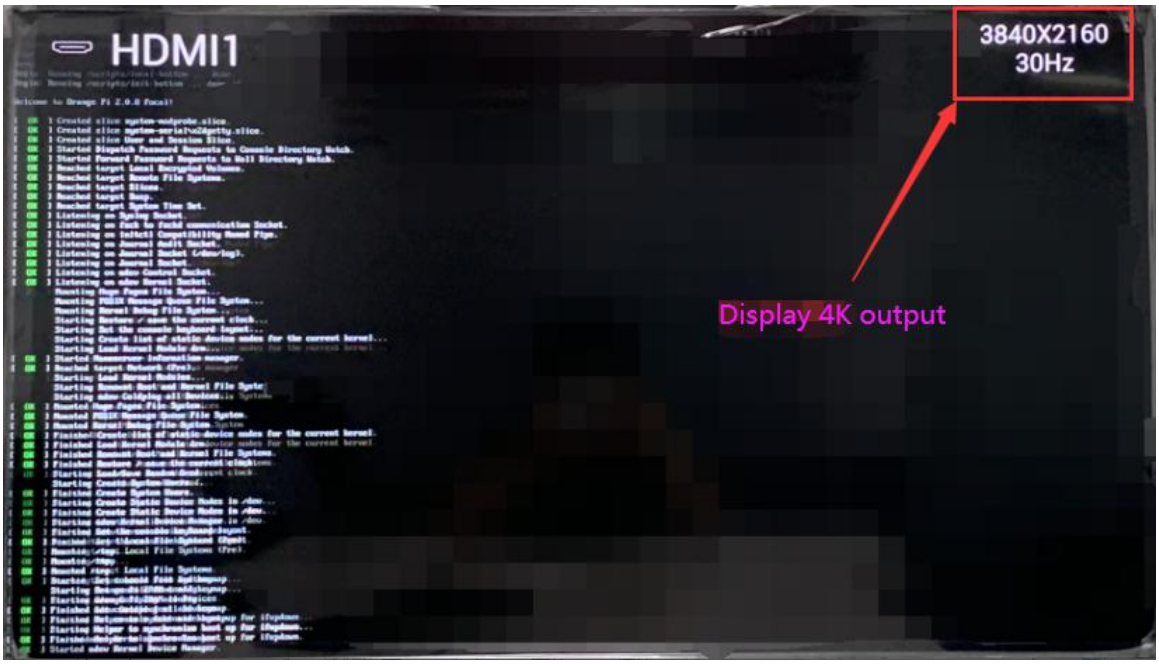
2) The disp_mode variable supports setting values as shown in the following table

disp_mode supported values	HDMI resolution	HDMI refresh rate
480i	720x480	60
576i	720x480	50
480p	720x480	60
576p	720x576	60
720p50	1280x720	50
720p60	1280x720	60
1080i50	1920x1080	50
1080i60	1920x1080	60
1080p24	1920x1080	24
1080p50	1920x1080	50
1080p60	1920x1080	60
2160p24	3840x2160	24
2160p25	3840x2160	25
2160p30	3840x2160	30

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

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

a. Received 4K TV display as shown below



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

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

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

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

3. 12. Modification method of framebuffer width and height in Linux4.9 system

Note: This method is only applicable to linux4.9 kernel systems, and linux5.10 kernel systems cannot be used

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

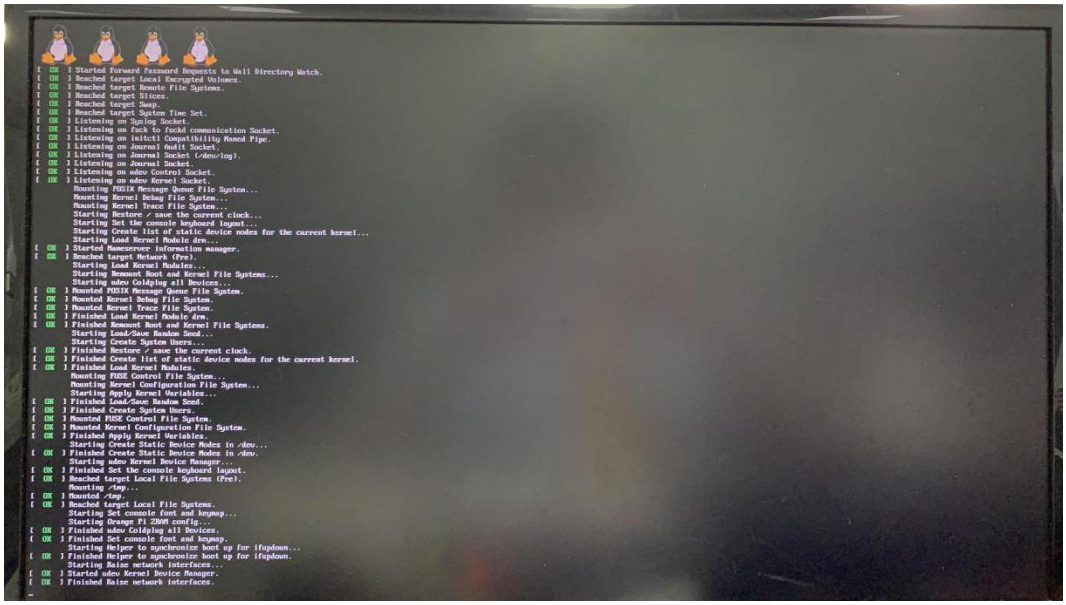
```
root@orangepi:~# cat /boot/orangepiEnv.txt
verbosity=1
console=both
disp_mode=1080p60
fb0_width=1280
fb0_height=720
```

2) The **reference values** corresponding to different resolutions of fb0_width and fb0_height are shown below. It should be noted that when the resolution of HDMI is set to **2160p24, 2160p25 and 2160p30**, Framebuffer only supports up to **1920x1080**

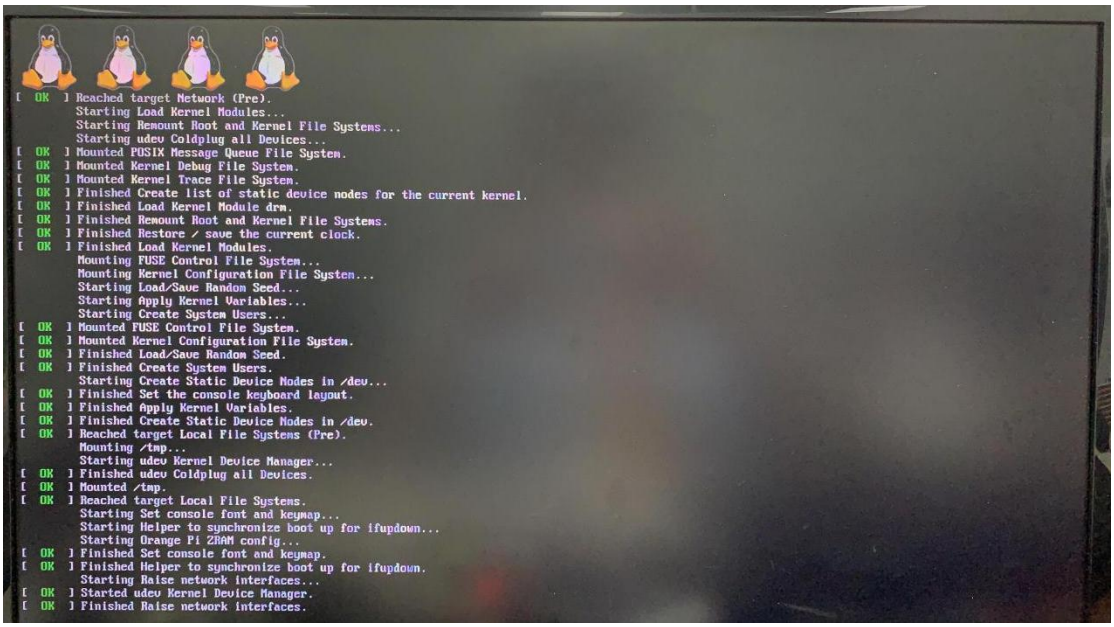
HDMI resolution	fb0_width	fb0_height
480p	720	480
576p	720	576
720p	1280	720
1080p	1920	1080
2160p	1920	1080

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

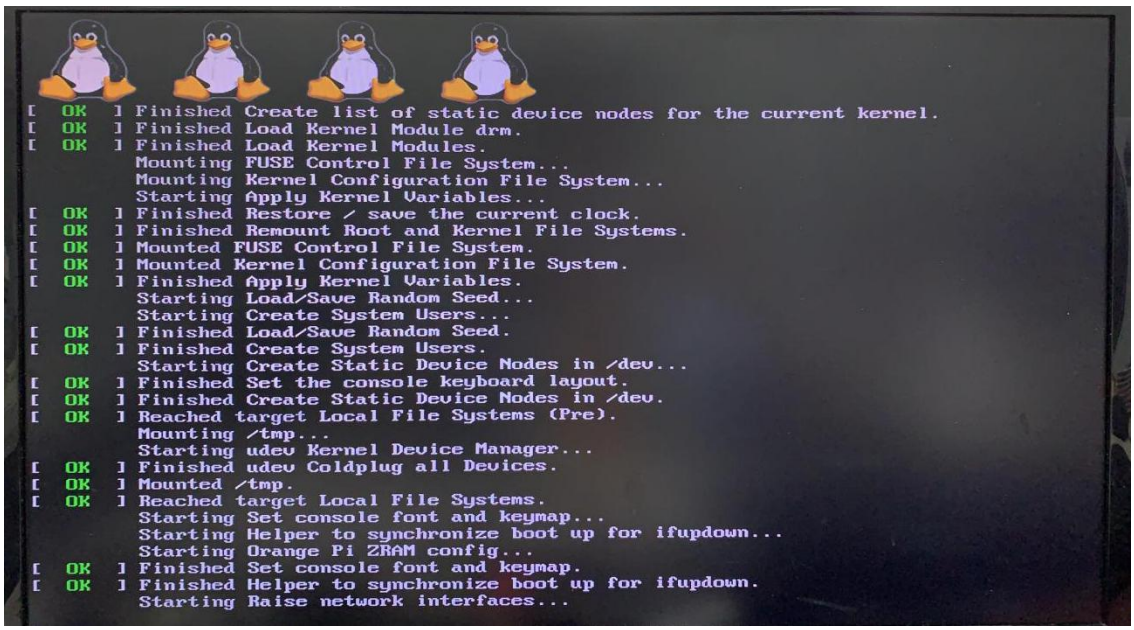
a. HDMI resolution is 1080p60, fb0_width and fb0_height are 1920x1080 display situation



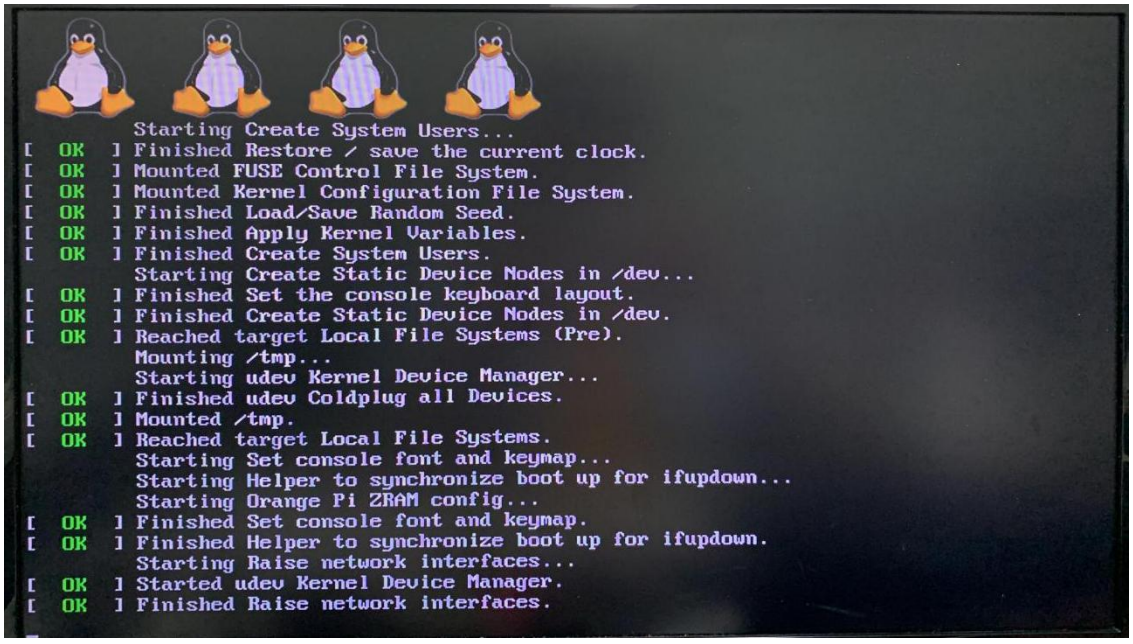
b. HDMI resolution is 1080p60, fb0_width and fb0_height are 1280x720 display



c. HDMI resolution is 1080p60, fb0_width and fb0_height are 720x576 display



d. HDMI resolution is 1080p60, fb0_width and fb0_height are 720x480 display



3. 13. Framebuffer cursor related settings

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

```
root@orangepi:~# echo 1 > /sys/class/graphics/fbcon/cursor_blink      #Cursor blinks
```

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

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

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

3. 14. 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. 14. 1. The server version image connects to WIFI by command

When the development board is not connected to the Ethernet or HDMI display, but only to the serial port, it is recommended to use the commands demonstrated in this section to connect to the WIFI network. Because nmtui can only display characters in some serial port software (such as minicom), 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 through SSH

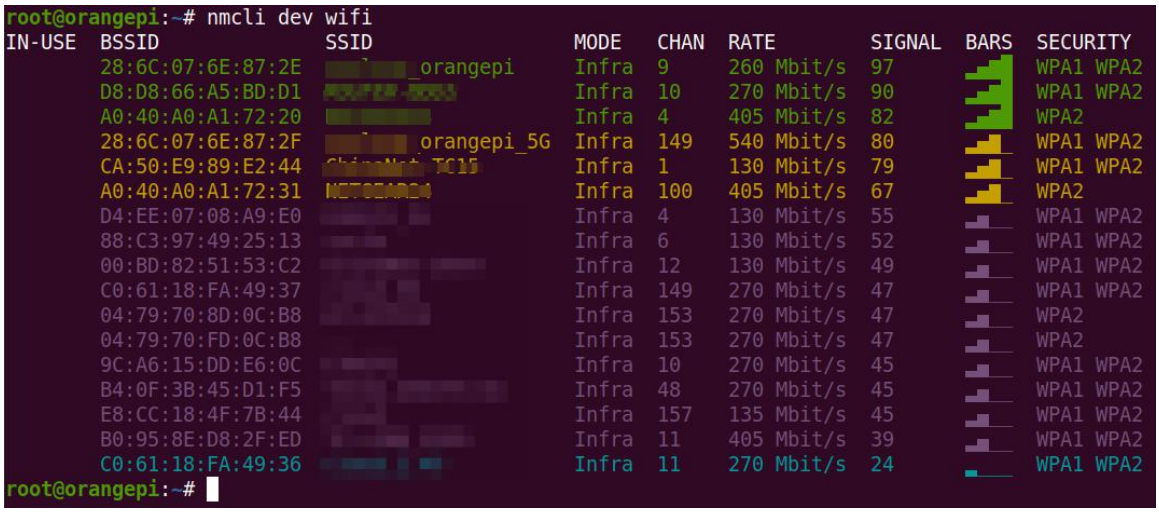
b. If the debugging serial port is connected to the development board, you can use the serial terminal to log in to the Linux system

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

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

```
root@orangepi:~# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.49  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
```

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

```
root@orangepi:~# ping www.baidu.com -I wlan0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.119 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
```

3.14.2. The server version image connects to WIFI in a graphical way

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

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

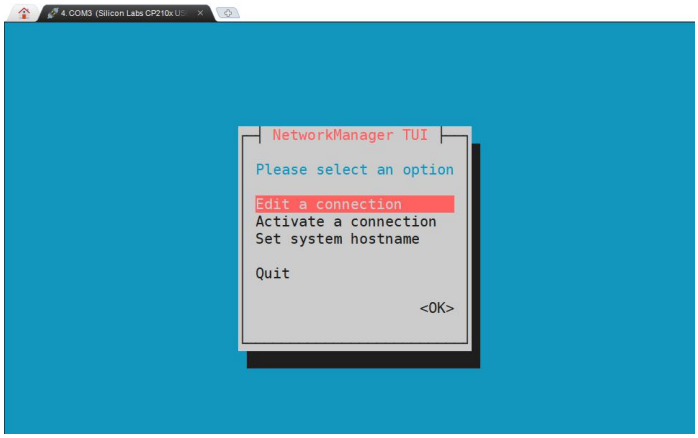
b. If the debugging serial port is connected to the development board, you can use the serial terminal to log in to the linux system (for serial port software, please use MobaXterm, and minicom cannot display the graphical interface)

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

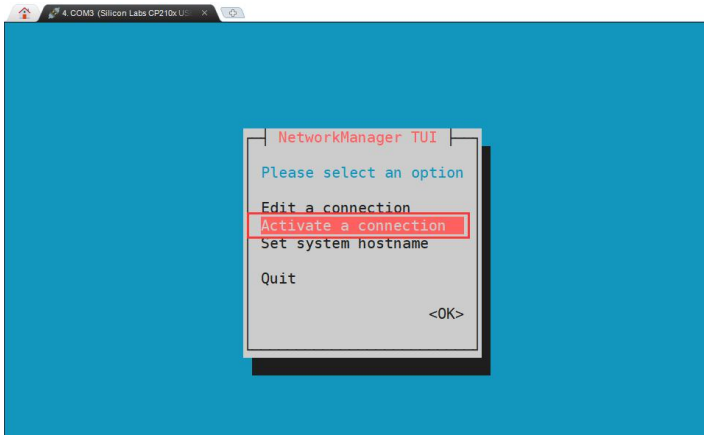
2) Then enter the nmtui command in the command line to open the wifi connection interface

```
root@orangepi:~# nmtui
```

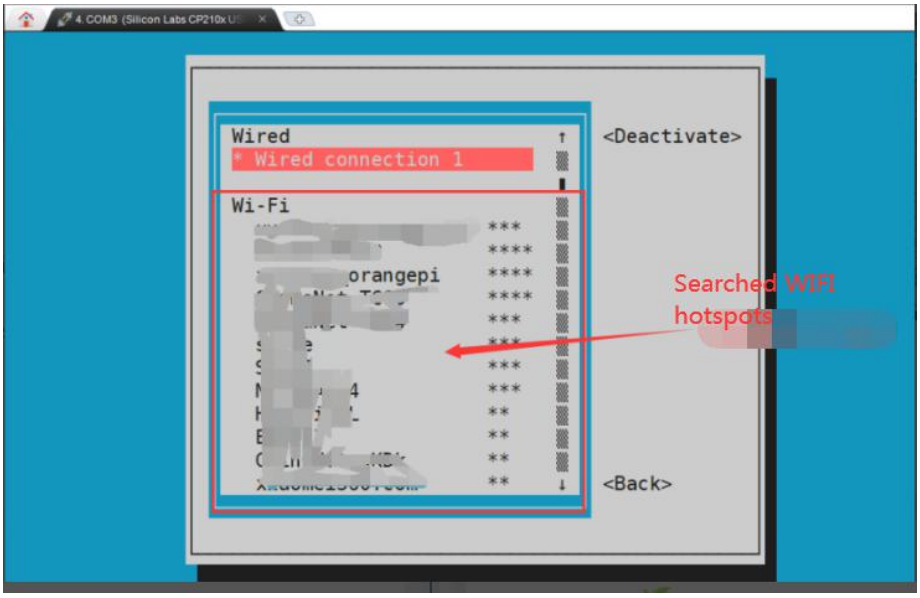
3) Enter the nmtui command to open the interface as shown below



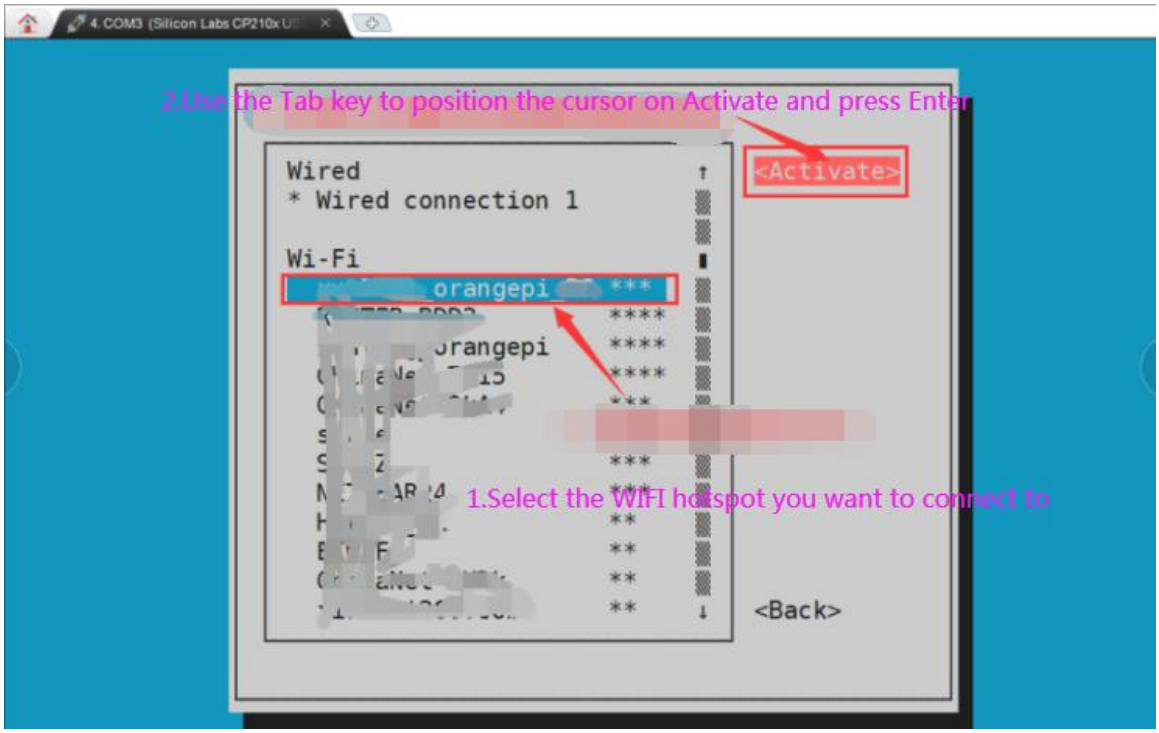
4) Select **Activate a connect** and press Enter



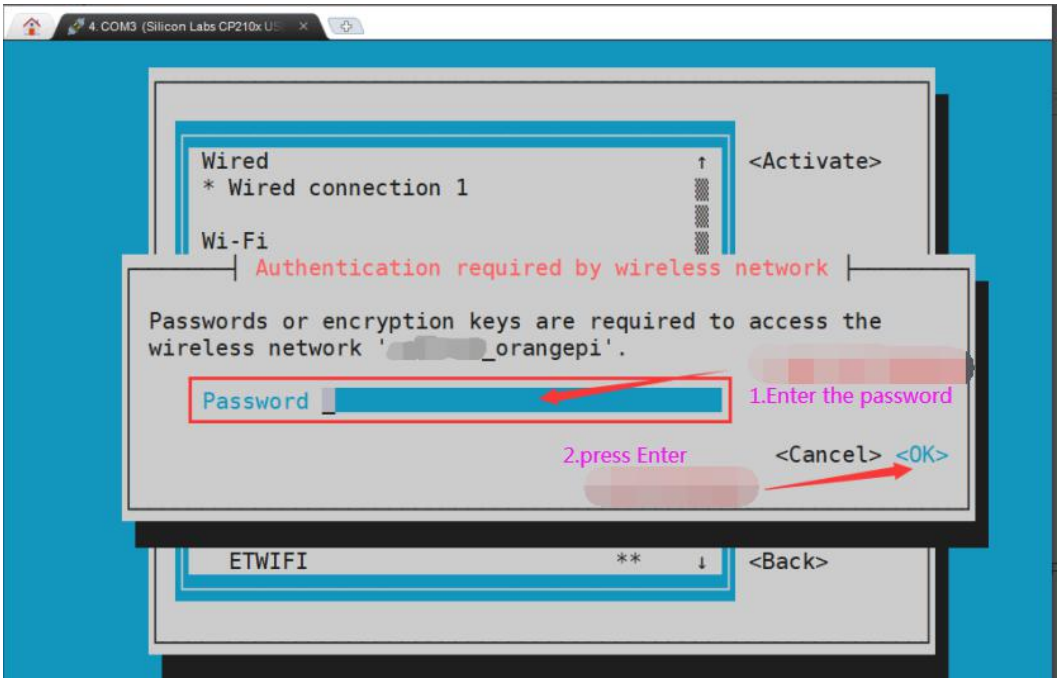
5) Then you can see all the searched WIFI hotspots



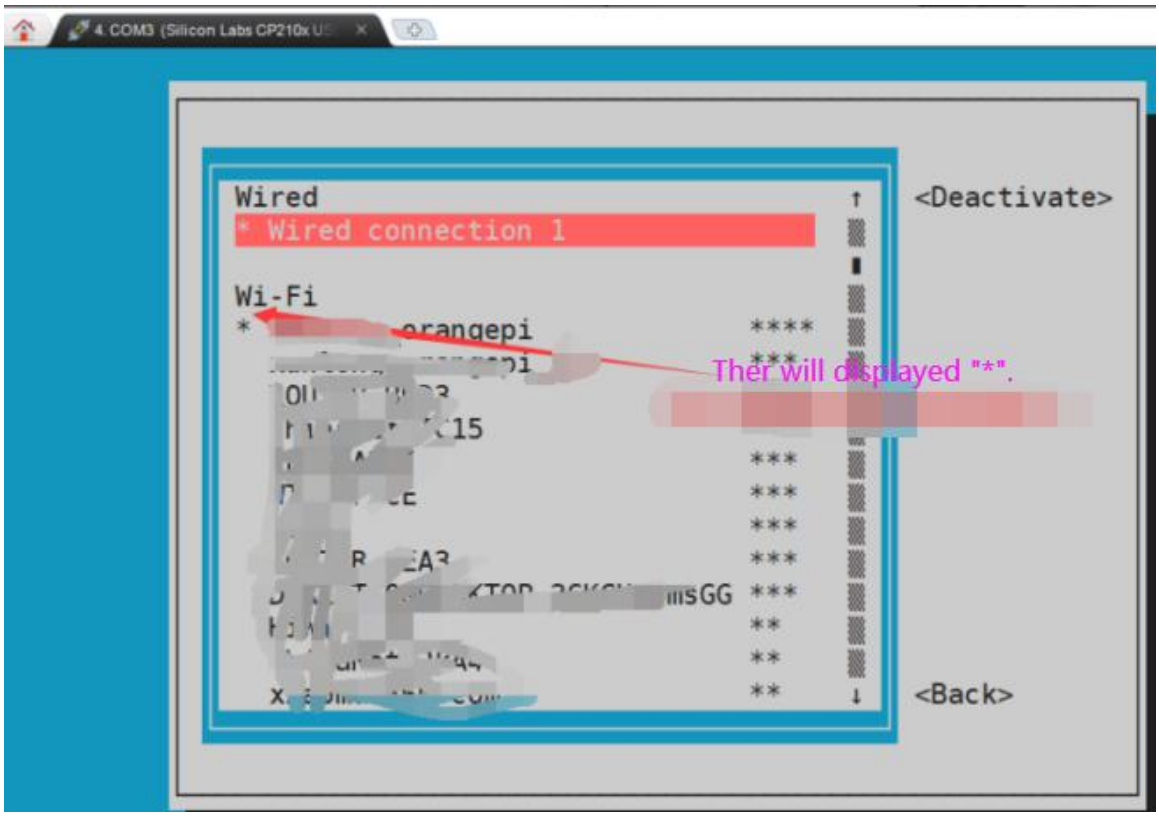
6) Select the WIFI hotspot you want to connect to, then use the Tab key to position the cursor on **Activate** and press Enter



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



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



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

```
root@orangepi:~# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.49  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

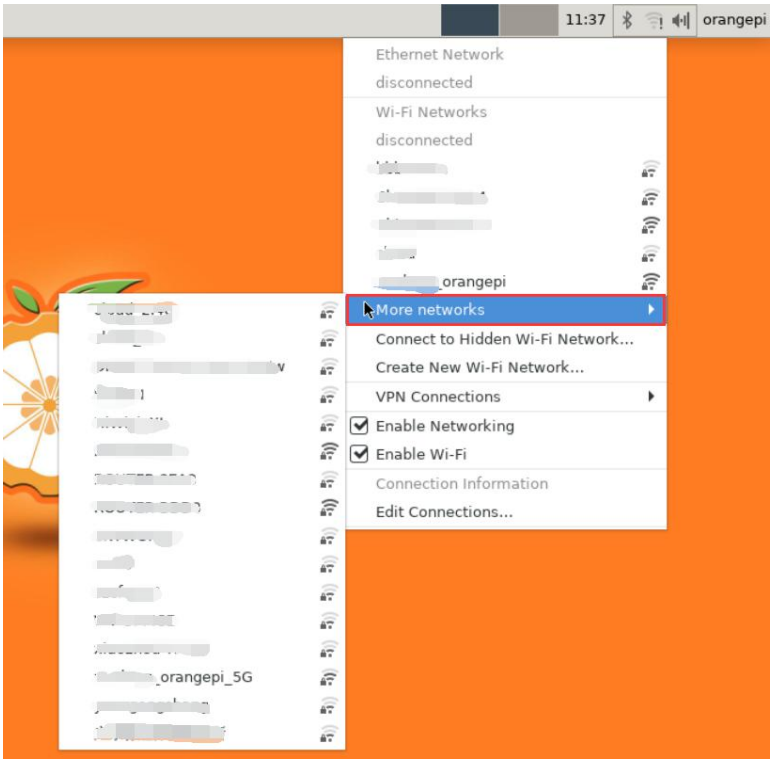
```
root@orangepi:~# ping www.baidu.com -I wlan0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.119 wlan0: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=15.0 ms
```

```
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. 14. 3. Test method of desktop version image

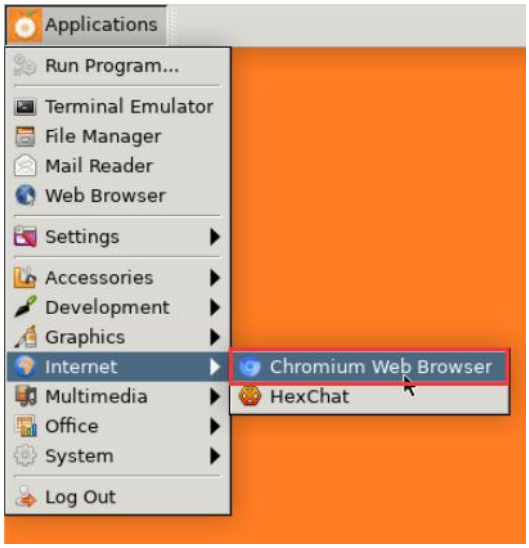
1) Click the network configuration icon in the upper right corner of the desktop (please do not connect the network cable when testing WIFI)

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

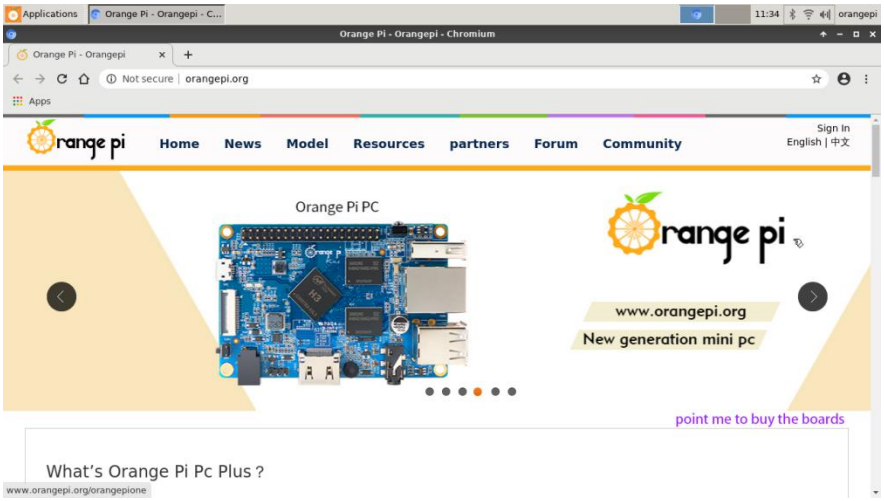


3) Then enter the password of the WIFI hotspot, and then click **Connect** to start connecting to the WIFI

4) After connecting to 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 can open other web pages, it means that the WIFI connection is normal

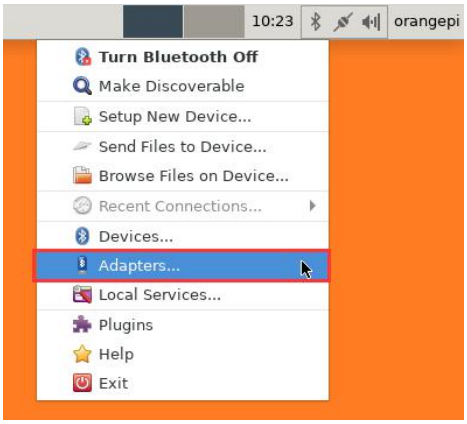


3. 15. How to use Bluetooth

3. 15. 1. Test method of desktop version image

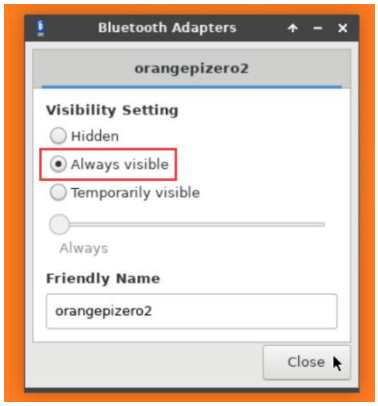
1) Click the Bluetooth icon in the upper right corner of the desktop

2) Then select the adapter

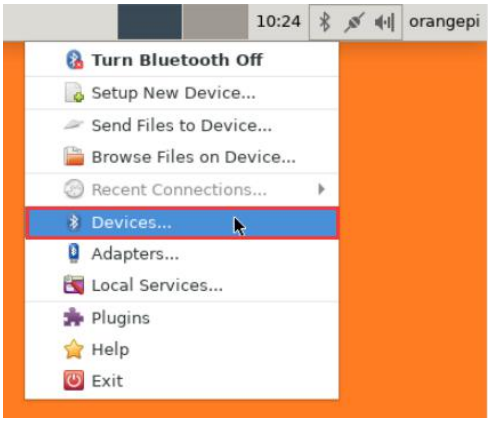


3) Set **Visibility Setting** to **Always visible** in the Bluetooth adapter setting interface, and

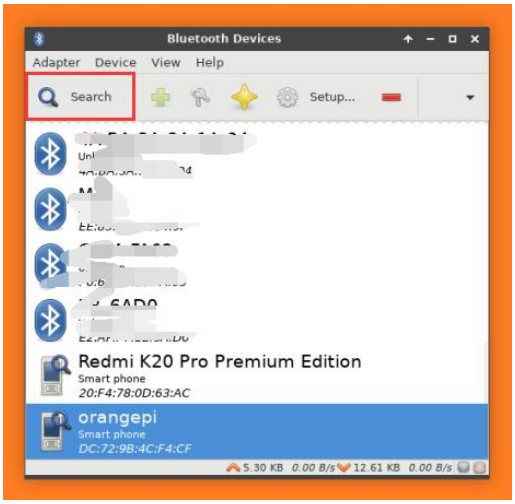
then click **close** to close



4) Then open the configuration interface of the Bluetooth device

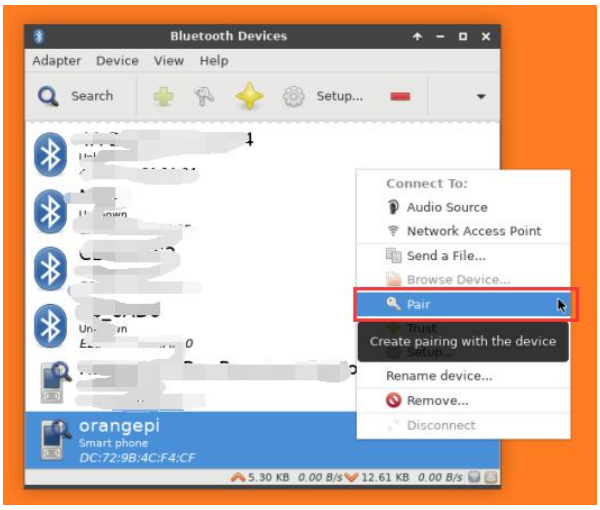


5) Click **Search** to start scanning surrounding Bluetooth devices

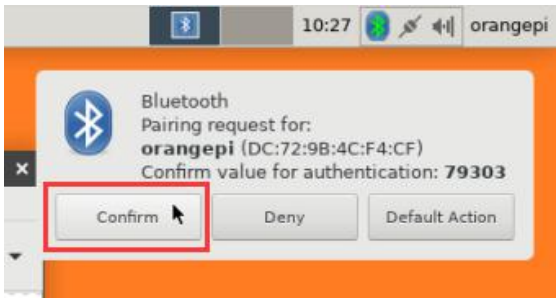


6) Then select the Bluetooth device you want to connect to, and then click the right

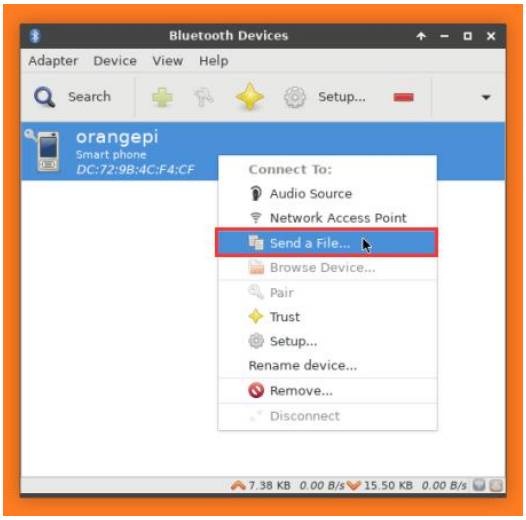
mouse button to pop up the operation interface of the Bluetooth device. Select **Pair** to start pairing. The demonstration here is pairing with an Android phone



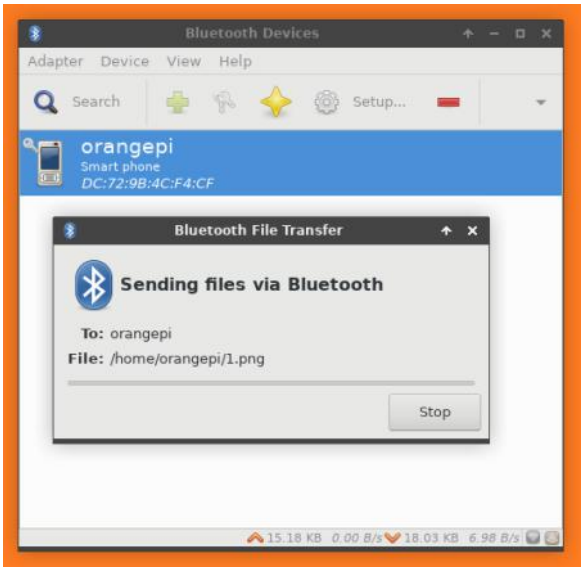
7) When pairing, a pairing confirmation box will pop up in the upper right corner of the desktop, select **Confirm** to confirm, and the phone also needs to be confirmed at this time



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 as follows



3. 15. 2. How to use the server version image

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

```
root@orangepi:~# hciconfig -a
hci0:   Type: Primary   Bus: UART
        BD Address: 10:11:12:13:14:15   ACL MTU: 1021:8   SCO MTU: 240:3
        UP RUNNING
        RX bytes:646 acl:0 sco:0 events:37 errors:0
        TX bytes:2650 acl:0 sco:0 commands:37 errors:0
        Features: 0xbf 0xff 0x8d 0xfe 0xdb 0x3d 0x7b 0xc7
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
```

```
        Link policy:
        Link mode: SLAVE ACCEPT
        Name: 'orangepi3-lts'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 5.0 (0x9)    Revision: 0x400
        LMP Version: 5.0 (0x9)    Subversion: 0x400
        Manufacturer: Spreadtrum Communications Shanghai Ltd (492)
```

2) Use bluetoothctl to scan for Bluetooth devices

```
root@orangepi:~# bluetoothctl
[NEW] Controller 10:11:12:13:14:15 orangepi3-lts [default]
Agent registered
[bluetooth]# power on           #Enable controller
Changing power on succeeded
[bluetooth]# discoverable on       #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
```

3.16. On-board LED light test instructions

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

	Yellow light	red light
u-boot startup phase	Turn off	Bright
Kernel boot to enter the system	Bright	Turn off
GPIO port	PL7	PL4

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

a. First enter the yellow light setting directory

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

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

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

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

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

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

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

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

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

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

a. First enter the red light setting directory

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

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

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

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

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

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

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

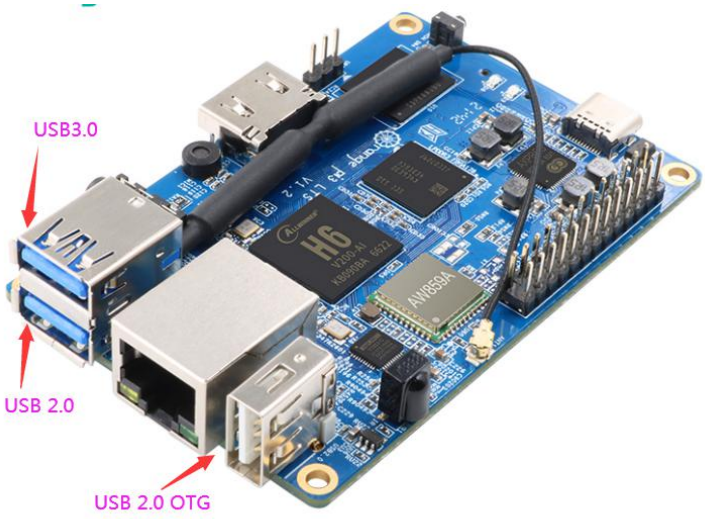
e. The command to set the red light to stop flashing is as follows

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

3. 17. USB interface test

3. 17. 1. USB2.0 and USB3.0 interface description

1) Orange Pi 3 LTS has 3 USB ports in total, including a USB3.0, a USB2.0 and a USB2.0 OTG port. The USB2.0 OTG port defaults to Host mode and can be used with a mouse, keyboard or U disk



3. 17. 2. Connect mouse or keyboard test

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

development board

2) When using a mouse, you need to connect the Orange Pi development board to the HDMI display

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

3. 17. 3. 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@orangepi:~# cat /proc/partitions | grep "sd*"
major minor  #blocks  name
   8        0   30044160 sda
   8        1   30043119 sda1
```

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

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

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

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

3. 18. USB wireless network card test

Note: This method is only applicable to linux5.10 kernel systems, there is no debugging for linux4.9 kernel systems

The USB wireless network cards that **have been tested** in the linux5.10 system are as follows. Please test other types of USB wireless network cards by yourself. If you can't

use them, you need to transplant the corresponding USB wireless network card drivers.

Serial number	model
1	RTL8723BU
2	RTL8821CU

3.18.1.1. RTL8723BU test

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

2) Then the Linux system will automatically load RTL8723BU related kernel modules, and you can see the following output through the lsmod command

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

3) Through the dmesg command, you can see the loading information of the RTL8723BU module

```
root@orangepi:~# dmesg | tail
[ 1583.908354] usb 3-1.1: 1e0: ff ff ff ff ff ff ff ff
[ 1583.908357] usb 3-1.1: 1e8: ff ff ff ff ff ff ff ff
[ 1583.908360] usb 3-1.1: 1f0: ff ff ff ff ff ff ff ff
[ 1583.908363] usb 3-1.1: 1f8: ff ff ff ff ff ff ff ff
[ 1583.908369] usb 3-1.1: RTL8723BU rev E (SMIC) 1T1R, TX queues 3, WiFi=1, BT=1, GPS=0, HI PA=0
[ 1583.908373] usb 3-1.1: RTL8723BU MAC: 00:13:ef:f4:58:ae
[ 1583.908377] usb 3-1.1: rtl8xxxu: Loading firmware rtlwifi/rtl8723bu_nic.bin
[ 1583.908536] usb 3-1.1: Firmware revision 35.0 (signature 0x5301)
[ 1584.013655] Bluetooth: hci1: RTL: fw version 0x1e4cc3ff
[ 1584.813524] rtl8xxxu 3-1.1:1.2 wlx0013eff458ae: renamed from wlan1
```

4) Then you can see the device node of RTL8723BU WIFI through the ifconfig command. Please refer to the **WIFI connection test** section for the WIFI connection and test method, and I won't repeat it here.

```
root@orangepi:~# ifconfig wlx0013eff458ae
```

```
wlx0013eff458ae: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:13:ef:f4:58:ae  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

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

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

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

3.18.1.2. RTL8821CU test

1) First insert the RTL8821CU wireless network card module into the USB interface of the development board

2) Then the linux system will automatically load RTL8821CU related kernel modules, and the following output can be seen through the lsmod command

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

3) Through the dmesg command, you can see the loading information of the

RTL8821CU module

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

4) Then you can see the device node of RTL8821CU WIFI through the ifconfig command. Please refer to the **WIFI connection test** section for the connection and test method of WIFI, which will not be repeated here.

```
root@orangepi:~# ifconfig wlxd0c0bf8742cd
wlxd0c0bf8742cd: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether d0:c0:bf:87:42:cd  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

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

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

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

3. 19. USB network card test

1) The USB network cards that **have been tested** and can be used are as follows

Serial number	model
1	RTL8152B USB 100M Ethernet card
2	RTL8153 USB Gigabit LAN

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

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

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

```
root@orangepi:~# ifconfig
enx00e04c362017: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

```
        inet 192.168.1.177  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::681f:d293:4bc5:e9fd  prefixlen 64  scopeid 0x20<link>
        ether 00:e0:4c:36:20:17  txqueuelen 1000  (Ethernet)
        RX packets 1849  bytes 134590 (134.5 KB)
        RX errors 0  dropped 125  overruns 0  frame 0
        TX packets 33  bytes 2834 (2.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4) The command to test network connectivity is as follows

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

3. 20. USB camera test

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

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

a. linux4.9 system display is as follows

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

b. linux5.10 system display as shown below

```
root@orangepi:~# lsmod | grep "uvc"
uvcvideo               102400  0
videobuf2_vmalloc       20480  1 uvcvideo
videobuf2_v4l2          28672  1 uvcvideo
```

```
videobuf2_common       53248  2 videobuf2_v4l2,uvcvideo
videodev              233472  3 videobuf2_v4l2,uvcvideo,videobuf2_common
mc                     49152  4
videodev,videobuf2_v4l2,uvcvideo,videobuf2_common
```

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

4) Use fswebcam to test the USB camera

a. Install fswebcam

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

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

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

b) --no-banner is used to remove the watermark of the photo

c) -r option 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@orangepi:~# fswebcam -d /dev/video0 --no-banner -r 1280x720 -S 5 ./image.jpg
```

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

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

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

5) Use motion to test the USB camera

a. Install the camera test software motion

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

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

```
root@orangepi:~# 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

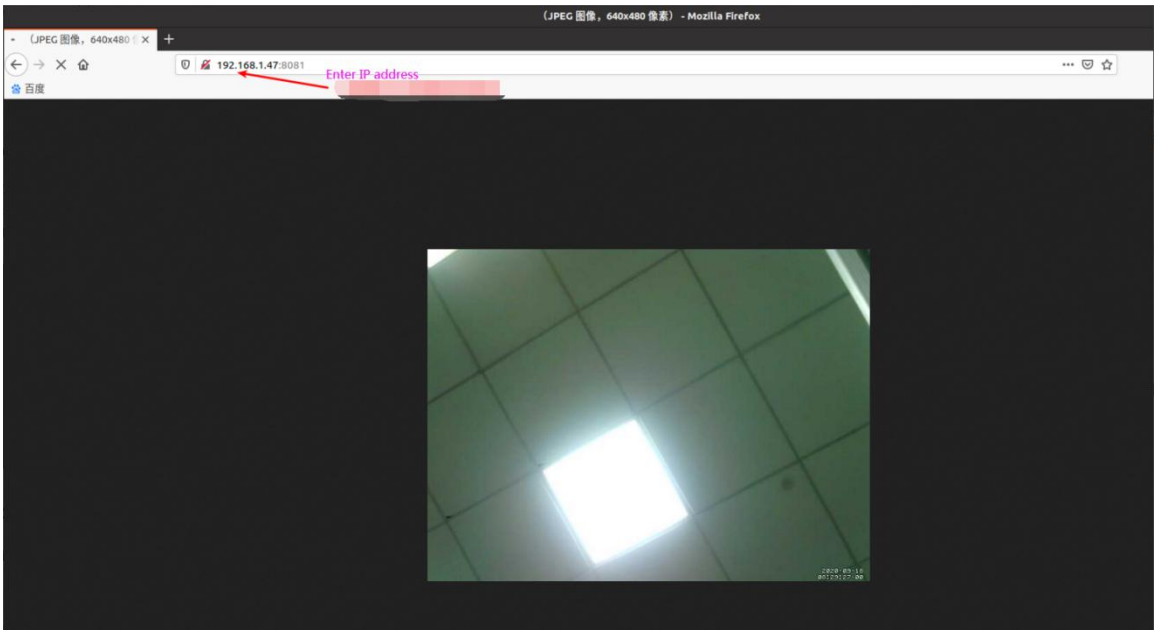
```
root@orangepi:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf        (This is a command)
```

d. Then restart the motion service

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

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

f. 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



6) Use mjpg-streamer to test the USB camera

a. Download mjpg-streamer

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

b. Install dependent packages

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

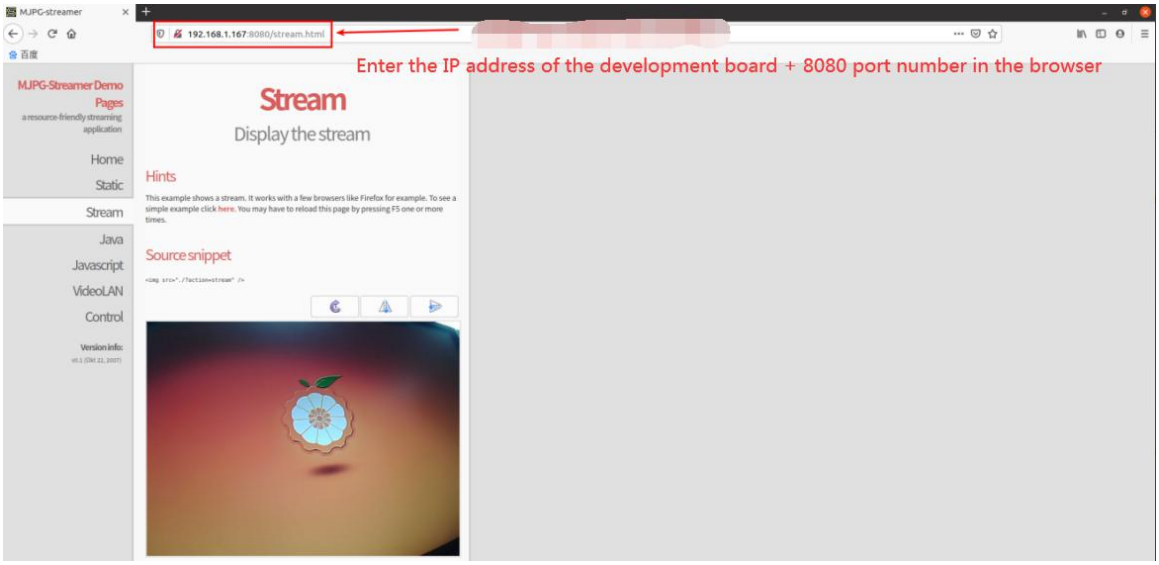
c. Compile and install mjpg-streamer

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

d. Then enter the following command to start mjpg_streamer

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

e. Then enter 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, which is much smoother than motion, and you will not feel any lag when using mjpg-streamer

3. 21. Audio test

Note: linux5.10 system does not support earphone playback and MIC recording, only linux4.9 system supports

3.21.1. linux4.9 system headphone jack play audio test

1) First insert the earphones in the audio interface

2) Through the **aplay -l** command, you can view the sound card devices supported by the linux system, where **sndacx00codec** is the sound card device required for headset playback

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

3) Then you can use the aplay command to play the test audio file that comes with the system. If the earphone can hear the sound, it means that the audio is playing normally

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

3.21.2. linux4.9 system MIC recording test

1) The recording command is as follows

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

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

```
root@orangepi:~# ls -lh
total 864K
-rw-r--r-- 1 root root 862K Dec    1 06:25 test.wav
root@orangepi:~# aplay -D hw:1,0 test.wav
```

3.21.3. 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) HDMI audio playback does not require other settings, you can hear the sound directly by using the aplay command to play the audio

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

3.22. Infrared receiving test

1) Install ir-keytable infrared test software

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

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

a. linux4.9 system output is as follows

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

b. linux5.10 system output is as follows

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

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

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

a. linux4.9 system output is as follows

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

b. linux5.10 system output is as follows

```
root@orangepi:~# ir-keytable -c -p NEC -t
Old keytable cleared
Protocols changed to nec
Testing events. Please, press CTRL-C to abort.
3161.772262: lirc protocol(nec): scancode = 0x45c
3161.772323: event type EV_MSC(0x04): scancode = 0x45c
3161.772323: event type EV_SYN(0x00).
```

3. 23. Temperature sensor

1) H6 has 2 temperature sensors in total, the command to check the temperature is as follows:

a. sensor0：CPU

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

b. sensor1：GPU

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

3. 24. How to install Docker

1) First install the following packages

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y apt-transport-https ca-certificates curl    \
  software-properties-common
```

2) Add the key of Alibaba Cloud docker

```
root@orangepi:~# curl -fsSL http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg \
  | sudo apt-key add    -
```

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

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

4) Install the latest version of docker-ce

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

5) Test docker

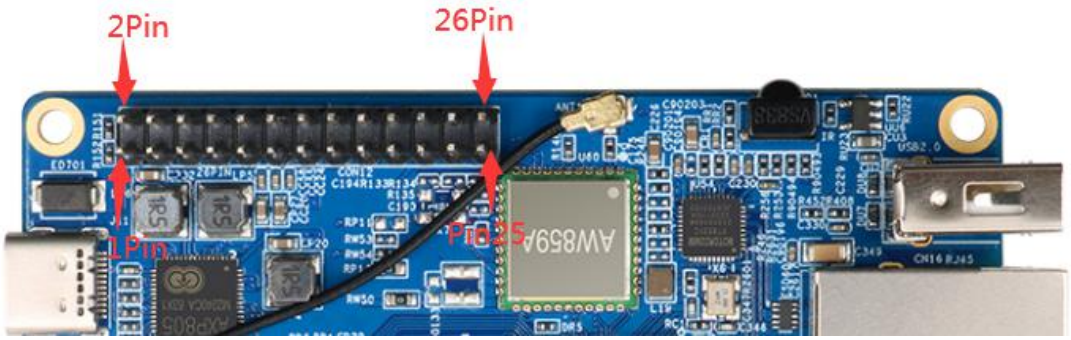
```
root@orangepi:~# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
256ab8fe8778: Pull complete
```

```
Digest:
sha256:7f0a9f93b4aa3022c3a4c147a449ef11e0941a1fd0bf4a8e6c9408b2600777c5
Status: Downloaded newer image for hello-world:latest

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

3. 25. 26 Pin interface pin description

1) Please refer to the figure below for the sequence of the 26 pin interface of the Orange Pi 3 LTS development board



2) The function of the 26 pin interface of the Orange Pi 3 LTS development board is shown in the table below

GPIO serial number	GPIO	function	Pin
		3.3V	1
122	PD26	TWI0-SDA	3
121	PD25	TWI0-SCK	5
118	PD22	PWM0	7
		GND	9
120	PD24	UART3_RX	11
119	PD23	UART3_TX	13
362	PL10	PL10	15
		3.3V	17
229	PH5	SPI1_MOSI	19

Pin	function	GPIO	GPIO serial number
2	5V		
4	5V		
6	GND		
8	PL2	PL2	354
10	PL3	PL3	355
12	PD18	PD18	114
14	GND		
16	PD15	PD15	111
18	PD16	PD16	112
20	GND		

230	PH6	SPI1_MISO	21
228	PH4	SPI1_CLK	23
		GND	25

22	PD21	PD21	117
24	SPI1_CS	PH3	227
26	PL8	PL8	360

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

3. 26. How to install wiringOP

1) Download the code of wiringOP

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

2) Compile wiringOP

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

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

```
root@orangepi3:~/wiringOP# gpio readall
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  122 |   0 |    SDA.0 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  121 |   1 |    SCL.0 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |  118 |   2 |    PWM.0 |  OFF | 0 |  7 || 8  | 0 | OFF  | PL02     | 3   | 354  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OFF  | PL03     | 4   | 355  |
 |  120 |   5 |    RXD.3 | ALT4 | 0 | 11 || 12 | 0 | OFF  | PD18     | 6   | 114  |
 |  119 |   7 |    TXD.3 | ALT4 | 0 | 13 || 14 |   |      | GND      |     |      |
 |  362 |   8 |     PL10 |  OFF | 0 | 15 || 16 | 0 | OFF  | PD15     | 9   | 111  |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | OFF  | PD16     | 10  | 112  |
 |  229 |  11 |   MOSI.1 | ALT2 | 0 | 19 || 20 |   |      | GND      |     |      |
 |  230 |  12 |   MISO.1 | ALT2 | 0 | 21 || 22 | 0 | OFF  | PD21     | 13  | 117  |
 |  228 |  14 |   SCLK.1 | ALT2 | 0 | 23 || 24 | 0 | ALT2 | CE.1     | 15  | 227  |
 |      |     |      GND |      |   | 25 || 26 | 0 | OFF  | PL08     | 16  | 360  |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
root@orangepi3:~/wiringOP#
```

3. 27. 26pin interface GPIO, I2C, UART, SPI test

wiringOP has been adapted to the Orange Pi 3 LTS development board, using wiringOP can test the functions of GPIO, I2C, UART and SPI

3. 27. 1. 26pin GPIO port test

1) Below, take pin 7-corresponding to GPIO as PD22-corresponding to wPi serial number as 2-as an example to demonstrate how to set the high and low levels of the GPIO port

```
root@orangepi3:~/wiringOP# gpio readall
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  122 |   0 |    SDA.0 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  121 |   1 |    SCL.0 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |  118 |   2 |    PWM.0 |  OFF | 0 |  7 || 8  | 0 | OFF  | PL02     | 3   | 354  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OFF  | PL03     | 4   | 355  |
 |  120 |   5 |    RXD.3 | ALT4 | 0 | 11 || 12 | 0 | OFF  | PD18     | 6   | 114  |
```

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

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

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

```
root@orangepi3:~/wiringOP# gpio readall
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  122 |   0 |    SDA.0 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  121 |   1 |    SCL.0 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |  118 |   2 |    PWM.0 |  OUT | 0 |  7 || 8  | 0 | OFF  | PL02     | 3   | 354  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OFF  | PL03     | 4   | 355  |
```

3) Then set the GPIO port to output 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@orangepi:~/wiringOP# gpio write 2 0
```

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

```
root@orangepi3:~/wiringOP# gpio readall
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  122 |   0 |    SDA.0 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  121 |   1 |    SCL.0 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |  118 |   2 |    PWM.0 |  OUT | 0 |  7 || 8  | 0 | OFF  | PL02     | 3   | 354  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OFF  | PL03     | 4   | 355  |
```

4) Then set the GPIO port to output 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@orangepi:~/wiringOP# gpio write 2 1
```

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

```
root@orangepi3:~/wiringOP# gpio readall
 +------+-----+----------+------+---+   OPi 3  +---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |  122 |   0 |    SDA.0 |  OFF | 0 |  3 || 4  |   |      | 5V       |     |      |
 |  121 |   1 |    SCL.0 |  OFF | 0 |  5 || 6  |   |      | GND      |     |      |
 |  118 |   2 |    PWM.0 |  OUT | 1 |  7 || 8  | 0 | OFF  | PL02     | 3   | 354  |
 |      |     |      GND |      |   |  9 || 10 | 0 | OFF  | PL03     | 4   | 355  |
```

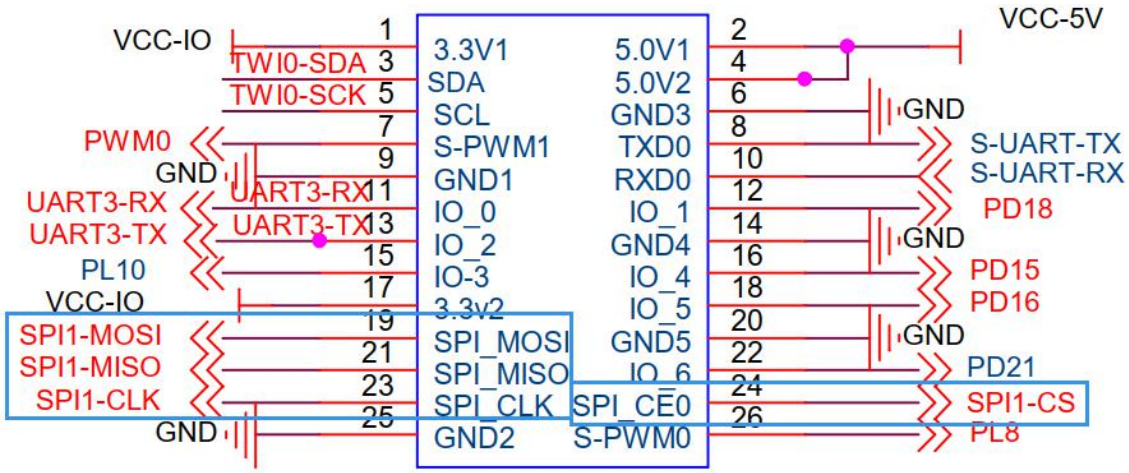
5) The setting method of other pins is similar, just modify the serial number of wPi to the serial number corresponding to the pin.

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

3.27.2. 26pin SPI test

1) In linux5.10 system, the spi controller in 26pin is turned off by default in dts. If you need to use spi, you need to open the configuration of spi first. The linux4.9 system is enabled by default and no additional configuration is required. linux5. The opening method of 10 system spi is as follows:

a. According to the schematic diagram of 26pin, the available spi of the development board is spi1



b. Then set overlays=spi-spidev1 in /boot/orangepiEnv.txt

```
overlays=spi-spidev1
```

c. Then restart the system. When booting, you can see the configuration output of SPI DT overlays in the boot log of u-boot

```
Applying kernel provided DT overlay sun50i-h6-spi-spidev1.dtbo
4191 bytes read in 6 ms (681.6 KiB/s)
Applying kernel provided DT fixup script (sun50i-h6-fixup.scr)
```

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

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

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

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

3) Do not short-circuit the mosi and miso 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@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF 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    | ................................
```

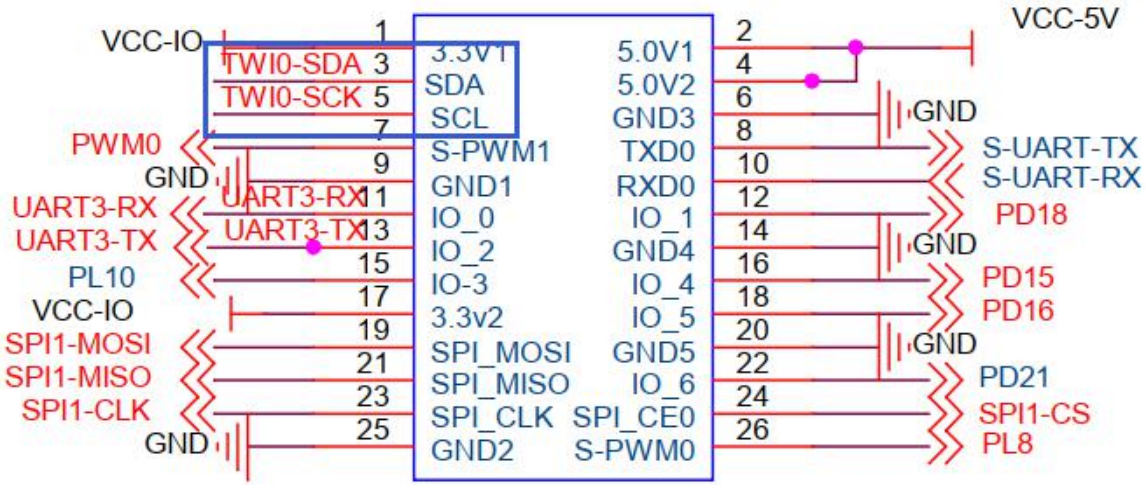
4) Then short-circuit the mosi (No. 19 pin in the 26pin interface) and miso (No. 21 pin in the 26pin interface) two pins of SPI1, and then run the output of spidev_test as follows, you can see the sending and receiving Same data

```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev1.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF 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. 27. 3. 26pin I2C test

1) The linux5.10 system turns off the i2c controller in 26pin by default in the dts. If you need to use i2c, you need to open the i2c configuration first. The linux4.9 system is open by default and no additional configuration is required. linux5. The opening method of 10 system i2c is as follows:

a. According to the schematic diagram of 26pin, the i2c available on the development board is i2c0



b. Then set overlays=i2c0 in /boot/orangepiEnv.txt to open the configuration of i2c0

```
overlays=i2c0
```

c. Then restart the system. When booting, you can see the configuration output of I2C DT overlays in the boot log of u-boot

```
Applying kernel provided DT overlay sun50i-h6-i2c0.dtbo
4191 bytes read in 5 ms (818.4 KiB/s)
Applying kernel provided DT fixup script (sun50i-h6-fixup.scr)
```

d. After the system is started, if there are more i2c device nodes under /dev, the configuration is correct

```
root@orangepi:~# ls /dev/i2c*
/dev/i2c-0  /dev/i2c-1  /dev/i2c-2  /dev/i2c-3
```

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

a) i2c0 in 26pin corresponds to /dev/i2c-0

```
root@orangepi3:~# ls /sys/class/i2c-adapter/i2c-* -lh
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-0 -> ../../devices/platform/soc/5002000.i2c/i2c-0
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-1 -> ../../devices/platform/soc/5002c00.i2c/i2c-1
lrwxrwxrwx 1 root root 0 Dec  7 12:56 /sys/class/i2c-adapter/i2c-2 -> ../../devices/platform/soc/7081400.i2c/i2c-2
lrwxrwxrwx 1 root root 0 Dec  7 12:57 /sys/class/i2c-adapter/i2c-3 -> ../../devices/platform/soc/6000000.hdmi/i2c-3
```

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

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

3) Then connect an i2c device to the i2c0 pin of the 26pin connector

	i2c0
Sda Pin	Corresponding to pin 3
Sck Pin	Corresponding to pin 5
Vcc Pin	Corresponding to pin 1
Gnd Pin	Corresponding to pin 6

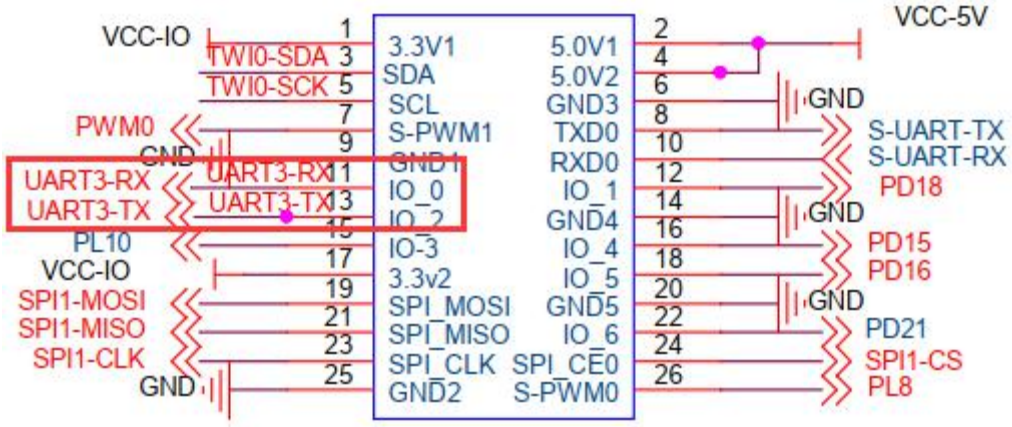
4) Then use the i2cdetect -y 0 command if the address of the connected i2c device can be detected, it means that i2c can be used normally

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

3. 27. 4. 26pin UART test

1) In linux5.10 system, the uart controller in 26pin is turned off by default in dts. If you need to use uart, you need to open the configuration of uart first. The linux4.9 system is enabled by default and no additional configuration is required. linux5. The opening method of 10 system uart is as follows:

a. According to the schematic diagram of 26pin, the uart available on the development board is uart3



b. Then set overlays=uart3 in /boot/orangepiEnv.txt to open the configuration of uart3

```
overlays=uart3
```

c. Then restart the system. When booting, you can see the configuration output of DT overlays related to UART in the boot log of u-boot

```
Applying kernel provided DT overlay sun50i-h6-uart3.dtbo
268 bytes read in 4 ms (65.4 KiB/s)
Applying kernel provided DT fixup script (sun50i-h6-fixup.scr)
```

d. After the system is started, you can see the information of ttyS3 under

/sys/class/tty, uart3 in 26pin corresponds to /dev/ttyS3

```
root@orangepi3:~# ls /sys/class/tty/ttyS* -lh
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS0 -> ../../devices/platform/soc/5000000.serial/tty/ttyS0
lrwxrwxrwx 1 root root 0 Dec  7 12:38 /sys/class/tty/ttyS1 -> ../../devices/platform/soc/5000400.serial/tty/ttyS1
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS2 -> ../../devices/platform/serial8250/tty/ttyS2
lrwxrwxrwx 1 root root 0 Dec  7 12:38 /sys/class/tty/ttyS3 -> ../../devices/platform/soc/5000c00.serial/tty/ttyS3
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS4 -> ../../devices/platform/serial8250/tty/ttyS4
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS5 -> ../../devices/platform/serial8250/tty/ttyS5
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS6 -> ../../devices/platform/serial8250/tty/ttyS6
lrwxrwxrwx 1 root root 0 Jan  1  1970 /sys/class/tty/ttyS7 -> ../../devices/platform/serial8250/tty/ttyS7
```

2) Then start to test the uart interface, first use the Dupont line to short-circuit the rx and tx of the uart3 interface to be tested

	uart3
Tx Pin	Corresponding to pin 13
Rx Pin	Corresponding to pin 11

3) Then modify the serial device node name opened by the serial test program serialTest in wiringOP to /dev/ttyS3

```
root@orangepi3-lts:~/wiringOP/examples# vim serialTest.c
```

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

  if ((fd = serialOpen ("/dev/ttyS3", 115200)) < 0)
  {
    fprintf (stderr, "Unable to open serial device: %s\n", strerror (errno)) ;
    return 1 ;
  }
```

4) Recompile the serial test program serialTest in wiringOP

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

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

```
root@orangepi:~/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. 28. 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 26pin interface of the Orange Pi development board through the DuPont cable, the wiring method is as follows

Pins of OLED module	Description	Development board 26pin interface corresponding pin
GND	Power ground	Pin 6
VCC	5V	Pin 2
SCL	I2C clock line	Pin 5
SDA	I2C data cable	Pin 3
RST	Connect to 3.3V	Pin 1
DC	Connect to GND	Pin 9
CS	Connect to GND	Pin 25

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

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y i2c-tools
root@orangepi:~# i2cdetect -y 0
```

```
root@orangepi3-lts:~# i2cdetect -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

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

```
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP
root@orangepi:~# cd wiringOP
root@orangepi:~/wiringOP# ./build clean && ./build
```

```
root@orangepi:~/wiringOP# cd examples
root@orangepi:~/wiringOP/examples# make oled_demo
root@orangepi:~/wiringOP/examples# ./oled_demo /dev/i2c-0
---------start--------
----------end---------
```

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

3.29. After running oled_demo, you can see the following output on the OLED screen

Note: This method is only applicable to linux4.9 kernel systems, and linux5.10 kernel systems are not adapted

3.29.1. 2.4 inch SPI LCD display

1) The link to the tested LCD display details page is as follows

http://www.lcdwiki.com/2.4inch_SPI_Module_ILI9341_SKU:MSP2402

2) The wiring method of the LCD display and the development board is as follows

TFT SPI module pins	Corresponding pins of development board 26pin	GPIO -- GPIO num
VCC	Pin 1	
GND	Pin 6	
CS	Pin 24	
RESET	Pin 12	PD18 -- 114
D/C	Pin 16	PD15 -- 111
SDI(MOSI)	Pin 19	

SCK	Pin 23	
LED	Pin 18	PD16 -- 112
SDO(MISO)	Pin 21	

3) After connecting the display to the development board, use the following command to load the fbtft_device kernel module

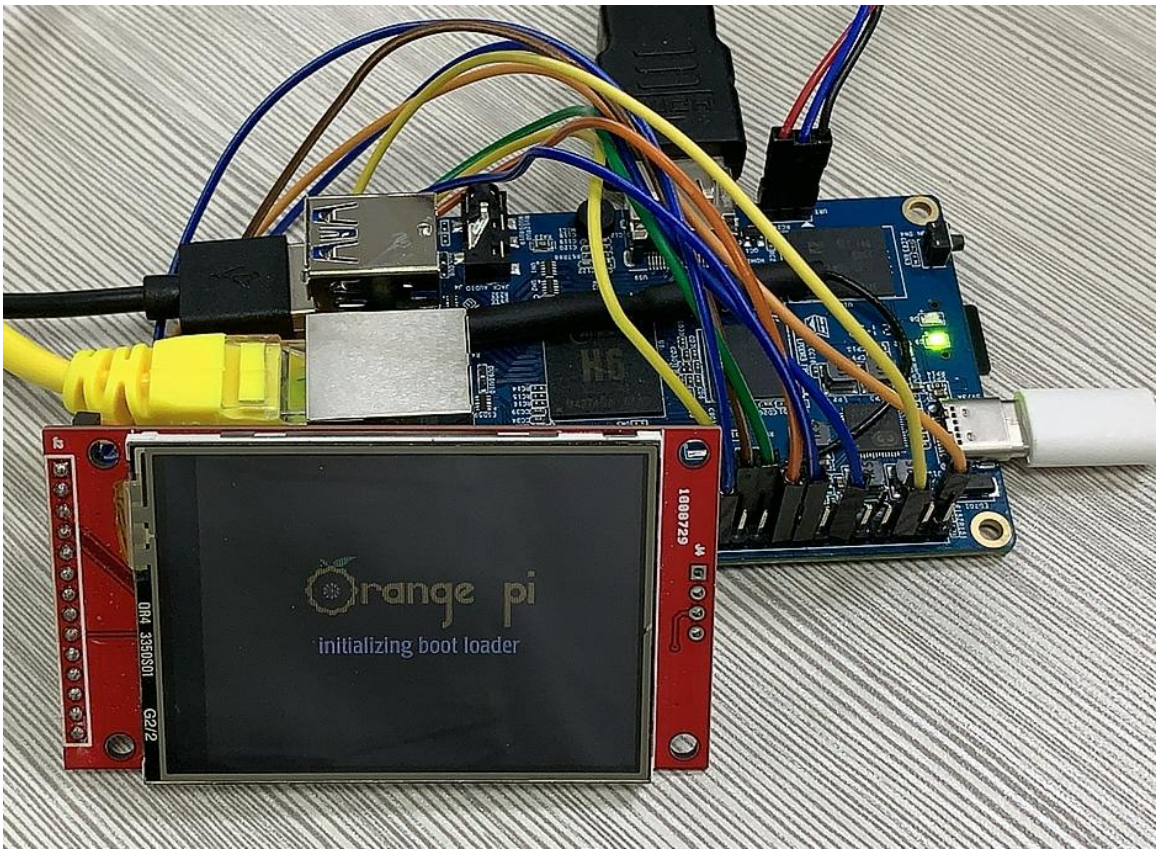
```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341    busnum=1 cs=0
gpios=reset:114,dc:111,led:112 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

4) When the fbtft_device kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD display is fb8

```
root@orangepi:~# dmesg | tail
[    191.788887] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
[    191.789343] spidev spi1.0: Deleting spi1.0
[    191.793283] fbtft_device: GPIOS used by 'fb_ili9341':
[    191.793301] fbtft_device: 'reset' = GPIO114
[    191.793313] fbtft_device: 'dc' = GPIO111
[    191.793324] fbtft_device: 'led' = GPIO112
[    191.793351] spi spi1.0: fb_ili9341 spi1.0 65000kHz 8 bits mode=0x00
[    191.807788] fb_ili9341: module is from the staging directory, the quality is unknown,
you have been warned.
[    192.083355] graphics fb8: fb_ili9341 frame buffer, 320x240, 150 KiB video memory,
64 KiB buffer memory, fps=20, spi1.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



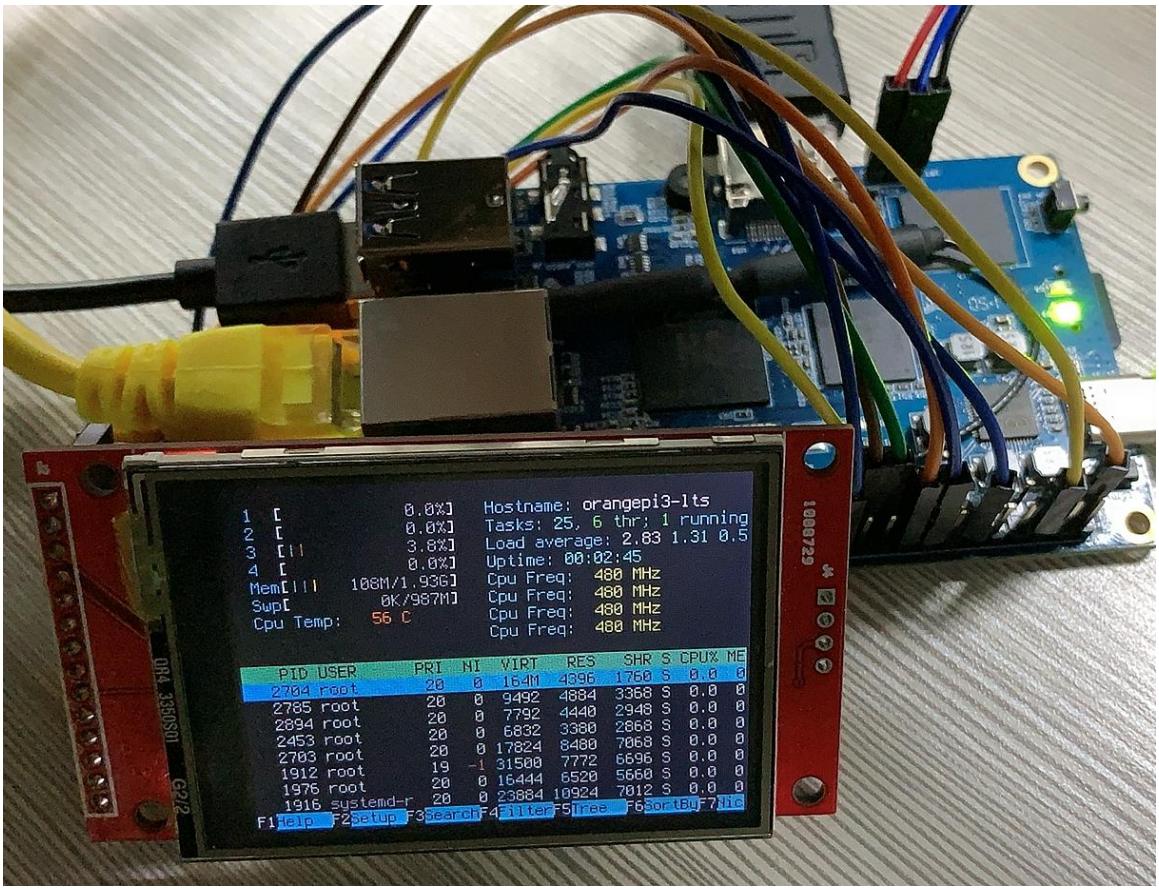
6) You can also map the output of tty1 to the fb device of the LCD display-fb8. After the mapping is completed, HDMI will no longer have image output

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

```
root@orangepi:~# con2fbmap 1 0
```

If you want to switch back to HDMI display, please use the following command

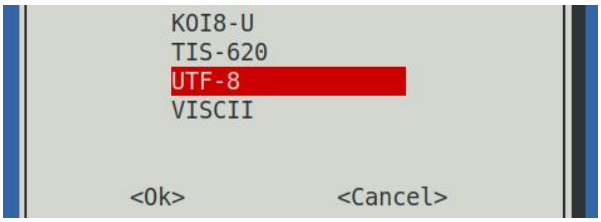


7) Because the default terminal font is too large, the display cannot display too much content, you can use the following method to reduce the terminal font

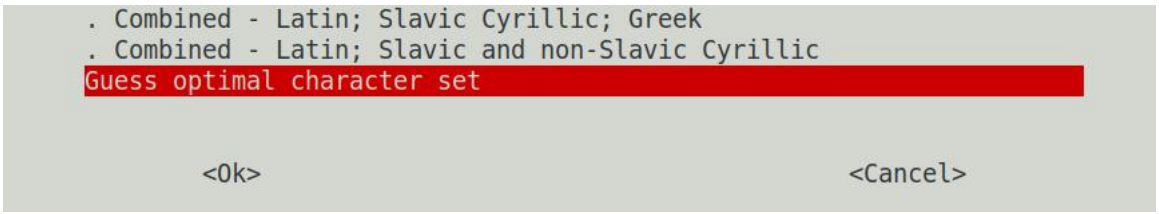
a. First run dpkg-reconfigure console-setup

```
root@orangepi:~# apt-get update
root@orangepi:~# apt-get install -y kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

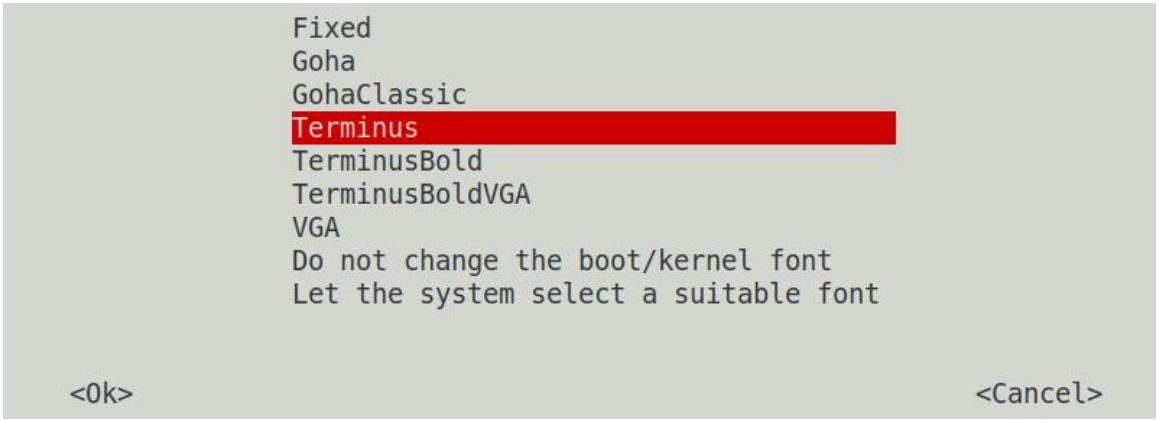
b. Terminal encoding select UTF-8



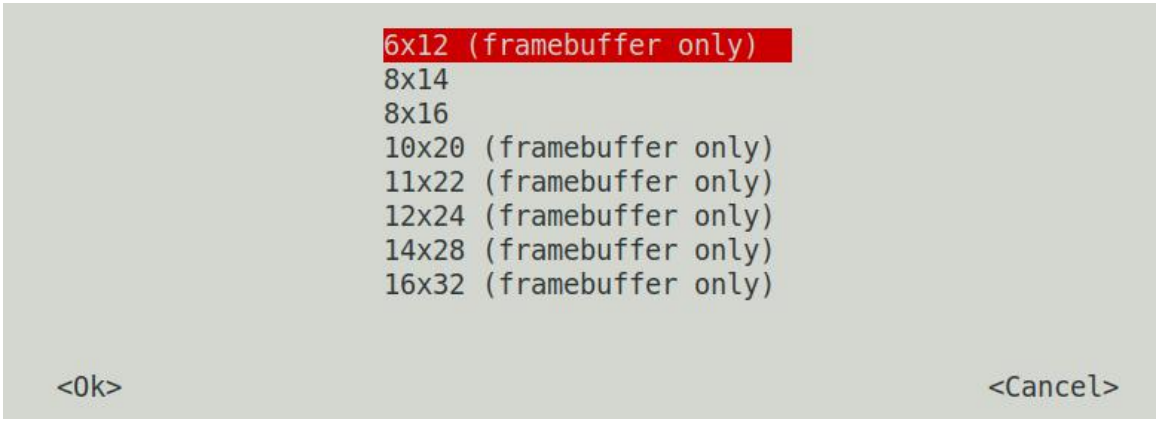
c. Then select Guess optimal character set



d. Then select Terminus



e. Finally select the font size as 6x12



f. After setting, you can see that the font on the LCD display becomes smaller

8) Set the method to automatically load the fbtft_device module at system startup

a. Create a new /etc/modules-load.d/fbtft.con configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new /etc/modprobe.d/fbtft.conf configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
```

```
options    fbtft_device custom name=fb_ili9341    busnum=1 cs=0
gpios=reset:114,dc:111,led:112 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to /boot/orangepiEnv.txt, and then restart the system to see the LCD display output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=cma=25M fbcon=map:8
```

3. 29. 2. 3.2 inch RPi SPI LCD display

1) The link to the tested LCD display details page is as follows

```
http://www.lcdwiki.com/3.2inch_RPi_Display
```

2) The wiring method of the LCD display and the development board is as follows

3) After connecting the LCD display to the development board, use the following command to load the fbtft_device kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341    busnum=1 cs=0
gpios=reset:119,dc:362 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

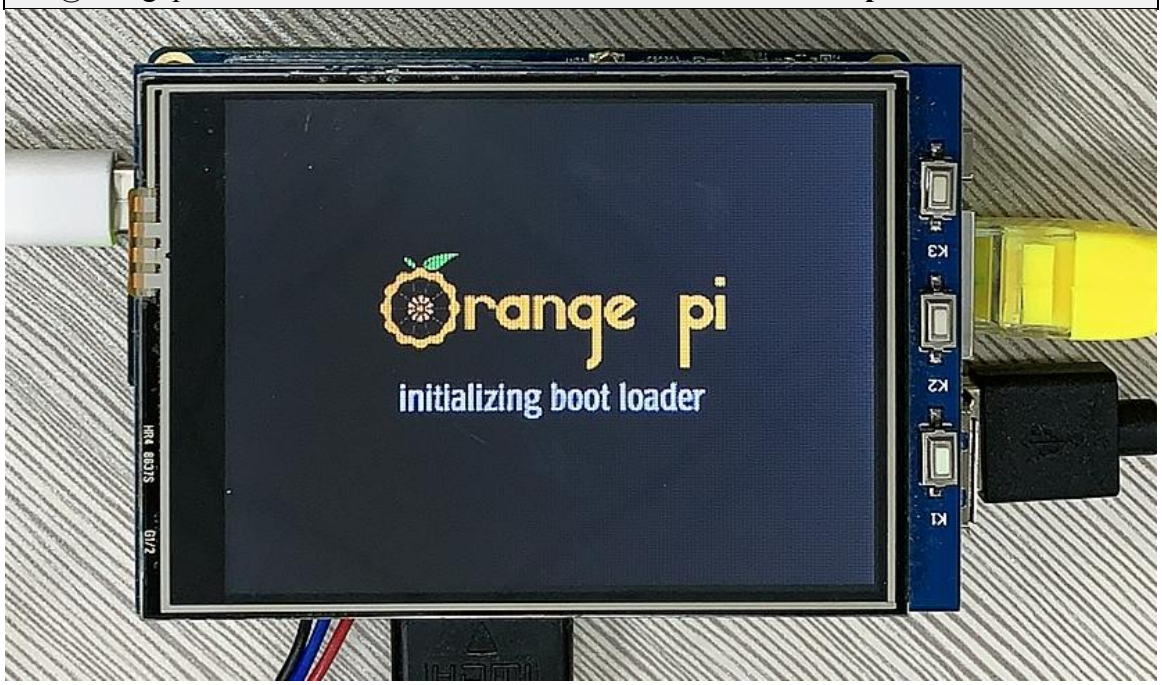
4) When the fbtft_device kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD screen is fb8

```
root@orangepi:~# dmesg | tail
[    97.884472] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
```

```
[      97.884933] spidev spi1.0: Deleting spi1.0
[      97.888354] fbtft_device: GPIOS used by 'fb_ili9341':
[      97.888373] fbtft_device: 'reset' = GPIO119
[      97.888385] fbtft_device: 'dc' = GPIO362
[      97.889430] spi spi1.0: fb_ili9341 spi1.0 65000kHz 8 bits mode=0x00
[      97.907191] fb_ili9341: module is from the staging directory, the quality is unknown,
you have been warned.
[      98.181286] Console: switching to colour frame buffer device 40x30
[      98.181938] graphics fb8: fb_ili9341 frame buffer, 320x240, 150 KiB video memory,
64 KiB buffer memory, fps=20, spi1.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD screen

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



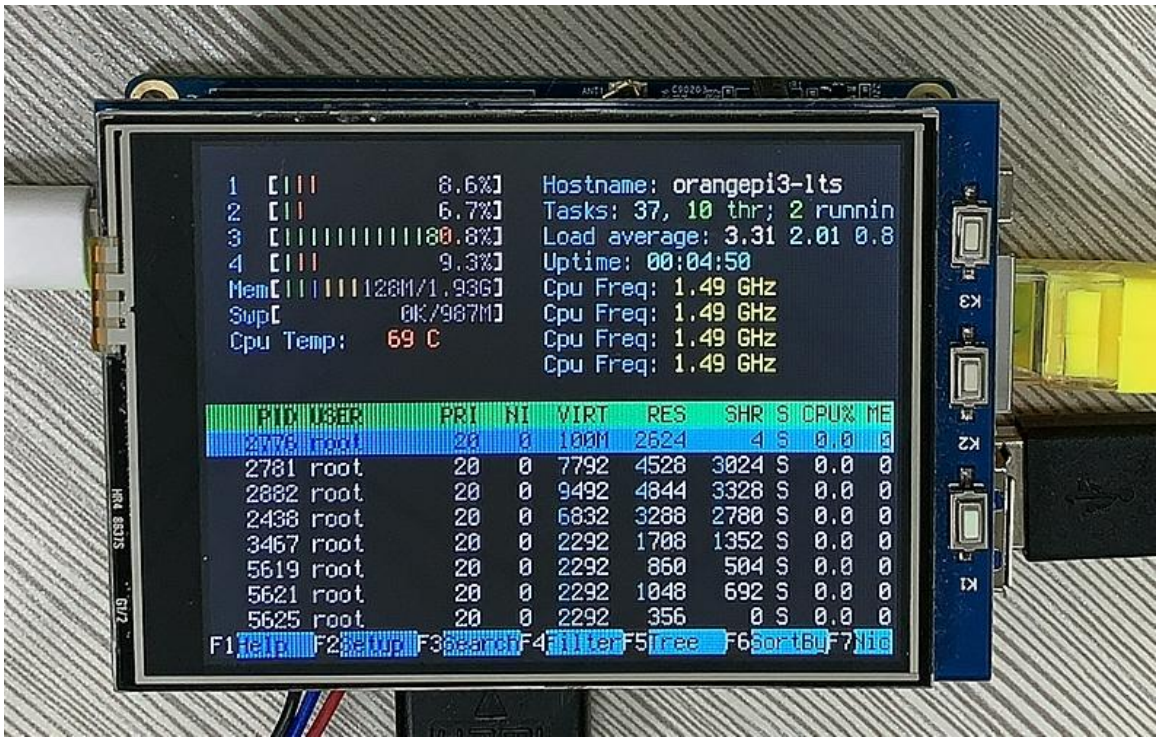
6) You can also map the output of tty1 to the fb device of the LCD screen-fb8. After the mapping is completed, HDMI will no longer have image output

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command

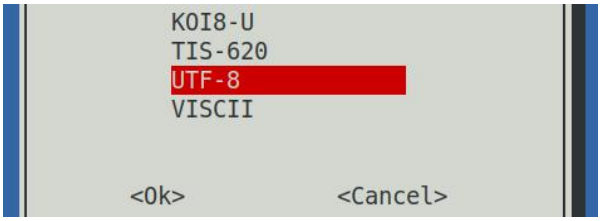


7) Because the default terminal font is too large, the screen cannot display too much content, you can use the following method to reduce the terminal font

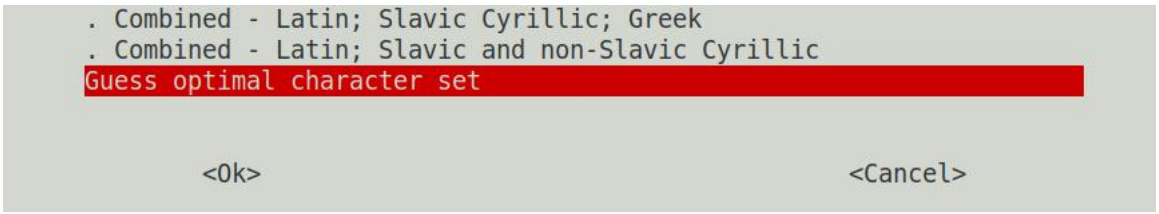
a. First run dpkg-reconfigure console-setup

```
root@orangepi:~# apt-get update
root@orangepi:~# apt-get install -y kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

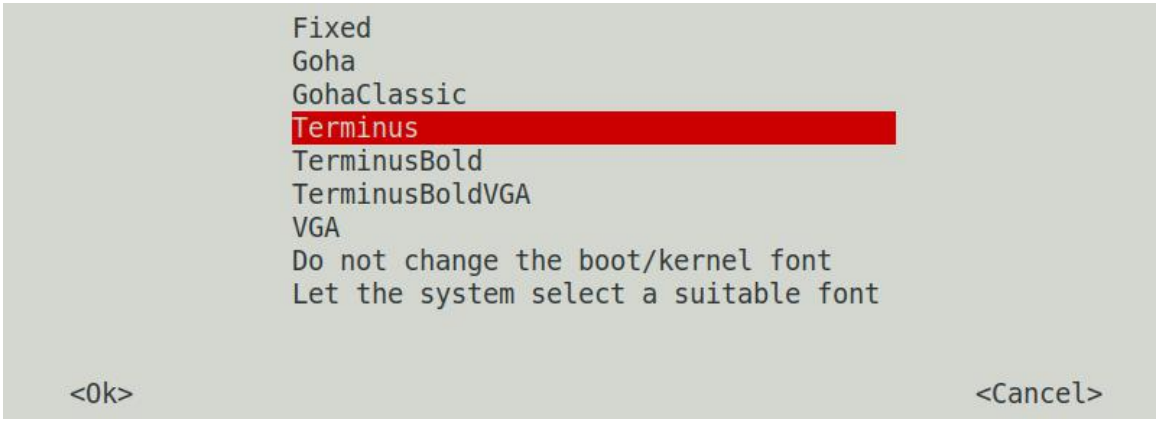
b. Terminal encoding select UTF-8



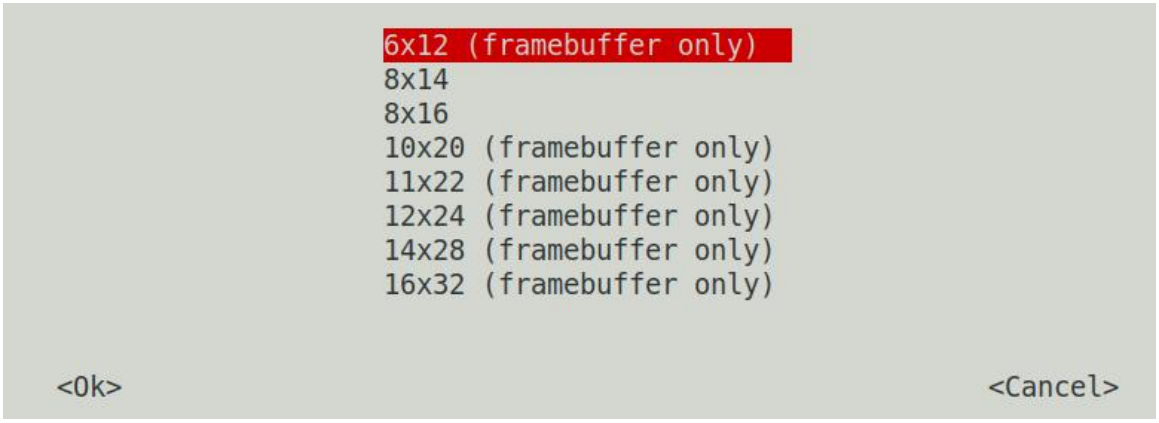
c. Then select Guess optimal character set



d. Then select Terminus



e. Finally select the font size as 6x12



f. After setting, you can see that the font on the LCD screen becomes smaller

8) Set the method to automatically load the fbtft_device module at system startup

a. Create a new /etc/modules-load.d/fbtft.con configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new /etc/modprobe.d/fbtft.conf configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9341    busnum=1 cs=0    gpios=reset:119,dc:362
```

```
rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD screen after booting, please add the following configuration in /boot/orangepiEnv.txt, and then restart the system to see the LCD screen output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=cma=25M fbcon=map:8
```

3. 29. 3. 3.5 inch SPI LCD display

1) The link to the tested LCD display details page is as follows

```
http://www.lcdwiki.com/3.5inch_SPI_Module_ILI9488_SKU:MSP3520
```

2) The wiring method of the LCD display and the development board is as follows

TFT SPI module pins	Corresponding pins of development board 26pin	GPIO -- GPIO num
VCC	1	
GND	6	
CS	24	
RESET	12	PD18 -- 114
DC/RS	16	PD15 -- 111
SDI(MOSI)	19	
SCK	23	
LED	18	PD16 -- 112
SDO(MISO)	21 Pin	

3) After connecting the display to the development board, use the following command to load the fbtft_device kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9488    busnum=1 cs=0
gpios=reset:114,dc:111,led:112 rotate=270 speed=65000000 bgr=1 txbuflen=65536
```

4) When the fbtft_device kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD display is fb8

```
root@orangepi:~# dmesg | tail
[    36.897250] spidev spi1.0: dh2228fv spi1.0 1200kHz 8 bits mode=0x00
[    36.897709] spidev spi1.0: Deleting spi1.0
[    36.900560] fbtft_device: GPIOS used by 'fb_ili9488':
[    36.900579] fbtft_device: 'reset' = GPIO114
[    36.900591] fbtft_device: 'dc' = GPIO111
[    36.900602] fbtft_device: 'led' = GPIO112
[    36.900629] spi spi1.0: fb_ili9488 spi1.0 65000kHz 8 bits mode=0x00
[    36.918499] fb_ili9488: module is from the staging directory, the quality is unknown,
you have been warned.
[    37.263271] graphics fb8: fb_ili9488 frame buffer, 480x320, 300 KiB video memory,
64 KiB buffer memory, fps=60, spi1.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install -y fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```

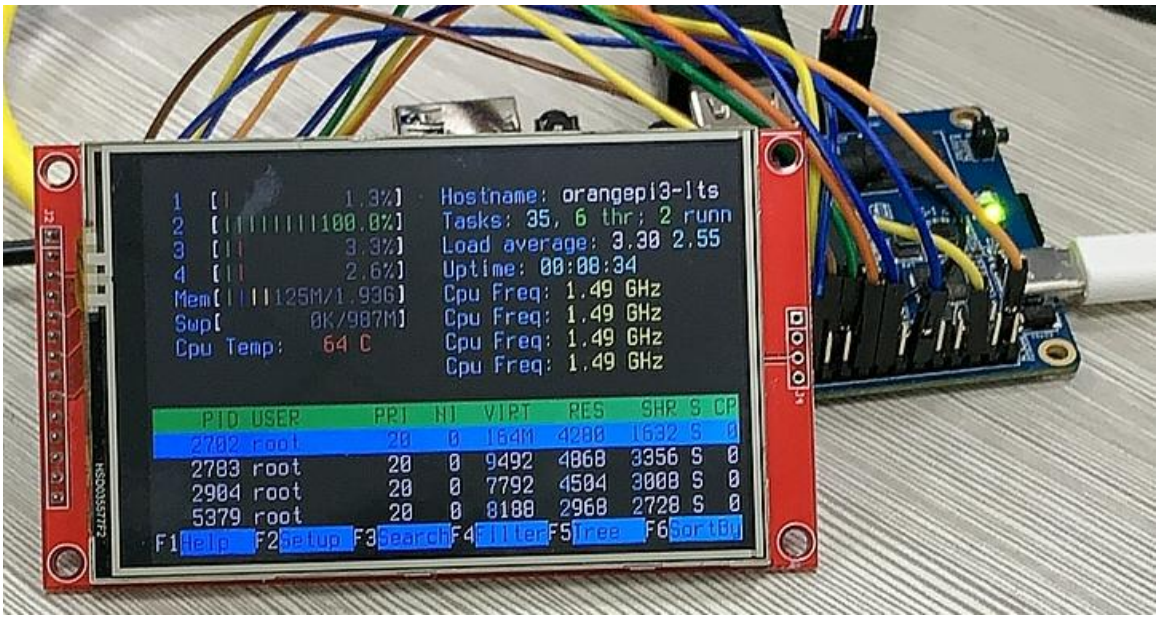
6) You can also map the output of tty1 to the fb device of the LCD display-fb8. After the mapping is completed, the LCD screen will display the output of the terminal, and HDMI will no longer have image output.

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command



7) Set the method to automatically load the fbtft_device module at system startup

a. Create a new /etc/modules-load.d/fbtft.con configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new /etc/modprobe.d/fbtft.conf configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9488    busnum=1 cs=0
gpios=reset:114,dc:111,led:112 rotate=270 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

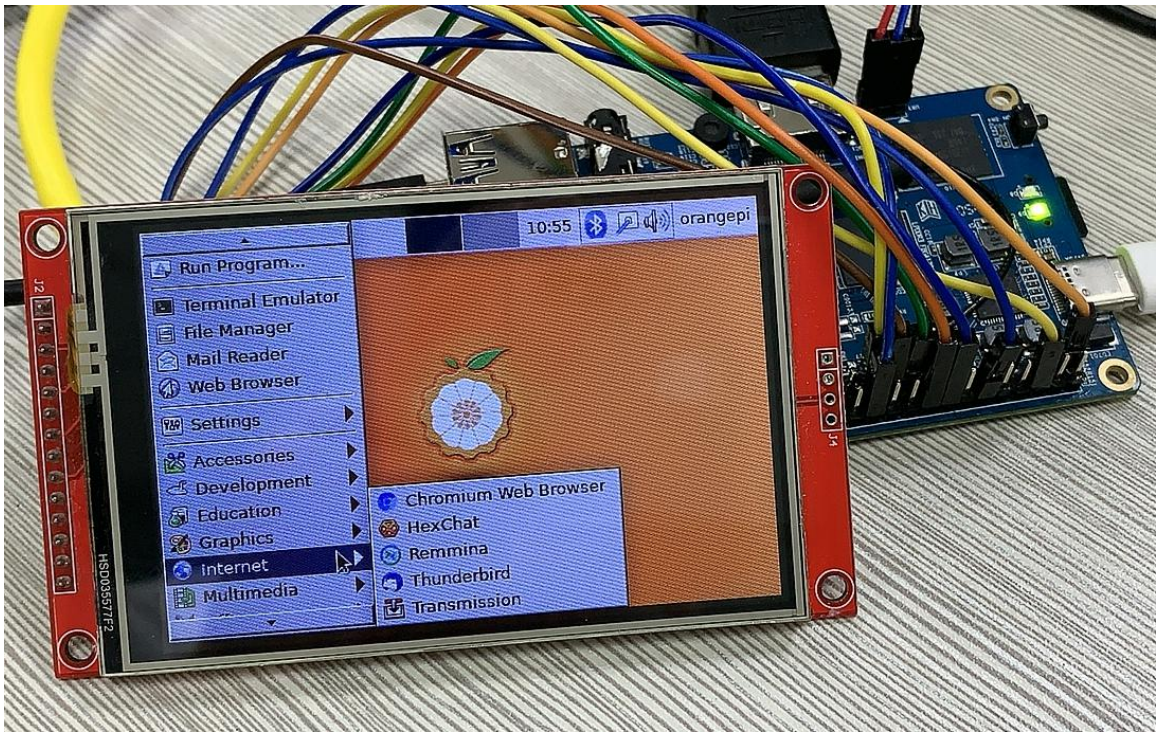
8) If you want the linux system to automatically map the console to the LCD display

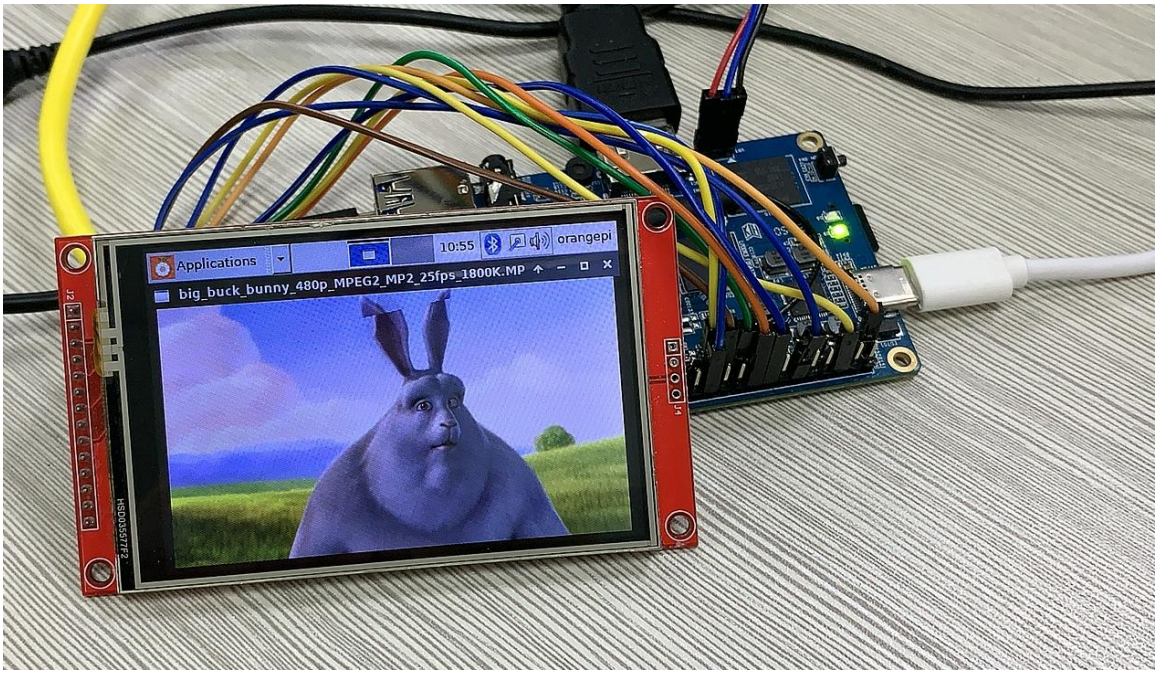
after booting, please add the following configuration to /boot/orangepiEnv.txt, and then restart the system to see the LCD display output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=cma=25M fbcon=map:8
```

9) If you need to display the desktop version of the system to the LCD screen, you can execute the following command, after a few seconds, the LCD screen will be able to see the desktop of the linux system

```
root@orangepi:~# FRAMEBUFFER=/dev/fb8 startx
```





10) If you want to automatically display the desktop to the LCD display after the Linux system is started, please add the following configuration file to the Linux system, and then restart the system to see the LCD display output

```
root@orangepi:~# cat /usr/share/X11/xorg.conf.d/99-fbdev.conf
Section "Device"
    Identifier "myfb"
    Driver "fbdev"
    Option "fbdev" "/dev/fb8"
EndSection
```

3. 30. The method of outputting the kernel print information to the 26pin serial port

The kernel console outputs to ttyS0 by default, which is the 3pin debug serial port on the development board. We can also set the kernel console output to be redirected to UART3 in the 26pin interface, the method is as follows:

1) The linux5.10 system needs to open the configuration of uart3 first. For the detailed configuration method, please refer to the 26pin UART test. The linux4.9 system is turned on by default and no additional configuration is required.

2) Then modify the console=ttyS0 in /boot/boot.cmd to console=ttyS3

```
root@orangepi:~# vim /boot/boot.cmd
```

```
if test "${console}" = "display" || test "${console}" = "both"; then setenv consoleargs "console=ttyS3,115200 console=tty1"; fi
if test "${console}" = "serial"; then setenv consoleargs "console=ttyS3,115200"; fi
if test "${bootlogo}" = "true"; then setenv consoleargs "bootsplash.bootfile=bootsplash.orangepi ${consoleargs}"; fi
```

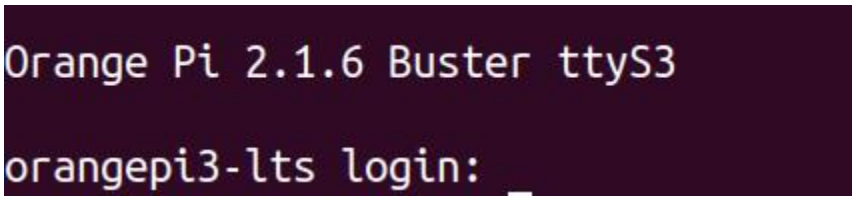
3) Then recompile /boot/boot.cmd to /boot/boot.scr (operate in the linux system of the development board)

```
root@orangepi:~# mkimage -C none -A arm -T script -d /boot/boot.cmd /boot/boot.scr
Image Name:
Created:          Tue Dec  8 02:35:43 2020
Image Type:       ARM Linux Script (uncompressed)
Data Size:        2448 Bytes = 2.39 KiB = 0.00 MiB
Load Address: 00000000
Entry Point:      00000000
Contents:
   Image 0: 2440 Bytes = 2.38 KiB = 0.00 MiB
```

4) Then connect the USB to TTL module to the UART3 pin of the 26pin interface through the Dupont line

a. Connect the GND of the USB to TTL module to the GND of the 26pin interface of the development board
b. Connect the RX of the USB to TTL module to the TX of the development board UART3
c. The TX of the USB to TTL module is connected to the RX of the development board UART3

5) Then restart the development board, you can see that the kernel console outputs to ttyS3 by default. Note that the output log of u-boot is still output to ttyS0, not ttyS3



3. 31. Hardware watchdog test

1) Download the code of wiringOP

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

2) Compile the watchdog test program

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

3) Run the watchdog test program

a. The second parameter 10 represents the counting time of the watchdog. If 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@orangepi:~/wiringOP/examples# ./watchdog 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
put_usr return,if 0,success:0
keep alive
keep alive
```

```
keep alive
```

3. 32. View the chipid of the H6 chip

1) The command to view the chipid of the h6 chip is as follows, the chipid of each chip is different, so you can use chipid to distinguish multiple development boards

```
root@orangepi:~# cat /sys/class/sunxi_info/sys_info | grep "chipid"
sunxi_chipid            : 38641f0f0141410900004c0000000000
```

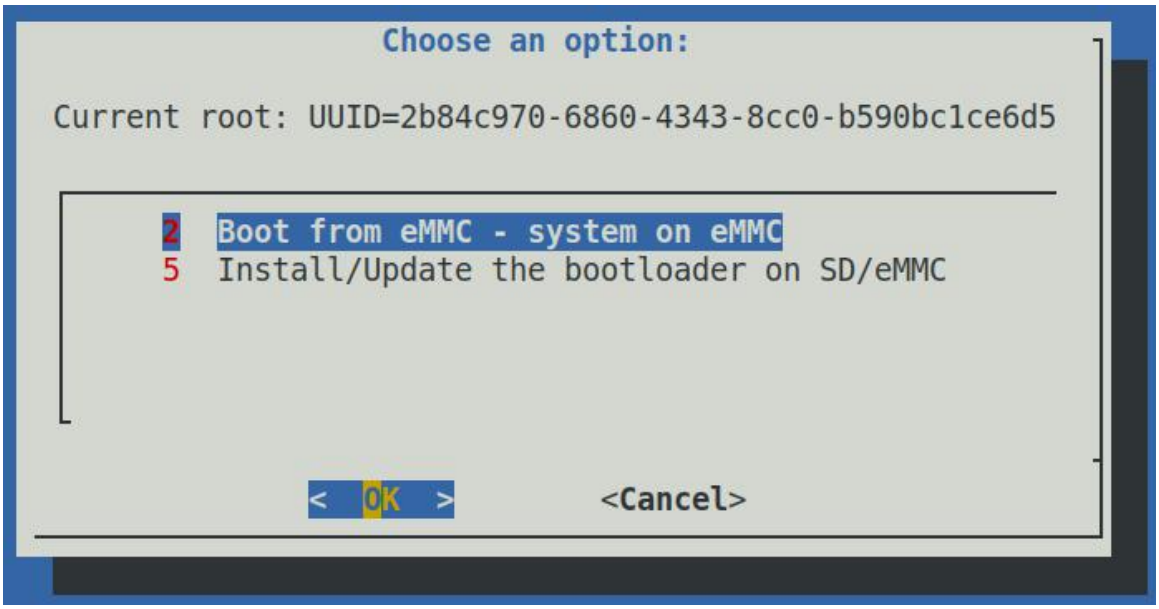
3. 33. Method of flashing linux image to eMMC

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

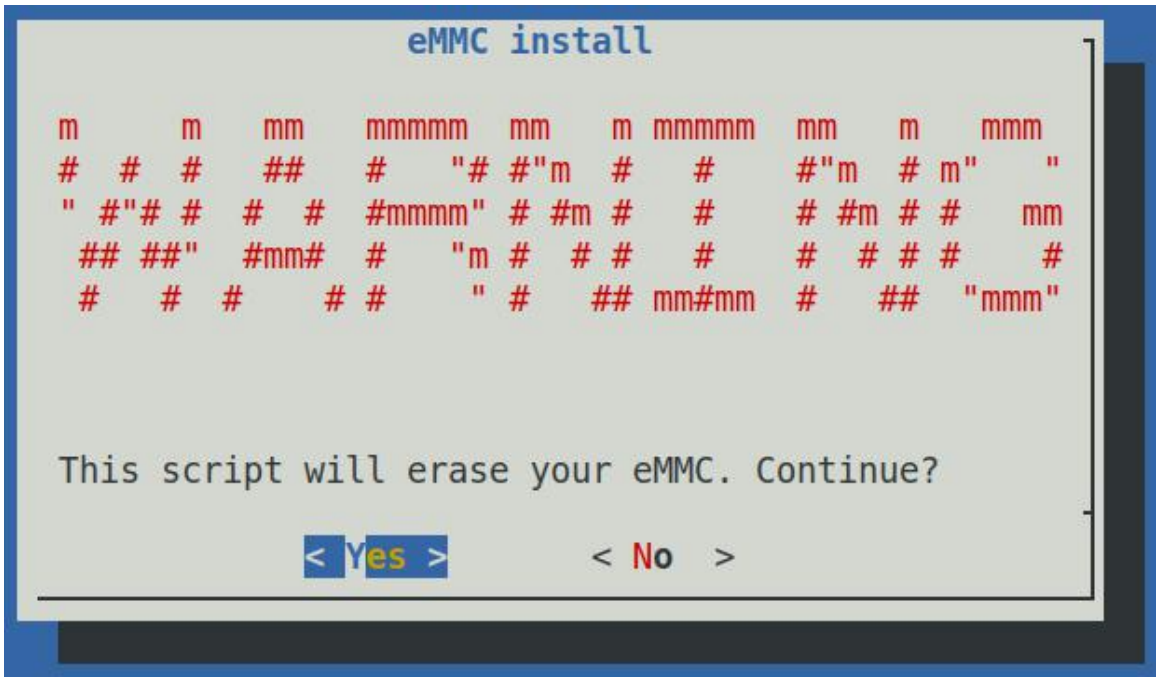
2) Then run the nand-sata-install script

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

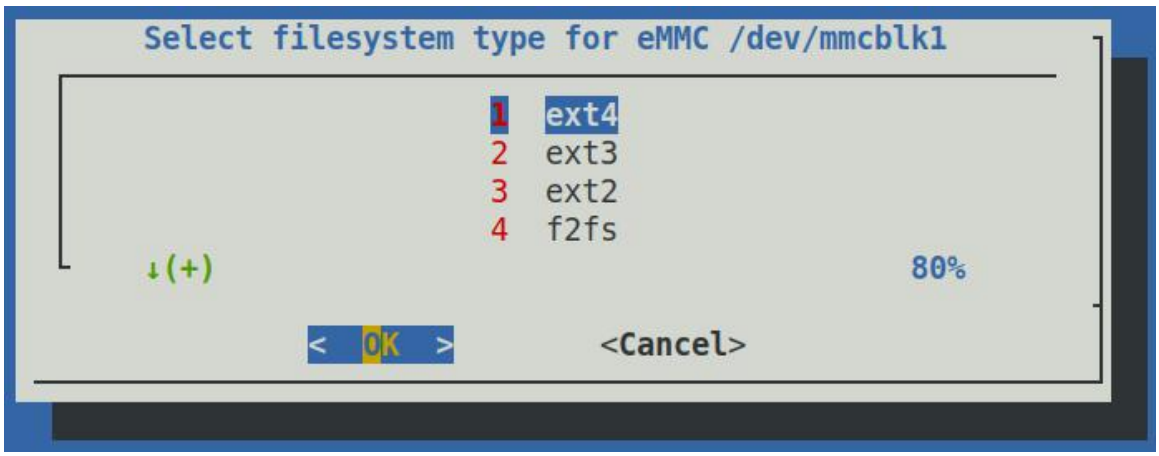
3) Then select 2 Boot from eMMC-system on eMMC

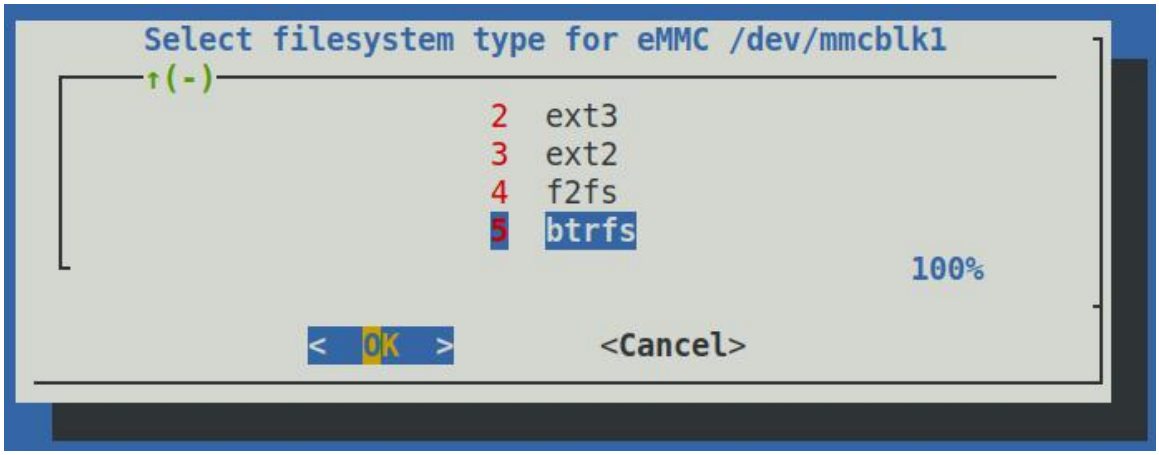


4) Then a warning will pop up, the script will erase all data on the eMMC, select <Yes> to continue

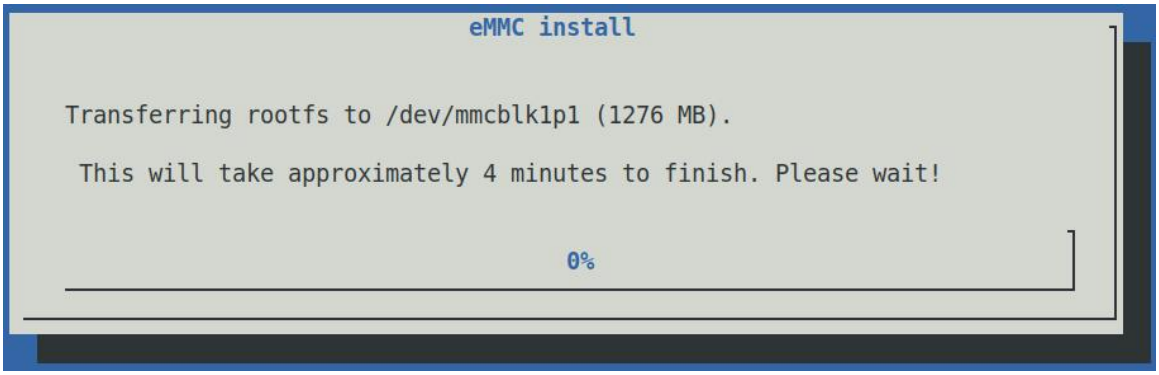


5) Then you will be prompted to select the type of file system, supporting five file systems ext2/3/4, f2fs and btrfs

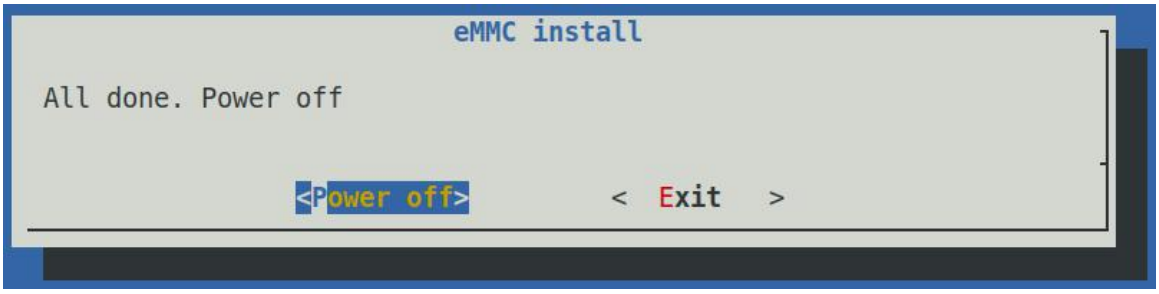




6) Then it will start to format the eMMC, and after the eMMC is formatted, it will start to burn the linux image to the eMMC



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 be started

3. 34. Boot and shutdown method

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 shut down

3) After shutting down, press and hold the power button on the development board for 2 seconds to boot

4) The command to restart the linux system is

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

4. Linux SDK instructions

4.1. Compile system requirements

1) The Linux SDK, orangepi-build, only supports running on a computer with Ubuntu 18.04 installed, so before downloading orangepi-build, please make sure that the Ubuntu version installed on your computer is Ubuntu 18.04. The command to view the Ubuntu version installed on the computer is as follows. If the Release field does not display 18.04, it means that the current Ubuntu version does not meet the requirements. Please change the system before proceeding with the following operations

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.5 LTS
Release:        18.04
Codename:       bionic
test@test:~$
```

2) If the computer is installed with a Windows system and does not have a computer with Ubuntu 18.04 installed, you can consider using VirtualBox or VMware to install an Ubuntu 18.04 virtual machine in the Windows system. But please be careful not to compile orangepi-build on the WSL virtual machine, because orangepi-build has not been tested in the WSL virtual machine, so there is no guarantee that orangepi-build can be used normally in WSL, and please do not use the Linux system of the development board Use orangepi-build in

4.2. Get the source code of linux sdk

4.2.1. Download orangepi-build from github

1) The linux sdk actually refers to the orangepi-build set of codes, and multiple versions of linux images can be compiled using orangepi-build. First download the code of orangepi-build, currently the H6 series development boards already support the legacy branch and the current branch

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

To download the code of orangepi-build through the git clone command, you do not need to enter the user name 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 The name and password are generally entered incorrectly in the address of the orangepi-build warehouse behind git clone. Please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account.

2) The legacy branch generally uses the BSP version of u-boot and kernel, and the current branch generally uses the u-boot and kernel close to the mainline version. The u-boot and linux kernel currently used by the Orange Pi 3 LTS development board are as follows

Branch	u-boot Version	Linux kernel Version
legacy	u-boot 2014.07	linux4.9
current	u-boot 2020.04	linux5.10

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

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

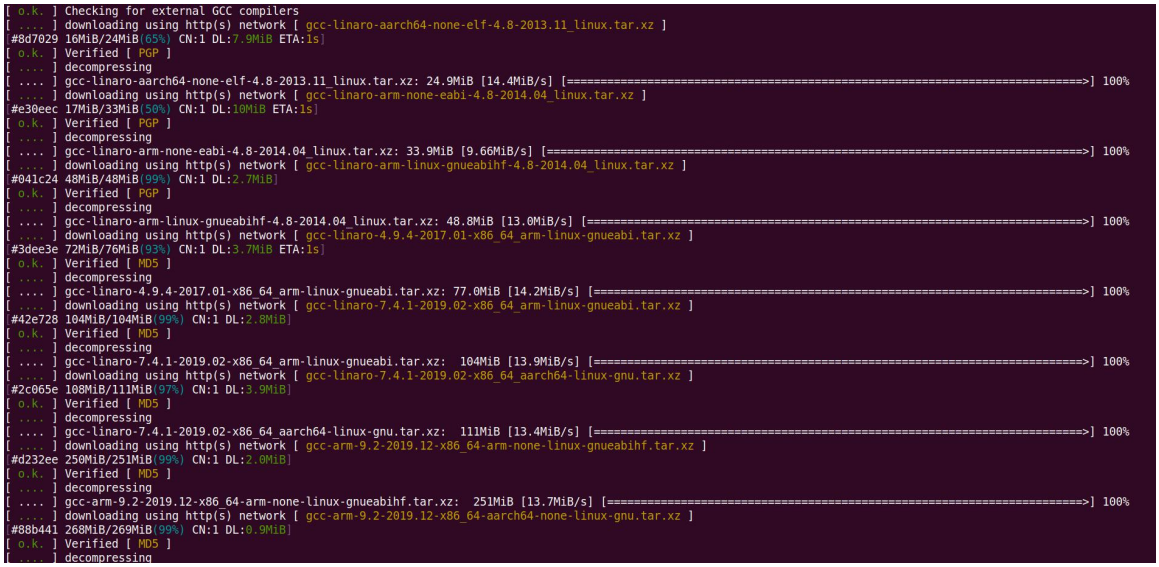
```
test@test:~/orangepi-build$ ls
build.sh  external  LICENSE  README.md  scripts
```

If you downloaded the code of orangepi-build from github, after downloading, you may find that orangepi-build does not contain the source code of u-boot and linux kernel, nor does it require cross-compilation to compile u-boot and linux kernel. Toolchain, this is normal, because these things are stored in other separate github repositories or some servers (the addresses will be detailed below).

Orangepi-build will specify the addresses of u-boot, linux kernel and cross-compilation tool chain in the script and configuration file. When orangepi-build is running, when it finds that there are no such things locally, it will automatically go to the corresponding place to download it.

4. 2. 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 H6 linux kernel source code is

a. linux4.9

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

b. linux5.10

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

5) The cross-compilation tool chain used to compile the H6 u-boot source code is

a. u-boot 2014.07

```
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
```

b. u-boot 2020.04

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

4. 2. 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 linux kernel and u-boot source code are stored in a separate git repository

a. The git repository where the linux kernel source code is stored is as follows, where sun50iw6 is the code name of the H6 SOC chip

a) linux4.9

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-4.9-sun50iw6
```

b) linux5.10

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10
```

b. The git repository where u-boot source code is stored is as follows, where sun50iw6 is the code name of the H6 SOC chip

a) u-boot 2014.07

```
https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2014.07-sun50iw6-linux4.9
```

b) u-boot 2020.04

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

If you are unfamiliar with orangepi-build and do not know the detailed process of compiling linux kernel and u-boot, please do not download and use the above linux kernel and u-boot source code to compile, because of the compilation script and configuration of orangepi-build The file will make some adjustments and optimizations for u-boot and linux. If you don't use orangepi-build to compile u-boot and linux, you may encounter problems of compilation failure or failure to start.

2) When orangepi-build runs for the first time, it will download the cross-compilation 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. **build.sh:** Compile the startup script
b. **external:** Contains the configuration files needed to compile the image, specific function scripts, and the source code of some programs. The rootfs compressed package cached during the compilation of the image is also stored in external
c. **kernel:** Stores the source code of the Linux kernel. The folder named orange-pi-4.9-sun50iw6 stores the kernel source code of the legacy branch of the H6 development board. The folder named orange-pi-5.10 stores the H6. The kernel source code of the current branch of the development board (If you only compile the linux image of the legacy branch, you can only see the kernel source code of the legacy branch; if you only compile the linux image of the current branch, you can only see the kernel of the current branch Source code), please do not manually modify the name of the folder of the kernel source code. If the build system is modified, the kernel source code will be downloaded again when the system is running.
d. **LICENSE:** GPL 2 license file
e. **README.md:** orangepi-build instruction file
f. **output:** store files such as u-boot, linux and other deb packages generated by compilation, compilation logs, and images generated by compilation
g. **scripts:** general scripts for compiling linux images
h. **toolchains:** store cross-compilation toolchains
i. **u-boot:** Store the source code of u-boot, the folder named v2014.07-sun50iw6-linux4.9 inside stores the u-boot source code of the legacy branch of the H6 development board, and the folder named v2020.04 is stored inside This is the u-boot source code of the current branch of the H6

development board (if you only compile the linux image of the legacy branch, you can only see the u-boot source code of the legacy branch; if you only compile the linux image of the current branch, then only You can see the u-boot source code of the current branch). Please do not modify the name of the u-boot source code folder manually. If the build system is modified, the u-boot source code will be re-downloaded when the system is running.

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

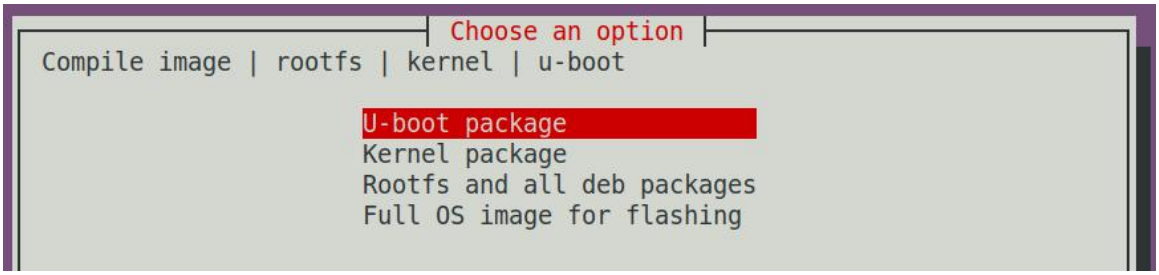
```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts
toolchains  u-boot  userpatches
```

4. 3. Compile u-boot

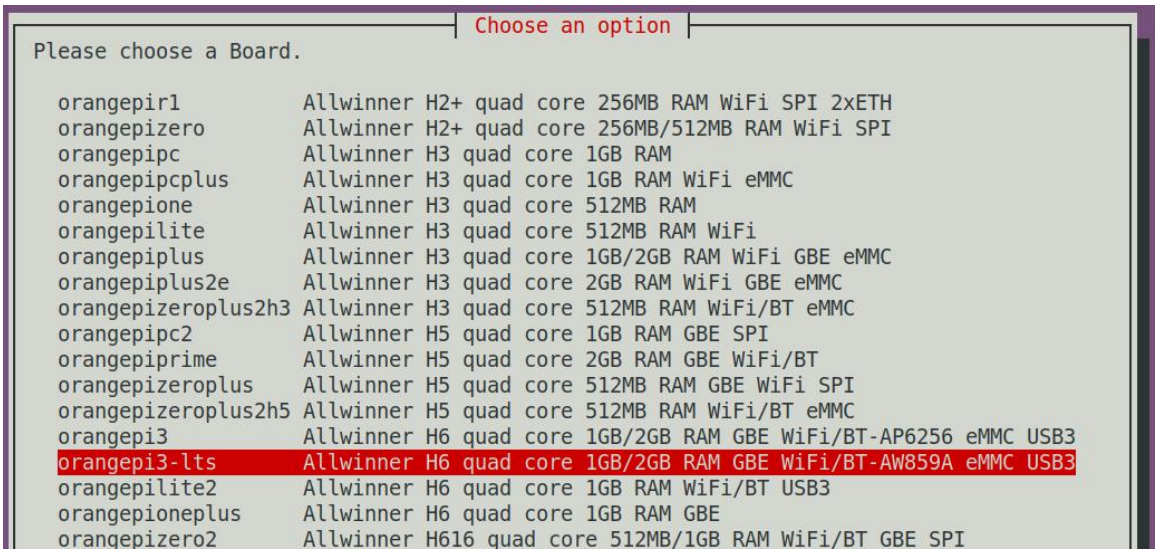
1) Run the build.sh script, remember to add sudo permissions

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

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



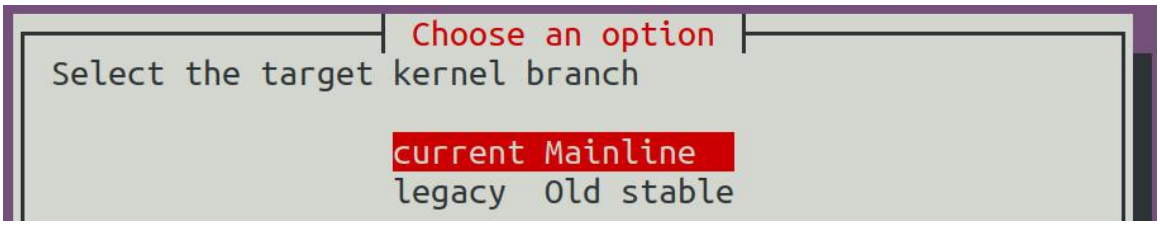
3) Then select the model of the development board



4) Then select the branch

a. current will compile u-boot v2020.04

b. legacy will compile u-boot v2014.07



5) Then it will start to compile u-boot, some of the information prompted during compilation are explained as follows (take the legacy branch as an example)

a. u-boot source version

[o.k.] Compiling u-boot [**v2014.07**]

b. The version of the cross-compilation toolchain

[o.k.] Compiler version [**arm-linux-gnueabi-gcc 4.9.4**]

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

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

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

[o.k.] File name [**linux-u-boot-legacy-orangepi3-lts_2.1.6_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 directly start compiling u-boot

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh    BOARD=orangepi3-lts
BRANCH=legacy BUILD_OPT=u-boot KERNEL_CONFIGURE=no ]
```

6) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-legacy-orangepi3-lts_2.1.6_arm64.deb
```

7) 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-orangepi3-lts_2.1.6_arm64.deb .      (Note that the command has
  a“.”)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-current-orangepi3-lts_2.1.6_armhf.deb    usr
```

b. The decompressed file is as follows

```
test@test:~orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-legacy-orangepi3-lts_2.1.6_arm64
    │   ├── boot0_sdcard.fex            //boot0 binary file
    │   └── boot_package.fex            //u-boot binary file
    └── u-boot
        ├── LICENSE
        ├── orangepi3-lts-u-boot.dts
        └── platform_install.sh

3 directories, 5 files
```

8) When the orangepi-bulid compilation system compiles the u-boot source code, it will first synchronize the u-boot source code with the u-boot source code 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"
```

9) 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-orangepi3-lts_2.1.6_arm64.deb root@192.168.1.xxx:/root
```

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

```
root@orangepi:~# apt purge -y linux-u-boot-orangepi3-lts-legacy
```

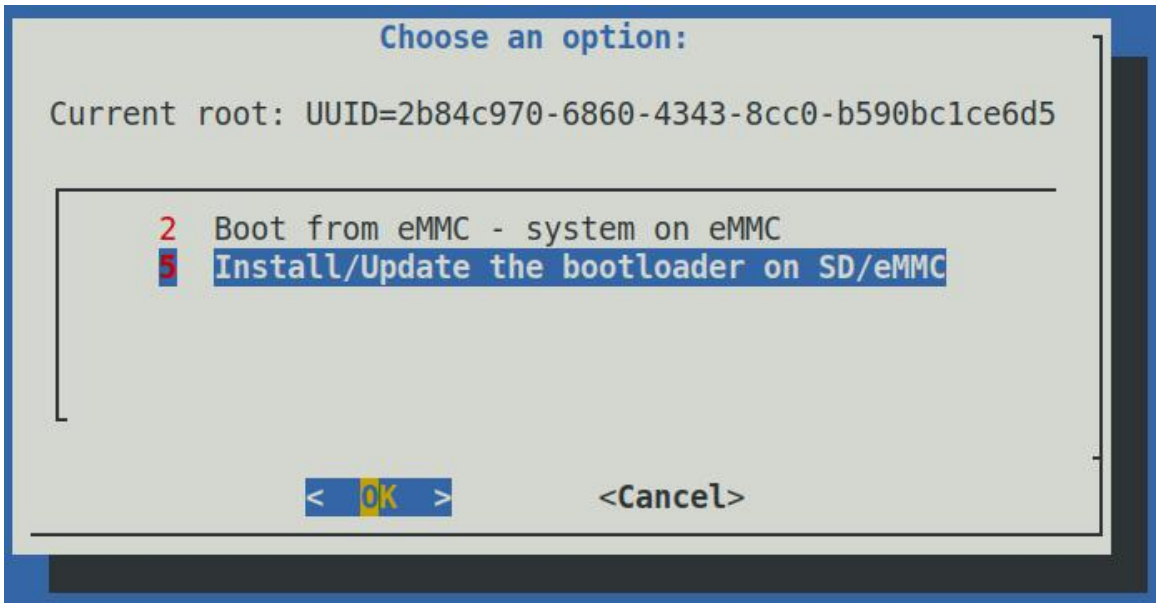
c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-legacy-orangepi3-lts_2.1.6_arm64.deb
```

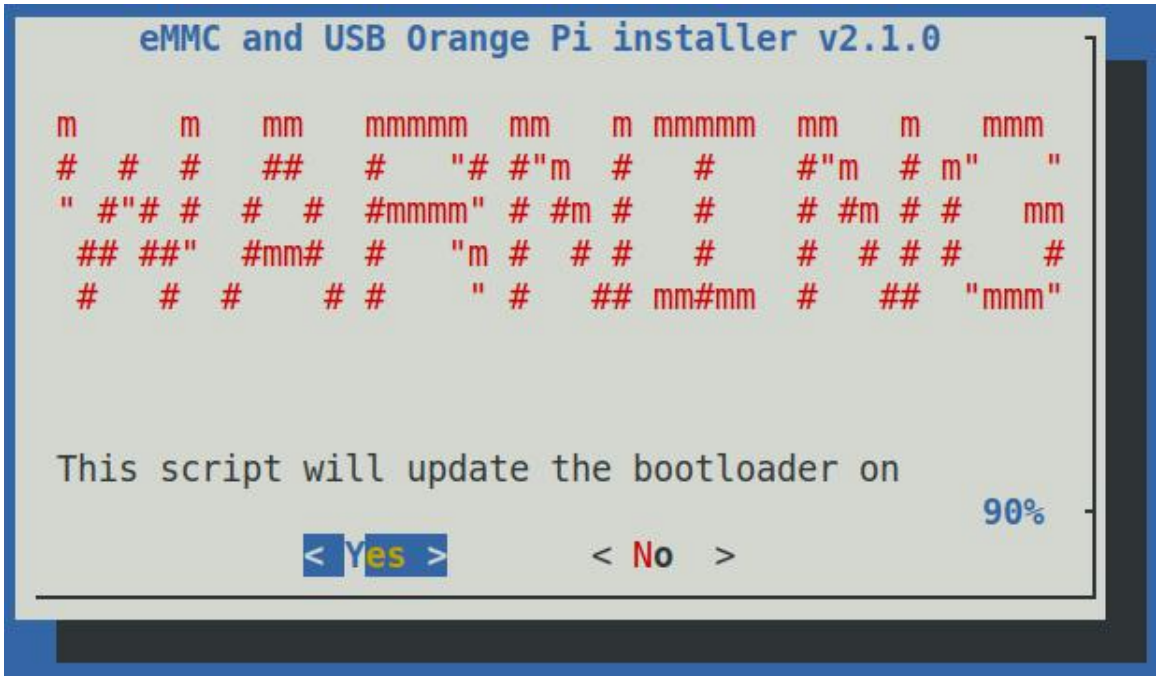
d. Then run the nand-sata-install script

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

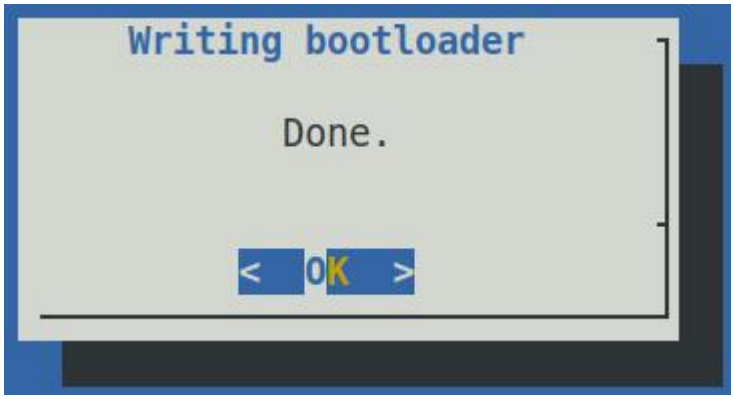
e. Then choose **5 Install/Update the bootloader on SD/eMMC**



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



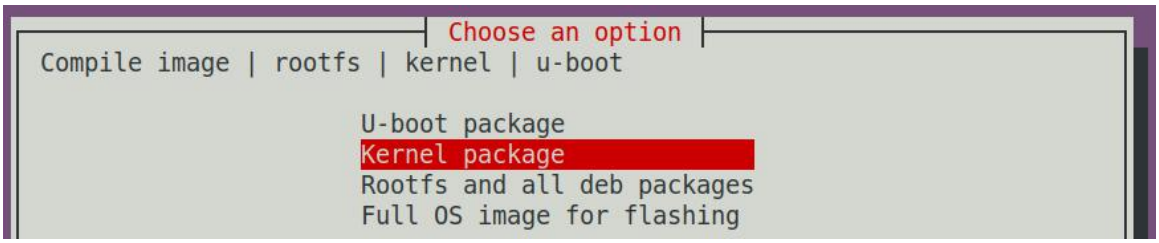
h. Then you can restart the development board to test whether the u-boot modification takes effect

4. 4. Compile the Linux kernel

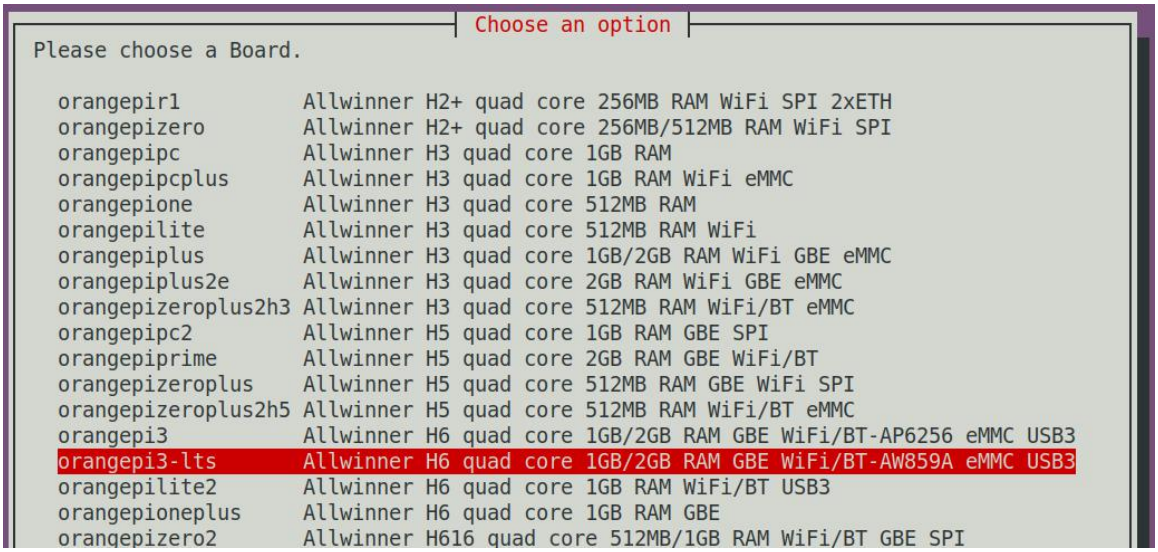
1) Run the build.sh script, remember to add sudo permissions

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

2) Select Kernel package, and then press Enter



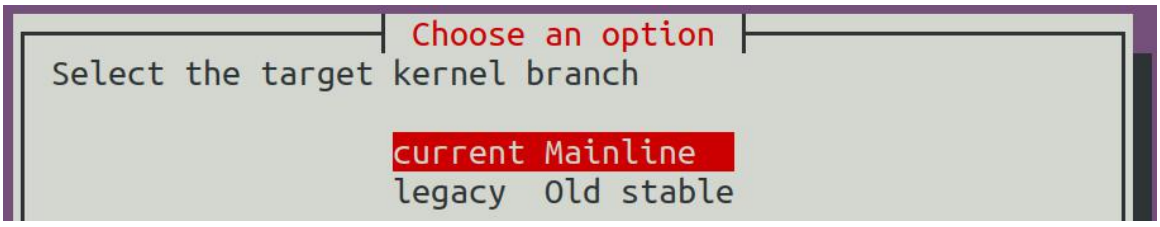
3) Then select the model of the development board



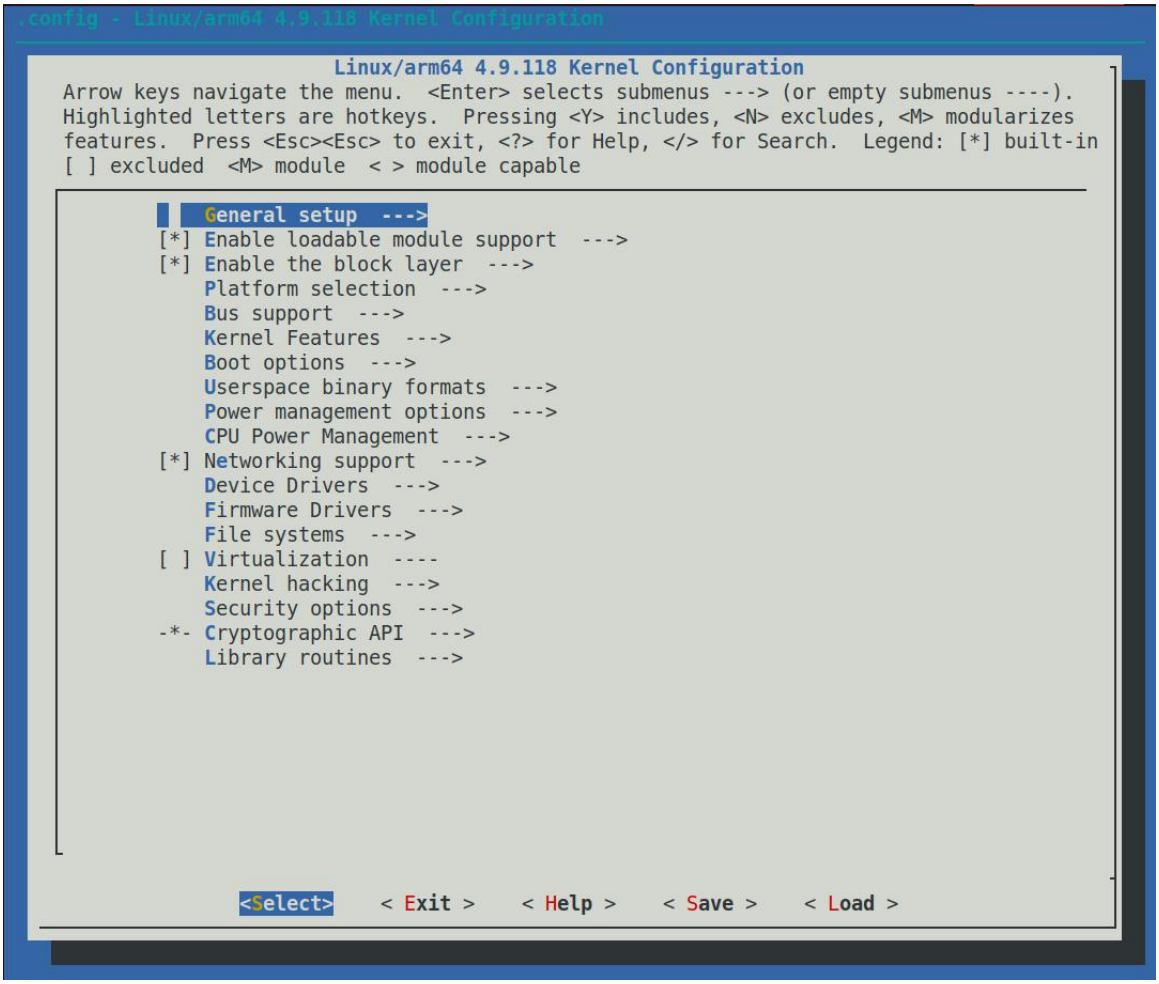
4) Then select the branch

a. current will compile linux 5.10

b. legacy will compile linux4.9



5) Then the kernel configuration interface opened through 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 directly. After exiting, the kernel source code will be compiled.



a. If you do not need to modify the kernel configuration options, 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
```

6) Part of the information prompted when compiling the kernel source code is explained as follows (take the legacy branch as an example)

a. The version of the linux kernel source code

[o.k.] Compiling legacy kernel [**4.9.118**]

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

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

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

[o.k.] Using kernel config file [**config/kernel/linux-sun50iw6-legacy.config**]

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

[o.k.] Exporting new kernel config [**output/config/linux-sun50iw6-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-sun50iw6_2.1.6_arm64.deb**]

g. Compile 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=orangepi3-lts**

```
BRANCH=legacy BUILD_OPT=kernel KERNEL_CONFIGURE=yes ]
```

7) The path of the kernel configuration file used by default when orangepi-build compiles the kernel is shown below

a. linux4.9-legacy branch

```
orangepi-build/external/config/kernel/linux-sun50iw6-legacy.config
```

b. linux5.10-current branch

```
orangepi-build/external/config/kernel/linux-5.10-sunxi64-current.config
```

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

a. **linux-dtb-legacy-sun50iw6_2.1.6_arm64.deb c**ontains dtb files used by the kernel

b. **linux-headers-legacy-sun50iw6_2.1.6_arm64.deb** contains kernel header files

c. **linux-image-legacy-sun50iw6_2.1.6_arm64.deb c**ontains kernel image and kernel module

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-legacy-sun50iw6_2.1.6_arm64.deb
output/debs/linux-headers-legacy-sun50iw6_2.1.6_arm64.deb
output/debs/linux-image-legacy-sun50iw6_2.1.6_arm64.deb
```

9) 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-sun50iw6_2.1.6_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-legacy-sun50iw6_2.1.6_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-legacy-sun50iw6_2.1.6_arm64.deb  usr
```

b. The decompressed file is as follows

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

```
│   ├── config-4.9.118-sun50iw6          //Configuration file used to compile the kernel source code
│   ├── System.map-4.9.118-sun50iw6
│   └── vmlinuz-4.9.118-sun50iw6         //Compile the generated kernel image file
├── etc
│   └── kernel
├── lib
│   └── modules                          //Compile the generated kernel module
├── linux-image-legacy-sun50iw6_2.1.6_arm64.deb
└── usr
    ├── lib
    └── share

8 directories, 4 files
```

10) When the orangepi-bulid compilation system compiles the linux kernel source code, it first synchronizes the linux kernel source code with the linux kernel source code of the github server, so if you want to modify the linux kernel source code, you first need to turn off the source code update function (you need to complete the compilation once This function can only be turned off after the linux kernel 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"
```

11) If you modify the kernel, you can use the following method to update the kernel and kernel modules of the Linux system on the development board

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-sun50iw6_2.1.6_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-sun50iw6
```

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

```
root@orangepi:~# dpkg -i linux-image-legacy-sun50iw6_2.1.6_arm64.deb
```

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

12) The method of installing the kernel header file into the linux system is as follows

a. Upload the deb package of the compiled linux header file to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-headers-legacy-sun50iw6_2.1.6_arm64.deb root@192.168.1.207:/root
```

b. Then log in to the development board and install the deb package of the linux header file just uploaded

```
root@orangepi:~# dpkg -i linux-headers-legacy-sun50iw6_2.1.6_arm64.deb
```

c. After installation, you can see the contents of the kernel header file just installed in /usr/src

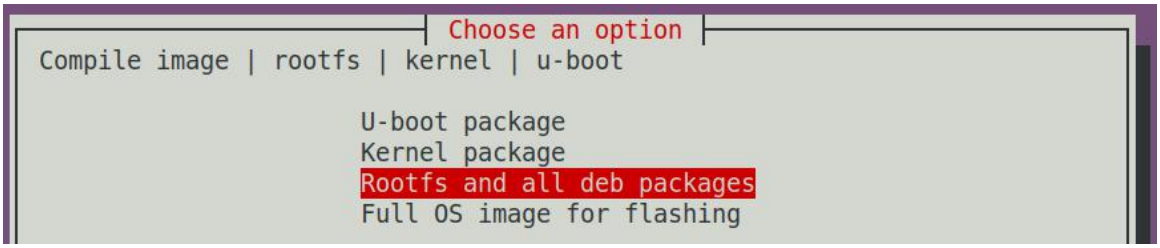
```
root@orangepi:~# ls /usr/src
linux-headers-4.9.118-sun50iw6
root@orangepi:~# ls /usr/src/linux-headers-4.9.118-sun50iw6
Documentation Module.symvers certs firmware init lib net security usr
Kconfig arch crypto fs ipc mm samples sound virt Makefile block
drivers include kernel modules scripts tools
```

4. 5. Compile rootfs

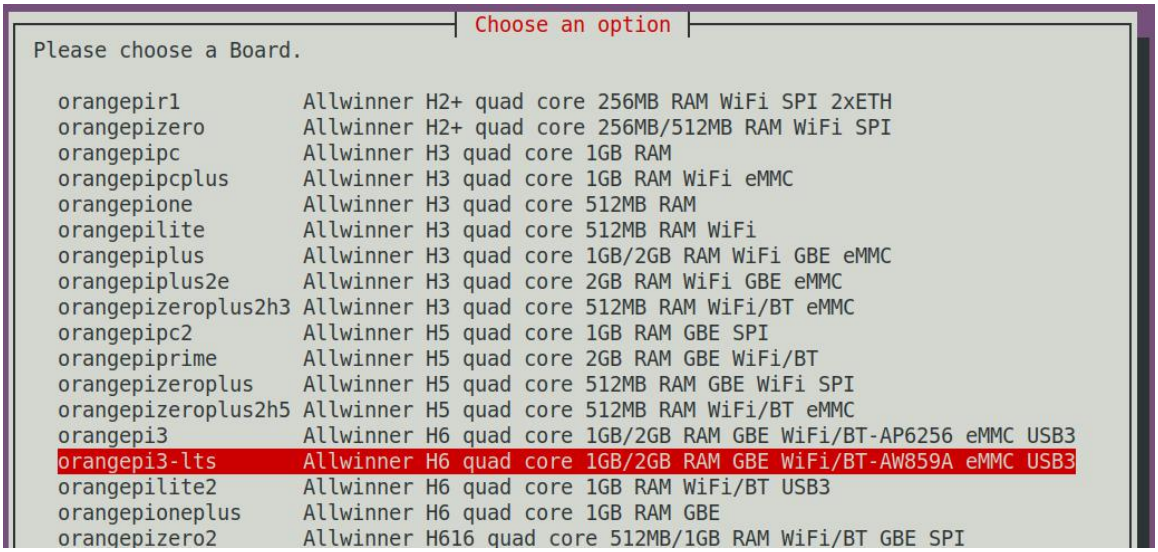
1) Run the build.sh script, remember to add sudo permissions

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

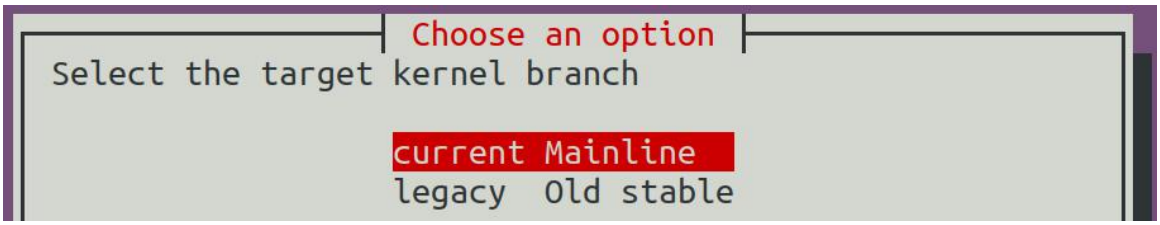
2) Select Rootfs and all deb packages, and then press Enter



3) Select Rootfs and all deb packages, and then press Enter



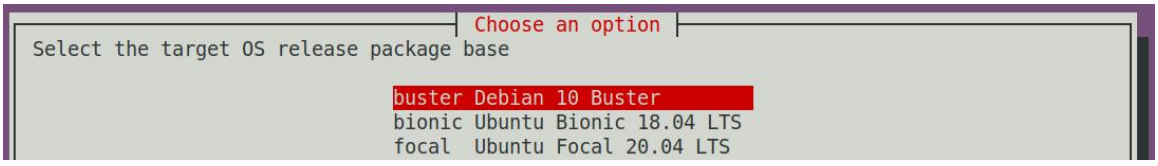
4) Then select the branch



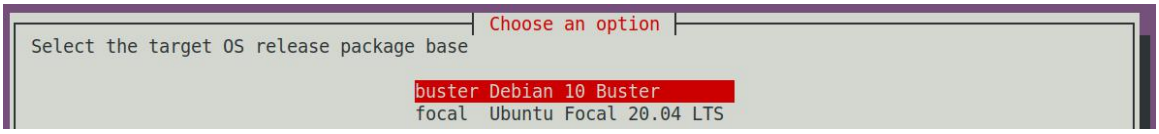
5) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04

a. The Linux distributions supported by linux4.9 are as follows



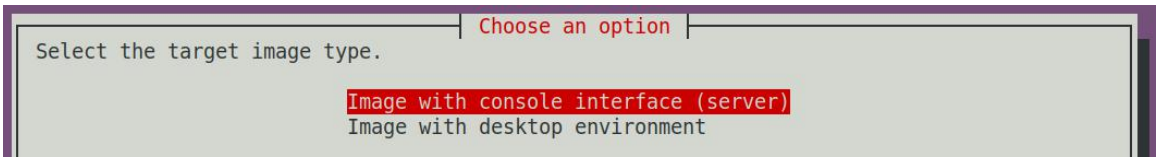
b. The Linux distributions supported by linxu5.10 are as follows



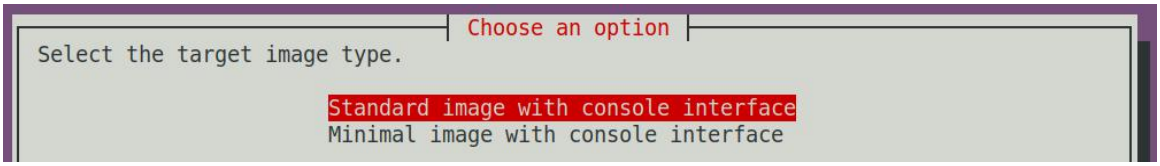
6) Then select the type of image

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



7) 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.



8) After selecting the type of image, rootfs will start to compile, and some of the information prompted during compilation are explained as follows

a. Type of rootfs

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

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 [**focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4**]

d. Compilation time

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

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

[o.k.] Repeat Build Options [**sudo ./build.sh BOARD=orangepi3-lts BRANCH=legacy BUILD_OPT=rootfs RELEASE=focal BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes**]

9) Repeat the command to compile rootfs, use the following command without selecting

through the graphical interface, you can start compiling rootfs directly

a. focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4 is a compressed package of rootfs, the meaning of each field of the name is as following:
   a) Focal represents the type of linux distribution of rootfs
   b) cli means rootfs is the server version type, if it is dekstop, it means the desktop version type
   c) arm64 represents the architecture type of rootfs
   d) 153618961f14c28107ca023429aa0eb9 is the MD5 hash value generated by the package names of all software packages installed by rootfs. As long as the list of software packages installed by rootfs is not modified, this value will not change. The compilation script will use this MD5 hash value. Determine whether you need to recompile rootfs
b. focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list list the package names of all packages installed by rootfs

```
test@test:~/orangepi-build$ ls external/cache/rootfs/
focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4
focal-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list
```

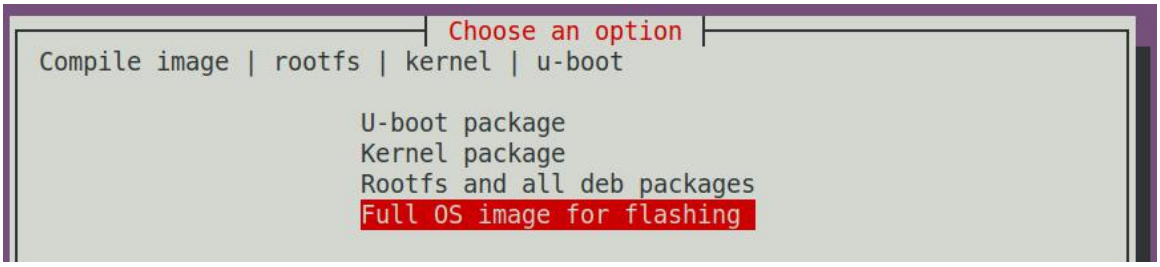
10) If the required rootfs already exists under **external/cache/rootfs**, then compiling rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it is already Rootfs with cache available,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

4. 6. Compile linux image

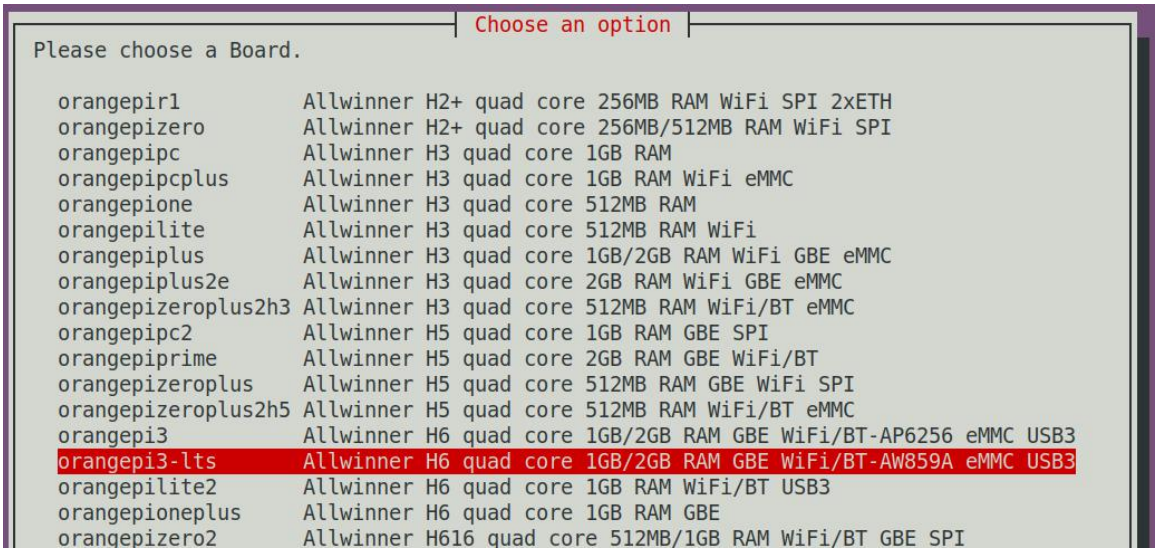
1) Run the build.sh script, remember to add sudo permissions

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

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



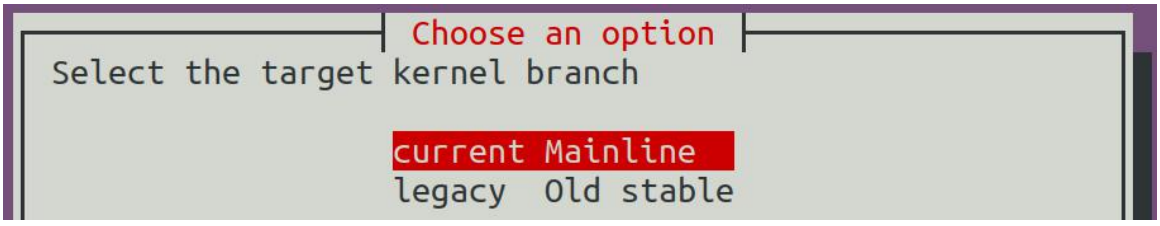
3) Then select the model of the development board



4) Then select the branch

d. current will compile linux 5.10

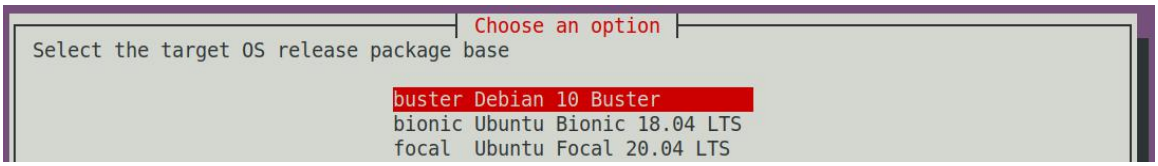
e. legacy will compile linux4.9



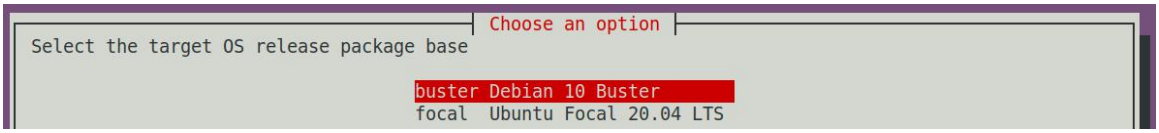
5) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04

a. The Linux distributions supported by linux4.9 are as follows



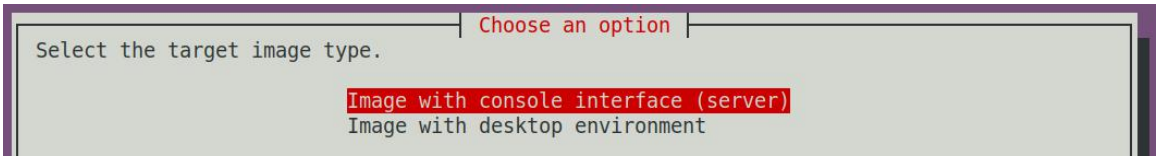
b. The Linux distributions supported by linxu5.10 are as follows



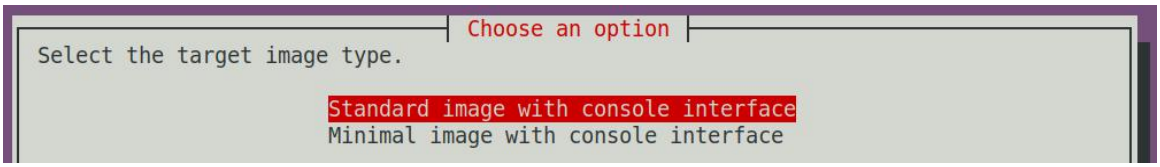
6) Then select the type of image

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

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



7) 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.



8) 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 the linux source code and generate linux-related deb packages

e. Make deb package of linux firmware

f. Make deb package of orangepi-config tool

g. Make board-level support deb packages

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, then re-create the rootfs, if it has been cached, then directly 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 configurations, etc.

l. Then make an image file and format the partition, the default type is ext4

m. Copy the configured rootfs to the image partition

n. Then update the initramfs

o. Finally, the bin file of u-boot is written to the image through the dd command

9) After compiling the image, the following information will be prompted (take the legacy branch as an example)

a. he storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/Orangepi3-lts_2.1.6_ubuntu_focal_server_linux4.9.118/Orangepi3-lts_2.1.6_ubuntu_focal_server_linux4.9.118.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=orangepi3-lts BRANCH=legacy BUILD_OPT=image RELEASE=focal  BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

5. Android system instructions

5. 1. Supported Android version

Android version	Kernel version
Android 9.0	linux4.9

5. 2. Android 9.0 function adaptation situation

Function	Statu
HDMI video	OK
HDMI audio	OK
USB2.0	OK
USB3.0	OK
TF card boot	OK
eMMC boot	OK
Network card	OK
Infrared	OK
WIFI	OK
WIFI hotsport	OK
Buletooth	OK
Headphone audio	OK
TV-OUT	OK
USB camera	OK
LED lights	OK
Temperature sensor	OK
Mali GPU	OK
Video codec	OK
Power button	OK
ADB debugging	OK

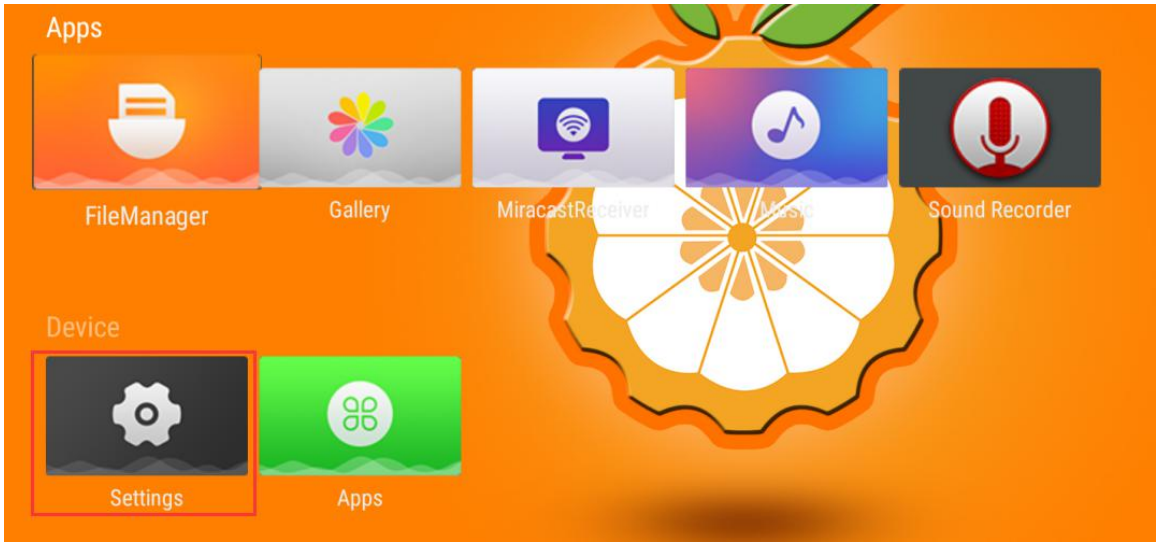
5. 3. Onboard LED light display description

	Yellow light	Red light
u-boot startup phase	Off	On
Kernel boot to enter the system	On	Off
GPIO	PL7	PL4

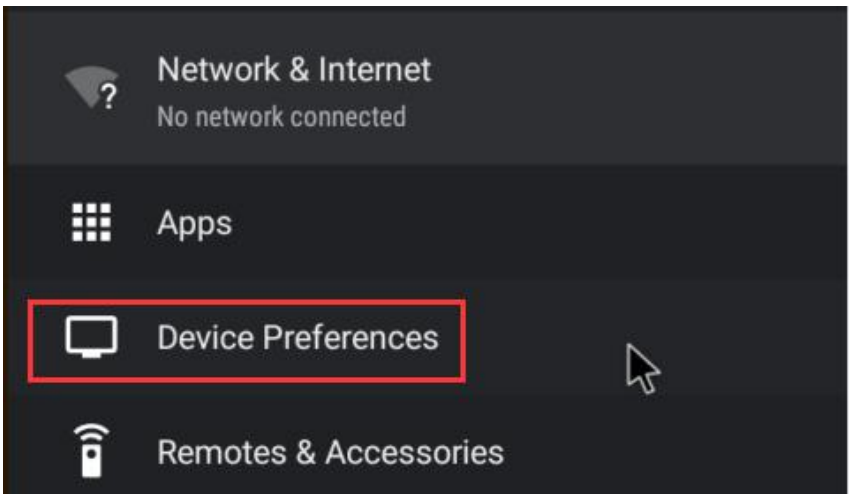
5. 4. How to use ADB

5. 4. 1. Turn on the USB debugging option

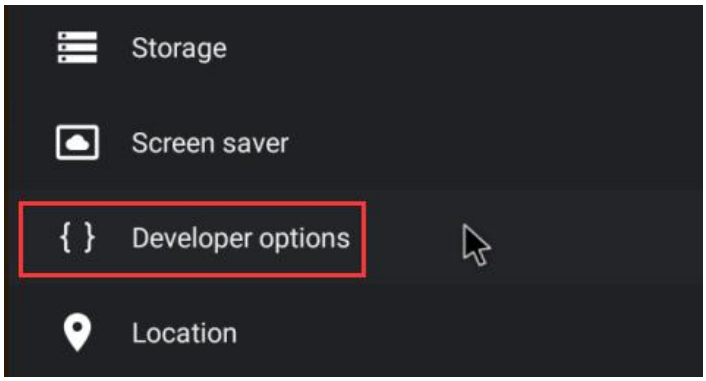
1) 1) First select Settings



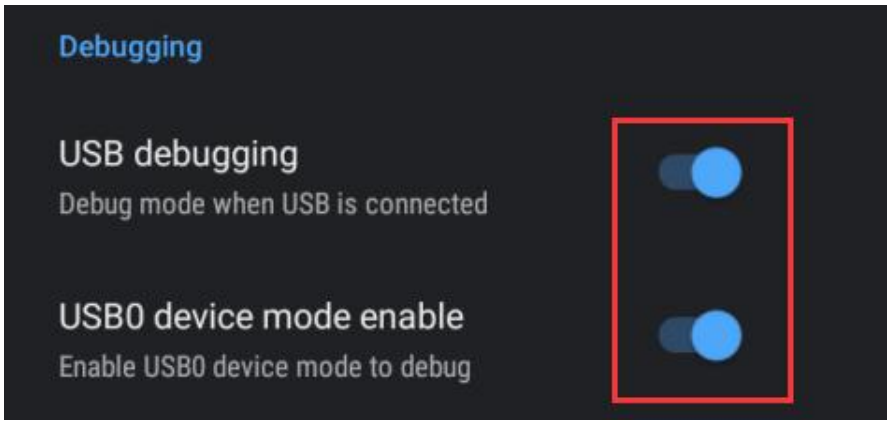
2) Then select **Device Preferences**



3) Then select Developer options



4) 4) Then turn on USB debugging and USB device mode enable



5. 4. 2. Use network connection adb debugging

1) To use the network adb, there is no need to use the USB interface data cable to connect the computer and the development board, but to communicate through the

network, so first make sure that the wired or wireless network of the development board has been connected, and then obtain the IP address of the development board. To be used later

2) Make sure that the **USB debugging** option is turned on

3) Make sure that the service.adb.tcp.port of the Android system is set to port number 5555 (it has been set by default)

```
petrel-p1:/ # getprop | grep "adb.tcp"
[service.adb.tcp.port]: [5555]
```

5) If service.adb.tcp.port is not set, you can use the following command to set the port number of the network adb

```
petrel-p1:/ # setprop service.adb.tcp.port 5555
petrel-p1:/ # stop adbd
petrel-p1:/ # start adbd
```

6) Install adb tool on Ubuntu PC

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

7) Then connect to the network adb on the Ubuntu PC

```
test@test:~$ adb connect 192.168.1.xxx      (The IP address needs to be modified to the IP address of the development board)
* daemon not running; starting now at tcp:5037
* daemon started successfully
connected to 192.168.1.xxx:5555

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

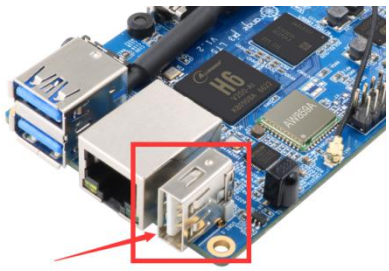
8) Then you can log in to the android system through the adb shell on the Ubuntu PC

```
test@test:~$ adb shell
petrel-p1:/ #
```

5.4.3. Use the data cable to connect adb for debugging

1) Make sure that the USB debugging option is turned on

2) Then you need to use a double-headed USB interface data cable to connect the development board to the USB interface of the computer. The interface that needs to be connected to the development board is shown in the right picture below



3) Install adb tool on Ubuntu PC

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

4) View the identified ADB device

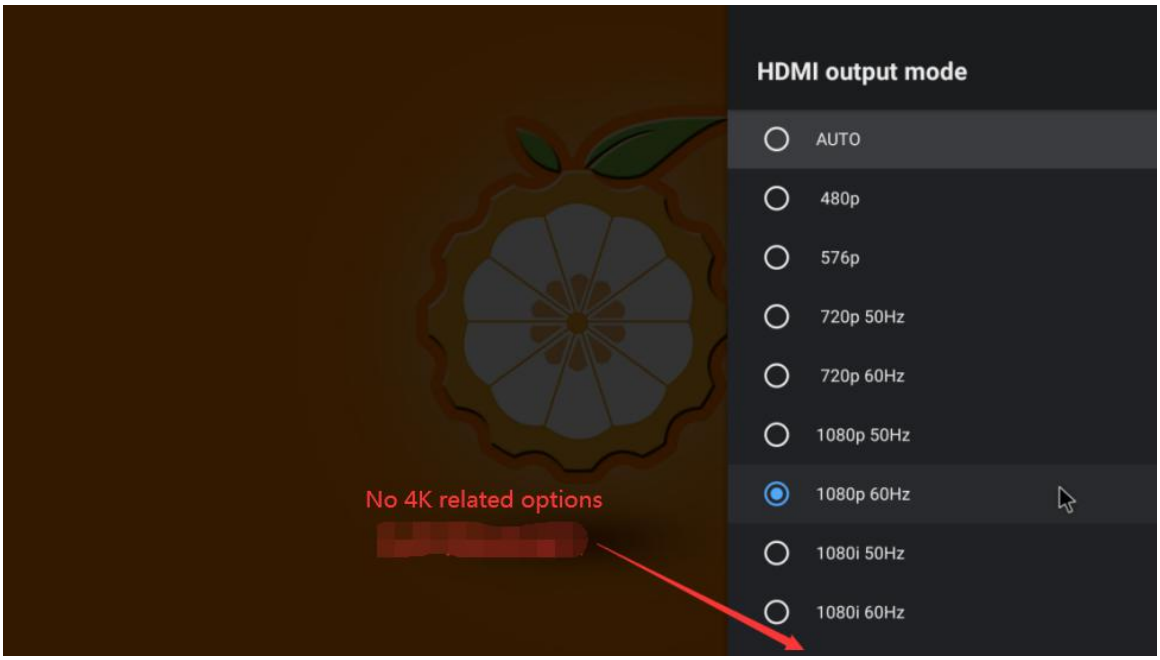
```
test@test:~$ adb devices
List of devices attached
* daemon not running; starting now at tcp:5037
* daemon started successfully
cc001421c7028941ed1     device
```

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

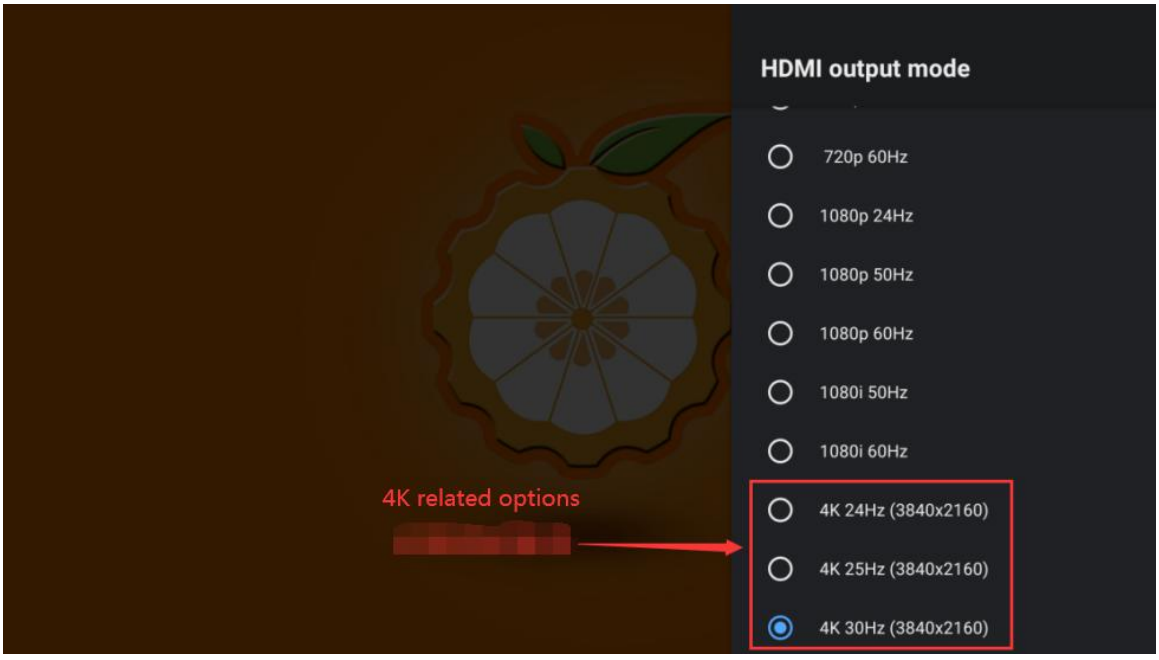
```
test@test:~$ adb shell
petrel-p1:/ #
```

5.5. HDMI 4K display description

1) If you connect the HDMI of Orange Pi 3 LTS to a TV or monitor that does not support 4K, you will not see 4K-related options when you view the resolution supported by HDMI in the settings



2) Only when you connect the HDMI of Orange Pi 3 LTS to a TV or monitor that supports 4K, you can see 4K-related options in the resolution supported by HDMI



5. 6. HDMI to VGA display test

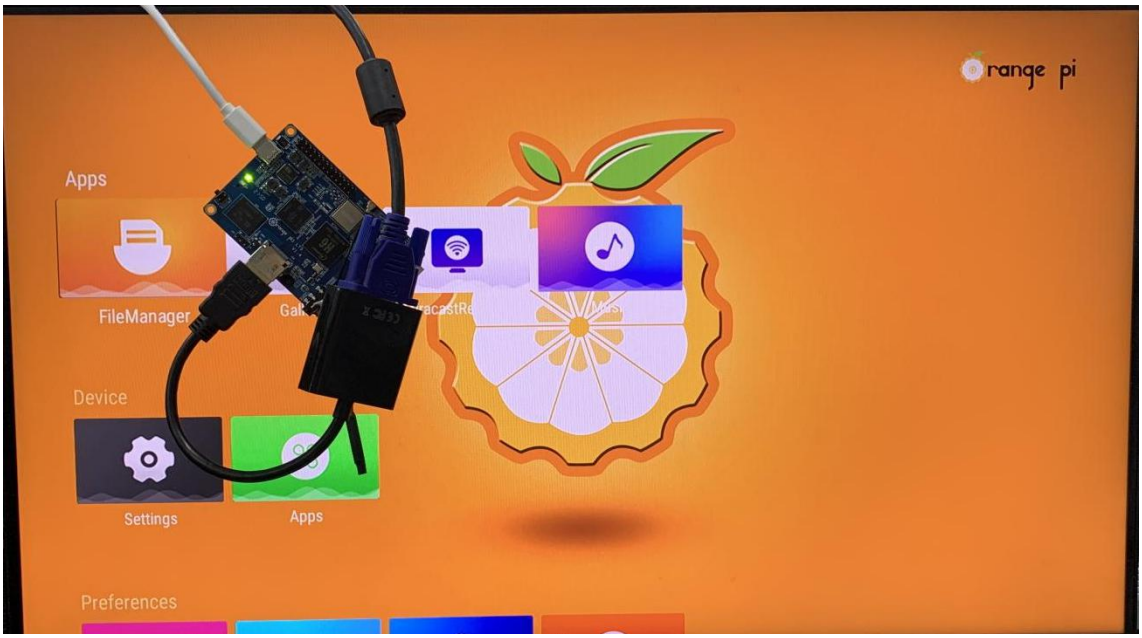
1) First, you need to prepare the following accessories

a. HDMI to VGA converter

b. A VGA cable

c. A monitor or TV that supports VGA interface

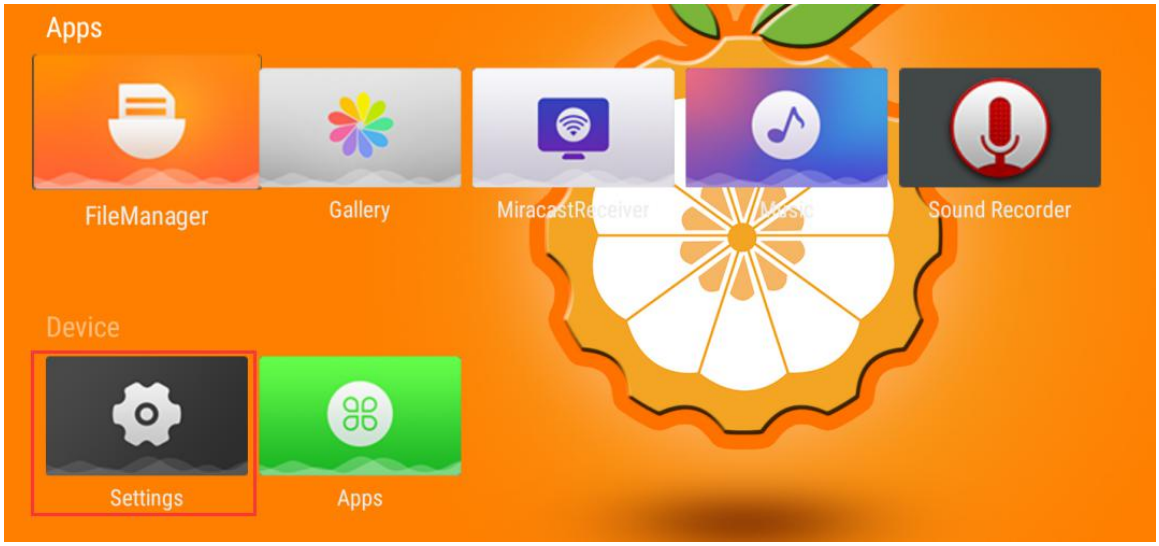
2) The HDMI to VGA display test is shown below



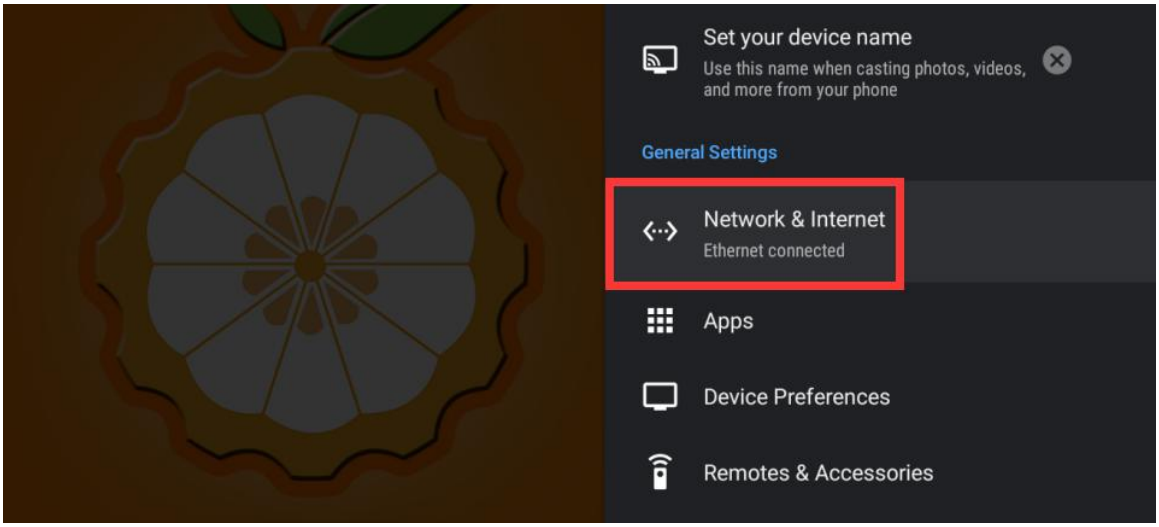
3) When using HDMI to VGA display, the development board and the Android system of the development board do not need to do any settings, only the HDMI interface of the development board can display normally. So if there is a problem with the test, please check if there is a problem with the HDMI to VGA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

5. 7. WI-FI connection method

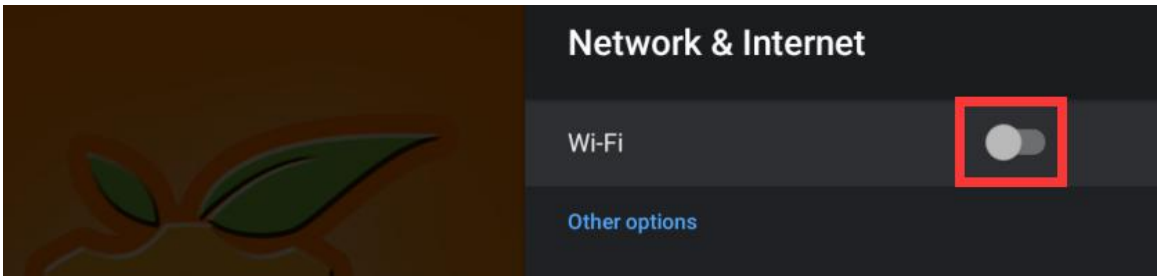
1) First select **Settings**



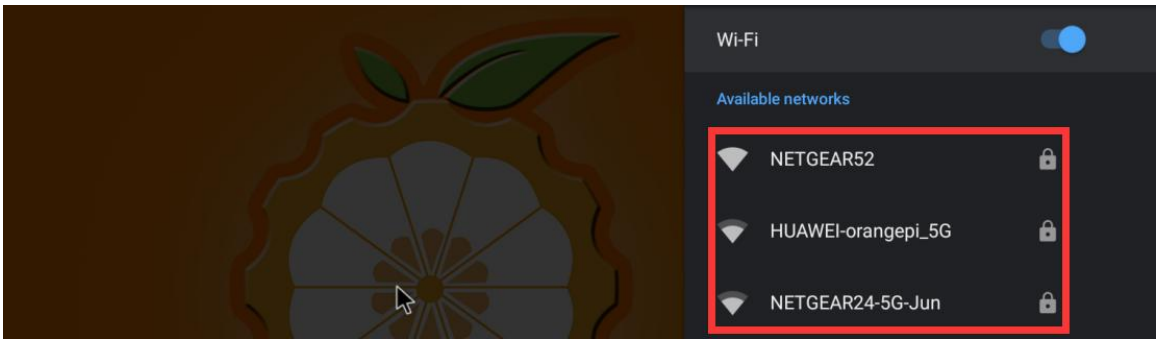
2) Then select **Network & Internet**



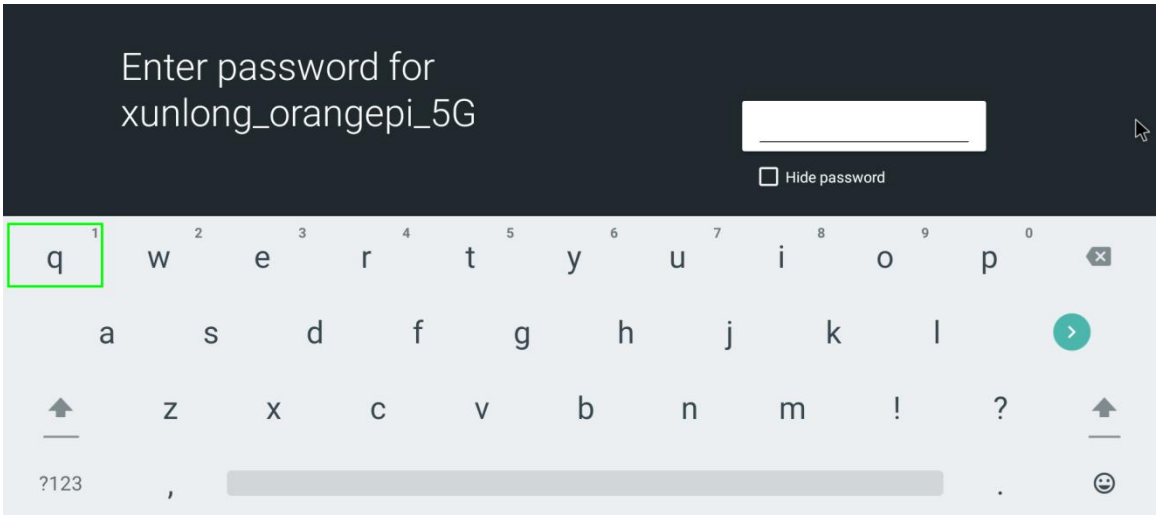
3) Then open WI-FI



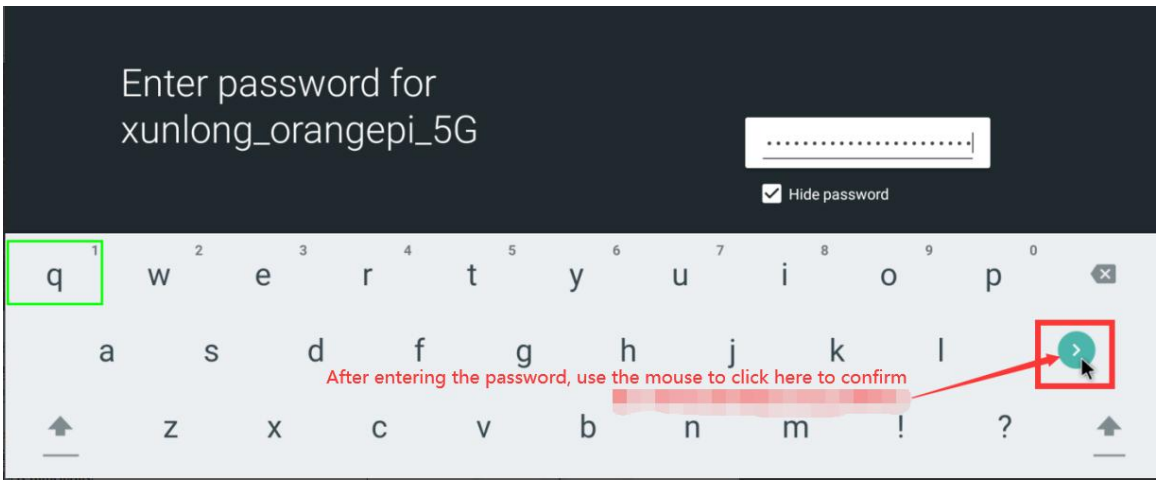
4) After opening WI-FI, you can see the searched signal under Available networks



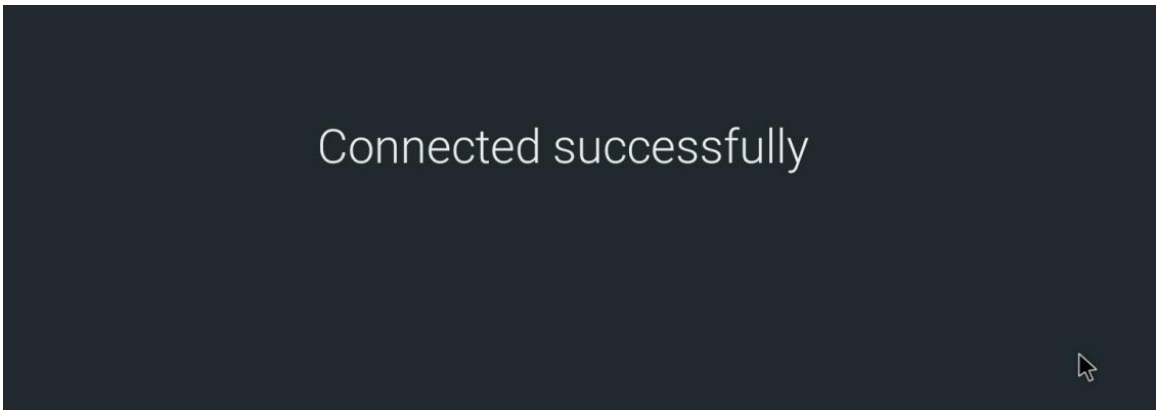
5) After selecting the WI-FI you want to connect to, the password input interface shown in the figure below will pop up



6) Then use the keyboard to enter the password corresponding to WI-FI, and then use the mouse to click the Enter button in the virtual keyboard to start connecting to WI-FI



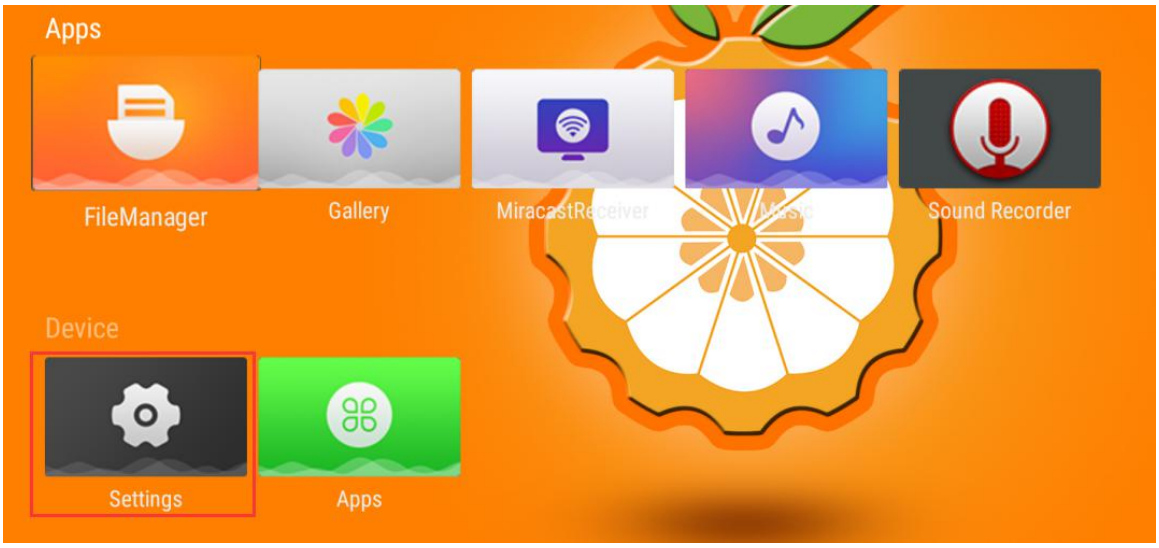
7) The display after successful WI-FI connection is as shown in the figure below



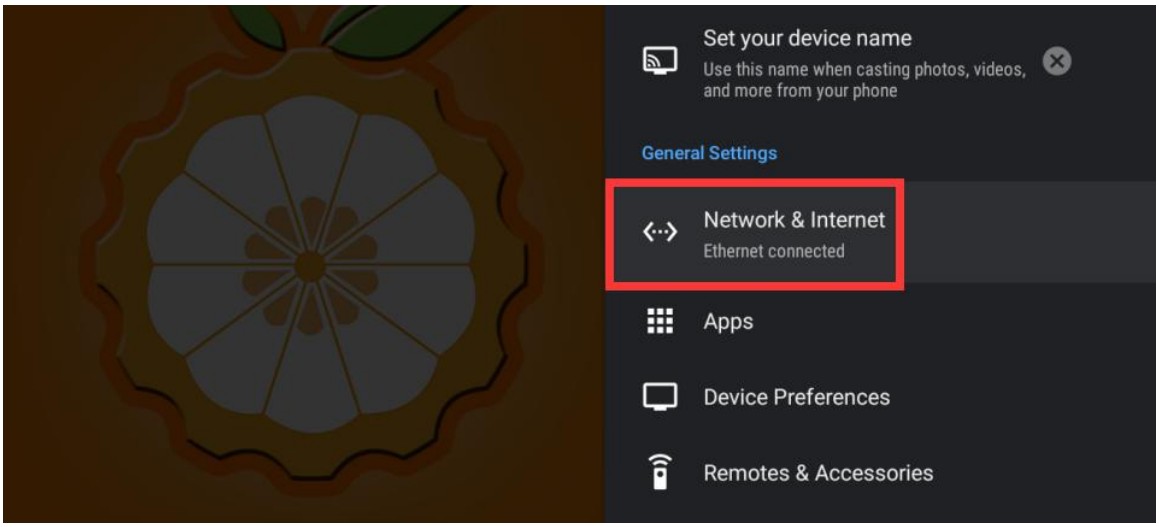
5.8. How to use WI-FI hotspot

1) First of all, please make sure that the Ethernet port is connected to the network cable and can access the Internet normally

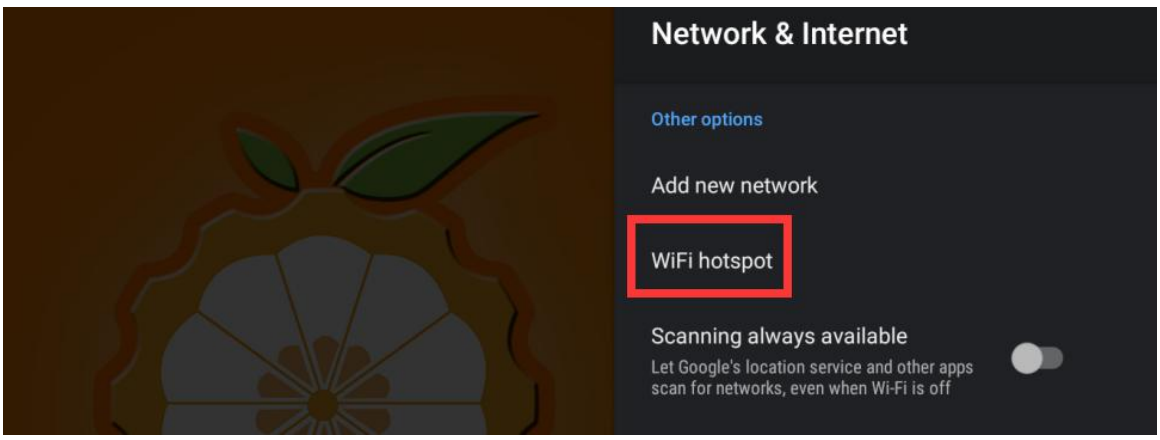
2) 2) Then select **Settings**



3) Then select **Network & Internet**

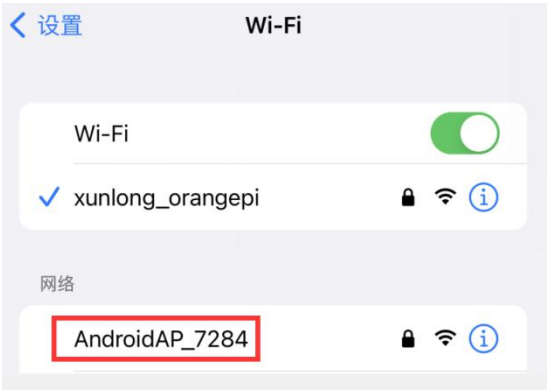


4) Then select **WIFI hotspot**



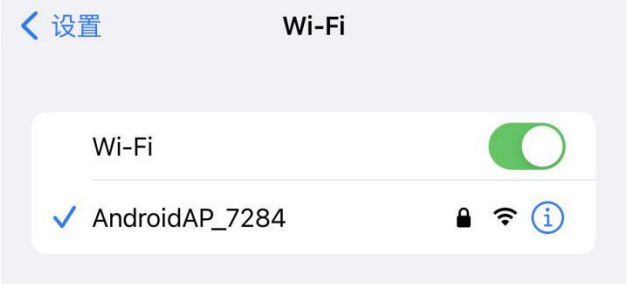
5) 5) Then turn on Hotspot Enable, you can also see the name and password of the generated hotspot in the figure below, remember them, and use them when connecting to the hotspot (If you need to modify the name and password of the hotspot, you need to turn off Hotspot Enable first, Then can be modified)

6) You can take out your mobile phone at this time. If everything is normal, you can find the WIFI hotspot with the same name (here, AndroidAP_7284) shown under Hotspot name in the WI-FI list of the mobile phone search. Then you can click AndroidAP_7284 to connect to the hotspot, the password can be seen under the Hotspot password in the picture above



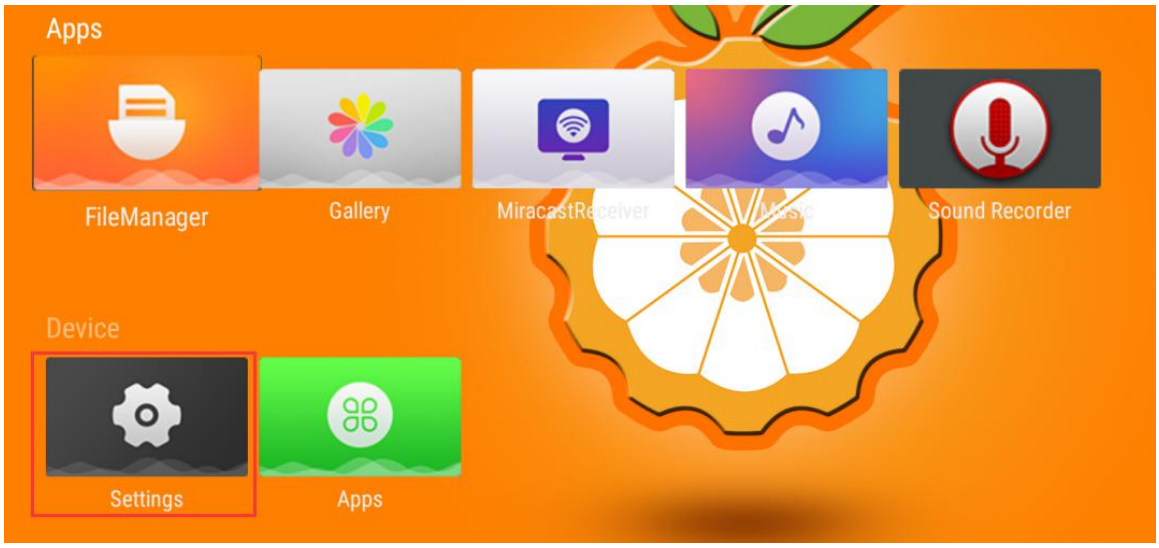
7) After the connection is successful, it will display as shown in the figure below (different mobile phone interface will be different, the specific interface is subject to your mobile phone display). At this point, you can open a webpage on your phone to see if you

can surf the Internet. If you can open the webpage normally, it means that the Wi-Fi Hotspot of the development board can be used normally.

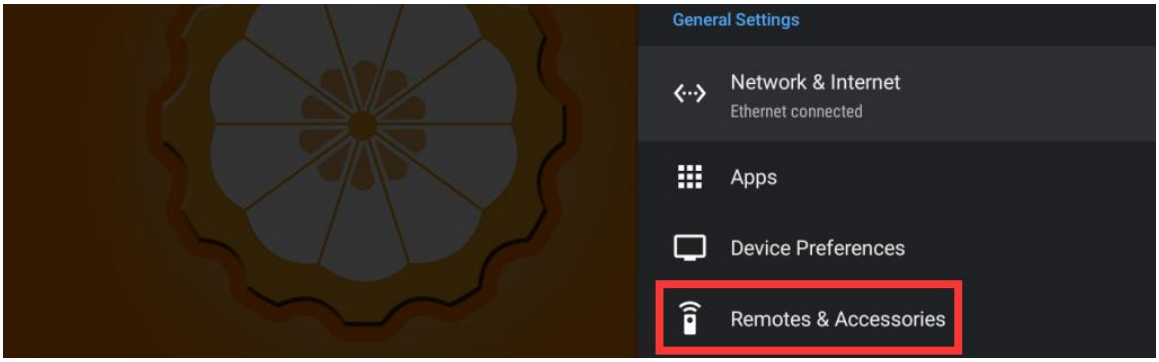


5. 9. Bluetooth connection method

1) First select **Settings**

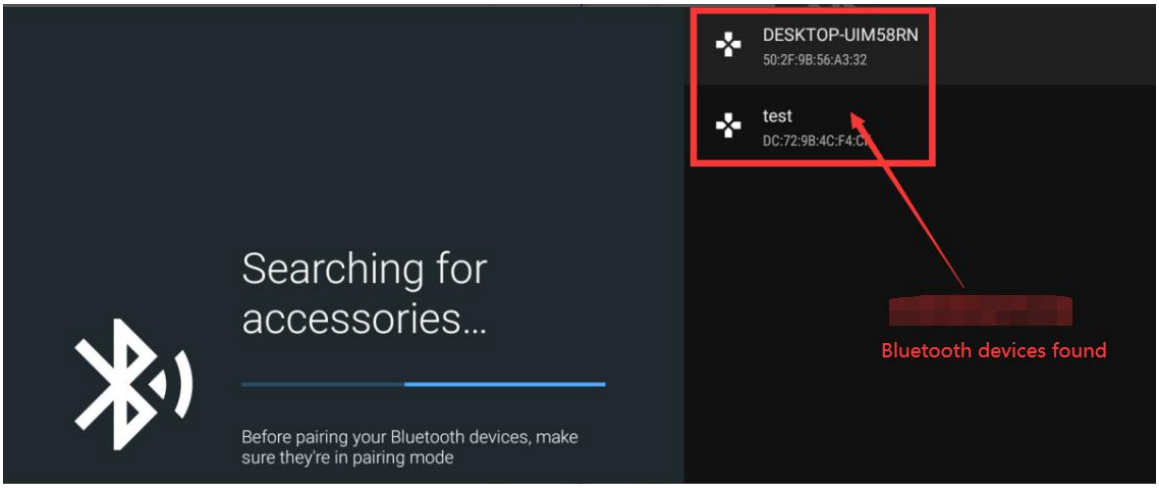


2) Then select **Remotes & Accessories**

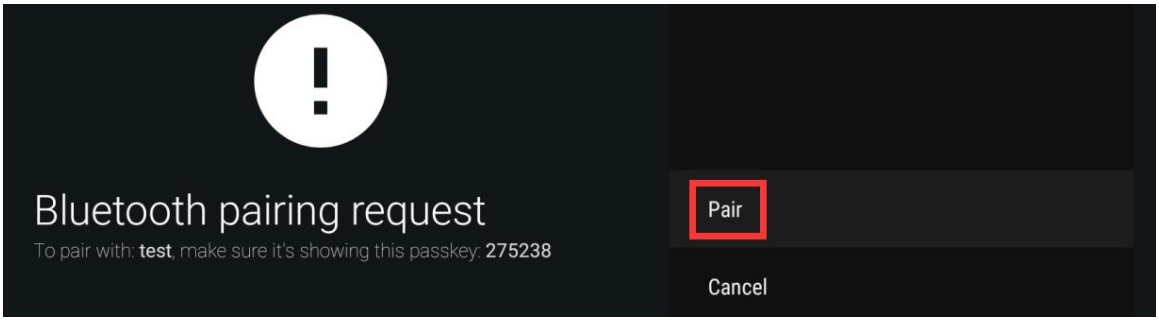


3) Then the system will start searching for surrounding Bluetooth devices

4) After searching for a Bluetooth device, the following interface will be displayed, and the searched Bluetooth device is in the list on the right

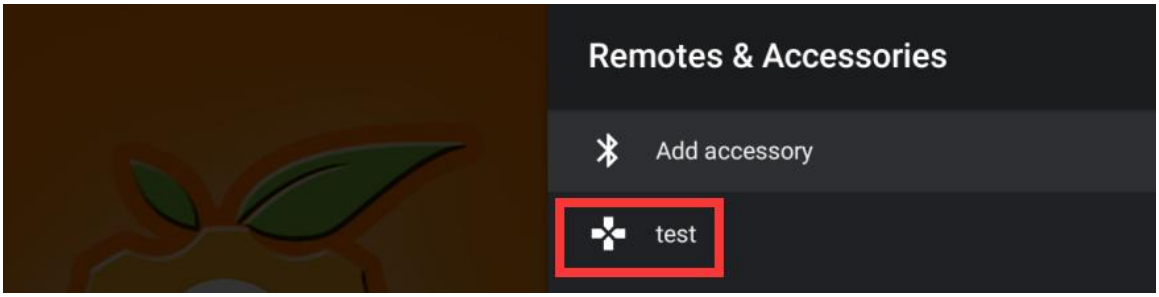


5) Then click on the Bluetooth device you want to connect to to start pairing. When the following interface pops up, please use the mouse to select the Pair option

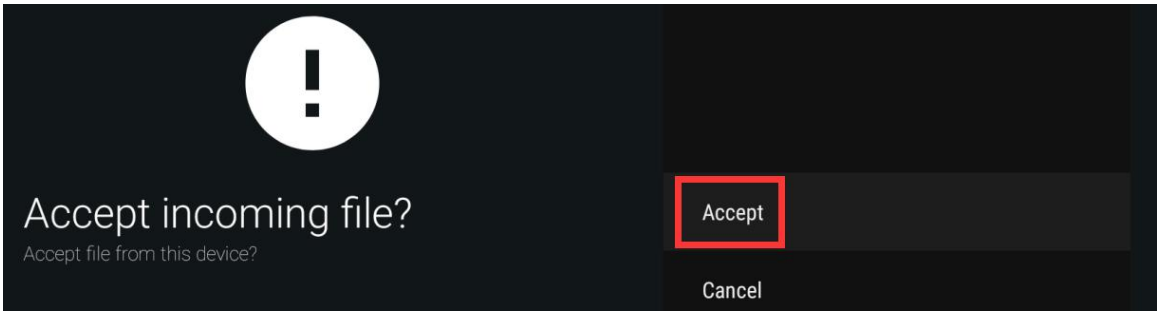


6) The test here is the configuration process of the development board and the Bluetooth of the Android phone. At this time, the following confirmation interface will pop up on the phone, and the pairing process will start after clicking the pairing button on the phone.

7) After the pairing is complete, open Remotes & Accessories to see the paired Bluetooth devices

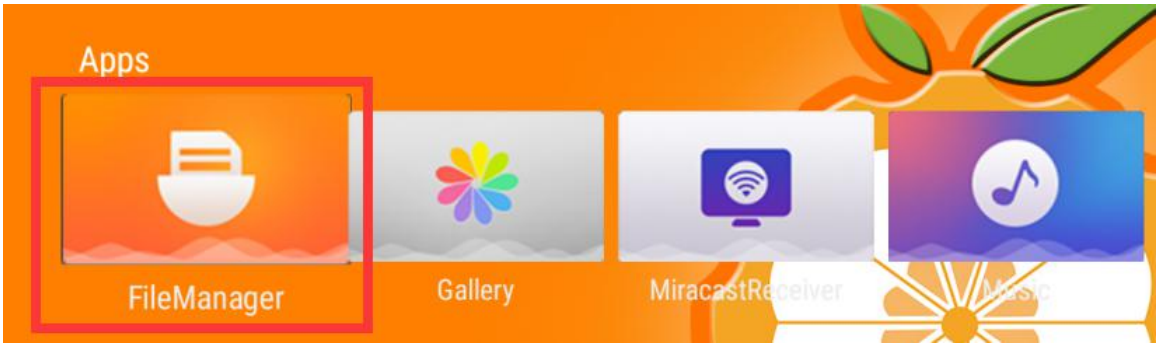


8) At this time, you can use the mobile phone's Bluetooth to send a picture to the development board. After sending, you can see the confirmation interface below in the Android system of the development board, and then click Accept to start receiving the picture sent by the mobile phone.



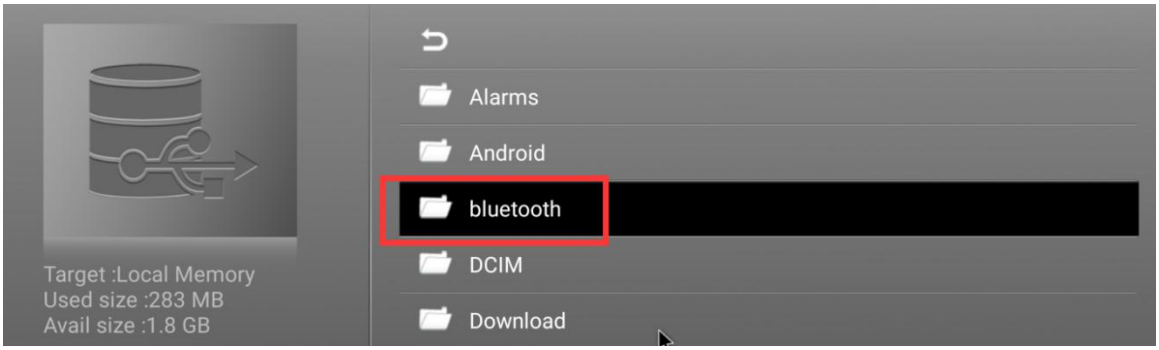
9) The pictures received by the Bluetooth of the Android system of the development board can be seen in the folder below

a. First select **FileManager**



b. Then select **Local Memory**

c. Then you can see the folder named bluetooth, the pictures received by the development board Bluetooth are saved in this folder, open this folder to see it



5. 10. How to use USB camera

1) First insert the USB camera into the USB interface of the development board, and then confirm that the kernel module related to the USB camera has been loaded normallypetrel-p1:/ # **lsmod**

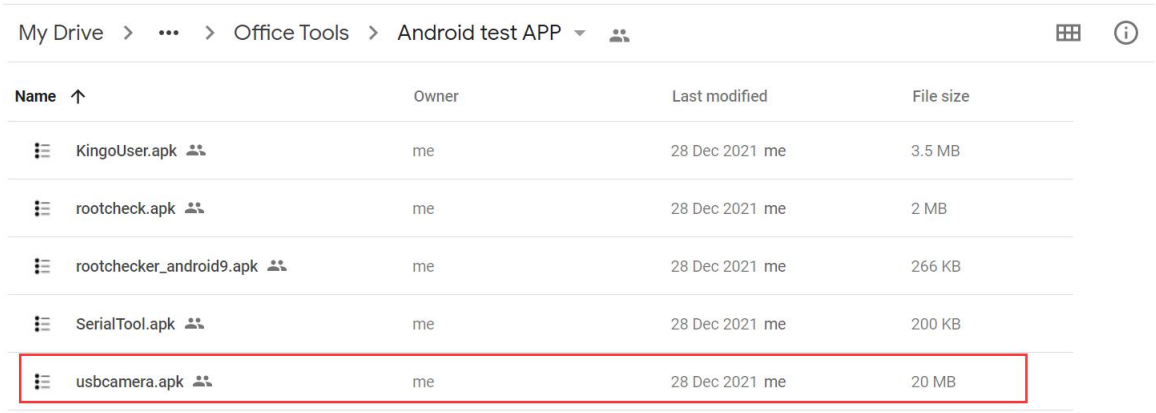
Module	Size	Used by
sprdwl_ng	381753	0
sprdbt_tty	29622	2
uwe5622_bsp_sdio	260394	2 sprdwl_ng,sprdbt_tty
uvcvideo	**86387**	**0**
videobuf2_v4l2	**19657**	**1 uvcvideo**
videobuf2_vmalloc	**7119**	**1 uvcvideo**
videobuf2_memops	**2671**	**1 videobuf2_vmalloc**
videobuf2_core	**38644**	**2 uvcvideo,videobuf2_v4l2**
sunxi_ir_rx	11332	0
mali_kbase	392286	22

2) If the USB camera is recognized normally, the corresponding video device node will be generated under /dev

```
petrel-p1:/ # ls /dev/video*
/dev/video0
petrel-p1:/ # ls /sys/class/video4linux/ -lh
total 0
lrwxrwxrwx 1 root root 0 2020-12-08 15:58 video0 -> ../../devices/soc/5311000.ehci3-controller/usb2/2-1/2-1:1.0/video4linux/video0
petrel-p1:/ #
```

3) Then make sure that the adb connection between the Ubuntu PC and the development board is normal

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



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

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

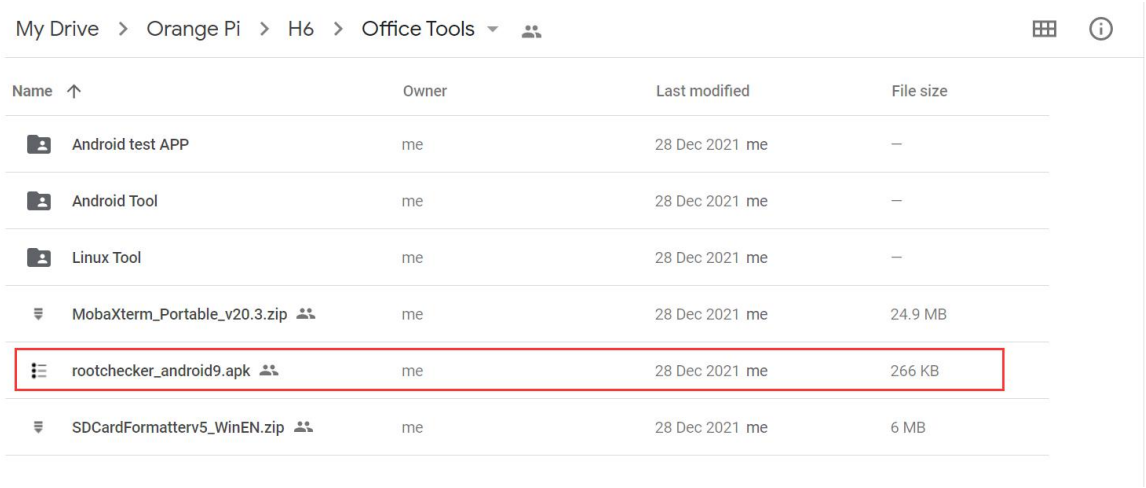
6) After installation, you can see the startup icon of the USB camera on the Android desktop

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

5. 11. Android system ROOT description

The Android 9.0 system released by Orange Pi is already ROOT, you can use the following method to test

1) Download **rootchecker_android9.apk** from the official tool on the Orange Pi 3 LTS data download page



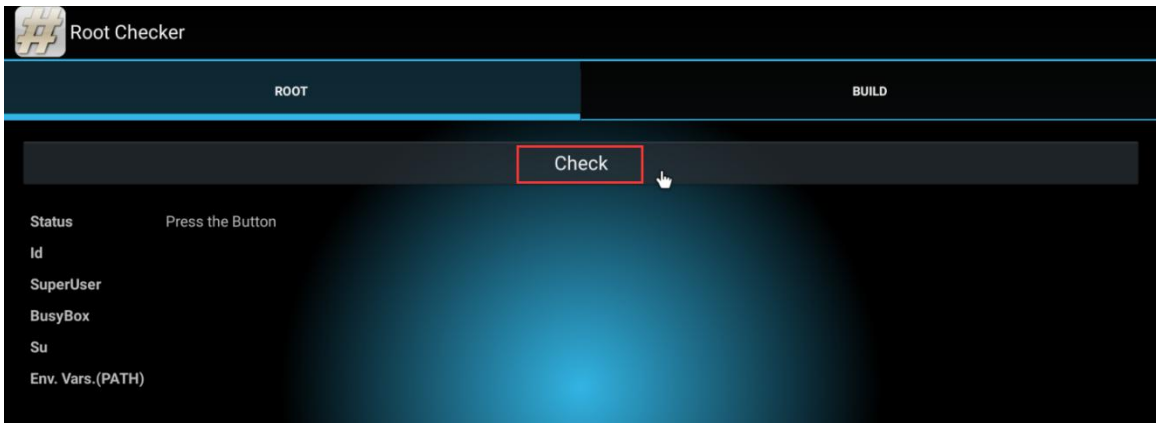
2) Then make sure that the adb connection between the Ubuntu PC and the development board is normal

3) Then use the adb command to install rootcheck_android9.apk to the Android system, of course, you can also use the U disk copy method to install

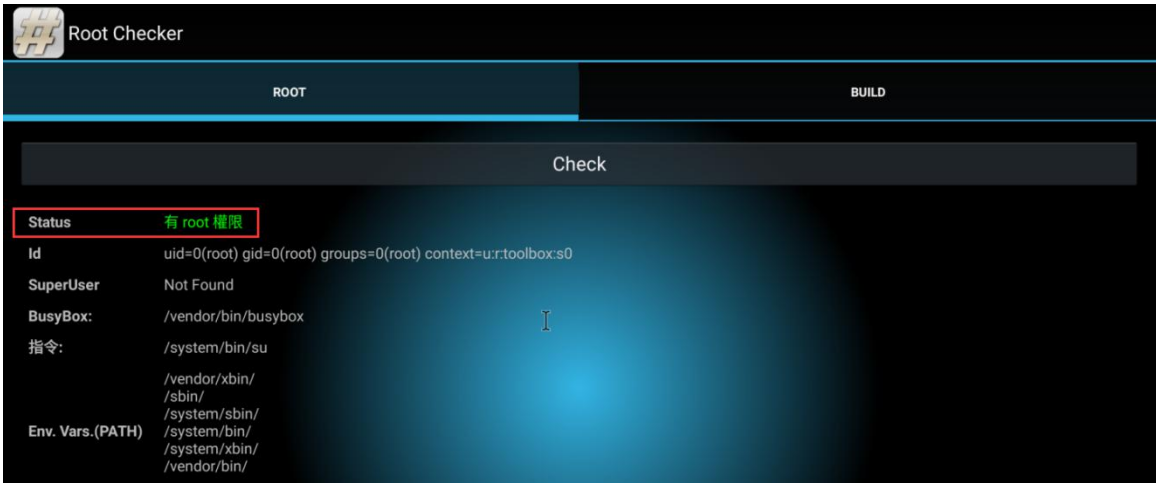
```
test@test:~$ adb install rootcheck_android9.apk
```

4) After installation, you can see the startup icon of the ROOT test tool on the Android desktop

5) After installation, you can see the startup icon of the ROOT test tool on the Android desktop



6) The display after clicking Check is as follows, you can see that the Android system has obtained ROOT permission



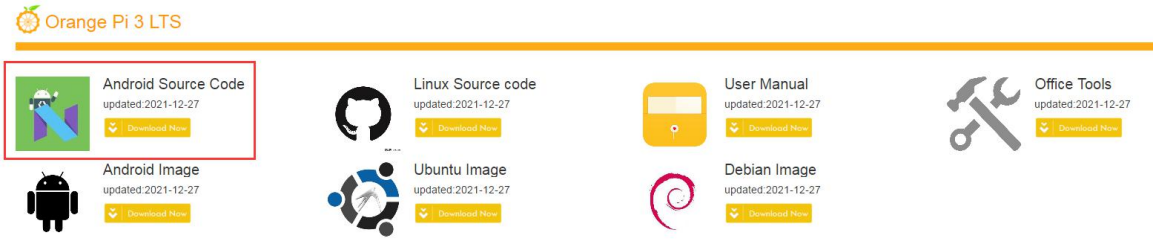
6. Android9.0 SDK instructions

1) The Android SDK is compiled on a PC with Ubuntu 18.04 installed. There may be some differences in other versions of Ubuntu systems. If you want to use other versions of Ubuntu to compile the Android source code, please test it yourself

2) The H6 Android9.0 SDK provided by Orange Pi is the original SDK released by the chip manufacturer. If you want to use the Android image compiled by the Android9.0 SDK on the Orange Pi development board, you need to adapt to different boards to ensure that all The function is used normally, Orange Pi based on the original SDK adaptation code is not provided

6.1. Download the source code of Android SDK

1) Download the Android 9.0 source code from orange pi official website Orange Pi 3 LTS data download area: http://www.orangepi.org/downloadresources/



2) After downloading the Android source code, please check whether the MD5 checksum is correct, if not, please download the source code again

```
test@test:~$ md5sum -c android.tar.gz.md5sum
android.tar.gz: confirm
test@test:~$ md5sum -c lichee.tar.gz.md5sum
longan.tar.gz: confirm
```

3) Then unzip the Android source code

a. android: store android-related source code
b. lichee: store the source code of linux kernel and u-boot, and other configuration files

```
test@test:~$ tar -zxf android.tar.gz
```

```
test@test:~$ tar -zxf lichee.tar.gz
test@test:~$ ls
android    lichee
```

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 -y git gnupg flex bison gperf build-essential \
zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 \
lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z1-dev ccache \
libgl1-mesa-dev libxml2-utils xsltproc unzip u-boot-tools
```

6. 3. Compile Android image

6. 3. 1. Compile the kernel

1) The compilation method of lichee is as follows

a. Compile options need to be configured when compiling for the first time

```
test@test:~$ cd ichee
```

```
test@test:~/lichee$ ./build.sh config

Welcome to mkscript setup progress
All available platform:
    0. android
    1. dragonboard
    2. linux
    3. camdroid
Choice [android]: 0
All available chip:
    0. sun3iw1p1
    1. sun50iw1p1
    2. sun50iw2p1
    3. sun50iw3p1
    4. sun50iw6p1
    5. sun50iw8p1
    6. sun8iw10p1
    7. sun8iw11p1
    8. sun8iw12p1
    9. sun8iw15p1
   10. sun8iw17p1
   11. sun8iw1p1
   12. sun8iw3p1
   13. sun8iw5p1
   14. sun8iw6p1
   15. sun8iw7p1
   16. sun8iw8p1
   17. sun8iw9p1
   18. sun9iw1p1
Choice [sun50iw6p1]: 4
All available kern_ver:
    0. linux-4.9
Choice [linux-4.9]: 0
All available board:
    0. fpga
    1. perf1_v1_0
```

```
    2. perf2_v1_0
    3. perf3_v1_0
    4. petrel-cmcc-p1
    5. petrel-h603-axpdummy
    6. petrel-iptv-p1
    7. petrel-p1-axp802
    8. petrel-p1-axpdummy
    9. petrel-p1
   10. pro_v1_0
   11. qc
   12. sata
Choice [petrel-p1]: 9
```

b. Then start to compile

```
test@test:~/lichee$ ./build.sh
```

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

```
sun50iw6p1 compile Kernel successful



INFO: build kernel OK.
INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: build rootfs OK.
INFO: ----------------------------------------
INFO: build lichee OK.
INFO: ----------------------------------------
```

6. 3. 2. Compile Android source code

1) The command to compile Android is as follows

```
test@test:~$ cd android
test@test:~/android$ source build/envsetup.sh
test@test:~/android$ lunch petrel_fvd_p1-eng
test@test:~/android$ extract-bsp
test@test:~/android$ make -j8
test@test:~/android$ pack
```

2) The pack command is used to package and generate Android firmware. If the compilation and packaging process passes smoothly, the following information will be prompted

```
Dragon execute image.cfg SUCCESS !
----------image is at----------

lichee/tools/pack/sun50iw6p1_android_petrel-p1_uart0.img

pack finish
```

3) The path where the generated Android image is stored is

```
lichee/tools/pack/sun50iw6p1_android_petrel-p1_uart0.img
```