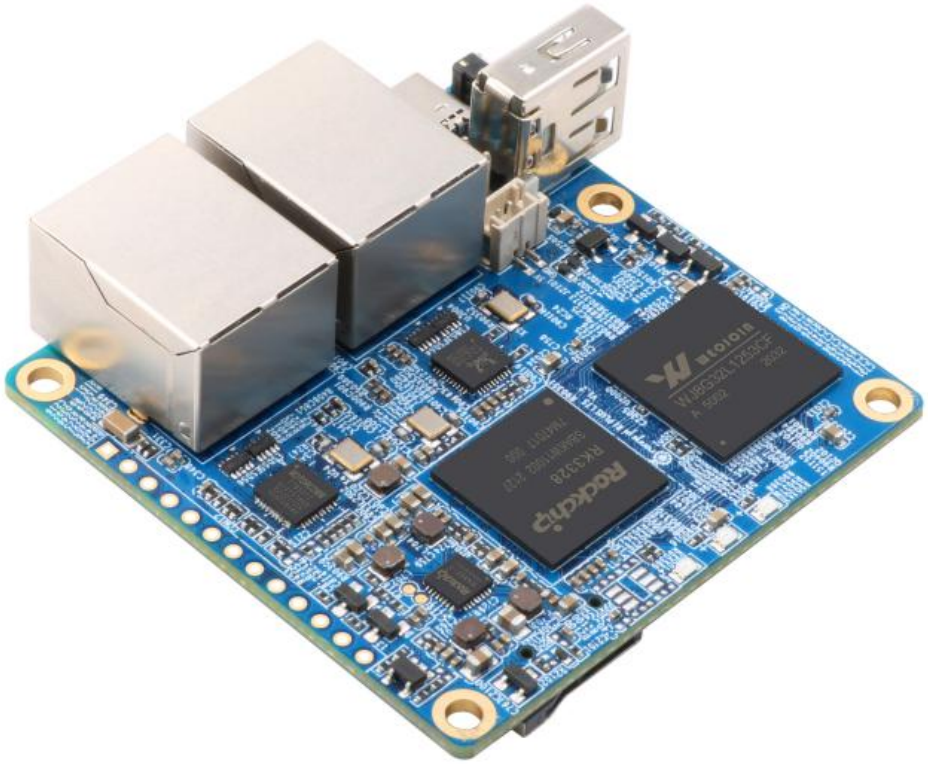
Orange Pi R1 Plus LTS
User Manual



目录

1. Basic features of Orange Pi R1 Plus LTS

1.1. What is Orange Pi R1 Plus LTS?

Orange Pi is an open source single-board computer, a new generation of arm-64 development board, it can run Android 9.0, Ubuntu, Debian and OpenWRT and other operating systems. Orange Pi R1 Plus LTS uses Rockchip rk3328 system-on-chip and has 1GB LPDDR3 memory

1.2. Purpose of Orange Pi R1 Plus LTS?

Typical Application:

- A router
- One switch

Of course there are other more functions, because Orange Pi is open source

1.3. Who is for?

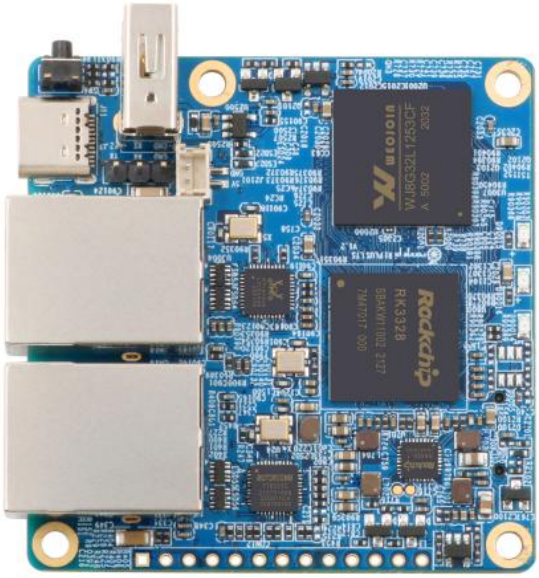
The Orange Pi development board is not only a consumer product, but also designed for anyone who wants to use technology to create and innovate. It is a very simple, interesting and practical tool, you can use it to create the world around you

1. 4. Hardware features of Orange Pi R1 Plus LTS

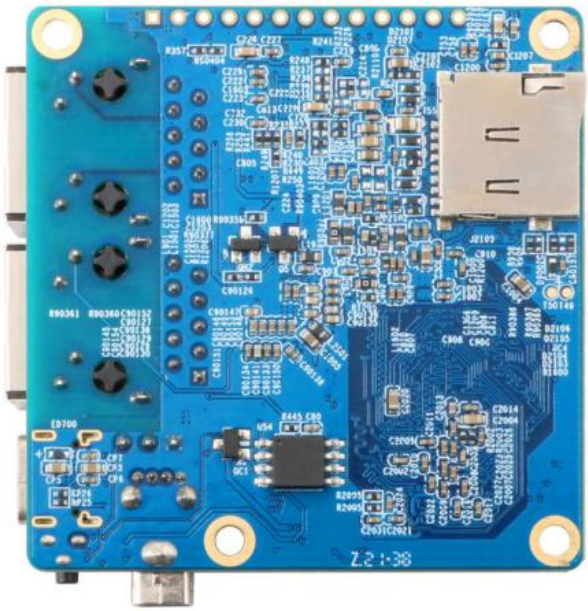
Hardware Specification introduction	
CPU	Rockchip RK3328 Quad-core ARM Cortex-A53 64-bit processor, main frequency speeds up to 1.5GHz
GPU	Mali-450MP2 Supports OpenGL ES 1.0/2.0
Power management chip	RK805
Memory(SDRAM)	1GB LPDDR3（ shared with GPU）
Onboard Storage	• Micro-SD Card slot • 16MB SPI Flash
Onboard Network	10M/100M/1000M Ethernet （YT8531C） 10M/100M/1000M USB Ethernet （RTL8153B）
Video Output	TV CVBS output（Via 13pin interface board）
Audio output	3.5mm audio port（Via 13pin interface board）
Power Source	USB Type C interface 5V2A input
USB	1x USB 2.0 HOST
Low-level peripherals	13pin header with IR pin、Tv-out、AUDIO(no MIC) 2xUSB2.0（not support）and 1 GPIO port
Debug serial port	UART-TX、UART-RX and GND
Button	1x Reset Button
Fan interface	1x cooling fan interface (5V)
LED	Power led & Status led
IR receiver	Support IR remote control (via 13pin interface board)
Supported OS	Android 9、Ubuntu、Debian、OpenWRT
Appearance specification introduction	
Dimension	56mm×57mm
Weight	30.5g

1. 5. Top view and Bottom view of Orange Pi R1 Plus LTS

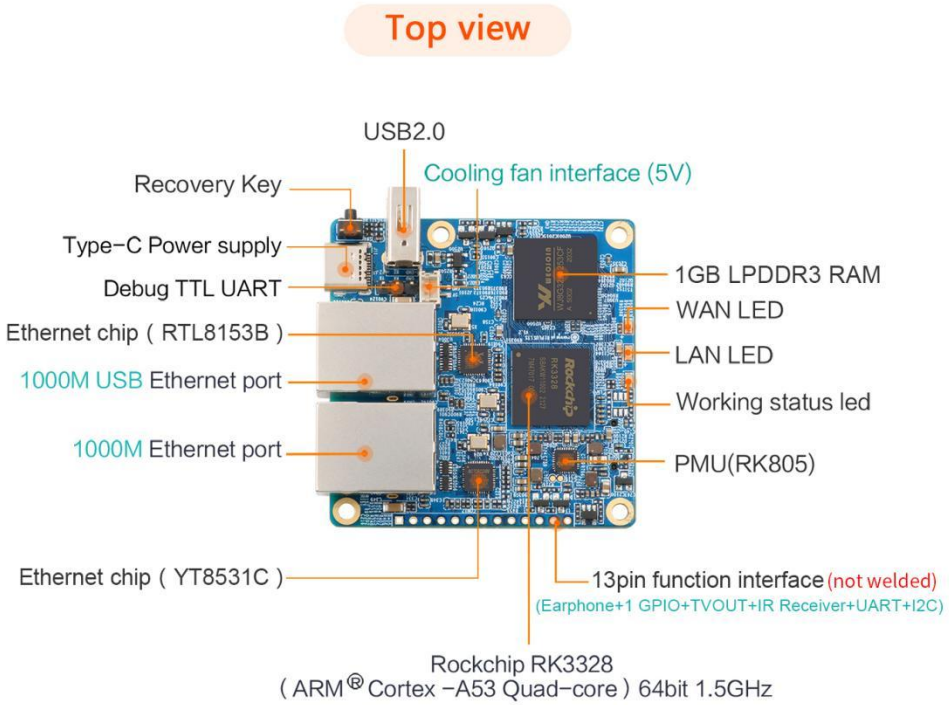
Top view：

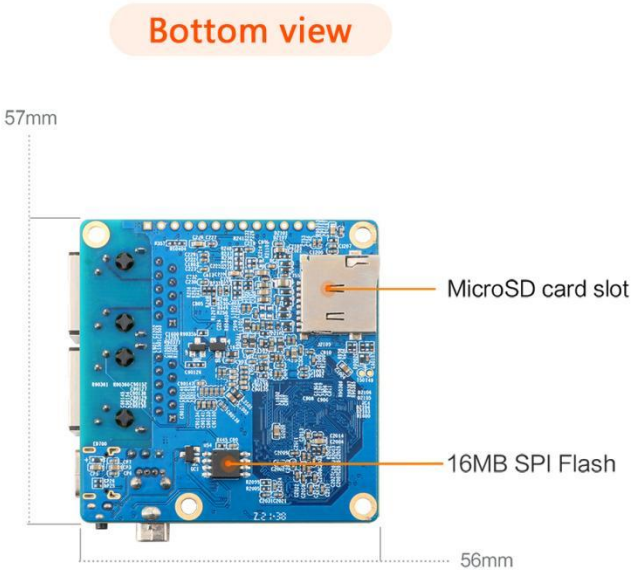


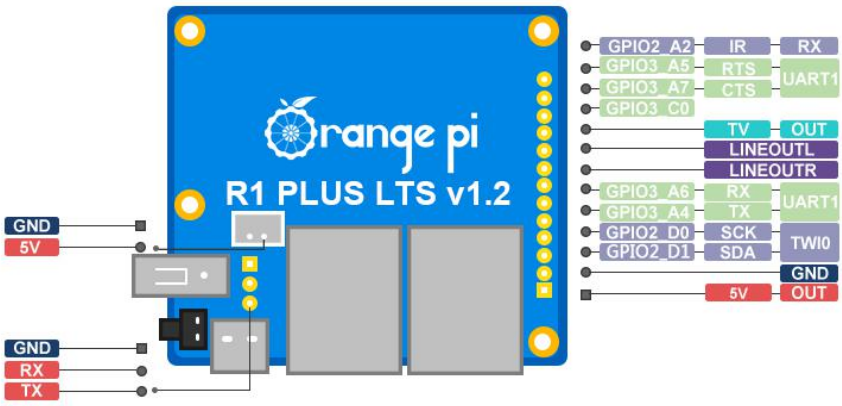
Bottom view：



1. 6. Orange Pi R1 Plus LTS interface details







2. Introduction to the use of the development board

2. 1. Prepare the necessary accessories

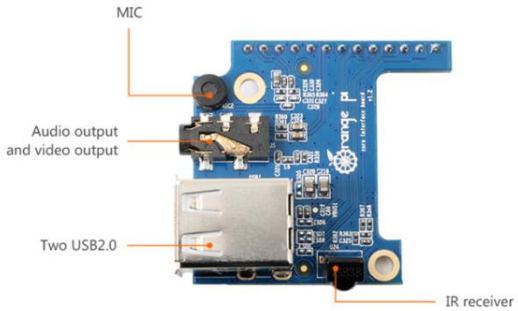
1) TF card, a high-speed card of class 10 or higher with a minimum capacity of 8GB, it is recommended to use SanDisk TF card. Orange Pi tests are all SanDisk TF cards. Other brands of TF cards may cause the system to fail to start.

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

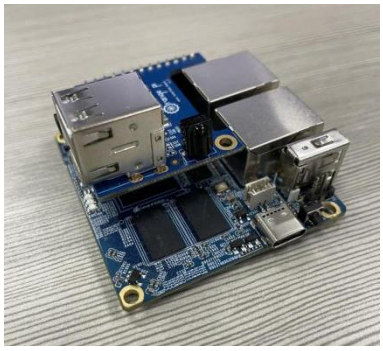
3) Power adapter, 5V/2A or 5V/3A high-quality Type C interface power adapter

4) 13pin interface board

a. The actual interface board is shown below



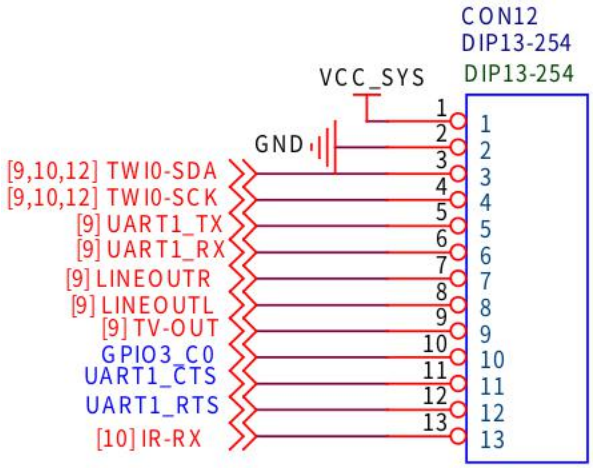
b. The way to insert the interface board into the development board is as follows, remember not to insert it backwards



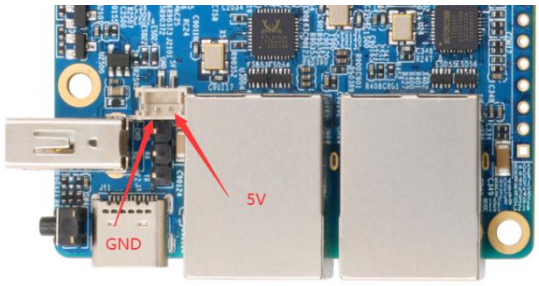
c. The 13pin header on the Orange Pi R1 Plus LTS development board can be connected to the interface board to expand the functions that are not on the development board. The functions included in the interface board are:

1	microphone	**not support**
2	Analog audio and video output interface	Connect TV through AV cable to output analog audio and video signals (Android system only)
3	USB2.0 x 2	**not support**
4	IR receiving function	Android system can be controlled by IR remote control

d.The schematic diagram of the 13pin header of Orange Pi R1 Plus LTS is shown below



5) **CPU fan, used to dissipate heat to the CPU, the interface voltage is 5V, the interface specification is 2pin, 1.5mm spacing**



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

7) 100M or 1000M network cable, used to connect the development board to the Internet

8) AV video cable, used to connect the development board to the TV through the CVBS interface to display video (only available for Android System)

9) USB to TTL module and DuPont cable, when using the serial port debugging function, you need USB to TTL module and DuPont cable 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, Android and OpenWRT source code

2	Windows PC	Used to burn Android, Linux and OpenWRT images

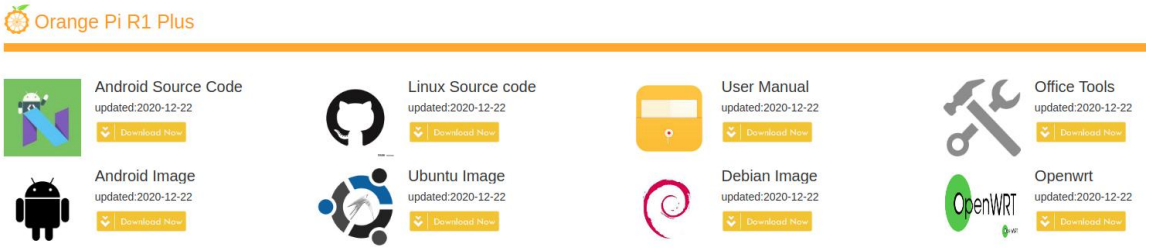
2. 2. Download the image and related information of the development board

1) The download URL of the Chinese version is

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

2) The download URL of the English version is

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



3) The information mainly contains

a. Android source code: saved on Baidu Cloud Disk and Google Cloud Disk

b. Linux source code: saved on github, the link address is

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

c. OpenWRT source code: saved on github, the link address is

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

d. User manual and schematic diagram: chip related data manual will also be placed here

e. Official tools: mainly include the software needed in the use of the development board

f. Android image: saved on Baidu Cloud Disk and Google Cloud Disk

g. Ubuntu image: saved on Baidu Cloud Disk and Google Cloud Disk

h. Debian image: saved on Baidu Cloud Disk and Google Cloud Disk

i. OpenWRT image: saved on Baidu Cloud Disk and Google Cloud Disk

2. 3. Method to burn Linux image or OpenWRT image to TF card based on Windows PC

1) The method of burning OpenWRT image based on Windows PC is the same as the method of burning Linux image. The following is an example of burning Linux image

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

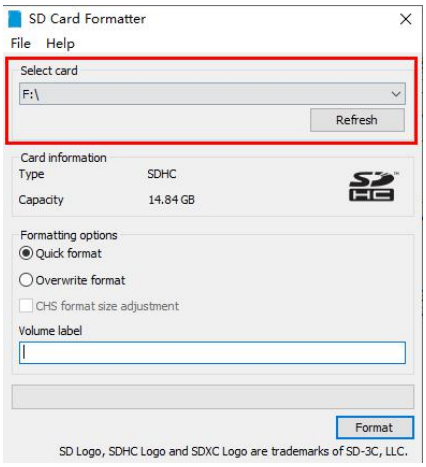
4) Then format the TF card

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

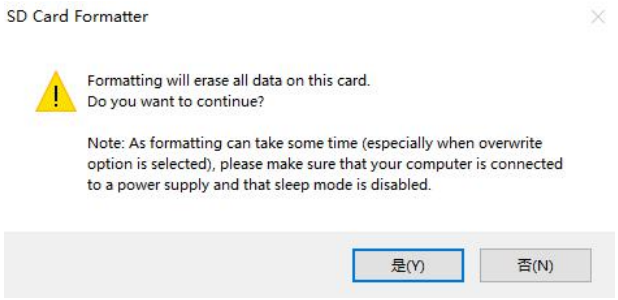
https://www.sdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip

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

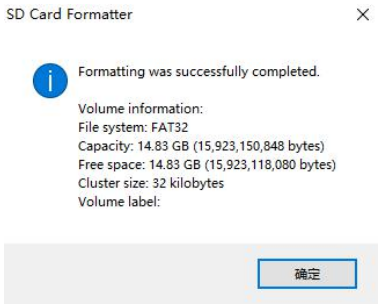
c. If the computer only has a TF card inserted, the “Select card” column will display the drive letter of the TF card. If multiple USB storage devices are inserted into the computer, you can select the drive letter corresponding to the TF card through the drop-down box



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



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



5) Download the Linux operating system image file compression package that 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

6) Use Win32Diskimager to burn Linux image to TF card

a. The download page of Win32Diskimager is

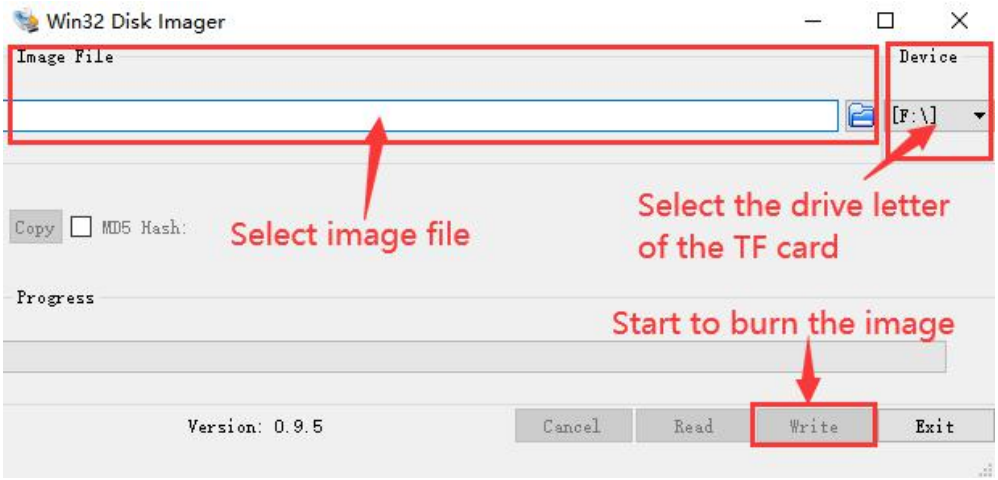
http://sourceforge.net/projects/win32diskimager/files/Archive/

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, then you can pull out the TF card and insert it into the development board to start

2. 4. Method of flashing Linux image or OpenWRT image to TF card based on Ubuntu PC

1) The method of flashing OpenWRT image based on Ubuntu PC is the same as that of flashing Linux image. The following is an example of flashing Linux image

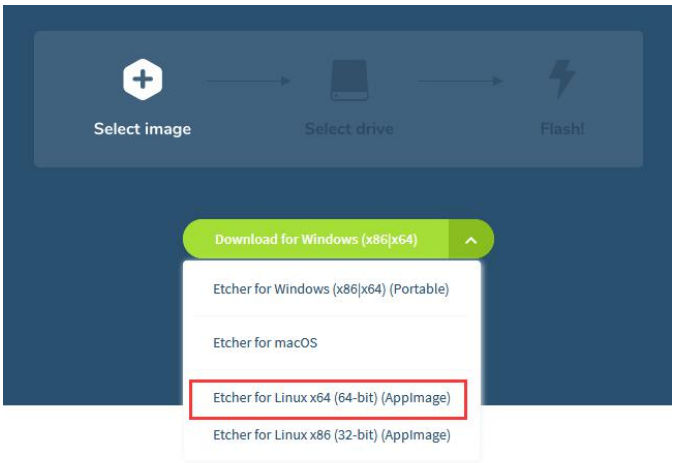
2) First prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

4) Download balenaEtcher software, the download address is

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

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



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

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

7) Download the Linux operating system image file compression package you want to burn from Orange Pi's 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 7z is as follows

```
test@test:~$ 7z x Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.7z
test@test:~$ ls Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.*
Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.7z
Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.img.sha  #Checksum file
Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.img  # image file
```

8) After decompressing the image, you can first use the sha256sum -c *.sha command to calculate whether the checksum is correct. If it is prompted that the downloaded image is correct, you can safely burn it to the TF card. If the checksum does not match, it indicates There is a problem with the downloaded image, please try to download again

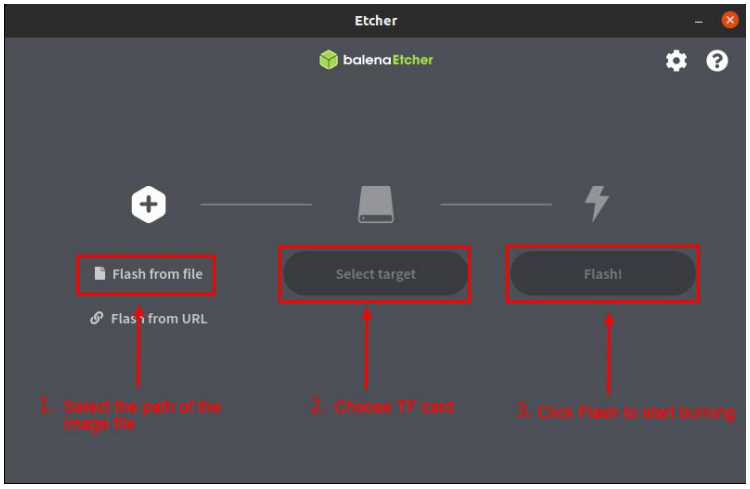
```
test@test:~$ sha256sum -c *.sha
Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.img: Success
```

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

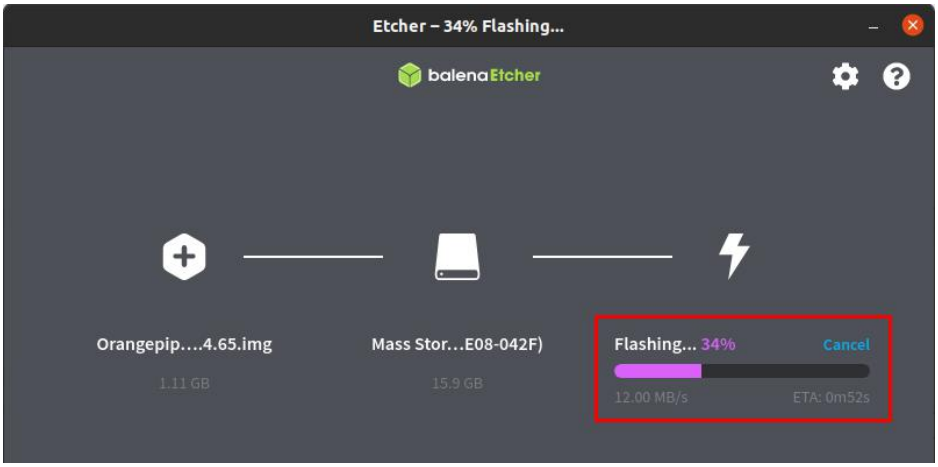
a. First select the path of the linux image file

b. Then select the device number of the TF card

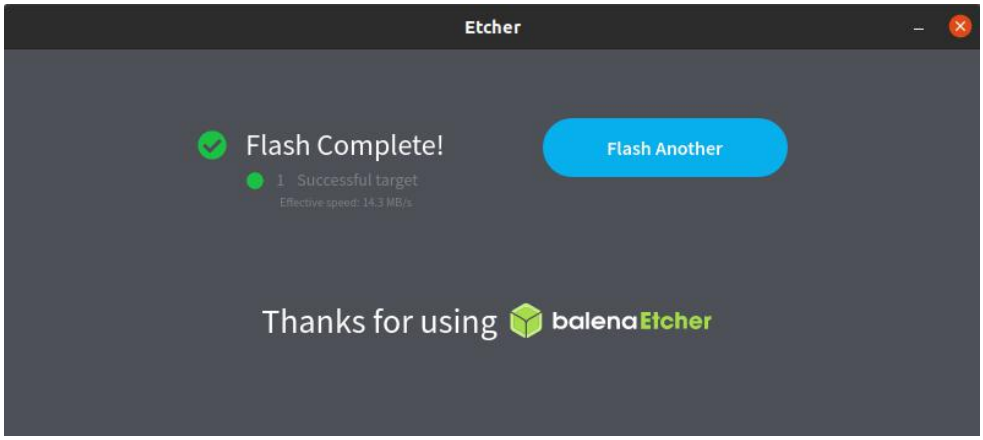
c. Finally click Flash to start burning



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



11) 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. 5. Method of flashing Android firmware to TF card

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

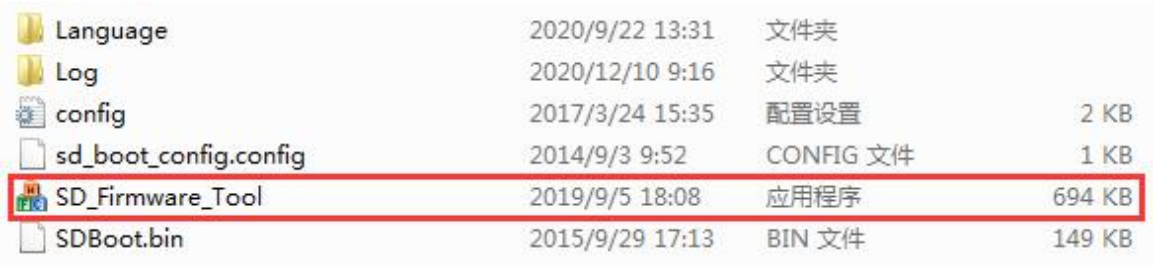
1) First prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

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

3) Download Android 9.0 firmware and SDDiskTool burning tool from Orange Pi's data download page

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

5) Use decompression software to decompress SDDiskTool_v1.59, this software does not need to be installed, find SD_Firmware_Tool in the decompressed folder and open it



6) After opening SD_Firmware_Tool, if the TF card is recognized normally, the name and capacity of the TF card will be displayed when the removable disk device is selected. Please make sure that the displayed TF card device information is consistent with the device information of the TF card you want to burn. There is no display, you can try to unplug the TF card

7) After confirming the device information, start writing the Android firmware to the TF card

a. First check "SD start" in the "Select function mode" column

b. Select the path of Android image in "Select Firmware Upgrade"

c. Then click the "Start Create" button

a. After clicking "Start to create", a warning box will pop up, select "Yes (Y)" to start burning

b. After starting to burn, the partition currently being burned will be displayed below

8) After burning, the display of SD_Firmware_Tool is as shown in the figure below. At this time, click the OK button to close SD_Firmware_Tool, and then you can unplug the TF card from the computer and insert it into the development board to start

2. 6. Start the Orange Pi development board

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

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

3) Connect a 5V/2A (5V/3A is also available) high-quality power adapter

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

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

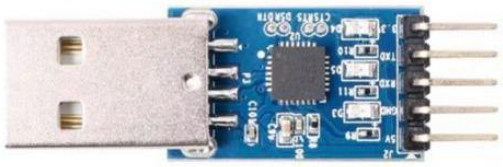
2) If you want to view the output information of the system through the debug serial port, please use the USB to TTL module and DuPont cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section on the use of the debug serial port

3) Then turn on the switch of the power adapter, if everything is normal, the serial terminal can see the output log of the system startup at this time

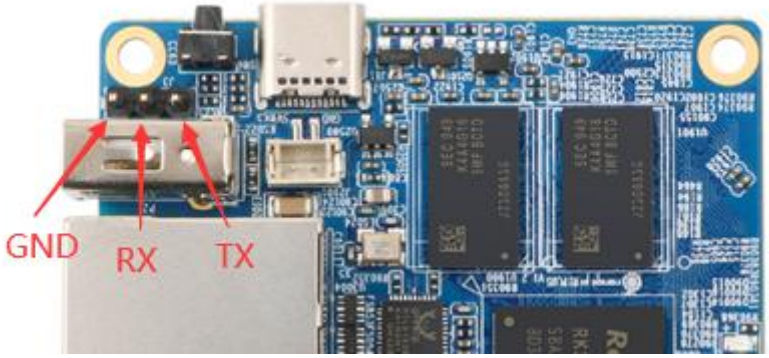
2. 7. How to use the debug serial port?

2. 7. 1. Debug serial port connection instructions

1) First, you need to prepare a USB to TTL module. This module can be bought in Orange Pi Store. If there are other similar USB to TTL modules, you can also insert the USB port of the USB to TTL module into the USB of the computer Interface



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



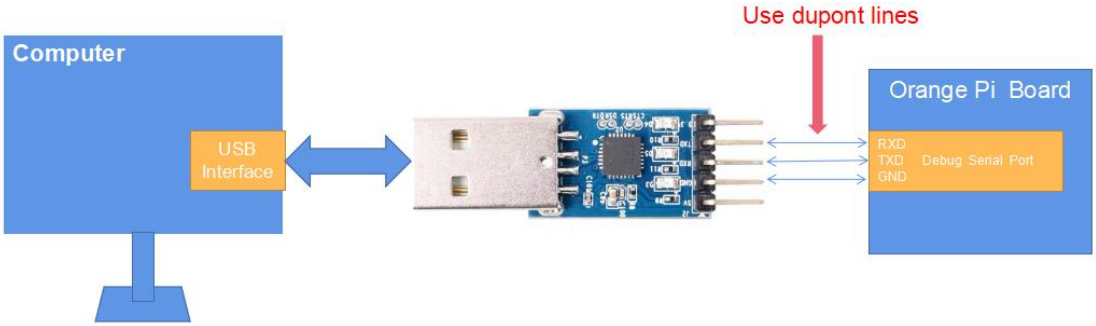
3) The GND, TX and RX pins of the USB to TTL module need to be connected to the debug serial port of the development board through a Dupont cable

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

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

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

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



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

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

USB to TTL module model	Host system	Support situation
CH340	win7	ok
	win10	ok
	ubuntu14.04	ok
	ubuntu18.04	ok
	ubuntu20.04	ok

CP2102	win7	ok
	win10	Not available
	ubuntu14.04	ok
	ubuntu18.04	Not available
	ubuntu20.04	Not available

2.7.2. How to use the debug serial port on 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

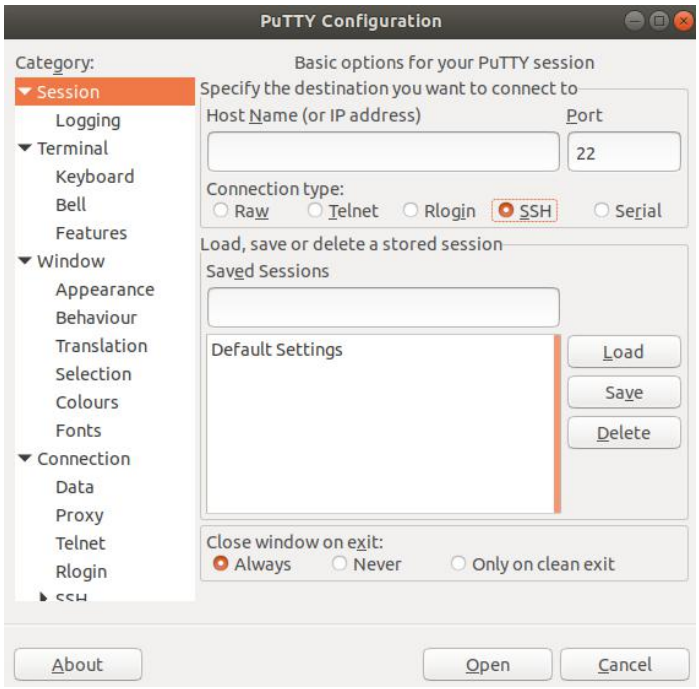
3) First install putty on Ubuntu PC

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

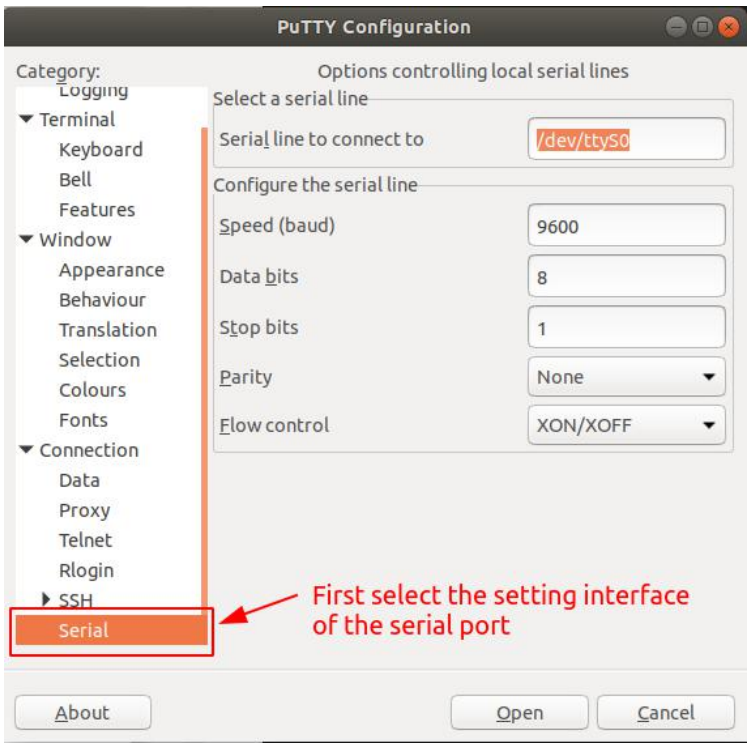
4) Then run putty, remember to add sudo permissions

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

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



6) First select the setting interface of the serial port



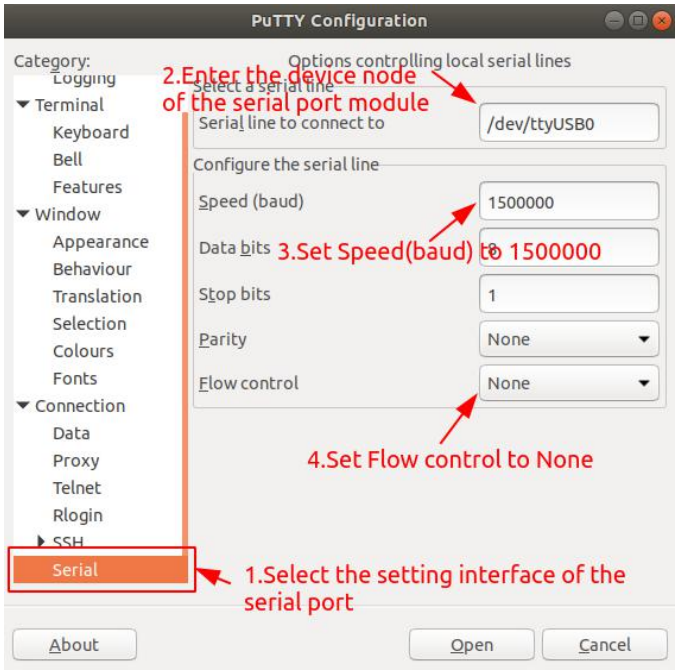
7) Then set the parameters of the serial port

a. Set the Serial line to connect to to /dev/ttyUSB0 (modify to the corresponding

node name, generally /dev/ttyUSB0)

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

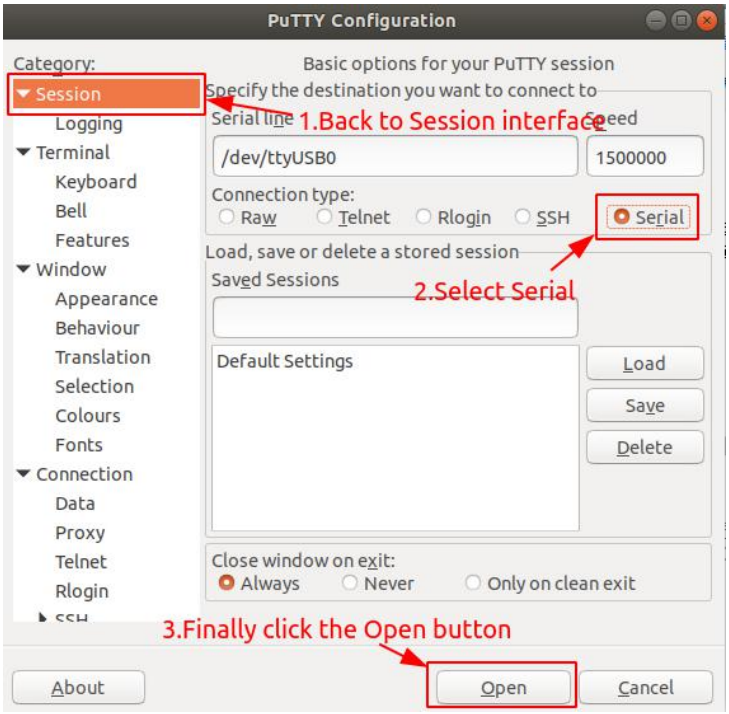
c. Set Flow control to None



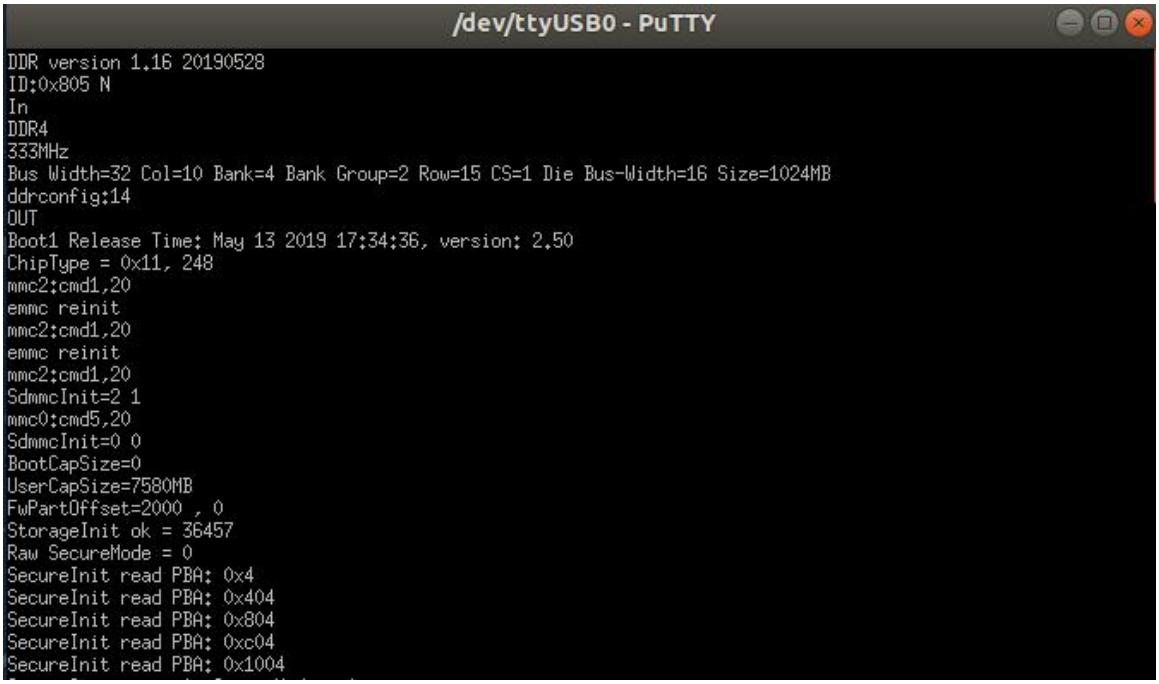
8) After setting the serial port setting interface, return to the Session interface

a. First select the Connection type as Serial

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



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



2. 7. 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

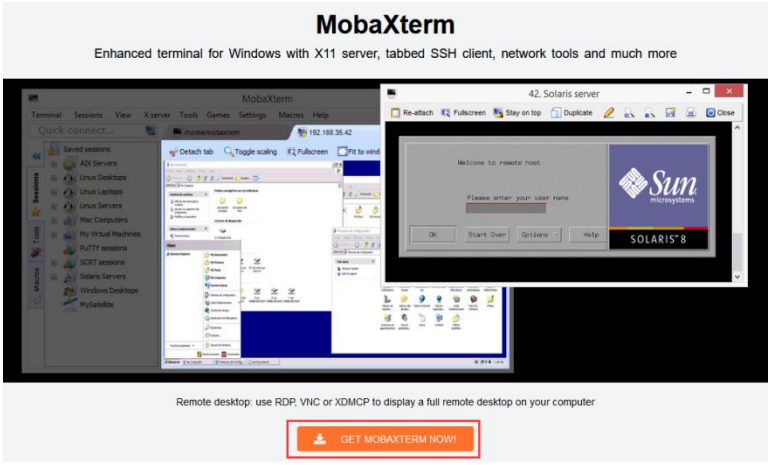
SecureCRT, MobaXterm, etc. The following demonstrates how to use MobaXterm. This software has a free version and can be used without purchasing a serial number.

2) Download MobaXterm

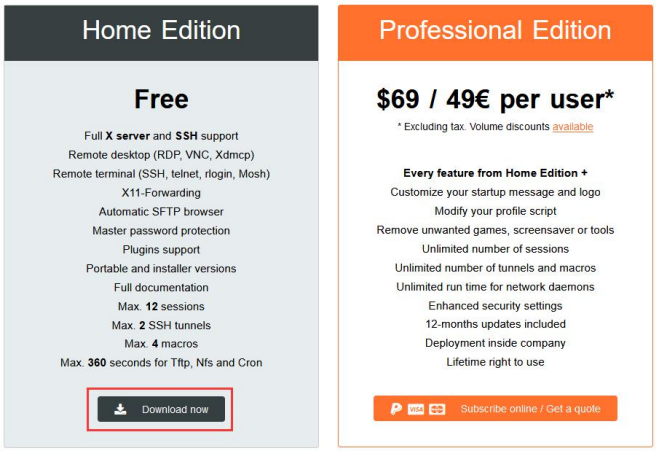
a. Download MobaXterm URL as follows

https://mobaxterm.mobatek.net/

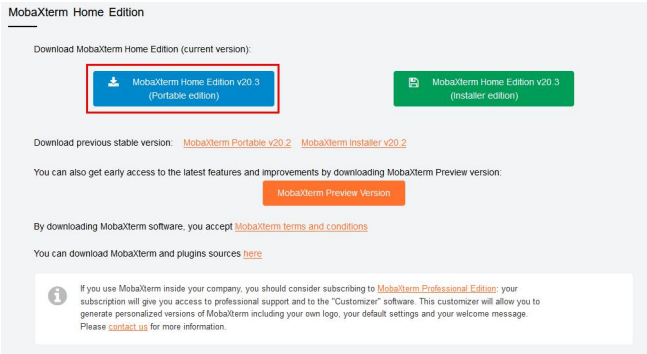
b. After entering the MobaXterm download page, click **GET XOBATERM NOW!**



c. Then choose to download the Home version



d. Then select the Portable version, after downloading, you don’t need to install it, 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 the serial connection are as follows

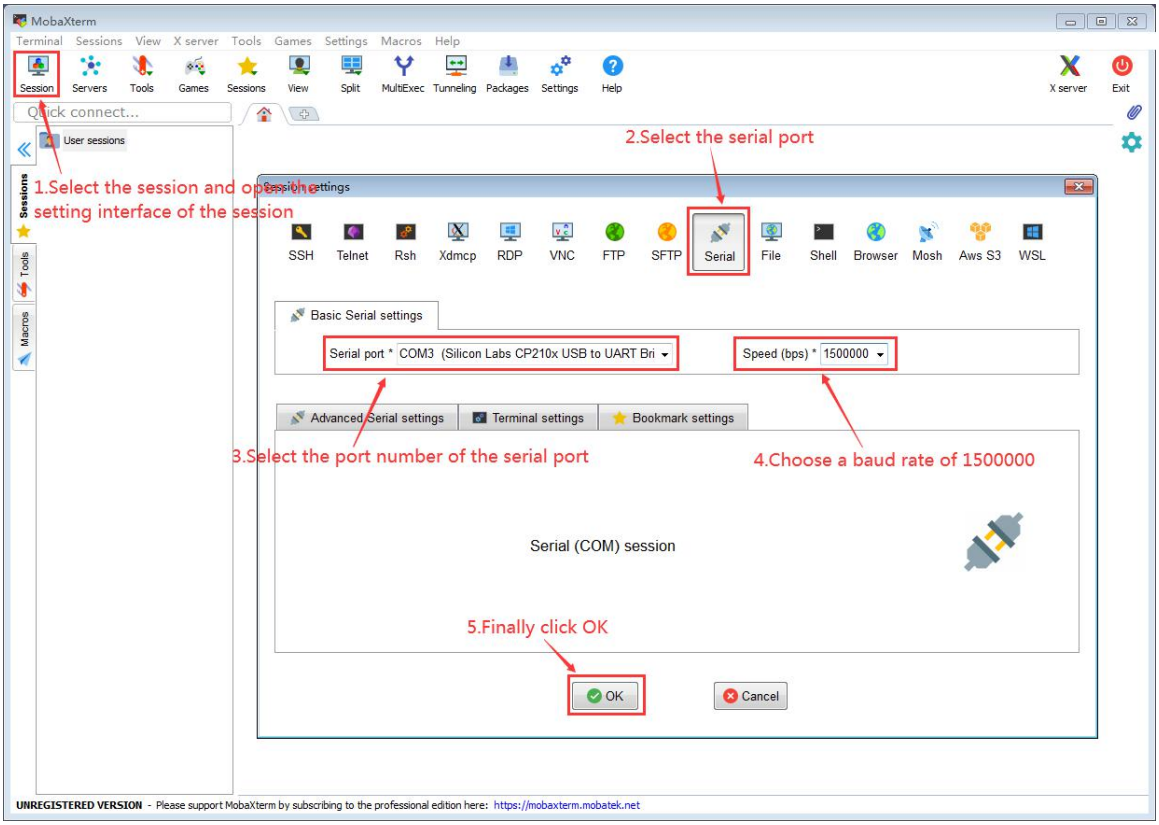
a. Open setting interface of the session

b. Select the serial port

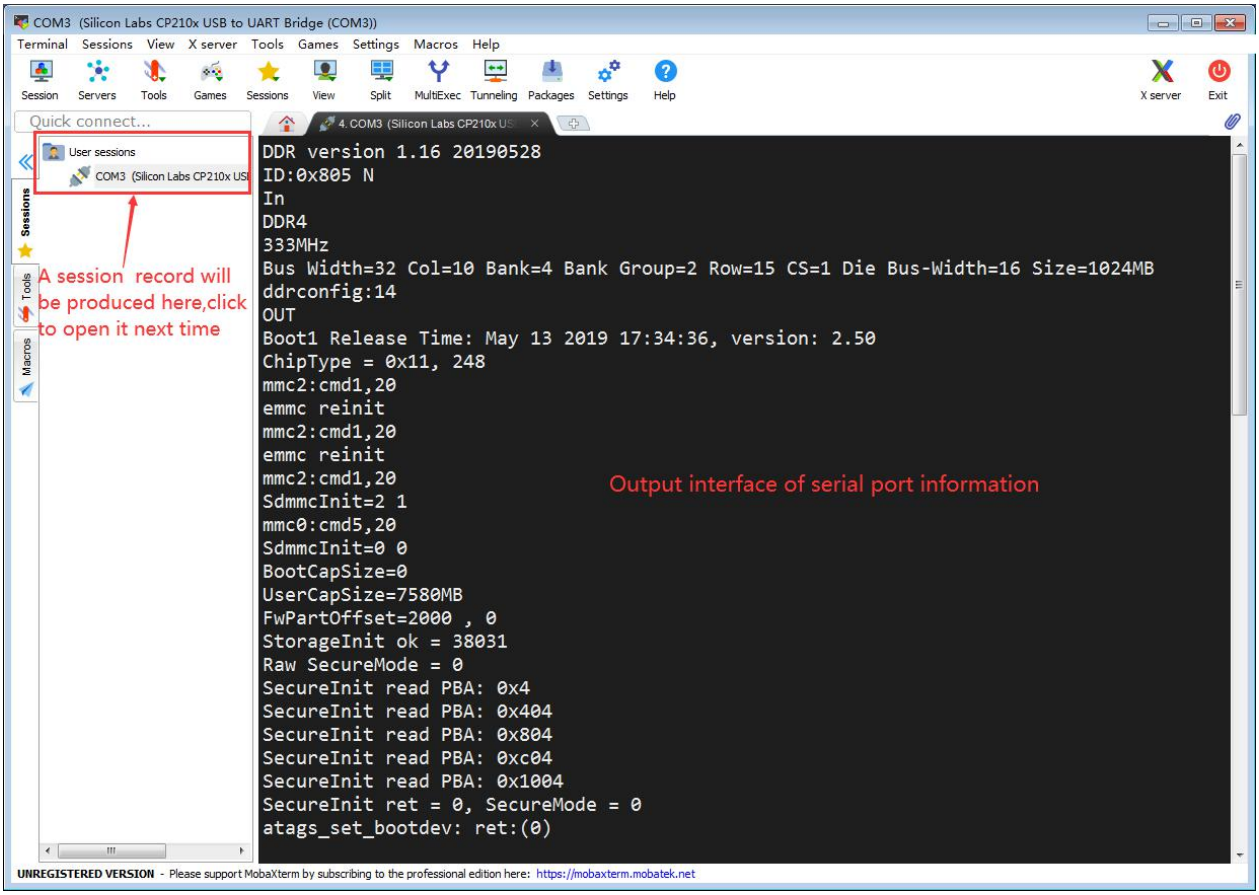
c. Select the port number of the serial port (choose the corresponding port number according to the actual situation), if you can't see the port number, please use the "360 driver master" to scan and install the USB to TTL serial chip driver

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

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



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



3. OpenWRT system instructions

3. 1. OpenWRT version

1）There are two branches of openwrt code on Github

OpenWRT version	Kernel version
OpenWRT openwrt-21.02 branch	Linux5.4
OpenWRT master branch	Linux5.10

2）Openwrt-21.02 is developed based on OpenWrt v21.02.1 Release version, and the function tends to be stable

3）The master branch is based on the snapshot version of openwrt, which is an unstable and under development version

3. 2. OpenWRT system default login account and password

It is recommended to change a safer password for web login and ssh login before use

Account	password
root	password

3. 2. 1. Modify root password

For example, to change the root password to 12345678, enter passwd root on the command line

```
root@OpenWrt:/# passwd root
Enter new UNIX password:
```

Follow the prompts to change the password

3. 3. Expand the rootfs in the TF card before the first boot

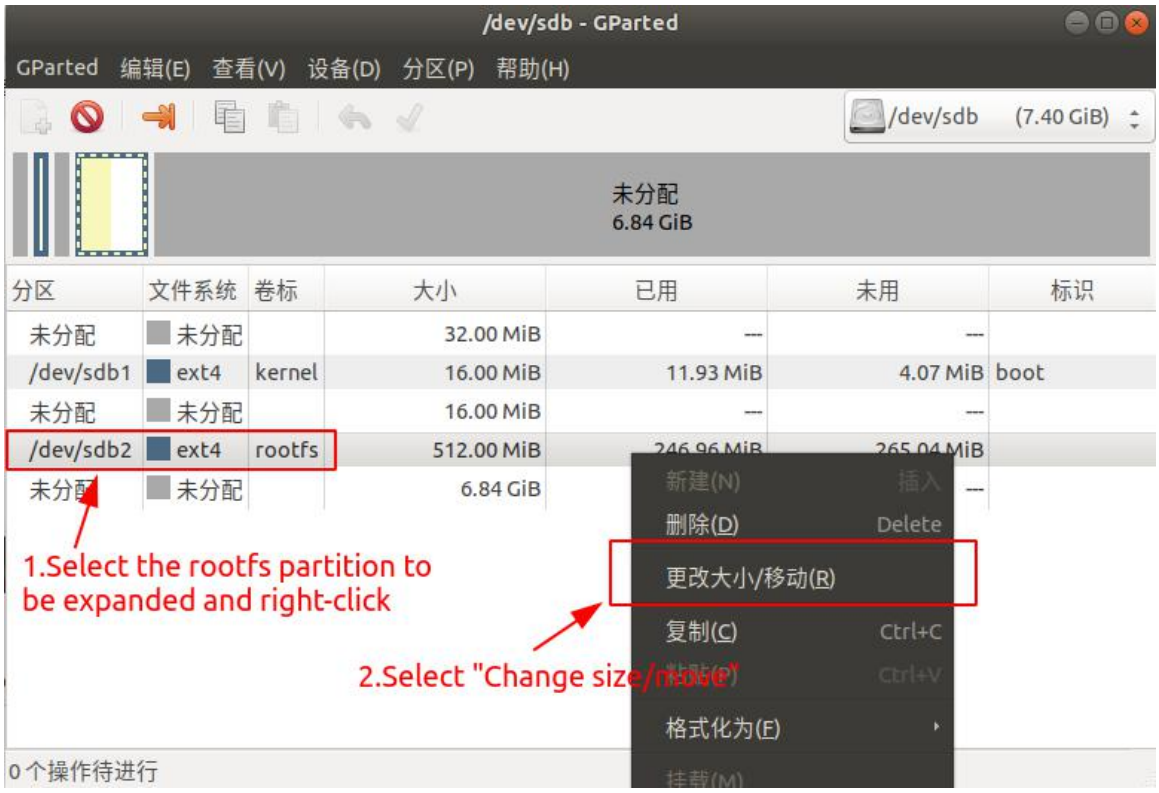
1) After burning the OpenWRT image, you need to manually expand the system rootfs to use the full space of the TF card

2) First install gparted on Ubuntu PC

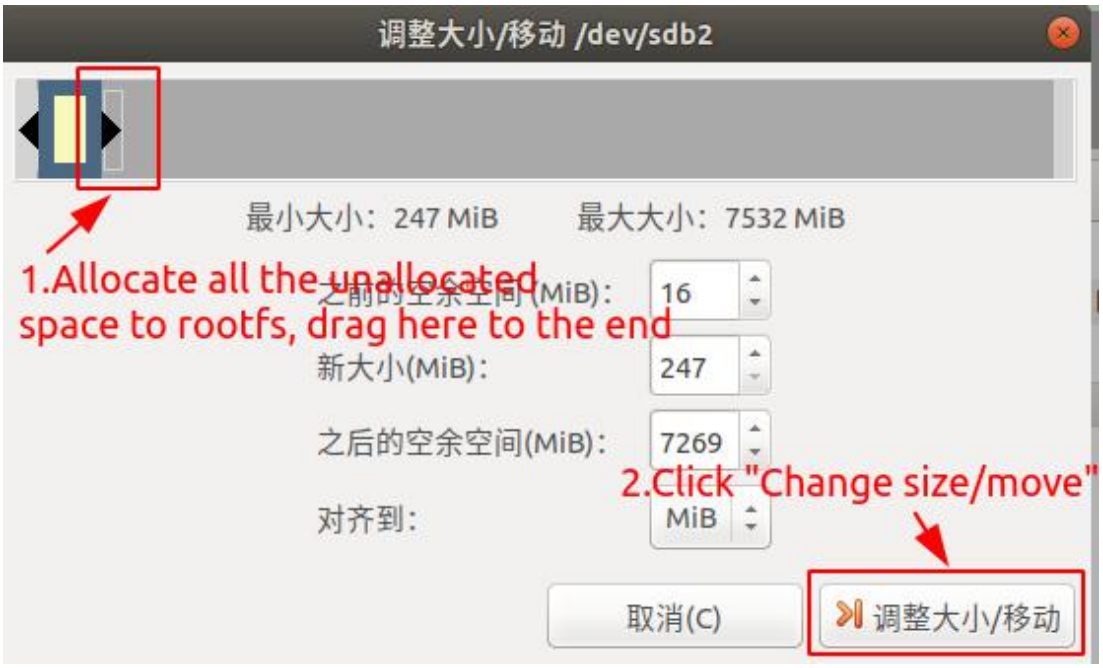
```
test@test:~$ sudo apt update
test@test:~$ sudo apt install gparted
```

3) Use a card reader to insert the TF card that has been burned with the OpenWRT image into the computer, and open gparted, select your TF card in the upper right corner, usually /dev/sdb

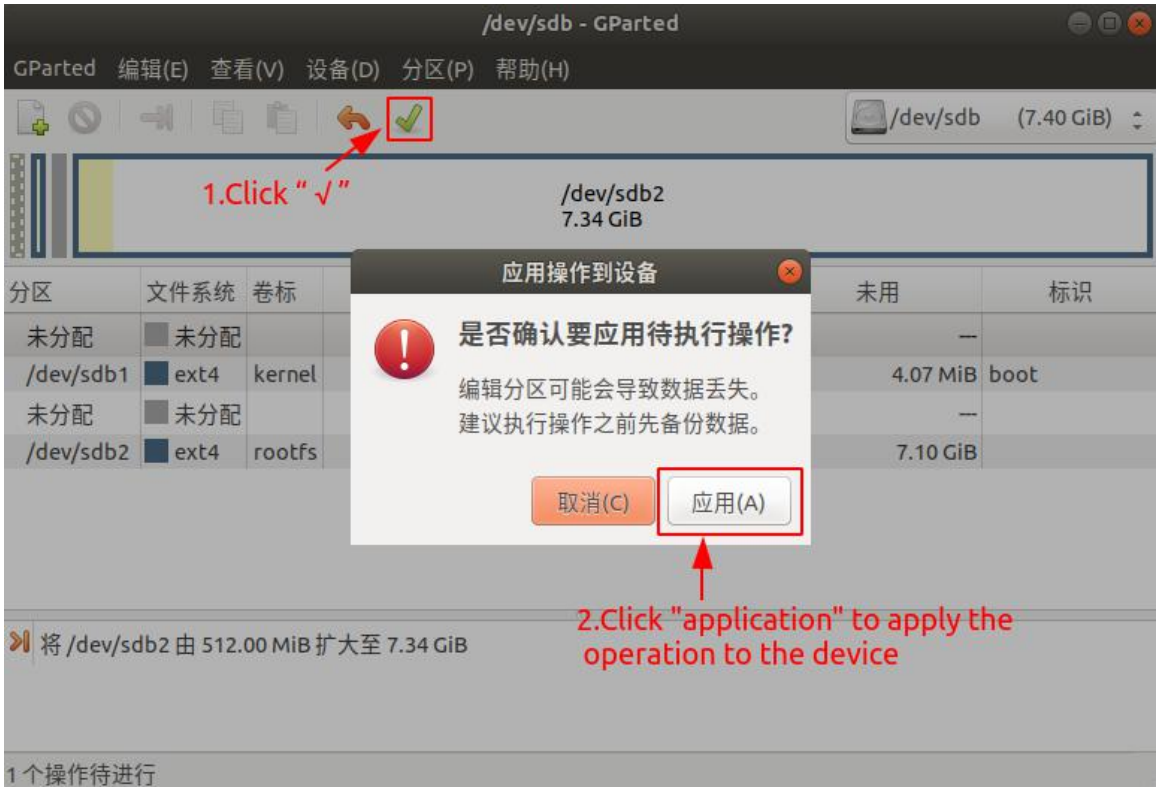
4) Select the rootfs partition to be expanded and Right-click, and select "Resize/Move"



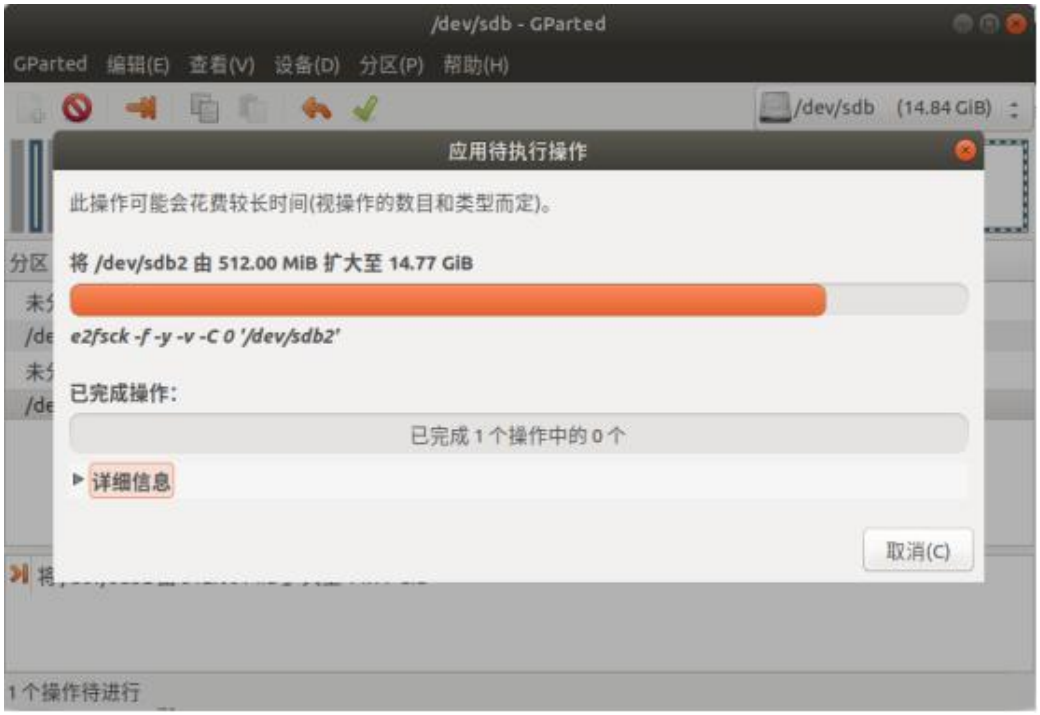
5) Allocate all the unallocated space to rootfs, drag here to the end, and then click "Resize/Move"



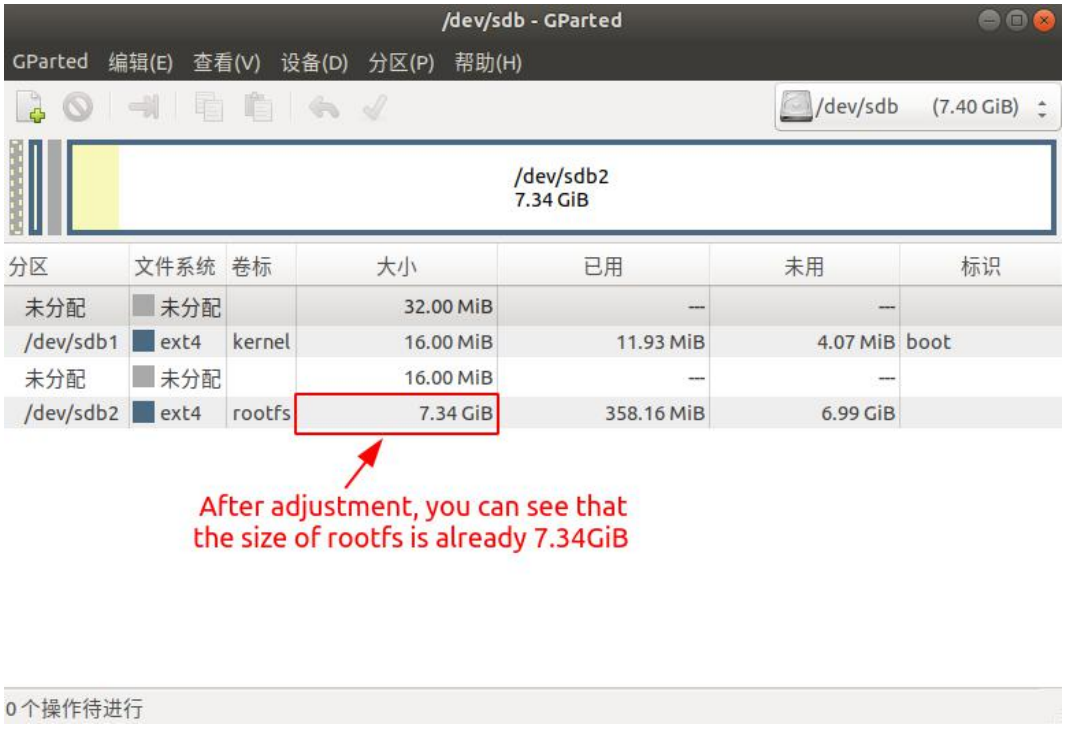
6) Click the "√" above, and click "application" to apply the operation to the device



7) The expansion process generally does not take long



8) After the expansion is completed, you can see that the rootfs size is the actual capacity of the TF card



9) After the system starts, execute the df -h command to check the size of rootfs. If it is

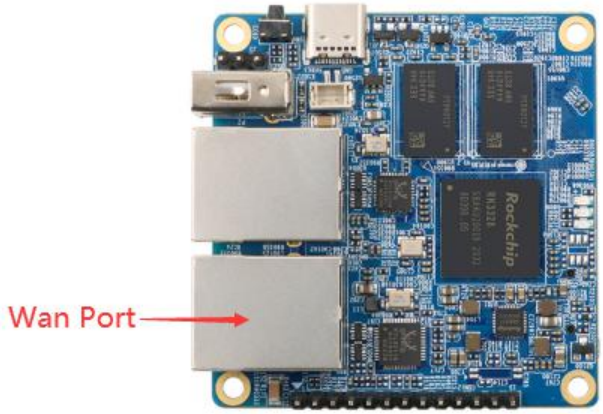
consistent with the actual capacity of the TF card, it can also indicate that the expansion is successful

```
rroot@OpenWrt:/# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                14.5G    238.7M     14.3G   2% /
tmpfs                   495.6M      2.8M    492.8M   1% /tmp
tmpfs                   512.0K         0    512.0K   0% /dev
cgroup                  495.6M         0    495.6M   0% /sys/fs/cgroup
```

3. 4. Ethernet port test

3. 4. 1. Wan port test

1) First, insert the network cable into the usb to 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@OpenWrt:/# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr C0:74:2B:FF:B3:41
          inet addr:192.168.1.87  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::c274:2bff:feff:b341/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1634 errors:0 dropped:84 overruns:0 frame:0
          TX packets:59 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

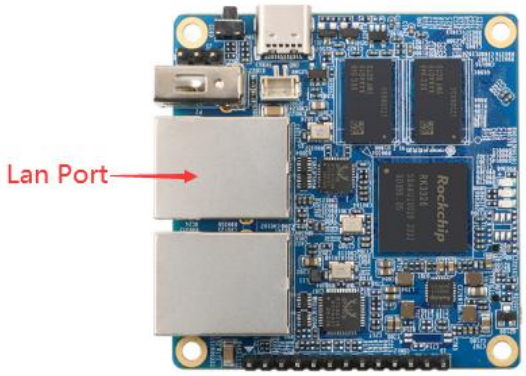
```
          RX bytes:142098 (138.7 KiB)    TX bytes:13503 (13.1 KiB)
          Interrupt:27
```

4) The command to test network connectivity is as follows

```
root@OpenWrt:/# ping www.baidu.com -I eth0
PING www.baidu.com (14.215.177.38): 56 data bytes
64 bytes from 14.215.177.38: seq=0 ttl=56 time=6.169 ms
64 bytes from 14.215.177.38: seq=1 ttl=56 time=5.473 ms
64 bytes from 14.215.177.38: seq=2 ttl=56 time=5.114 ms
64 bytes from 14.215.177.38: seq=3 ttl=56 time=5.992 ms
^C
--- www.baidu.com ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 5.114/5.693/6.169 ms
```

3.4.2. Lan port test

1) First connect the network cable to the Lan port of the development board and the Ubuntu PC



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

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

```
test@ubuntu:~# ifconfig enp5s0
eenp5s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
          inet 192.168.2.174  netmask 255.255.255.0  broadcast 192.168.2.255
          inet6 fd50:32f6:413:0:8050:834c:16d4:d493 prefixlen 64 scopeid 0x0<global>
          inet6 fe80::686:bb73:c7ae:9e9a  prefixlen 64  scopeid 0x20<link>
```

```
        inet6 fd50:32f6:413::d00  prefixlen 128  scopeid 0x0<global>
        inet6 fd50:32f6:413:0:6bbf:bc59:16e2:1f76 prefixlen 64 scopeid 0x0<global>
        ether 40:b0:76:60:17:c3  txqueuelen 1000  (以太网)
        RX packets 4331701  bytes 2941416494 (2.9 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13649801  bytes 5762726379 (5.7 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4) The command to test network connectivity is as follows

```
test@ubuntu:~# ping 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=64 time=0.439 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=64 time=0.455 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=64 time=0.390 ms
64 bytes from 192.168.2.1: icmp_seq=4 ttl=64 time=0.473 ms
^C
--- 192.168.2.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3066ms
rtt min/avg/max/mdev = 0.390/0.439/0.473/0.034 ms
```

5) At this time, if the Wan port is connected to the Internet, the Ubuntu PC can also be connected to the Internet

```
test@ubuntu:~# ping www.baidu.com
PING www.a.shifen.com (14.215.177.38) 56(84) bytes of data.
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=55 time=7.32 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=55 time=7.72 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=55 time=7.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=55 time=7.05 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 7.055/7.478/7.808/0.305 ms
```

3.5. SSH remote login to the development board

The OpenWRT system has SSH remote login enabled by default, and allows root users to log in to the system. Before ssh login, you need to make sure that the Ethernet is connected, and then use the ifconfig command or check the router to get the IP address of the development board

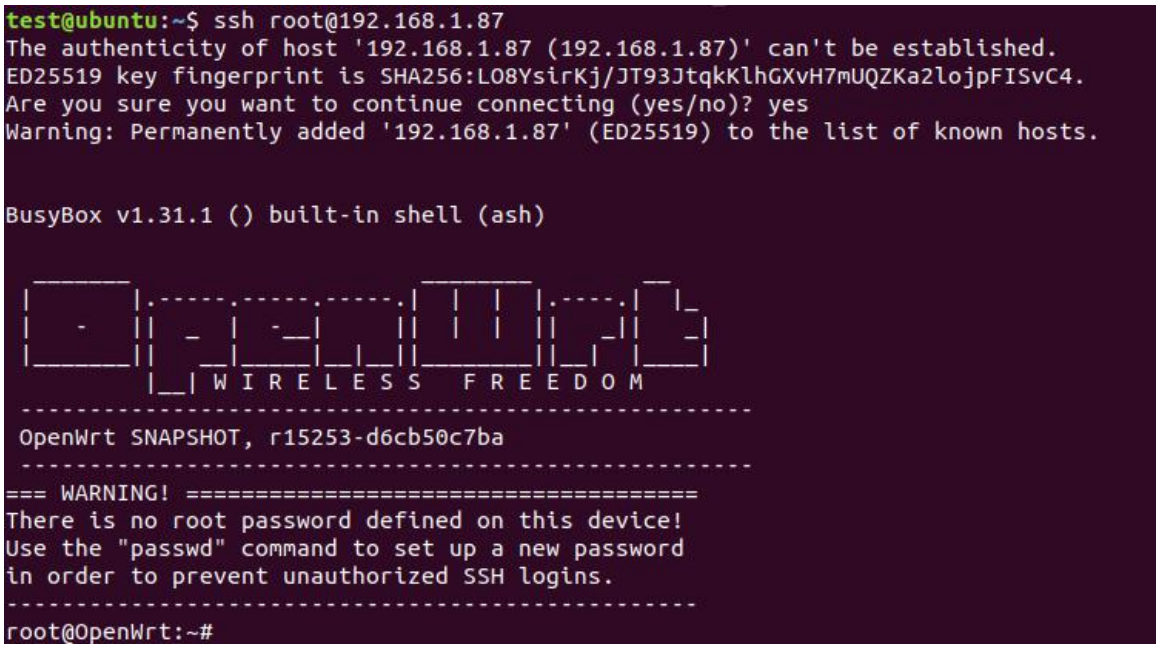
3.5.1. SSH remote login development board under Ubuntu

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@192.168.1.87** (Need to be replaced with the IP address of the development board)

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



3.5.2. SSH remote login development board under Windows

1) First get the IP address of the development board

2) MobaXterm can be used to remotely log in to the development board under windows, first create a new ssh session

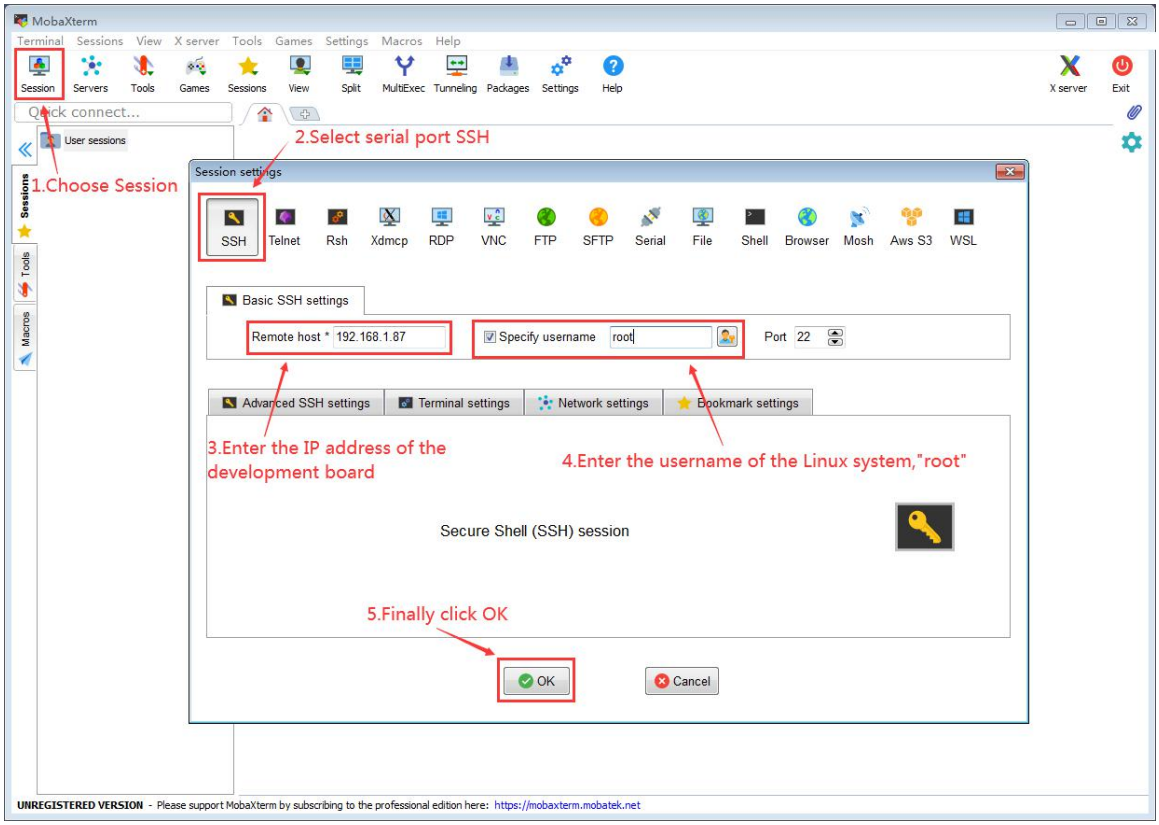
a. Choose Session

b. Then select serial port SSH

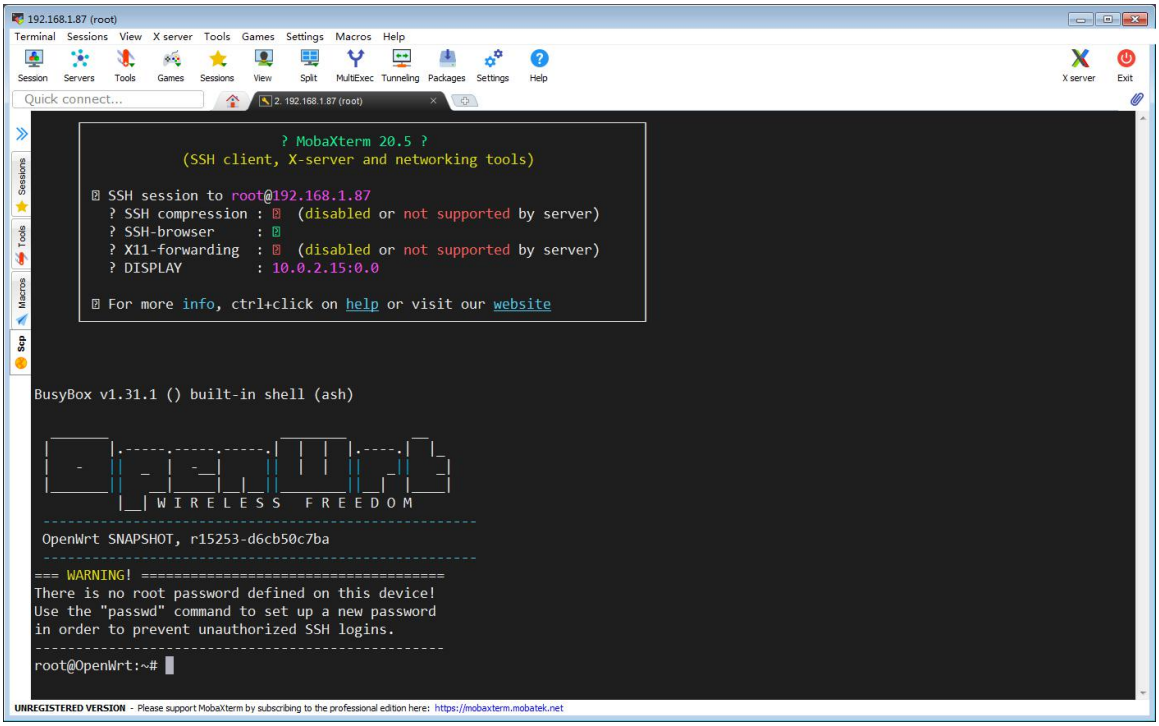
c. Then enter the IP address of the development board in Remote host

d. Then enter the username of the Linux system, “root”

e. Finally click OK



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



3. 6. USB interface test

3. 6. 1. 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 is successfully recognized

```
root@OpenWrt:~# cat /proc/partitions | grep "sd*"
major minor  #blocks  name
   8        0   15126528 sda
```

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

```
root@OpenWrt:~# mount /dev/sda /mnt/
root@OpenWrt:~# 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@OpenWrt:~# df -h | grep "sd"
/dev/sda                14.4G       187.2M        14.2G   1% /mnt
```

3. 7. Onboard LED light test instructions

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

Power status light (red light)	The system starts, the red light is always on
Wan port status light (yellow light)	Wan port is connected to the network cable, the yellow light flashes, the Wan port is unplugged, the yellow light is off
Lan port status light (yellow light)	Lan port is connected to the network cable, the yellow light flashes, the Lan port is unplugged, the yellow light is off

2) The method of setting the red/yellow light on and off and flashing is as follows (the red light is taken as an example below)

a. First enter the red light setting directory

```
root@OpenWrt:~# cd /sys/class/leds/orangepi-r1plus:red:sys
```

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

```
root@OpenWrt:/sys/class/leds/orangepi-r1plus:red:sys# echo 0 > brightness
```

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

```
root@OpenWrt:/sys/class/leds/orangepi-r1plus:red:sys# echo 1 > brightness
```

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

```
root@OpenWrt:/sys/class/leds/orangepi-r1plus:red:sys# echo heartbeat > trigger
```

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

```
root@OpenWrt:/sys/class/leds/orangepi-r1plus:red:sys# echo none > trigger
```

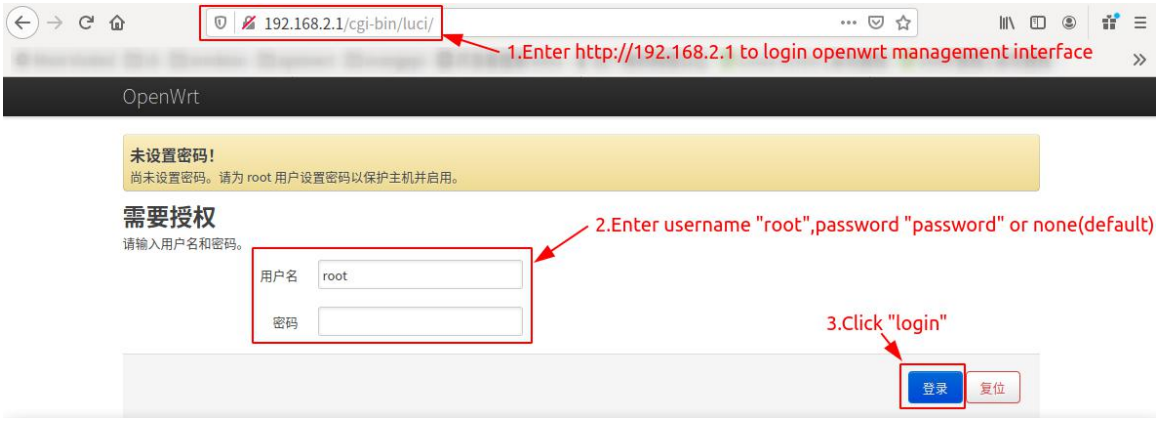
3. 8. Log in to the OpenWRT management page

3. 8. 1. Log in through the lan port

1) Connect the WAN port of OrangePi R1 Plus to your main router with a network cable

2) Connect the computer to the LAN port of OrangePi R1 Plus with a network cable,

enter http://192.168.2.1 to login OpenWRT management interface, and enter the username “root”and password “password” or none(default), and then click “login”



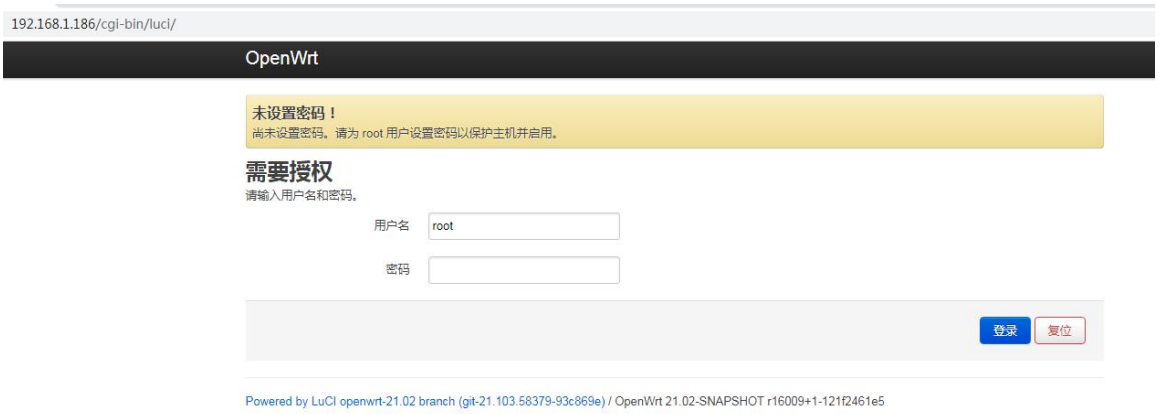
3. 8. 2. Log in through the wan port

1) Connect the WAN port of OrangePi R1 Plus to your main router with a network cable

2) Connect the WAN port of the computer to your main router with a network cable

3) Use the ifconfig command or check the router to obtain the IP address of OrangePi R1 Plus

4) Enter the IP address of OrangePi R1 Plus in the computer browser to enter the OpenWRT management page



3. 9. Install packages

There are two ways to install the software package on the OpenWRT system

3. 9. 1. Install via opkg in the terminal

1) Update the list of available packages

```
root@OpenWrt:/# opkg update
```

2) Get the software list

```
root@OpenWrt:/# opkg list
```

3) Install the specified software package

```
root@OpenWrt:/# opkg install < Package name >
```

4) View installed software

```
root@OpenWrt:/# opkg list-installed
```

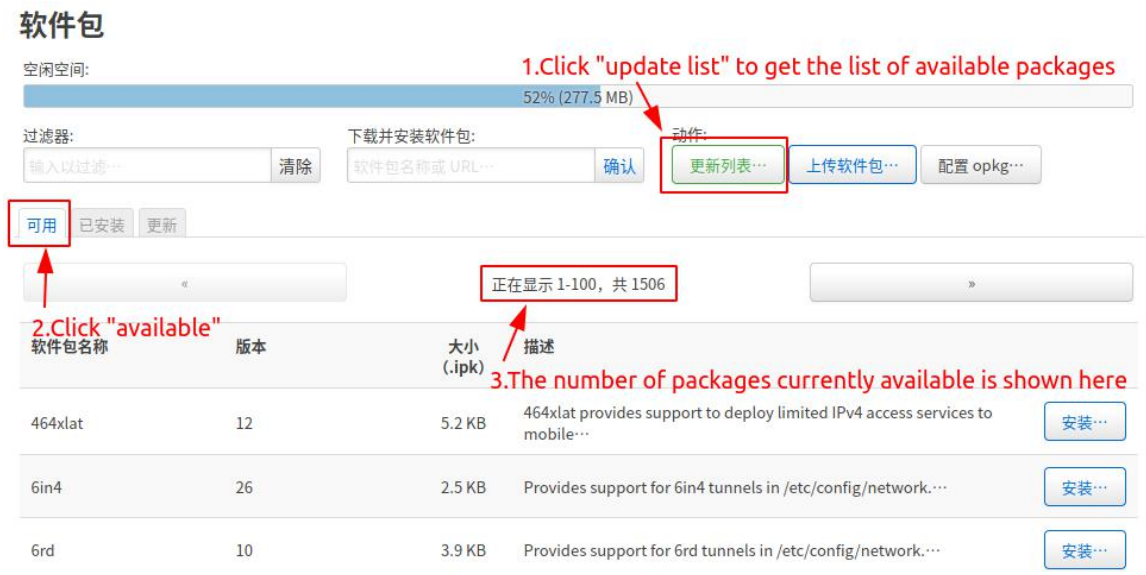
5) Uninstall software

```
root@OpenWrt:/# opkg remove < Package name >
```

3. 9. 2. Install on the package management page

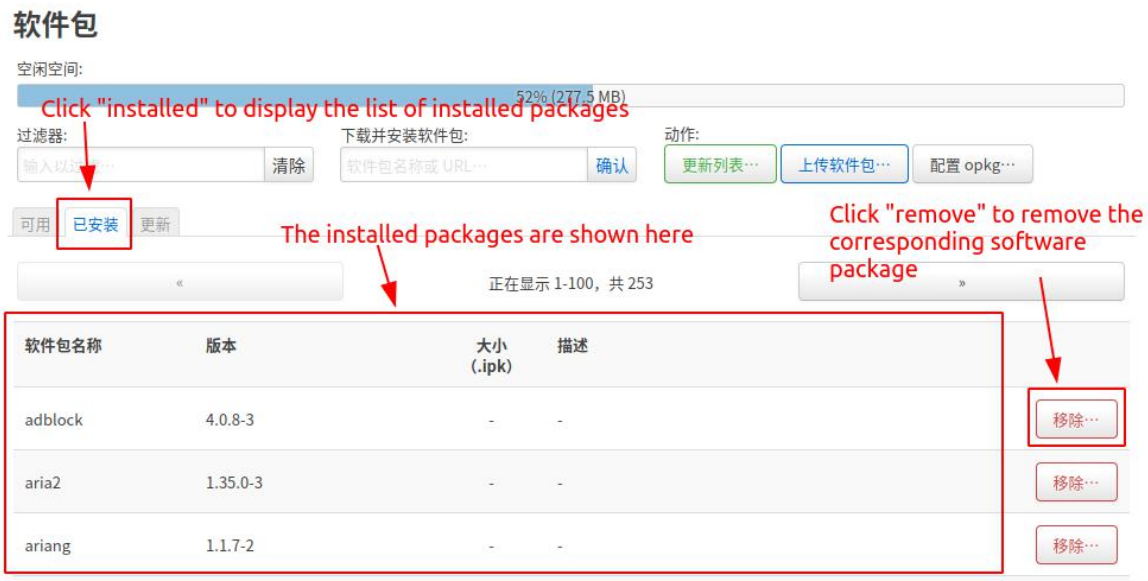
1) Click "System" -> "Package" to enter the software package management page

2) Click "Update List" to get a list of available packages, then click “available”, and then the number of packages currently available is shown here.



3) Find the software that needs to be installed and click "install"

4) After the installation is complete, you can see the installed software on the "Installed" page, and you can remove the software



3. 9. 3. Install packages from unofficial sources

Through the above method, OpenWRT can easily install thousands of software packages from software sources, which can meet most of the needs of users. However, sometimes we still need to find other packages from outside the official source to meet some special needs. When installing third-party software packages, you need to be aware that some software packages have system architecture restrictions. The system architecture of Orange Pi R1 Plus LTS is aarch64_generic, which also belongs to armv8. You need to select the software package of the corresponding architecture when downloading. Find and download on github or other forums

3. 9. 4. Common installation errors

1) The software package system architecture is wrong, you need to download the software package with the suffix aarch64_generic

```
root@OpenWrt:~# opkg install v2ray-core-mini_4.35.1-2_aarch64_cortex-a53.ipk
```

```
Unknown package 'v2ray-core-mini'.
Collected errors:
 * pkg_hash_fetch_best_installation_candidate: Packages for v2ray-core-mini found, but incompatible with the architectures configured
 * opkg_install_cmd: Cannot install package v2ray-core-mini.
```

2) The related library has been provided by the old version of the software, causing the installation to fail. You can choose to continue using the old version of the library, or uninstall the old version of the software and install the new version of the software, as shown below, when there are other libraries dependent In case of libnettle7, it will prompt that libnettle7 cannot be uninstalled. At this time, you need to add the --force-depends parameter, ignore the dependencies and uninstall libnettle7 directly.

```
root@OpenWrt:~# opkg install libnettle8
Installing libnettle8 (3.6-1) to root...
Downloading
https://downloads.openwrt.org/snapshots/packages/aarch64_generic/base/libnettle8_3.6-1_aarch64_generic.ipk
Collected errors:
 * check_data_file_clashes: Package libnettle8 wants to install file /usr/lib/libhogweed.so.6
        But that file is already provided by package  * libnettle7
root@OpenWrt:~# opkg remove libnettle7 --force-depends    #卸载 libnettle7
root@OpenWrt:~# opkg install libnettle8                   #安装 libnettle8
```

3. 10. Mount external storage devices

3. 10. 1. Mount in the terminal

1) Connect the U disk (or other storage device) to OrangePi R1 Plus

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

```
root@OpenWrt:~# cat /proc/partitions | grep "sd*"
major minor  #blocks  name
   8        0   15126528 sda
```

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

```
root@OpenWrt:~# mount /dev/sda /mnt/
root@OpenWrt:~# ls /mnt/
test.txt
```

3. 10. 2. Mount on the mount point management page

1) Connect the U disk (or other storage device) to OrangePi R1 Plus

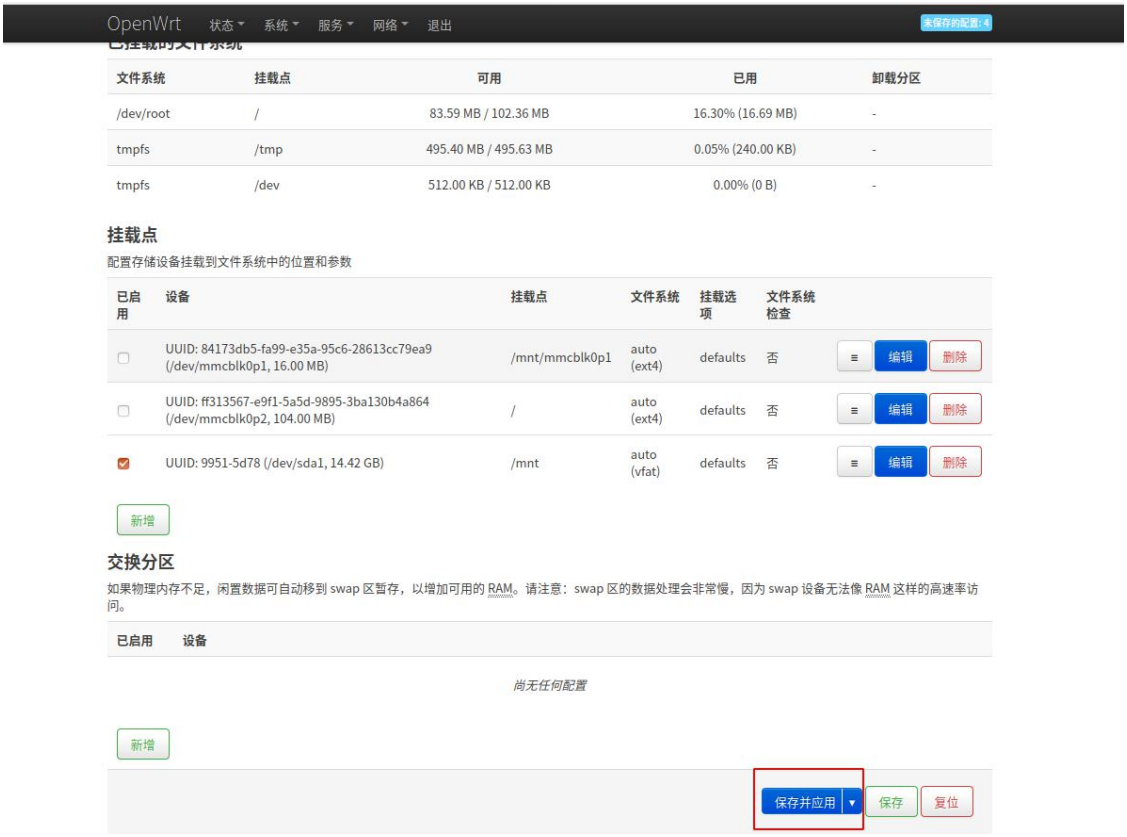
2) Click "System" -> "Mount Point" in OpenWRT to enter the mount point setting page

3) Find the mount point storage device settings at the bottom of the interface, click the "Add" button to add a mount point

4) Select the actual connected device /dev/sda1 in the UUID column of the general settings (choose according to your own device)

5) Use custom in the mount point column, fill in the target directory to be mounted to, here is the /mnt directory as an example, check the "enabled" above, and then click "save"

6) After setting, click "Save and Apply" to make the mount point effective



7) After saving, you can see in "Mounted File System", the storage device has been mounted

3. 11. Using Aria2

1) Click "Service" -> "Aira2" to enter the Aira2 management page

2) Check "Enabled" to enable Aira2, set the directory for downloading files in "Download Catalog", and click "Save and Apply" to save the settings

3) After saving, click "AriaNg" to enter the downloader, click the "New" button, and enter the download link in the box to download

3. 12. Use samba network share

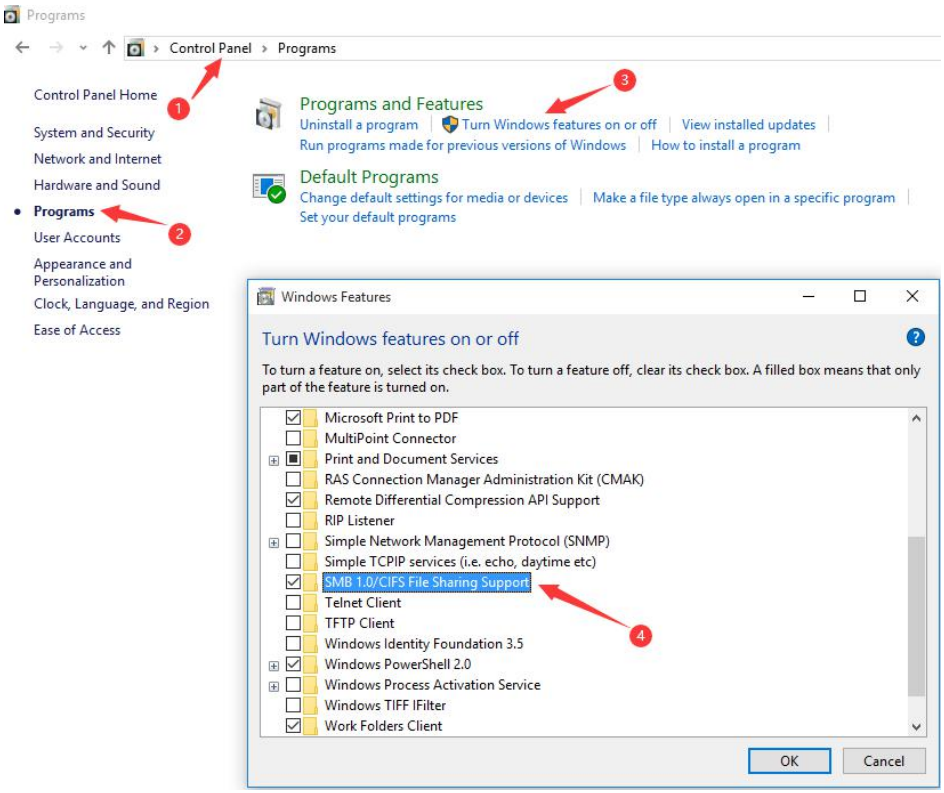
1) Click "Service" -> "Network" to enter the samba network share management page

2) Click "Add" at the bottom of the directory that needs to be shared. Here, take the setting of the shared/mnt directory as an example. After enter the name, path and permissions of the shared directory, click "Save and Apply" to save the configuration.

3) To access Samba under windows10 system, sharing needs to confirm whether window10 has started network discovery and sharing. If it is not started, first set the following settings

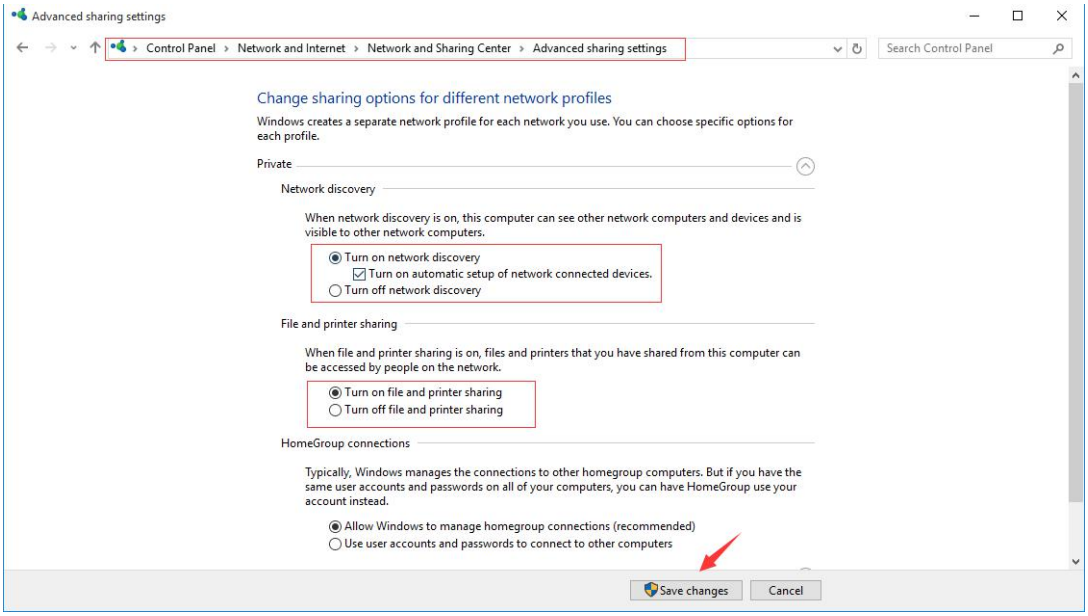
a) Enable Samba v1/v2 access

Go to "Control Panel" -> "Programs" -> "Turn Windows features on or off", select "SMB 1.0/CIFS File Sharing Support", and click "OK" to apply the configuration

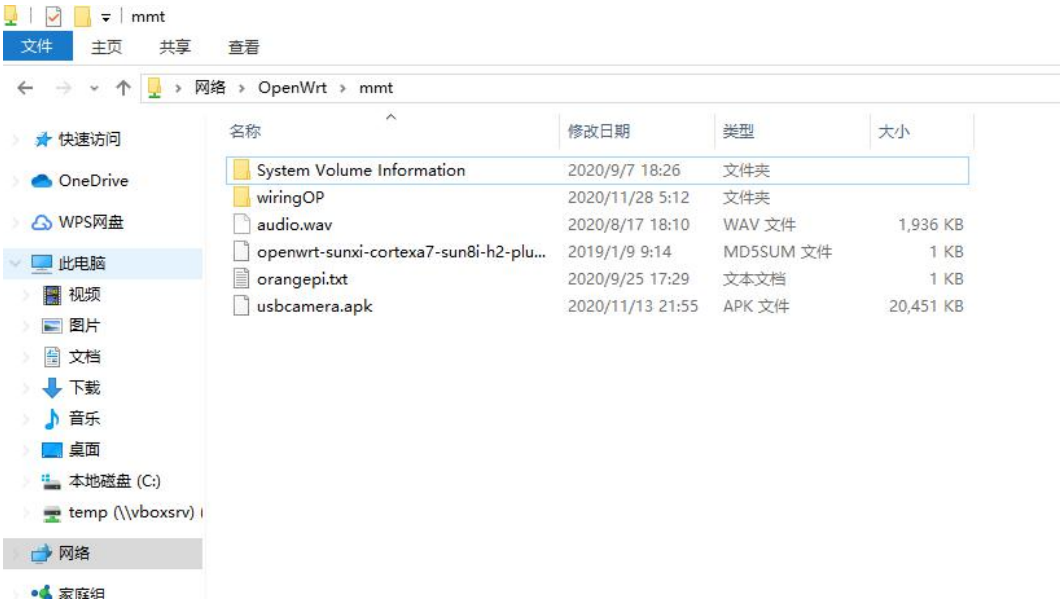


b) Turn on network discovery

Go to "Control Panel" -> "Network and Internet" -> "Network and Sharing Center" -> "Advanced Sharing Settings", open "Turn on Network Discovery" and "Turn on File and Printer Sharing"



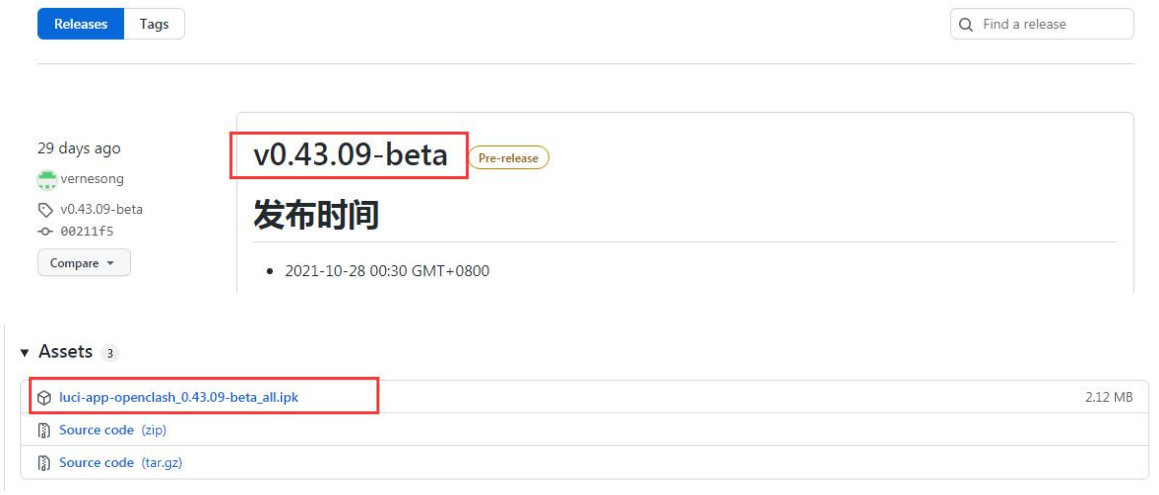
4) After the setting is completed, enter \\OpenWrt in the address bar of the explorer to access the shared directory, the user name is root, and the password is password (you don't need a password)



3. 13. Install luci-app-openclash

1) First download the package of luci-app-openclash on github, the github repository address is https://github.com/vernesong/OpenClash

2) 选择软件包版本并下载



3) Before installing luci-app-openclash, you need to install the following dependencies

```
root@OpenWrt:/# opkg update
root@OpenWrt:/# opkg remove dnsmasq && opkg install dnsmasq-full
root@OpenWrt:/# opkg install coreutils-nohup bash iptables dnsmasq-full \
curl ca-certificates ipset ip-full iptables-mod-tproxy iptables-mod-extra \
libcap libcap-bin ruby ruby-yaml kmod-tun （这是一条命令）
```

4) There are two ways to install luci-app-openclash

a. Copy luci-app-openclash_0.43.09-beta_all.ipk to the system via U disk, and execute the following command to install

```
root@OpenWrt:/# opkg install luci-app-openclash_0.43.09-beta_all.ipk
```

b. In the OpenWRT management interface, in System -> Package, click "Upload Package..." to upload and install

5) Restart after the installation is complete, re-enter the OpenWRT management interface, click "Service" -> "OpenClash" to enter the settings, specific use needs to be configured by yourself

3. 14. v2ray instructions

OpenWRT official software source does not include v2ray, if you need to use it, you can install it according to the following method

1) Import the certificate

```
root@OpenWrt:/#
wget -O kuoruan-public.key http://openwrt.kuoruan.net/packages/public.key
root@OpenWrt:/# opkg-key add kuoruan-public.key
```

2) Import the v2ray installation source

```
root@OpenWrt:/#
```

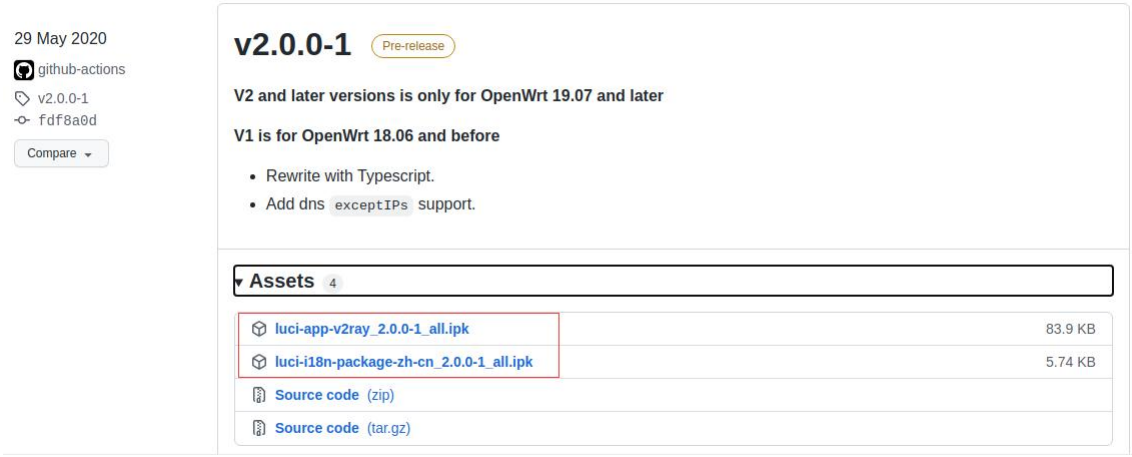
```
echo "src/gz kuoruan_packages http://openwrt.kuoruan.net/packages/releases/\
$(. /etc/openwrt_release ; echo $DISTRIB_ARCH)" >> /etc/opkg/customfeeds.conf
(This is a command)
```

3) Update the package and install v2ray

```
root@OpenWrt:/# opkg update
root@OpenWrt:/# opkg install v2ray-core
```

4) Download luci-app-v2ray and the Chinese package on github, the github warehouse address is

https://github.com/kuoruan/luci-app-v2ray/releases



5) There are two ways to install luci-app-v2ray and Chinese package

a. Copy the ipk file to the system via a USB flash drive, and execute the following command to install

```
root@OpenWrt:/# opkg install luci-app-v2ray_2.0.0-1_all.ipk
root@OpenWrt:/#
opkg install luci-i18n-package-zh-cn_2.0.0-1_all.ipk
```

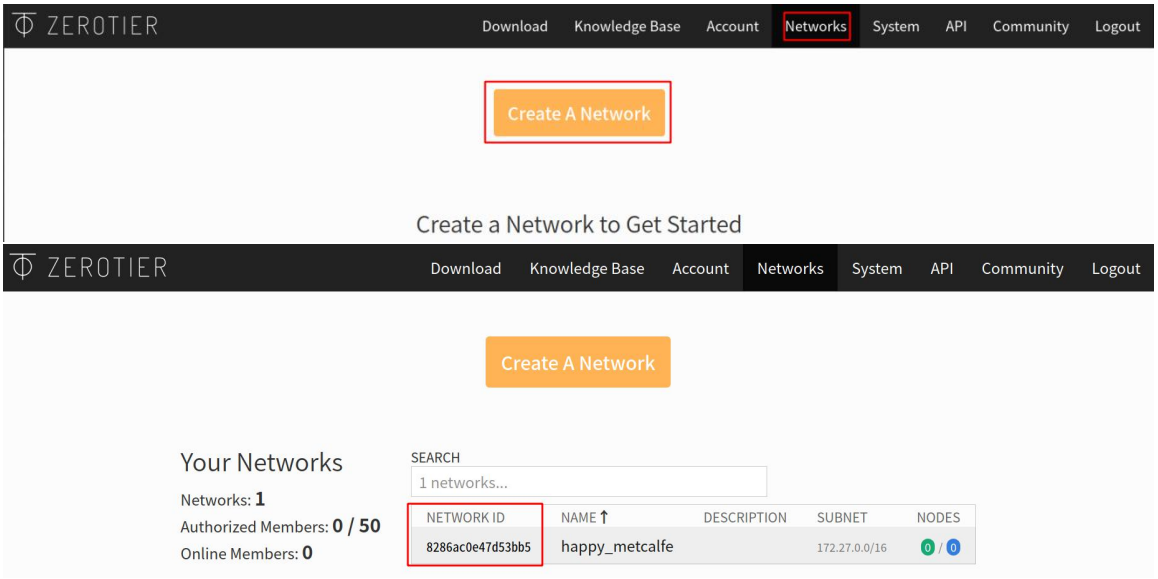
b. In the OpenWRT management interface, in System -> Package, click "Upload Package..." to upload and install

6) After the installation is complete, you can configure it on the OpenWRT management page

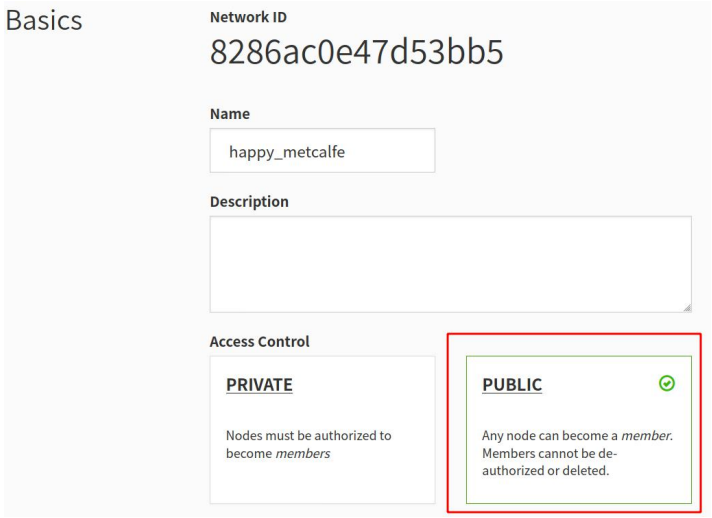
3. 15. zerotier instructions for use

The OpenWRT system has been pre-installed with the zerotier client. After creating a virtual local area network on the zerotier official website, the client can directly join it through the Network ID. The specific operations are as follows

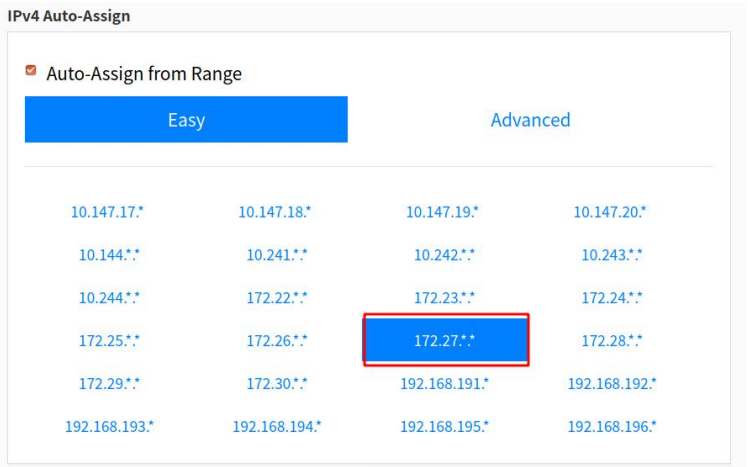
1) Log in to the zerotier official website https://my.zerotier.com/network, and click Network->Create A Network after registering and logging in to create a virtual local area network



2) Click to enter the web console page, you can set the privacy option to public, so that the added network node does not need to be verified



3) The address is automatically assigned below. Here you can choose the network segment yourself, here is 172.27.*.*



4) Enter the following command in the OpenWRT terminal to join the virtual local area network created above, where 8286ac0e47d53bb5 is the Network ID of the virtual local area network created above

```
root@OpenWrt:/# zerotier-one -d                    #Start the zerotier client
root@OpenWrt:/# zerotier-cli join 8286ac0e47d53bb5  #Join the network
```

5) Enter ifconfig in the terminal, you can see that there is already a newly added ztks54inm2 device with an IP address of 172.27.214.213

```
root@OpenWrt:/# ifconfig
ztks54inm2 Link encap:Ethernet  HWaddr F6:4E:DE:BF:D8:52
           inet addr:172.27.214.213  Bcast:172.27.255.255  Mask:255.255.0.0
```

```
        inet6 addr: fe80::e82f:d0ff:fe5a:867e/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:2800  Metric:1
        RX packets:18 errors:0 dropped:0 overruns:0 frame:0
        TX packets:48 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1720 (1.6 KiB)  TX byte81 (8.2 KiB)
```

6) Install the zerotier client on another device (here, Ubuntu 18.04 is taken as an example), execute the following command to install it, and restart the computer after the installation is complete

```
test@ubuntu:~$ curl -s https://install.zerotier.com | sudo bash
```

7) After restarting, join the virtual LAN according to the Network ID, and you can also see that the ip address assigned by zerotier has been obtained. At this time, the Ubuntu PC and OrangePi R1 Plus are in the same LAN, and the two can communicate freely.

```
test@ubuntu:~$ sudo zerotier-cli join 8286ac0e47d53bb5
test@ubuntu:~$ ifconfig
ztks54inm2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 2800
        inet 172.27.47.214  netmask 255.255.0.0  broadcast 172.27.255.255
        inet6 fe80::5ce1:85ff:fe2b:6918  prefixlen 64  scopeid 0x20<link>
        ether f6:fd:87:68:12:cf  txqueuelen 1000  (以太网)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 46  bytes 10006 (10.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

8) Test whether the two terminals can communicate

```
root@OpenWrt:/# ping 172.27.47.214 -I ztks54inm2
PING 172.27.47.214 (172.27.47.214): 56 data bytes
64 bytes from 172.27.47.214: seq=0 ttl=64 time=1.209 ms
64 bytes from 172.27.47.214: seq=1 ttl=64 time=1.136 ms
64 bytes from 172.27.47.214: seq=2 ttl=64 time=1.203 ms
64 bytes from 172.27.47.214: seq=3 ttl=64 time=1.235 ms
^C
--- 172.27.47.214 ping statistics ---
```

```
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.136/1.195/1.235 ms
```

9) Zerotier other commonly used commands

```
root@OpenWrt:/# zerotier-one -d                    #启动 zerotier 客户端
root@OpenWrt:/# zerotier-cli status                #获取地址和服务状态
root@OpenWrt:/# zerotier-cli join # Network ID     #加入网络
root@OpenWrt:/# zerotier-cli leave # Network ID    #离开网络
root@OpenWrt:/# zerotier-cli listnetworks          #列出网络
```

3. 16. Shadowsocks-libev instructions

The OpenWRT system has been pre-installed with Shadowsocks-libev, click "Service" -> "Shadowsocks-libev" on the OpenWRT management page to enter the configuration interface, and you can configure it yourself

3. 17. View OpenWRT system version information

```
root@OpenWrt:~# cat /etc/os-release
NAME="OpenWrt"
VERSION="21.02.1"
ID="openwrt"
ID_LIKE="lede openwrt"
PRETTY_NAME="OpenWrt 21.02.1"
VERSION_ID="21.02.1"
```

```
HOME_URL="https://openwrt.org/"
BUG_URL="https://bugs.openwrt.org/"
SUPPORT_URL="https://forum.openwrt.org/"
BUILD_ID="r16325-88151b8303"
OPENWRT_BOARD="rockchip/armv8"
OPENWRT_ARCH="aarch64_generic"
OPENWRT_TAINTS="no-all"
OPENWRT_DEVICE_MANUFACTURER="OpenWrt"
OPENWRT_DEVICE_MANUFACTURER_URL="https://openwrt.org/"
OPENWRT_DEVICE_PRODUCT="Generic"
OPENWRT_DEVICE_REVISION="v0"
OPENWRT_RELEASE="OpenWrt 21.02.1 r16325-88151b8303"
```

4. OpenWRT SDK instructions

4. 1. Download the source code of OpenWRT SDK

4. 1. 1. Download OpenWRT from github

1) At present, there are two branches of openwrt code on github. The openwrt-21.02 branch is adapted based on the official stable version of openwrt, and its functions tend to be stable. The master branch is adapted based on the official snapshot version of openwrt and is a version under development. Please download the corresponding branch according to your needs

2) Download the openwrt-21.02 branch code

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install git
test@test:~$
git clone https://github.com/orangepi-xunlong/openwrt.git -b openwrt-21.02
```

3) Download the master branch code

```
test@test:~$ git clone https://github.com/orangepi-xunlong/openwrt.git -b master
```

4) After the OpenWRT code is downloaded, the following files and folders will be

included

```
test@test:~/openwrt$ ls
BSDmakefile Config.in include Makefile README.md scripts toolchain
Config feeds.conf.default LICENSE package rules.mk target tools
```

4. 2. Compile OpenWRT

4. 2. 1. Compile OpenWRT Source Code

1) First install the following dependent software (currently only tested to compile on Ubuntu 18.04 and need to install the following software, if you compile on other versions of the system, please install the dependent software yourself according to the error message)

```
test@test:~/openwrt$ sudo apt update
test@test:~/openwrt$ sudo apt install make libncurses5-dev g++ gcc gawk
```

2) Then execute ./scripts/feeds update -a and ./scripts/feeds install -a to download dependency packages

```
test@test:~/openwrt$ ./scripts/feeds update -a
test@test:~/openwrt$ ./scripts/feeds install -a
```

Note: This step and the later make compile will download many packages with foreign sources. Because of network problems, it is very likely that the download will fail and cause compilation errors. Therefore, it is recommended to use the source code package of Baidu Netdisk, which already contains the ones that need to be downloaded. The software package, no need to download, after decompression, you can go directly to step 2

3) Choose to use the OrangePi R1 Plus Configuration file

```
test@test:~/openwrt$ cp configs/OrangePi_R1_Plus_defconfig .config
test@test:~/openwrt$ make defconfig
```

4) Start compiling

Execute make V=s to start compilation

```
test@test:~/openwrt$ make V=s
```

5) When you need to select a new software package

```
test@test:~/openwrt$ make menuconfig
```

6) Save your personal package configuration for next use

```
test@test:~/openwrt$ ./scripts/diffconfig.sh > ./configs/my_config
```

7) Image generation location

openwrt/bin/targets/rockchip/armv8/
openwrt-rockchip-armv8-xunlong_orangepi-r1plus-ext4-sysupgrade.img.gz

5. Linux system instructions

5.1. Supported Linux Release version and kernel version

Release version	Kernel version	Server version	Desktop version
Ubuntu 18.04	Linux5.10	Supported	Not Supported
Ubuntu 20.04	linux5.10	Supported	Not Supported
Debian 10	linux5.10	Supported	Not Supported

5.2. linux5.10 kernel driver adaptation situation

Functions	Status
USB2.0	OK
TF card boot	OK
Gigabit Ethernet	OK
USB to Gigabit LAN	OK
USB camera	OK
LED Lights	OK
13pin GPIO	OK
I2C	OK
SPI Nor Flash	OK
UART	OK
Reset button	OK

5.3. Linux system default login account and password

Account	Password
root	orangepi
orangepi	orangepi

5.4. 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 call the orangepi-resize-filesystem script through the orangepi-resize-filesystem.service systemd service 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@orangepir1plus-lts:~# 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, after burning the image to the TF, remember not to start the linux system

b. Then insert the TF card into the Ubuntu PC (Windows does not work), the Ubuntu PC will usually automatically mount the TF card partition, if the automatic mounting is normal, use the ls command to see the following output, the TF card partition

name and the following command The names shown are not necessarily the same, please modify according to the actual situation

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

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

```
test@test:~$ sudo -i
[sudo] test password：
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 delete the TF card, then unplug the TF and plug it into the development board to start. When the linux system starts, when the file .no_rootfs_resize in the /root directory is detected, the rootfs will no longer be automatically expanded

f. After prohibit automatic expansion of rootfs, you can see that the available capacity of the TF card is only about 168M

```
root@orangepir1plus-lts:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            417M     0  417M   0% /dev
tmpfs            98M  2.9M   96M   3% /run
/dev/mmcblk0p1  1.3G  1.1G  168M  87% /
tmpfs           490M     0  490M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           490M     0  490M   0% /sys/fs/cgroup
tmpfs           490M  4.0K  490M   1% /tmp
/dev/zram0       49M  1.3M   44M   3% /var/log
tmpfs            98M     0   98M   0% /run/user/0
```

5.5. How to modify the linux log level (loglevel)?

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

```
Starting kernel ...

Uncompressing Linux... done, booting the kernel.

Orange Pi 2.1.0 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 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. TF card inserted into the development board to start

```
root@orangepir1plus-lts:~# sed -i "s/verbosity=1/verbosity=7/" /boot/orangepiEnv.txt
root@orangepir1plus-lts:~# 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

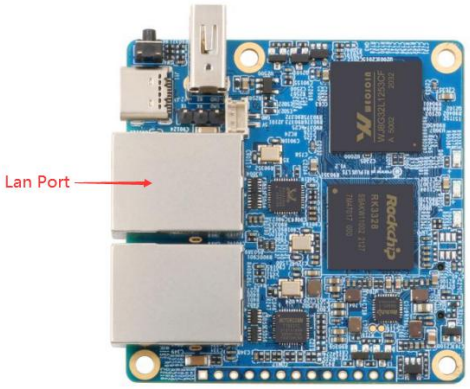
```
root@orangepir1plus-lts:~# cat /boot/orangepiEnv.txt
verbosity=7
console=serial
```

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

5.6. Ethernet port test

5.6.1. Lan Port Test

1) First, plug the network cable into the onboard 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@orangepir1plus-lts:~# ifconfig lan0
lan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.96  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::be68:b89f:def0:5261  prefixlen 64  scopeid 0x20<link>
        ether c0:74:2b:ff:b3:46  txqueuelen 1000  (Ethernet)
        RX packets 5737  bytes 329470 (329.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 57  bytes 5500 (5.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

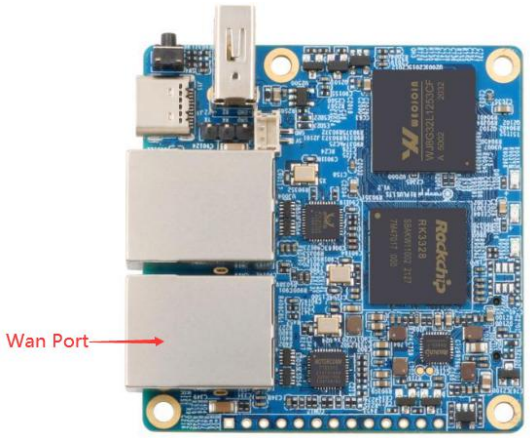
4) The command to test network connectivity is as follows

```
root@orangepir1plus-lts:~# ping www.baidu.com -I lan0
PING www.a.shifen.com (180.101.49.42) from 192.168.1.96 lan0: 56(84) bytes of data.
64 bytes from 180.101.49.42 (180.101.49.42): icmp_seq=1 ttl=53 time=27.2 ms
64 bytes from 180.101.49.42 (180.101.49.42): icmp_seq=2 ttl=53 time=26.0 ms
64 bytes from 180.101.49.42 (180.101.49.42): icmp_seq=3 ttl=53 time=25.8 ms
```

```
64 bytes from 180.101.49.42 (180.101.49.42): icmp_seq=4 ttl=53 time=26.7 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 25.839/26.478/27.254/0.583 ms
```

5. 6. 2. Wan Port Test

1) First insert the network cable into the usb to Ethernet interface of the development board, and make sure 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@orangepir1plus-lts:~# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.62  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::9db4:d13e:5e66:ee6a  prefixlen 64  scopeid 0x20<link>
        ether 66:2e:7d:b9:f7:74  txqueuelen 1000  (Ethernet)
        RX packets 2055  bytes 150800 (150.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 145  bytes 13816 (13.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 29
```

4) The command to test network connectivity is as follows

```
root@orangepir1plus-lts:~# ping www.baidu.com -I eth0
PING www.a.shifen.com (14.215.177.39) from 192.168.1.62 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=2 ttl=56 time=6.51 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=6.45 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=56 time=6.44 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=5 ttl=56 time=6.59 ms
^C
--- www.a.shifen.com ping statistics ---
13 packets transmitted, 12 received, 7% packet loss, time 12032ms
rtt min/avg/max/mdev = 6.252/6.596/7.067/0.226 ms
```

5.7. 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 is connected, and then use the ifconfig command or check the router to obtain the IP address of the development board

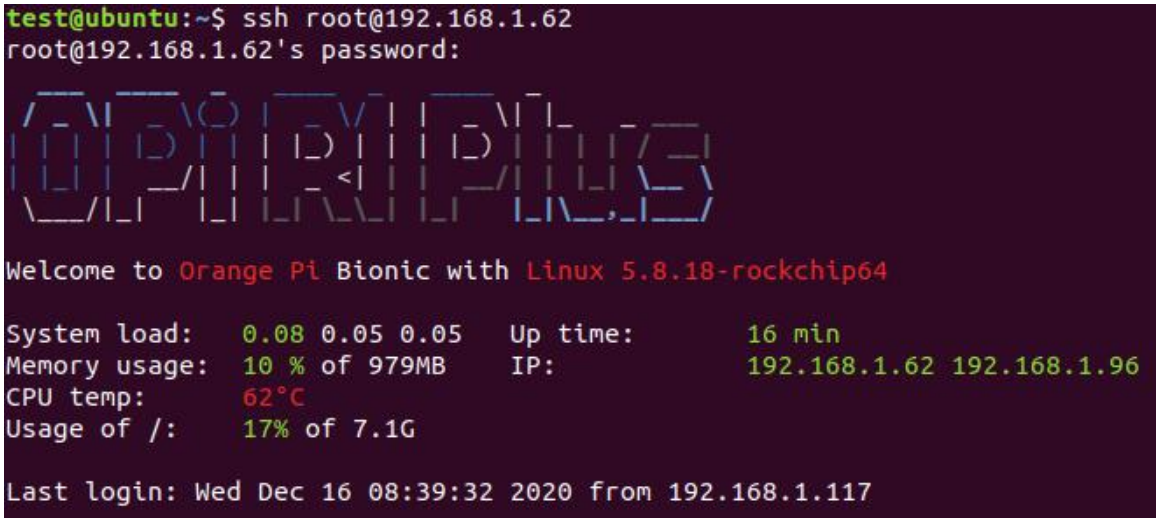
5.7.1. SSH remote login development board under Ubuntu

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@192.168.1.62        (Need to be replaced with the IP address of the development board)
root@192.168.1.62's password:        （Enter the password here, the default password is orangepi）
```

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



4) If the following error is prompted during ssh login

```
test@test:~$ ssh root@192.168.1.62
Connection reset by 192.168.1.62 port 22
```

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

```
root@orangepir1plus-lts:~# rm /etc/ssh/ssh_host_*
root@orangepir1plus-lts:~# dpkg-reconfigure openssh-server
```

5.7.2. SSH remote login development board under Windows

1) First get the IP address of the development board

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

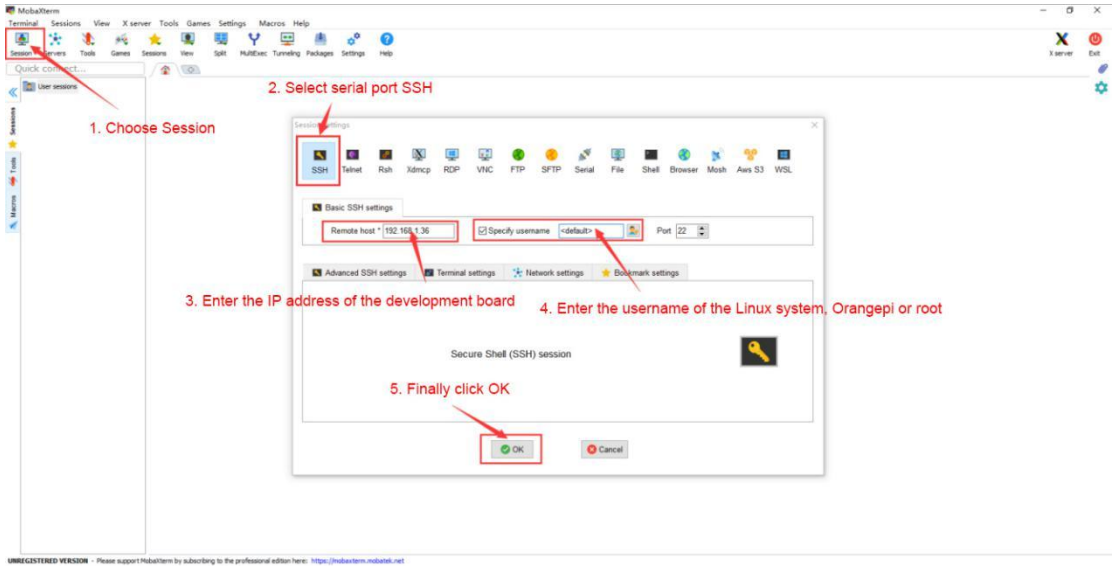
a. Choose **Session**

b. Then select serial port SSH

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

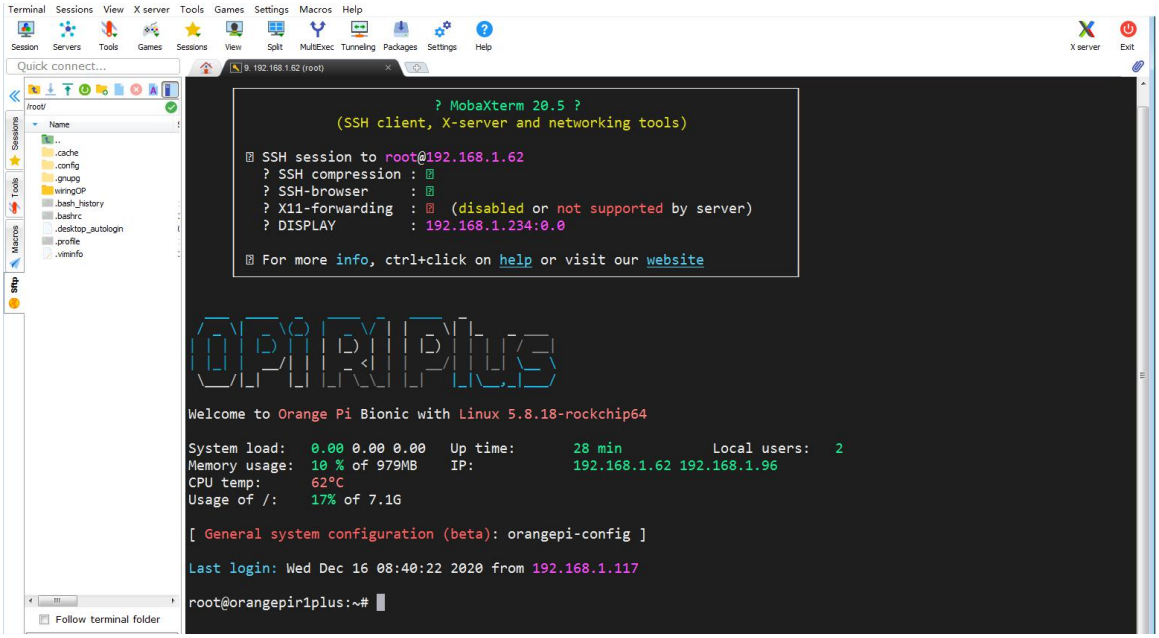
d. Then enter the username **root** or **orangepi** of the Linux system in **Specify username**

e. Finally click **OK**



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

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



5. 8. Onboard LED light test instructions

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

Power status light (red light)	The system starts, the red light flashes
Wan port status light (yellow light)	Wan port is connected to the network cable, the yellow light is always on, the Wan port is unplugged, the yellow light is off
Lan port status light (yellow light)	The Lan port is connected to the network cable, the yellow light is always on, and the Lan port is unplugged, the yellow light is off

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

a) First enter the red light setting directory

```
root@orangepir1plus-lts:~# cd /sys/class/leds/status_led
```

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

```
root@orangepir1plus-lts:/sys/class/leds/status_led# echo 0 > brightness
```

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

```
root@orangepir1plus-lts:/sys/class/leds/status_led# echo 1 > brightness
```

d) The command to set the red light to stop flashing is as follows

```
root@orangepir1plus-lts:/sys/class/leds/status_led# echo none > trigger
```

5.9. USB Port Test

5.9.1. 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 is successfully recognized

```
root@orangepir1plus-lts:~# 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@orangepir1plus-lts:~# mount /dev/sda1 /mnt/
root@orangepir1plus-lts:~# ls /mnt/
test.txt
```

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

```
root@orangepir1plus-lts:~# df -h | grep "sd"
/dev/sda1       29G  208K   29G   1% /mnt
```

5.10. USB wireless network card test

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 cannot use them, you need to transplant the corresponding USB wireless network card drivers.

Serial number	Model
1	RTL8723BU
2	RTL8821CU

5.10.1. RTL8723BU test

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

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

```
root@orangepir1plus-lts:~# lsmod | grep "rtl8"
rtl8xxxu                   126976  0
mac80211                   925696  1 rtl8xxxu
```

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

```
root@orangepir1plus-lts:~# dmesg | tail
[ 3128.895618] usb 3-1: 1d8: ff ff ff ff ff ff ff ff
[ 3128.895633] usb 3-1: 1e0: ff ff ff ff ff ff ff ff
[ 3128.895648] usb 3-1: 1e8: ff ff ff ff ff ff ff ff
[ 3128.895663] usb 3-1: 1f0: ff ff ff ff ff ff ff ff
[ 3128.895679] usb 3-1: 1f8: ff ff ff ff ff ff ff ff
[ 3128.895703] usb 3-1: RTL8723BU rev E (SMIC) 1T1R, TX queues 3, WiFi=1, BT=1, GPS=0, HI PA=0
[ 3128.895720] usb 3-1: RTL8723BU MAC: 00:13:ef:f4:58:ae
[ 3128.895737] usb 3-1: rtl8xxxu: Loading firmware rtlwifi/rtl8723bu_nic.bin
[ 3128.896110] usb 3-1: Firmware revision 35.0 (signature 0x5301)
[ 3132.191050] rtl8xxxu 3-1:1.2 wlx0013eff458ae: renamed from wlan0
```

4) Then you can see the device node of RTL8723BU WIFI through the ifconfig command, please refer to the WIFI connection test section for WIFI connection and test method

```
root@orangepir1plus-lts:~# 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 a Bluetooth device 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 test section

```
root@orangepir1plus-lts:~# hciconfig
hci0:    Type: Primary    Bus: USB
         BD Address: 00:13:EF:F4:58:AE    ACL MTU: 1021:8    SCO MTU: 255:16
         UP RUNNING
         RX bytes:1631 acl:0 sco:0 events:144 errors:0
         TX bytes:26662 acl:0 sco:0 commands:144 errors:0
```

5.10.2. RTL8821CU test

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

2) Then the Linux OS will automatically load RTL8821CU related kernel modules, and you can see the following output through the lsmod command

```
root@orangepir1plus-lts:~# lsmod | grep "8821"
8821cu                  2043904  0
cfg80211                 897024  3 8821cu,mac80211,rtl8xxx
```

3) Through the dmesg command, you can see the loading information of the RTL8821CU module

```
root@orangepir1plus-lts:~# dmesg | tail
[ 3987.552017] usb 2-1: Product: 802.11ac NIC
[ 3987.552032] usb 2-1: Manufacturer: Realtek
[ 3987.552046] usb 2-1: SerialNumber: 123456
[ 3987.560377] Bluetooth: hci0: RTL: examining hci_ver=08 hci_rev=000c lmp_ver=08
lmp_subver=8821
[ 3987.561349] Bluetooth: hci0: RTL: rom_version status=0 version=1
[ 3987.561370] Bluetooth: hci0: RTL: loading rtl_bt/rtl8821c_fw.bin
[ 3987.561818] Bluetooth: hci0: RTL: loading rtl_bt/rtl8821c_config.bin
[ 3987.562148] Bluetooth: hci0: RTL: cfg_sz 10, total sz 21678
```

```
[ 3987.974248] Bluetooth: hci0: RTL: fw version 0x826ca99e
[ 3987.998204] rtl8821cu 2-1:1.2 wlxd0c0bf8742cd: renamed from wlan0
```

4) Then you can see the device node of RTL8821CU WIFI through the ifconfig command, please refer to the WIFI connection test section for WIFI connection and test method

```
root@orangepir1plus-lts:~# 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 the Bluetooth device node through the hciconfig command. The node whose Bus type is USB is the Bluetooth node of RTL8821CU. For the Bluetooth test method, please refer to the Bluetooth test section

```
root@orangepir1plus-lts:~# hciconfig
hci0:   Type: Primary  Bus: USB
        BD Address: D0:C0:BF:87:42:CE  ACL MTU: 1021:8  SCO MTU: 255:12
        UP RUNNING
        RX bytes:1350 acl:0 sco:0 events:138 errors:0
        TX bytes:24230 acl:0 sco:0 commands:138 errors:0
```

5.11. WIFI connection test

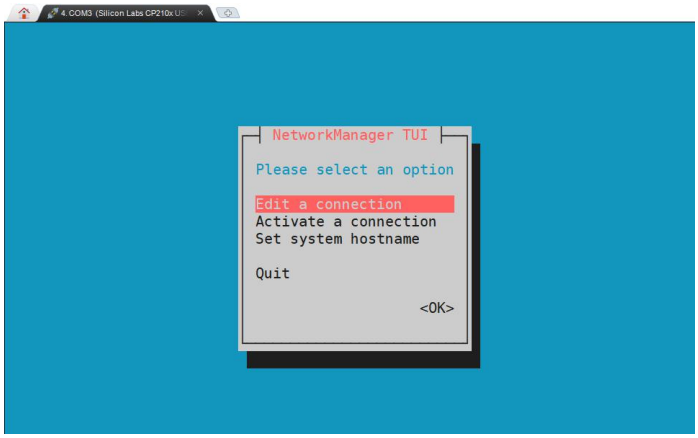
5.11.1. Linux OS test method

1) First use the serial terminal to log in to the Linux OS (Serial software please use MobaXterm, the graphical interface cannot be displayed using minicom)

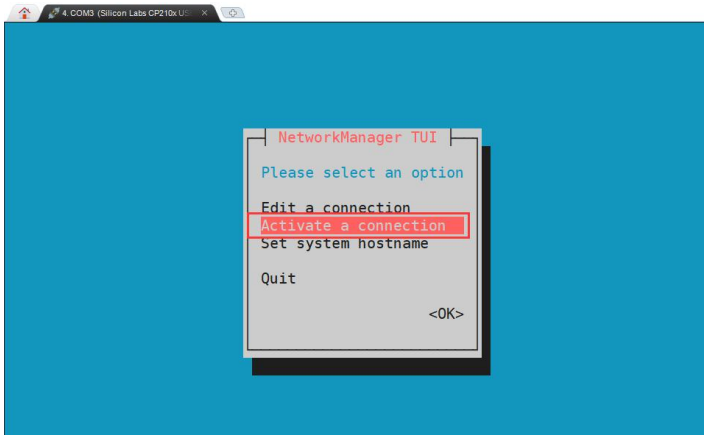
2) Then enter nmtui in the command line to open the wifi connection interface

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

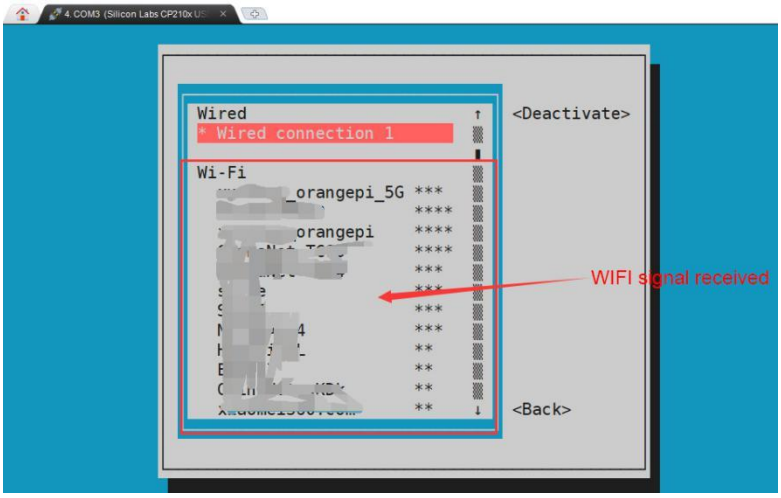
3) Enter the interface opened by nmtui as shown below



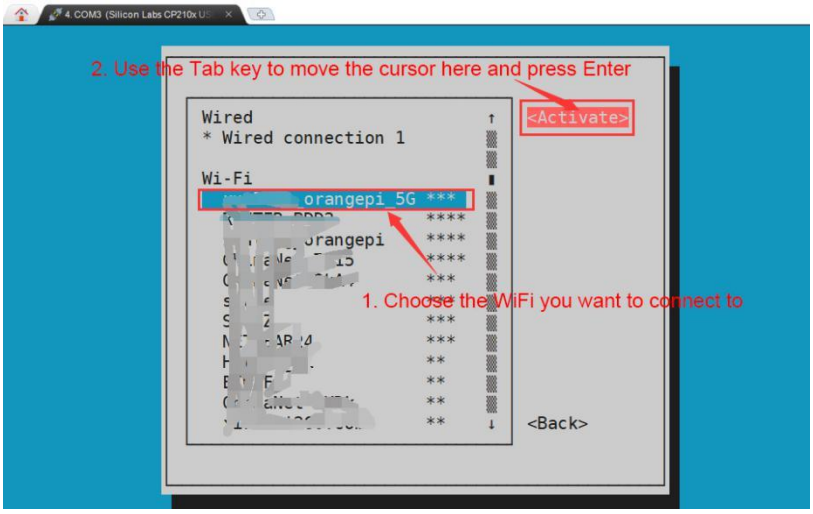
4) Select **Activate a connect** and press Enter



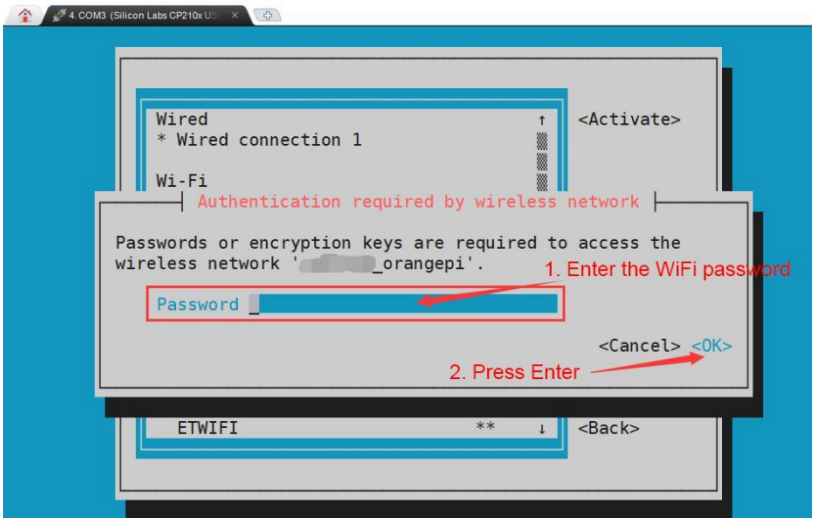
5) Then you can see all the searched WIFI hotspots



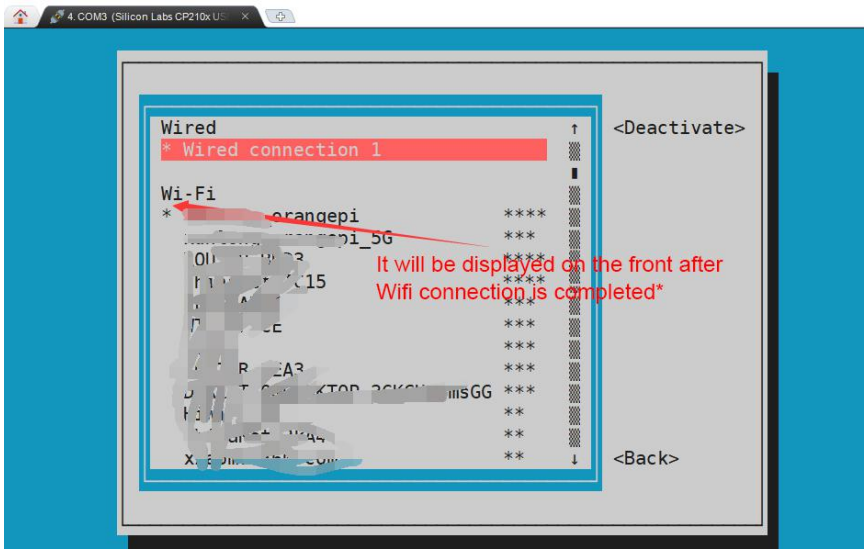
6) Select the WIFI hotspot you want to connect to, then use the Tab key to position the cursor on **Activate** and press Enter



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



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



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

```
root@orangepir1plus-lts:~# ifconfig wlxd0c0bf8742cd
wlxd0c0bf8742cd: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.198  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::db3a:3ea6:b98d:3fc9  prefixlen 64  scopeid 0x20<link>
        ether d0:c0:bf:87:42:cd  txqueuelen 1000  (Ethernet)
        RX packets 528  bytes 91350 (91.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 26  bytes 3802 (3.8 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@orangepir1plus-lts:~# ping www.baidu.com -I wlxd0c0bf8742cd
PING www.a.shifen.com (14.215.177.39) from 192.168.1.198 wlxd0c0bf8742cd: 56(84) bytes of data.
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=1 ttl=56 time=7.39 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=2 ttl=56 time=7.72 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=3 ttl=56 time=8.79 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=4 ttl=56 time=10.2 ms
64 bytes from 14.215.177.39 (14.215.177.39): icmp_seq=5 ttl=56 time=8.83 ms
^C
--- www.a.shifen.com ping statistics ---
```

```
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 7.391/8.607/10.296/1.021 ms
```

5.12. USB wireless network card Bluetooth test

5.12.1. Linux OS test method

1) The bluez tool is required to use the Bluetooth connection. Use the following command to install the bluez tool

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install bluez
```

2) First check the Bluetooth device information through hciconfig -a, the following information appears, indicating that the Bluetooth initialization is normal

```
root@orangepir1plus-lts:~# hciconfig -a
hci0:   Type: Primary   Bus: USB
        BD Address: 00:13:EF:F4:58:AE   ACL MTU: 1021:8   SCO MTU: 255:16
        UP RUNNING PSCAN
        RX bytes:15875 acl:28 sco:0 events:367 errors:0
        TX bytes:28282 acl:28 sco:0 commands:232 errors:0
        Features: 0xff 0xff 0xff 0xfa 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH HOLD SNIFF PARK
        Link mode: SLAVE ACCEPT
        Name: 'orangepir1plus-lts'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 4.0 (0x6)   Revision: 0x1e4c
        LMP Version: 4.0 (0x6)   Subversion: 0xc3ff
        Manufacturer: Realtek Semiconductor Corporation (93)
```

3) Use bluetoothctl to turn on Bluetooth to scan surrounding devices

```
root@orangepir1plus-lts:~# bluetoothctl
[NEW] Controller 00:13:EF:F4:58:AE orangepir1plus-lts [default]
```

```
Agent registered
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# discoverable on
Changing discoverable on succeeded
[CHG] Controller 00:13:EF:F4:58:AE Discoverable: yes
[bluetooth]# pairable on
Changing pairable on succeeded
[bluetooth]# scan on
[NEW] Device 6A:31:DF:62:08:78 6A-31-DF-62-08-78
[NEW] Device 56:73:C1:98:C6:63 56-73-C1-98-C6-63
[NEW] Device 9C:2E:A1:42:71:11 小米手机
[NEW] Device 44:F2:1B:B8:76:7B Roy
```

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

```
[bluetooth]# pair 44:F2:1B:B8:76:7B
Attempting to pair with 44:F2:1B:B8:76:7B
[CHG] Device 44:F2:1B:B8:76:7B  Connected: yes
Request confirmation
[Roy]1m[agent] Confirm passkey 996955  (yes/no): yes
[CHG] Device DC:72:9B:4C:F4:CF Paired: yes
Pairing successful
[bluetooth]# paired-devices
Device 44:F2:1B:B8:76:7B Roy
```

5. 13. 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

```
root@orangepir1plus-lts:~# lsmod | grep "uvc"
uvcvideo               110592  0
videobuf2_vmalloc       20480  1 uvcvideo
```

```
videobuf2_v4l2          36864  1 uvcvideo
videobuf2_common        65536  2 videobuf2_v4l2,uvcvideo
videodev               311296  3 videobuf2_v4l2,uvcvideo,videobuf2_common
mc                      65536  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@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install v4l-utils
root@orangepir1plus-lts:~# v4l2-ctl  --list-devices
USB 2.0 Camera: HD USB Camera (usb-ff5d0000.usb-1):
    /dev/video0
    /dev/video1
```

4) Use fswebcam to test the USB camera

a. Install fswebcam

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt-get install fswebcam
```

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

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

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

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

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

```
root@orangepir1plus-lts:~# fswebcam -d /dev/video0 --no-banner -r 1280x720 -S 5 ./image.jpg
```

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

```
root@orangepir1plus-lts:~# scp image.jpg test@192.168.1.55:/home/test （Modify the IP address and path according to the actual situation)
```

5) Use motion to test the USB camera

a. Install the camera test software motion

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install motion
```

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

```
root@orangepir1plus-lts:~# sed -i \
"s/start_motion_daemon=no/start_motion_daemon=yes/" \
/etc/default/motion      (这是一条命令)
```

c. Modify**/etc/motion/motion.conf** configuration

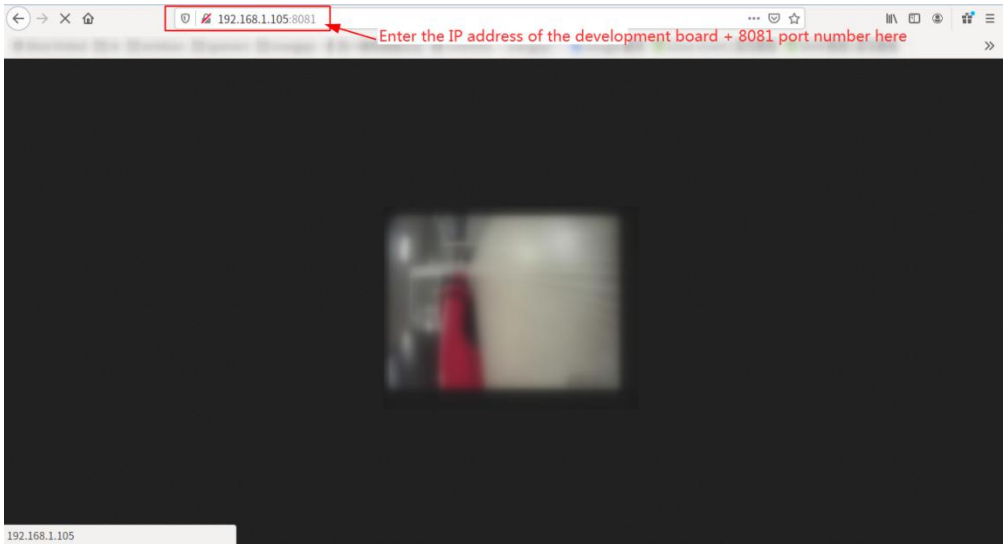
```
root@orangepir1plus-lts:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf      (这是一条命令)
```

d. Then restart the motion service

```
root@orangepir1plus-lts:~# /etc/init.d/motion restart
[ ok ] Restarting motion (via systemctl): motion.service.
```

e. Please make sure that the Orange Pi development board can connect to the network normally before using motion, 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.

g. 

5. 14. Temperature sensor

1) RK3328 has a total of 1 temperature sensor, the command to check the temperature is

as follows

```
root@orangepir1plus-lts:~# cat /sys/class/thermal/thermal_zone0/type
soc-thermal
root@orangepir1plus-lts:~# cat /sys/class/thermal/thermal_zone0/temp
61664
```

5. 15. How to install Docker

1) This method is not applicable on debian10 system

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

```
root@orangepir1plus-lts:~# apt remove docker docker-engine docker-ce docker.io
```

3) Then install the following packages

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install -y apt-transport-https ca-certificates curl
software-properties-common (这是一条命令)
```

4) Add the key of Alibaba Cloud docker

```
root@orangepir1plus-lts:~# curl -fsSL
http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg
  | sudo apt-key add - (This is a command)
```

5) Add the corresponding docker source in the system source of Ubuntu

```
root@orangepir1plus-lts:~# add-apt-repository "deb [arch=arm64]
 https://mirrors.aliyun.com/docker-ce/linux/ubuntu $(lsb_release -cs) stable"
  (This is a command)
```

6) Install the latest version of docker-ce

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install docker-ce
```

7) Verify the status of docker

```
root@orangepir1plus-lts:~# systemctl status docker
● docker.service - Docker Application Container Engine
```

```
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-08-24 10:29:22 UTC; 26min ago
     Docs: https://docs.docker.com
 Main PID: 3145 (dockerd)
    Tasks: 15
   CGroup: /system.slice/docker.service
           └─3145 /usr/bin/dockerd -H fd://
--containerd=/run/containerd/containerd.sock
```

8) Test docker

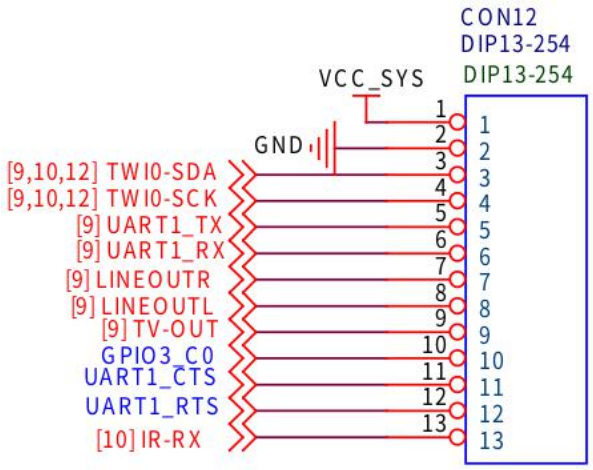
```
root@orangepir1plus-lts:~# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
256ab8fe8778: Pull complete
Digest:
sha256:7f0a9f93b4aa3022c3a4c147a449ef11e0941a1fd0bf4a8e6c9408b2600777c5
Status: Downloaded newer image for hello-world:latest

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

5. 16. 13Pin transfer board interface pin description

1) Please refer to the figure below for the sequence of the 13 pin interface of the Orange Pi R1 Plus LTS development board

2) The schematic diagram of the 13pin interface of the Orange Pi R1 Plus LTS development board is shown below



3) The function description of the 13 pin adapter board interface pins of Orange Pi R1 Plus LTS development board is as follows

a. When the 13pin pin is connected to the adapter board, it can be additionally provided

a) TV-OUT audio and video output

b) Infrared receiving function

c) The 3rd, 4th, 5th, 6th, 10th, 11th and 12th pins of the 13pin interface cannot be used after the adapter board is connected

d) **Also note that the MIC and 2*USB on the 13pin adapter board cannot be used on Orange Pi R1 Plus LTS**

b. When the 13 pin interface of the Orange Pi R1 Plus LTS development board is not connected to the adapter board, pins 3, 4, 5, 6, 10, 11, 12 and 13 can be used as ordinary GPIO

Pin	Function	GPIO	GPIO serial number
1	VCC_SYS		
2	GND		
3	TWI0-SDA	GPIO2_D1	89
4	TWI0-SCK	GPIO2_D0	88
5	UART1_TX	GPIO3_A4	100
6	UART1_RX	GPIO3_A6	102
7	LINEOUTR		
8	LINEOUTL		
9	TV-OUT		

10	GPIO3_C0	GPIO3_C0	112
11	UATR1_CTS	GPIO3_A7	103
12	UART1_RTS	GPIO3_A5	101
13	IR-RX	GPIO2_A2	66

5. 17. How to install wiringOP

1) Download the code of wiringOP

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install git
root@orangepir1plus-lts:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

2) Compile wiringOP

```
root@orangepir1plus-lts:~# cd wiringOP
root@orangepir1plus-lts:~/wiringOP# ./build clean
root@orangepir1plus-lts:~/wiringOP# ./build
```

3) The output of the test gpio readall command is as follows, where the physical pins 1 to 13 correspond to the 13 Pin pins on the development board, and the pins 7, 8, 9, and 14 cannot be used on WiringOP. Use Please ignore when

```
root@orangepir1plus:~# gpio readall
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     | 5V       |      |   |  1 || 2  |   |      | GND      |     |      |
 |   89 |   0 | SDA.0    |   IN | 1 |  3 || 4  | 1 | IN   | SCK.0    | 1   | 88   |
 |  100 |   2 | TXD.1    | ALT5 | 1 |  5 || 6  | 1 | ALT5 | RXD.1    | 3   | 102  |
 |      |     |          |      |   |  7 || 8  |   |      |          |     |      |
 |      |     |          |      |   |  9 || 10 | 1 | ALT3 | GPIO3_C0 | 4   | 112  |
 |  103 |   5 | CTS.1    | ALT5 | 1 | 11 || 12 | 1 | ALT5 | RTS.1    | 6   | 101  |
 |   66 |   7 | GPIO2_A2 |   IN | 1 | 13 || 14 |   |      |          |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
root@orangepir1plus:~#
```

5. 18. 13pin interface GPIO, I2C, UART test

wiringOP has been adapted to the Orange Pi R1 Plus LTS development board, using wiringOP can test the functions of GPIO, I2C and UART

5.18.1. 13pin GPIO port test

1) Below, take pin 5-corresponding to GPIO as GPIO3_A4--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@orangepir1plus:~# gpio readall
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     | 5V       |      |   |  1 || 2  |   |      | GND      |     |      |
 |   89 |   0 | SDA.0    |   IN | 1 |  3 || 4  | 1 | IN   | SCK.0    | 1   | 88   |
 |  100 |   2 | TXD.1    | ALT5 | 1 |  5 || 6  | 1 | ALT5 | RXD.1    | 3   | 102  |
 |      |     |          |      |   |  7 || 8  |   |      |          |     |      |
 |      |     |          |      |   |  9 || 10 | 1 | ALT3 | GPIO3_C0 | 4   | 112  |
 |  103 |   5 | CTS.1    | ALT5 | 1 | 11 || 12 | 1 | ALT5 | RTS.1    | 6   | 101  |
 |   66 |   7 | GPIO2_A2 |   IN | 1 | 13 || 14 |   |      |          |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
```

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

```
root@orangepir1plus-lts:~/wiringOP# gpio mode 2 out
```

Use gpio readall to see that the mode of pin 5 has changed to out

```
root@orangepir1plus:~# gpio mode 2 out
root@orangepir1plus:~# gpio readall
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     | 5V       |      |   |  1 || 2  |   |      | GND      |     |      |
 |   89 |   0 | SDA.0    |   IN | 1 |  3 || 4  | 1 | IN   | SCK.0    | 1   | 88   |
 |  100 |   2 | TXD.1    |  OUT | 0 |  5 || 6  | 1 | ALT5 | RXD.1    | 3   | 102  |
 |      |     |          |      |   |  7 || 8  |   |      |          |     |      |
 |      |     |          |      |   |  9 || 10 | 1 | ALT3 | GPIO3_C0 | 4   | 112  |
 |  103 |   5 | CTS.1    | ALT5 | 1 | 11 || 12 | 1 | ALT5 | RTS.1    | 6   | 101  |
 |   66 |   7 | GPIO2_A2 |   IN | 1 | 13 || 14 |   |      |          |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
```

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@orangepir1plus-lts:~/wiringOP# gpio write 2 0
```

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

```
root@orangepir1plus:~# gpio readall
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     | 5V       |      |   |  1 || 2  |   |      | GND      |     |      |
 |   89 |   0 | SDA.0    |   IN | 1 |  3 || 4  | 1 | IN   | SCK.0    | 1   | 88   |
 |  100 |   2 | TXD.1    |  OUT | 0 |  5 || 6  | 1 | ALT5 | RXD.1    | 3   | 102  |
 |      |     |          |      |   |  7 || 8  |   |      |          |     |      |
 |      |     |          |      |   |  9 || 10 | 1 | ALT3 | GPIO3_C0 | 4   | 112  |
 |  103 |   5 | CTS.1    | ALT5 | 1 | 11 || 12 | 1 | ALT5 | RTS.1    | 6   | 101  |
 |   66 |   7 | GPIO2_A2 |   IN | 1 | 13 || 14 |   |      |          |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
```

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@orangepir1plus-lts:~/wiringOP# gpio write 2 1
```

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

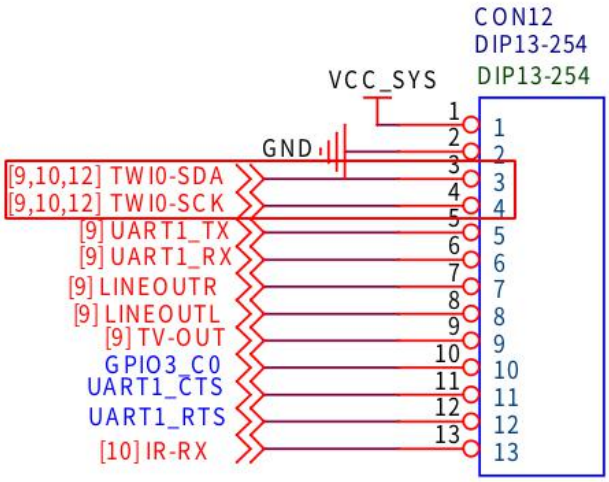
```
root@orangepir1plus:~# gpio readall
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     | 5V       |      |   |  1 || 2  |   |      | GND      |     |      |
 |   89 |   0 | SDA.0    |   IN | 1 |  3 || 4  | 1 | IN   | SCK.0    | 1   | 88   |
 |  100 |   2 | TXD.1    |  OUT | 1 |  5 || 6  | 1 | ALT5 | RXD.1    | 3   | 102  |
 |      |     |          |      |   |  7 || 8  |   |      |          |     |      |
 |      |     |          |      |   |  9 || 10 | 1 | ALT3 | GPIO3_C0 | 4   | 112  |
 |  103 |   5 | CTS.1    | ALT5 | 1 | 11 || 12 | 1 | ALT5 | RTS.1    | 6   | 101  |
 |   66 |   7 | GPIO2_A2 |   IN | 1 | 13 || 14 |   |      |          |     |      |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+ R1 Plus  +---+---+--+----------+-----+------+
```

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

5. 18. 2. 13pin I2C test

1) The linux5.10 OS turns off the i2c controller in 13pin by default in dts. If you need to use i2c, you first need to open the configuration of i2c first. The opening method of i2c in Linux5.10 OS is as follows:

a. According to the schematic diagram of 13pin, 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 rockchip-i2c0.dtbo
2698 bytes read in 8 ms (329.1 KiB/s)
Applying kernel provided DT fixup script (rockchip-fixup.scr)
```

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

```
root@orangepir1plus-lts:~# ls /dev/i2c*
/dev/i2c-0  /dev/i2c-1
```

e. The correspondence between different i2c device nodes is shown below, where

a) i2c0 in 13pin corresponds to /dev/i2c-0

```
root@orangepir1plus:~# ls /sys/class/i2c-adapter/i2c-* -lh
lrwxrwxrwx 1 root root 0 Dec 18 03:45 /sys/class/i2c-adapter/i2c-0 -> ../../devices/platform/ff150000.i2c/i2c-0
lrwxrwxrwx 1 root root 0 Dec 18 03:45 /sys/class/i2c-adapter/i2c-1 -> ../../devices/platform/ff160000.i2c/i2c-1
root@orangepir1plus:~#
```

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

```
root@orangepir1plus-lts:~# apt update
root@orangepir1plus-lts:~# apt install i2c-tools
```

3) Then connect an i2c device to the i2c0 pin of the 13pin connector

	i2c0

sda pin	corresponds to pin 3
sck pin	corresponds to pin 4
vcc pin	corresponds to pin 1
gnd pin	corresponds to pin 2

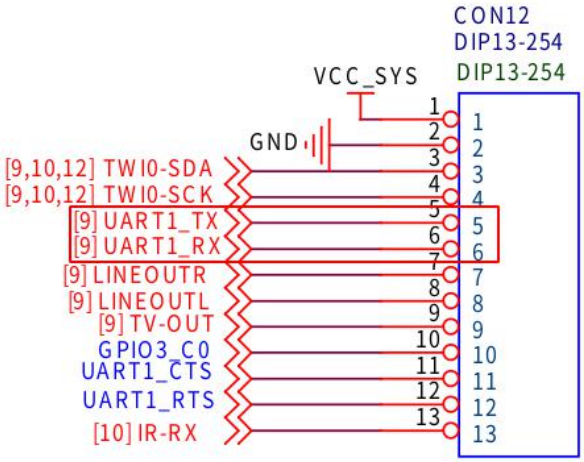
4) Then use the i2cdetect -y 0 command if the address of the connected i2c device can be detected, it means that i2c can be used normally

```
root@orangepir1plus:~# i2cdetect -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- 38 -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- 51 -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

5. 18. 3. 13pin UART test

1) The uart controller in 13pin is closed by default in the Linux5.10 OS in dts. If you need to use uart, you need to open the configuration of uart first. The opening method of uart in Linux5.10 OS is as follows:

a. According to the schematic diagram of 13pin, the uart available on the development board is uart1



b. Then set overlays=uart1 in /boot/orangepiEnv.txt to open the configuration of uart1

```
overlays=uart1
```

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

```
Applying kernel provided DT overlay rockchip-uart1.dtbo
2698 bytes read in 8 ms (329.1 KiB/s)
Applying kernel provided DT fixup script (rockchip-fixup.scr)
```

d. After the system starts, you can see the information of ttyS1 under /sys/class/tty, uart1 in 13pin corresponds to /dev/ttyS1

```
root@orangepir1plus:~# ls /sys/class/tty/ttyS* -lh
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS0 -> ../../devices/platform/serial8250/tty/ttyS0
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS1 -> ../../devices/platform/ff120000.serial/tty/ttyS1
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS2 -> ../../devices/platform/ff130000.serial/tty/ttyS2
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS3 -> ../../devices/platform/serial8250/tty/ttyS3
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS4 -> ../../devices/platform/serial8250/tty/ttyS4
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS5 -> ../../devices/platform/serial8250/tty/ttyS5
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS6 -> ../../devices/platform/serial8250/tty/ttyS6
lrwxrwxrwx 1 root root 0 Jan 21  2016 /sys/class/tty/ttyS7 -> ../../devices/platform/serial8250/tty/ttyS7
root@orangepir1plus:~#
```

2) Then start to test the uart interface, **first use the Dupont line to short-circuit the rx and tx of the uart1 interface to be tested**

	uart1
tx pin	corresponds to pin 5
rx pin	corresponds to pin 6

3) Then modify the serial device node name opened by the serial test program serialTest in wiringOP to **/dev/ttyS1**

```
root@orangepir1plus-lts:~/wiringOP/examples# vim serialTest.c
```

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

  if ((fd = serialOpen ("/dev/ttyS1", 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@orangepir1plus-lts:~/wiringOP/examples# make serialTest
```

```
[CC] serialTest.c
[link]
root@orangepir1plus-lts:~/wiringOP/examples#
```

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

```
root@orangepir1plus-lts:~/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
```

5.19. Method of outputting kernel print information to 13pin serial port

The kernel console outputs to ttyS2 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 UART1 in the 13pin interface, as follows:

1) For the linux5.10 OS, you need to open the configuration of uart1 first. For the detailed configuration method, see 13pin UART test

2) Then modify console=ttyS2 in /boot/boot.cmd to console=ttyS1

```
root@orangepir1plus-lts:~# vim /boot/boot.cmd
```

```
if test "${console}" = "display" || test "${console}" = "both"; then setenv consoleargs "console=ttyS1,1500000 console=tty1"; fi
if test "${console}" = "serial" || test "${console}" = "both"; then setenv consoleargs "console=ttyS1,1500000 ${consoleargs}"; fi
if test "${earlycon}" = "on"; then setenv consoleargs "earlycon ${consoleargs}"; fi
if test "${bootlogo}" = "true"; then setenv consoleargs "bootsplash.bootfile=bootsplash.orangepi ${consoleargs}"; fi
```

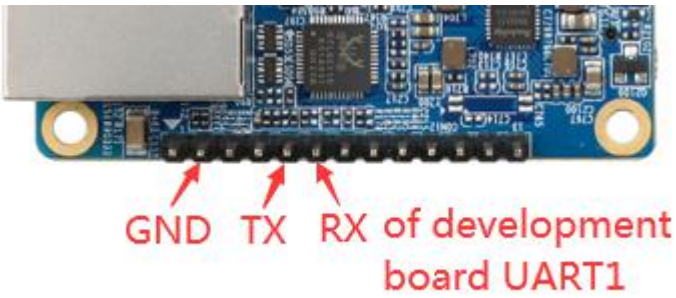
3) Then recompile /boot/boot.cmd to /boot/boot.scr (operate in the linux OS of the development board)

```
root@orangepir1plus-lts:~# mkimage -C none -A arm -T script -d /boot/boot.cmd /boot/boot.scr
```

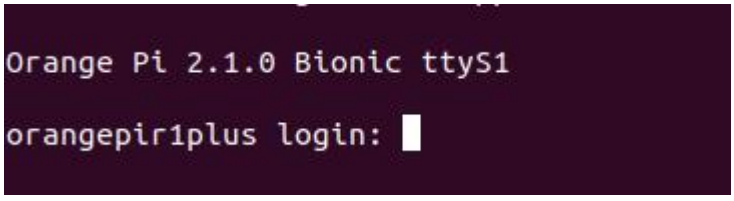
```
Image Name:
Created:          Tue Dec    8 02:35:43 2020
Image Type:       ARM Linux Script (uncompressed)
Data Size:        2448 Bytes = 2.39 KiB = 0.00 MiB
Load Address: 00000000
Entry Point:    00000000
Contents:
    Image 0: 2440 Bytes = 2.38 KiB = 0.00 MiB
```

4) Then connect the USB to TTL module to the UART1 pin of the 13pin interface through the Dupont line

a. Connect the GND of the USB to TTL module to the GND of the 13pin interface of the development board
b. Connect the RX of the USB to TTL module to the TX of the development board UART1
c. The TX of the USB to TTL module is connected to the RX of the development board UART1



5) Then restart the development board, you can see that the kernel console outputs to ttyS1 by default. Note that the output log of u-boot is still output to ttyS2, not ttyS1



5. 20. View the serial number of the RK3328 chip

1) The command to view the serial number of the RK3328 chip is as follows, the serial number of each chip is different, so you can use the serial number to distinguish multiple

development boards

```
root@orangepir1plus-lts:~# cat /sys/devices/platform/board/info
Hardware        : ORANGEPI-R1PLUS
Revision        : 0002
Serial          : 9b25e2e5a704467c

Model           : OrangePi R1PLUS
Manufacturer    : Shenzhen Xunlong Software CO.,Limited
```

5.21. Method to restart the system

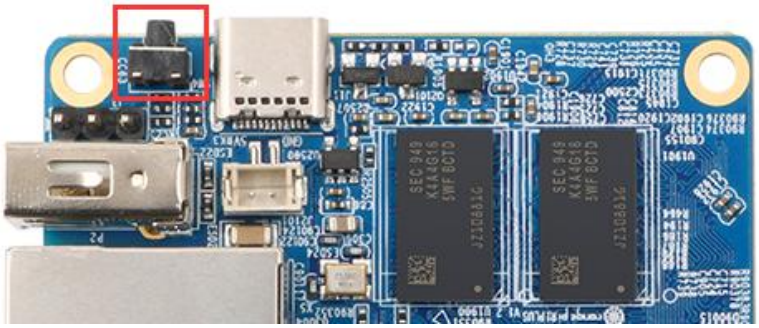
1) Restart using the reboot command

```
root@orangepir1plus-lts:~# reboot
```

2) Use the poweroff command to shut down. If you need to start up, you need to re-plug the power supply

```
root@orangepir1plus-lts:~# poweroff
```

3) Short press the reset button on the development board to restart the system



6. Linux SDK instructions

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

Linux SDK on WSL

6.1. Get the source code of linux sdk

6.1.1. Download orangepi-build from github

1) First download the code of orangepi-build. The code of orangepi-build is modified based on the armbian build system. Currently, the rk3328 series development board supports the current branch.

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

2) The current branch generally uses u-boot and kernel close to the mainline version. The u-boot and linux kernel currently used by the rk3328 series development board are as follows

Branch	u-boot version	linux kernel version
current	u-boot 2020.10	linux5.10.44

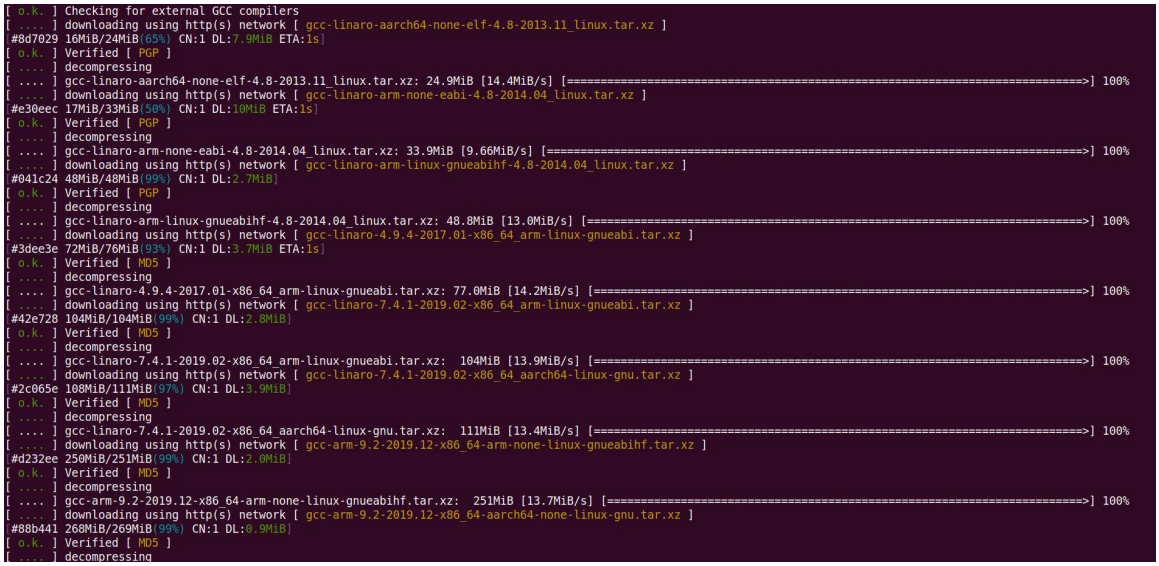
3) After orangepi-build is downloaded, it will contain the following files and folders

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

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

6.1.2. Download cross compilation toolchain

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 rk3328 linux5.10 kernel source code is

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

5) The cross-compilation tool chain used to compile the rk3328 u-boot 2020.10 source code is

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

6.1.3. Orangepi-build complete directory structure description

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

a. The git repository where the linux5.10 kernel source code is stored is as follows

https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.10-rockchip64/

b. The git repository of b.u-boot 2020.10 source code is as follows

https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2020.10-rockchip64

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

a. **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**: Store the source code of the linux kernel. The folder named **orange-pi-5.10-rockchip64** stores the kernel source code of the current branch of the rk3328 development board. Please do not manually modify the name of the kernel source folder. If you modify it The kernel source code will be re-downloaded when the build 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**: The folder named **v2020.10-rockchip64** in the u-boot source code stores the u-boot source code of the current branch of the rk3328 development board. Please do not modify the name of the u-boot source code folder manually. If you modify it The u-boot source code will be re-downloaded when the compilation

system is running

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

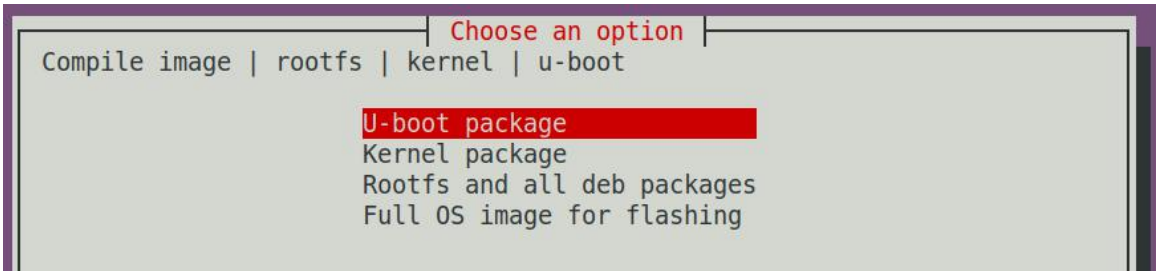
```
test@test:~/orangepi-build$ ls
build.sh  external  kernel  LICENSE  output  README.md  scripts
toolchains  u-boot  userpatches
```

6. 2. Compile u-boot

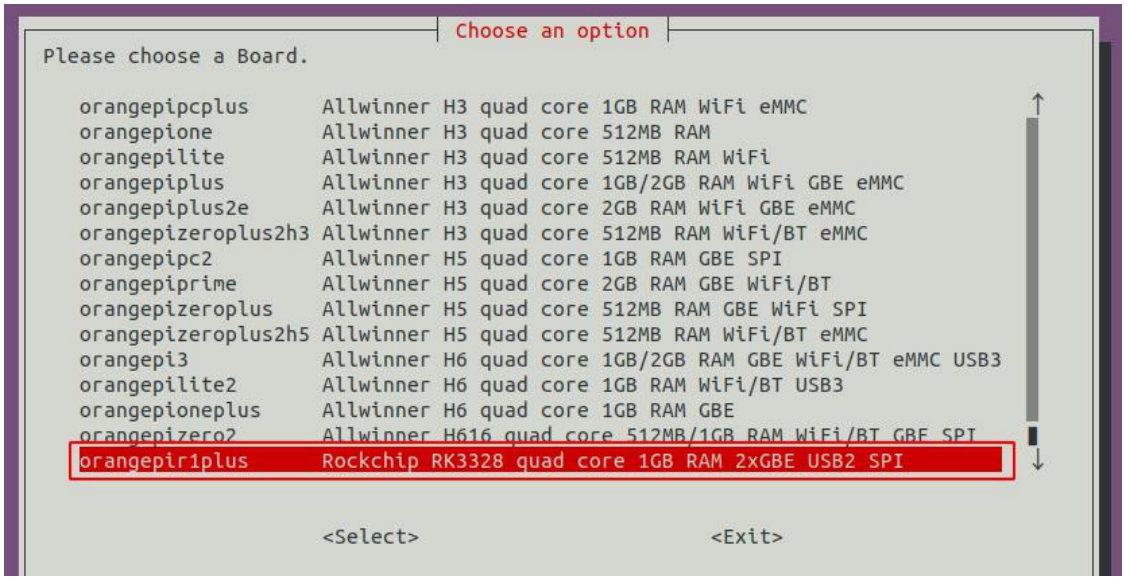
1) Run the build.sh script, remember to add sudo permissions

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

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



3) Then select the model of the development board



4) Then it will start to compile u-boot, some of the information prompted during

compilation are explained as follows

a. u-boot source version

```
[ o.k. ] Compiling u-boot [ v2020.10 ]
```

b. The version of the cross-compilation toolchain

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

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

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

d. Package name of u-boot deb package generated by compiling

```
[ o.k. ] File name [ linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb ]
```

e. Compilation time

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

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

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh    BOARD=orangepir1plus-lts
BRANCH=current BUILD_OPT=u-boot BUILD_DESKTOP=no
KERNEL_CONFIGURE=yes ]
```

5) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb
```

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

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \
linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb .      (Note that there is a
  "." at the end of the command)
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb    usr
```

b. The decompressed file is as follows

```
test@test:~orangepi-build/output/debs/u-boot$ tree usr
usr
└── lib
    ├── linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64
```

```
    ├──idbloader.bin
    ├──uboot.img          //u-boot binary file
    └──trust.bin
└── u-boot
    ├── LICENSE
    ├── orangepi_r1_plus_rk3328_defconfig
    └── platform_install.sh

3 directories, 6 files
```

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

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

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

a. Upload the compiled u-boot deb package to the Linux OS of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb root@192.168.1.xxx:/root
```

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

```
root@orangepir1plus-lts:~# apt purge -y linux-u-boot-orangepir1plus-lts-current
```

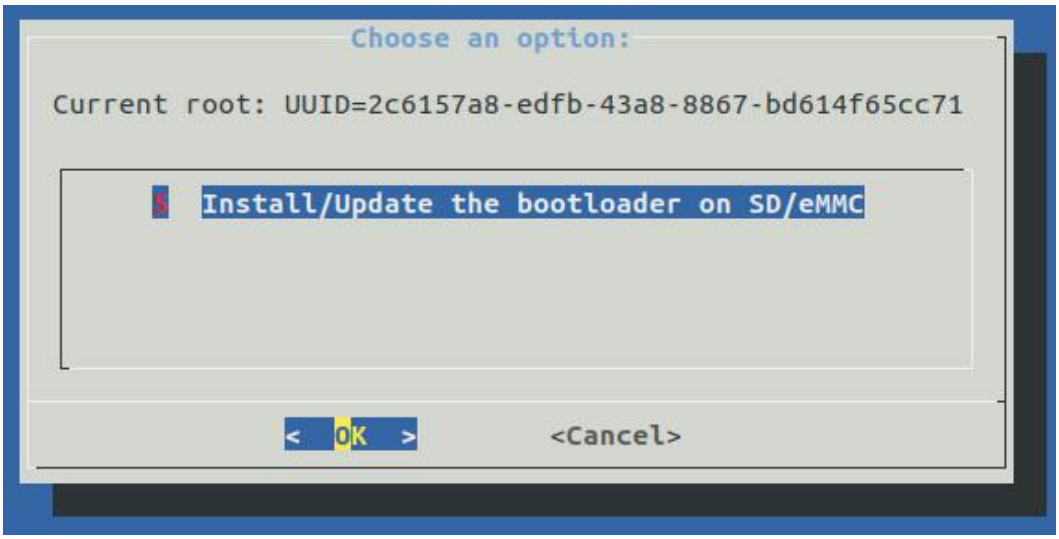
c. Install the new u-boot deb package just uploaded

```
root@orangepir1plus-lts:~#
dpkg -i linux-u-boot-current-orangepir1plus-lts_2.1.0_arm64.deb
```

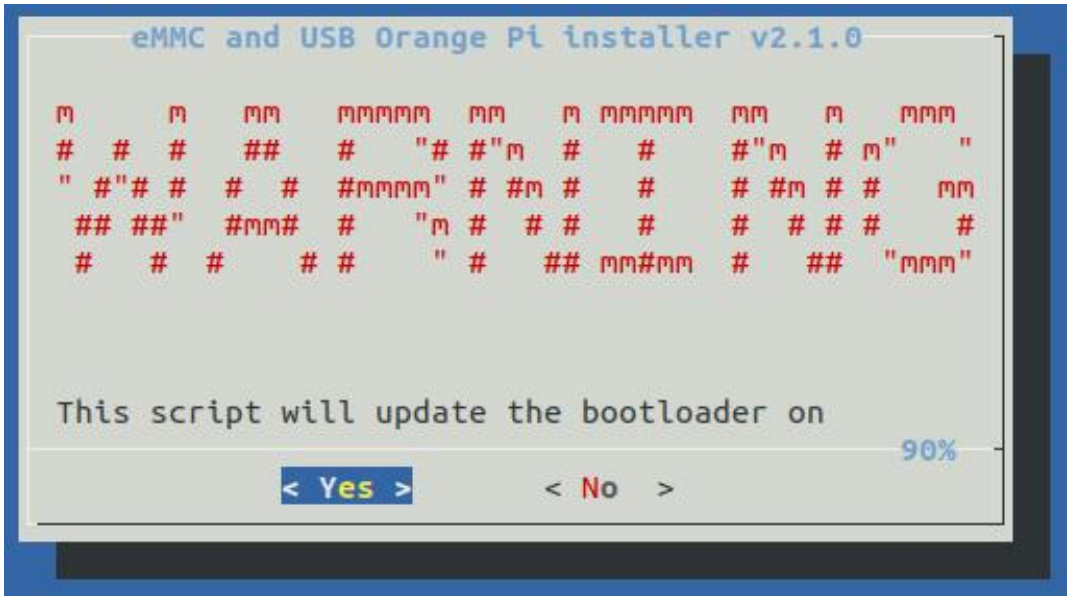
d. Then run the nand-sata-install script

```
root@orangepir1plus-lts:~# nand-sata-install
```

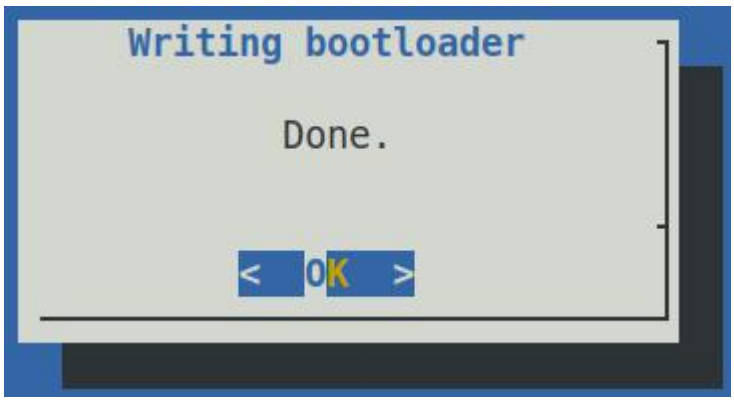
e. Then select 5 **Install/Update the bootloader on SD/eMMC**



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



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



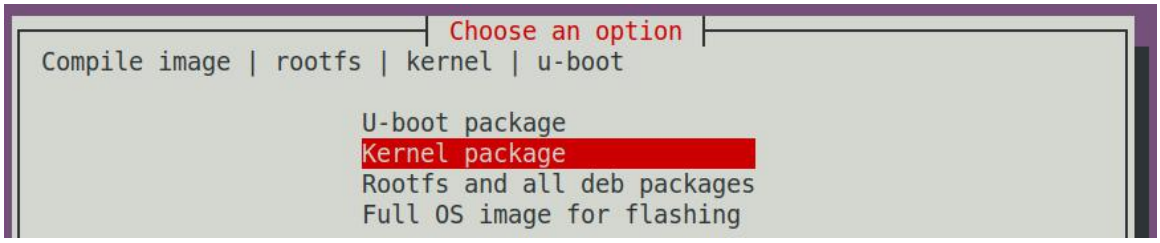
h. Then you can restart the development board to test whether the u-boot modification takes effect

6. 3. Compile the Linux kernel

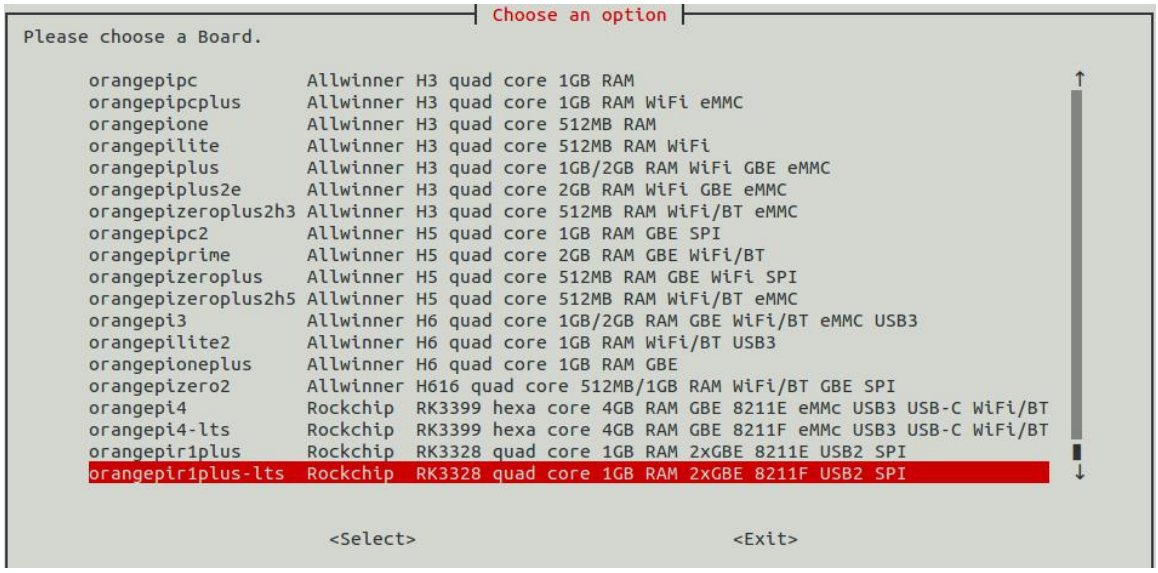
1) Run the build.sh script, remember to add sudo permissions

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

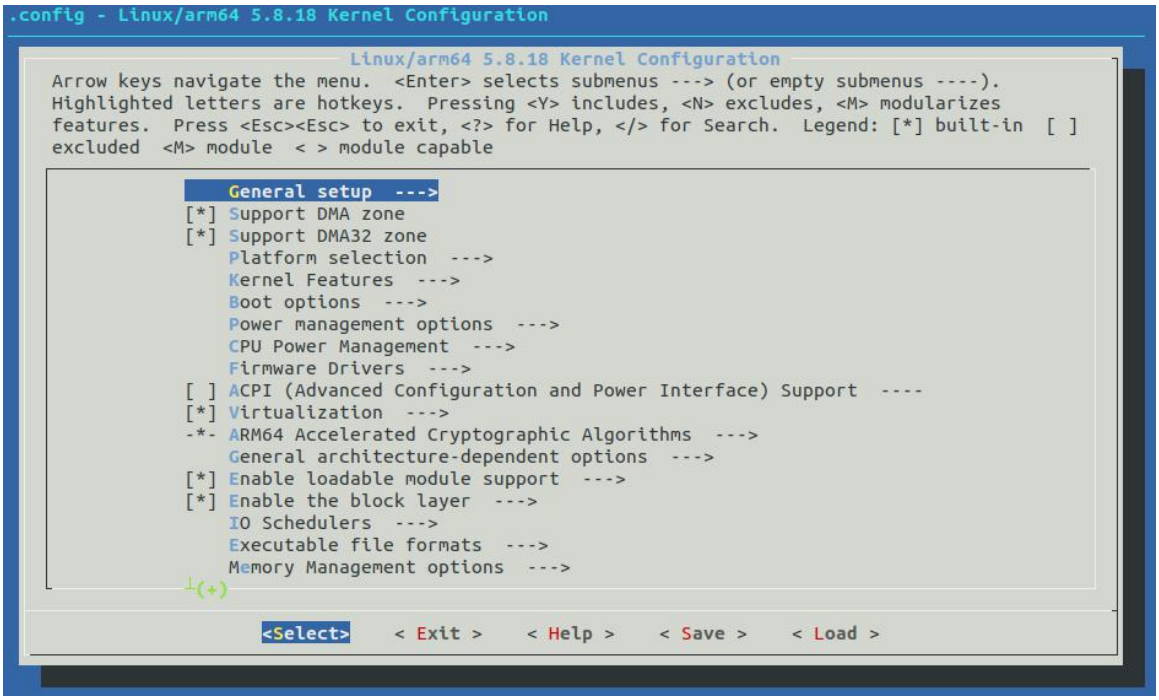
2) Select **Kernel package**, then press Enter



3) Then select the model of the development board



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



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

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

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

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

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

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

a. Linux kernel source code version

```
[ o.k. ] Compiling legacy kernel [ 5.10.44 ]
```

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-rockchip64-current.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-rockchip64-current.config ]
```

e. The path of the deb package related to the compiled kernel

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

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

```
[ o.k. ] File name [ linux-image-current-rockchip64_2.1.4_arm64.deb ]
```

g. Compile time

```
[ o.k. ] Runtime [ 25 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=orangepir1plus-lts
BRANCH=current BUILD_OPT=kernel RELEASE=bionic BUILD_DESKTOP=no
KERNEL_CONFIGURE=yes ]
```

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

a.linux-dtb-current-rockchip64_2.1.4_arm64.deb contains the dtb file used by the kernel

b.linux-headers-current-rockchip64_2.1.4_arm64.deb contains kernel header files

c.linux-image-current-rockchip64_2.1.4_arm64.deb contains kernel image and kernel module

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-current-rockchip64_2.1.4_arm64.deb
output/debs/linux-headers-current-rockchip64_2.1.4_arm64.deb
output/debs/linux-image-current-rockchip64_2.1.4_arm64.deb
```

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

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-current-rockchip64_2.1.4_arm64.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-current-rockchip64_2.1.4_arm64.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot  etc  lib  linux-image-current-rockchip64_2.1.4_arm64.deb  usr
```

b. The decompressed file is as follows

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

```
│   ├── config-5.10.44-rockchip64        //编译内核源码使用的配置文件
│   ├── System.map-5.10.44-rockchip64
│   └── vmlinuz-5.10.44-rockchip64       //编译生成的内核镜像文件
├── etc
│   └── kernel
├── lib
│   └── modules                          //编译生成的内核模块
├── linux-image-current-rockchip64_2.1.4_arm64.deb
└── usr
    ├── lib
    └── share
8 directories, 4 files
```

8) When the orangepi-bulid compilation system compiles the linux kernel source code, it first synchronizes the linux kernel source code with the linux kernel source code of the github server, so if you want to modify the 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 be turned off after the linux kernel source code, otherwise it will be prompted that the linux kernel 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) If you modify the kernel, you can use the following method to update the kernel and kernel modules of the development board Linux OS

a. Upload the compiled linux kernel deb package to the Linux OS of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-image-current-rockchip64_2.1.4_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@orangepir1plus-lts:~# apt purge -y linux-image-current-rockchip64
```

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

```
root@orangepir1plus-lts:~# dpkg -i linux-image-current-rockchip64_2.1.4_arm64.deb
```

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

10) The method of installing the kernel header file into the Linux OS is as follows

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

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-headers- current-rockchip64_2.1.4_arm64.deb root@192.168.1.xxx:/root
```

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

```
root@orangepir1plus-lts:~# dpkg -i linux-headers-current-rockchip64_2.1.4_arm64.deb
```

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

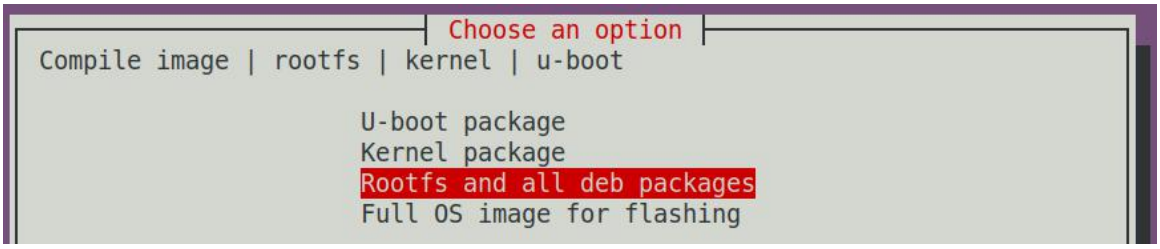
```
root@orangepir1plus-lts:~# ls /usr/src
linux-headers-5.10.44-rockchip64
root@orangepir1plus-lts:~# ls /usr/src/linux-headers-5.10.44-rockchip64
arch crypto fs ipc lib Module.symvers scripts tools block Documentation include Kconfig Makefile net security usr certs drivers init kernel mm samples sound virt
```

6. 4. Compile rootfs

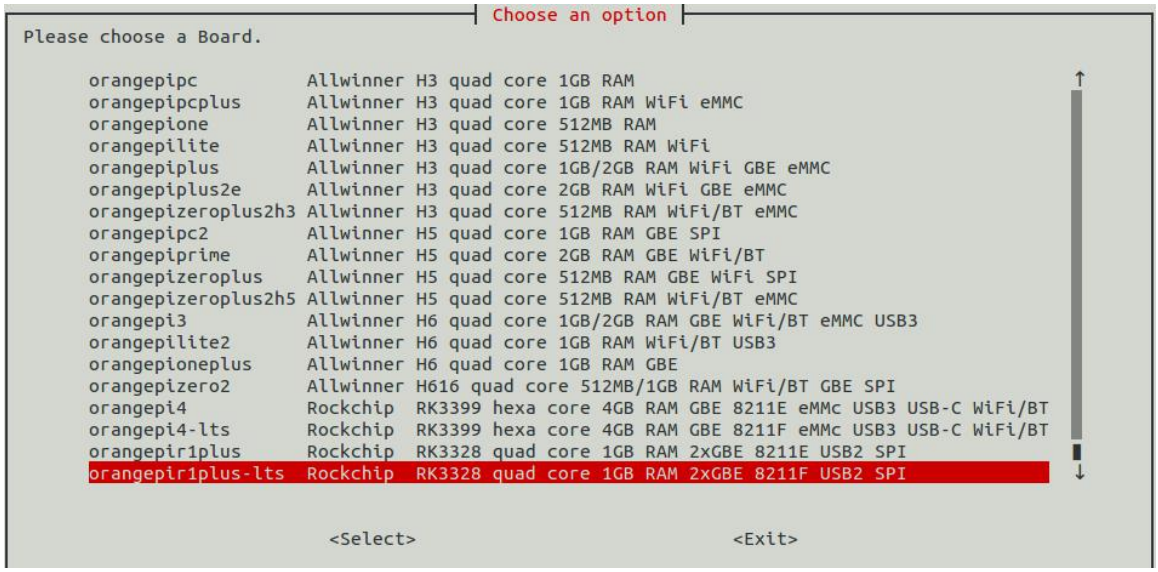
1) Run the build.sh script, remember to add sudo permissions

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

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



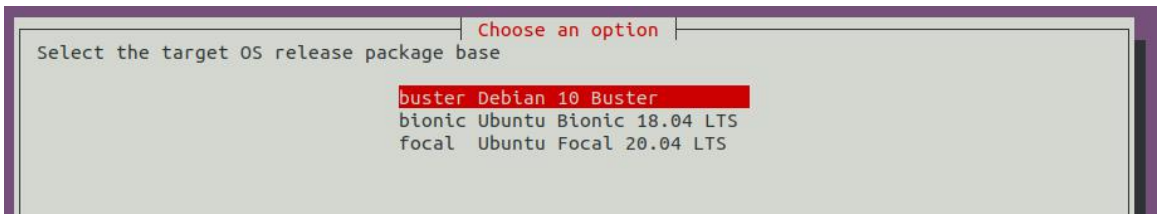
3) Then select the model of the development board



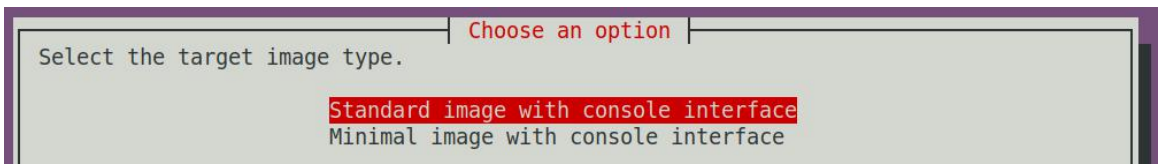
4) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04

The Linux distributions supported by linxu5.10 are as follows



5) If you are compiling the server version of the image, 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.



6) After selecting the type of image, rootfs will be compiled, 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 bionic]
```

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 [ bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4 ]
```

d. Compilation time

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

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

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh    BOARD=orangepir1plus-lts
BRANCH=current BUILD_OPT=rootfs RELEASE=bionic
BUILD_MINIMAL=no    BUILD_DESKTOP=no
KERNEL_CONFIGURE=yes    ]
```

7) View the compiled rootfs compressed package

a. **bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4**is a compressed package of rootfs, the meaning of each field of the name is

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

b) **cli** cli indicates that rootfs is a server 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. **bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list** lists the package names of all packages installed by rootfs

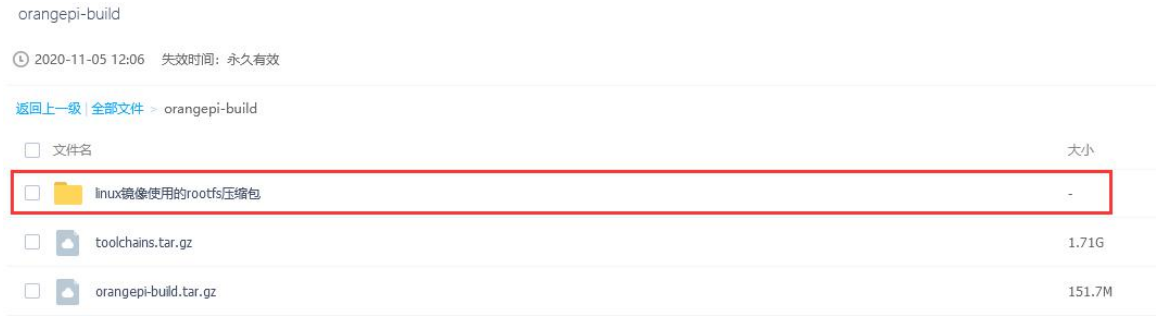
```
test@test:~/orangepi-build$ ls external/cache/rootfs/
bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4
bionic-cli-arm64.153618961f14c28107ca023429aa0eb9.tar.lz4.list
```

8) 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

download and compilation time

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

链接：https://pan.baidu.com/s/1vWQmCmSYdH7iCDFyKpJtVw
提取码：zero

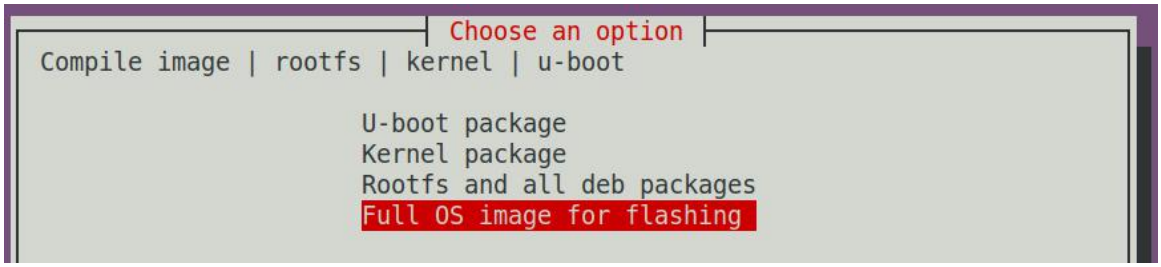


6. 5. Compile linux image

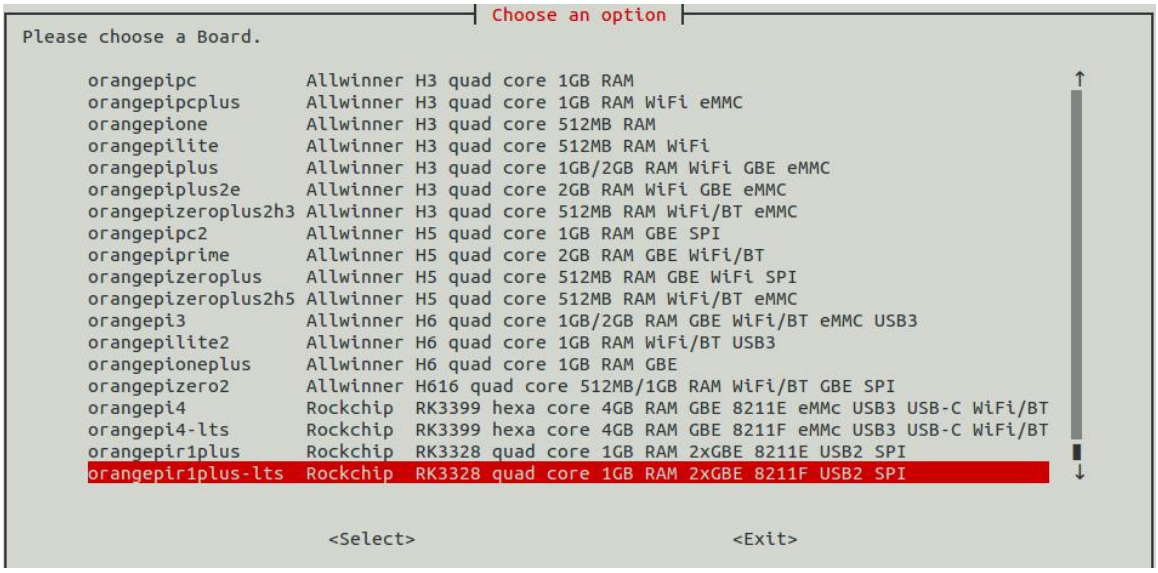
1) Run the build.sh script, remember to add sudo permissions

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

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



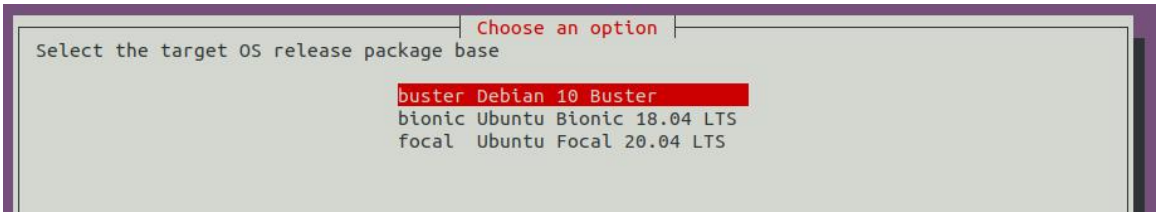
3) Then select the model of the development board



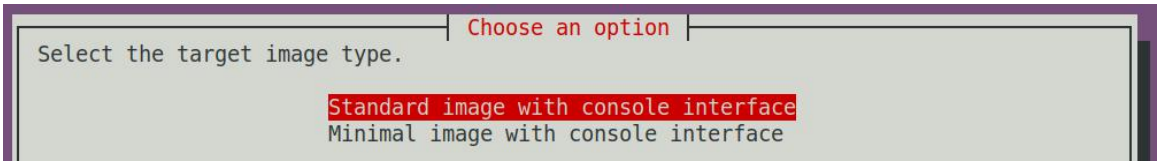
4) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04

The Linux distributions supported by linxu5.10 are as follows



5) If you are compiling the server version of the image, 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.



6) 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 is cached, only update the code)
c. Compile u-boot source code and generate u-boot deb package
d. Compile linux source code, generate linux related deb package
e. Make deb package of linux firmware
f. Make deb package of orangepi-config tool
g. Make board-level support deb package
h. Check whether the rootfs has been cached, if there is no cache, then re-create the rootfs, if it has been cached, just unzip and use
i. Install the previously generated deb package into rootfs
j. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages, modifying system configurations, etc.
k. Then make an image file and format the partition, the default type is ext4
l. Then copy the configured rootfs to the image partition
m. Then update the initramfs
n. Finally, the bin file of u-boot is written to the image through the dd command

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

a. The storage path of the compiled image

```
[ o.k. ] Done building
[ output/images/Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44/Orangepir1plus-lts_2.1.4_ubuntu_bionic_server_linux5.10.44.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=orangepir1plus-lts BRANCH=current BUILD_OPT=image RELEASE=bionic BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

7. Android OS instructions

7.1. Supported Android version

Android version	Kernel version
Android 9.0	Linux4.4

7.2. Android 9.0 function adaptation situation

Function	状态
USB2.0	OK
TF card start	OK
USB to Gigabit network	OK
Gigabit network card	OK
Infrared	OK
CVBS video	OK
Headphone audio	OK
LED light	OK
Reset button	OK
ADB debugging	OK

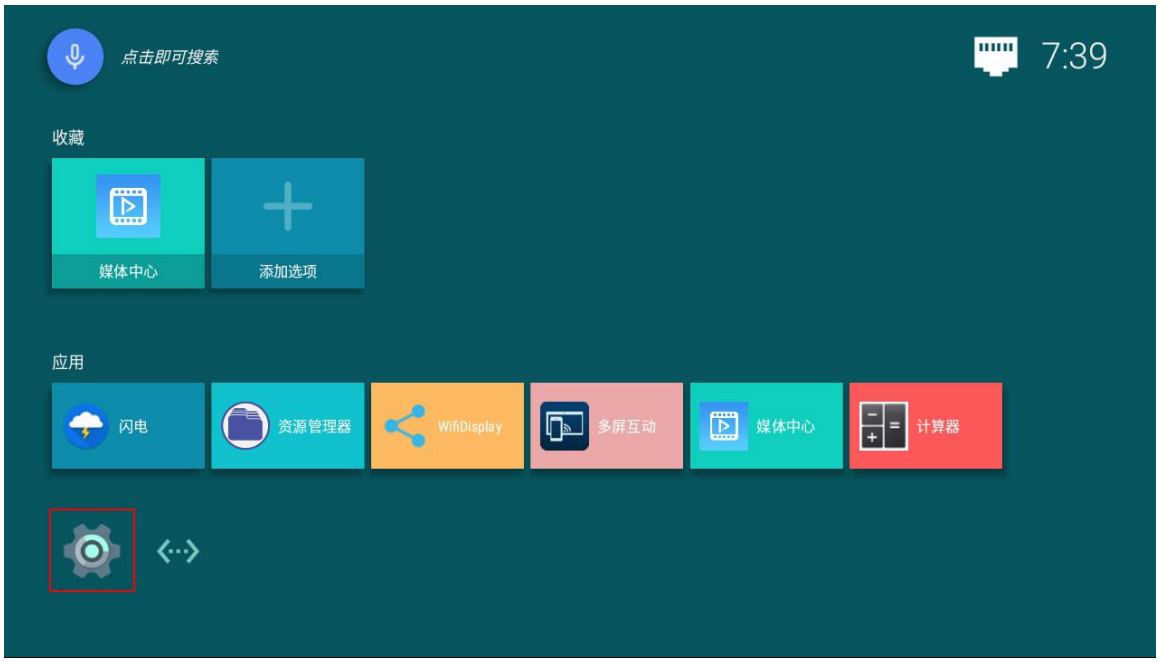
7.3. Onboard LED light display description

Power status light (red light)	The system starts, the red light flashes
Wan port status light (yellow light)	Wan port is connected to the network cable, the yellow light is always on, the Wan port is unplugged, the yellow light is off
Lan port status light (yellow light)	Lan port is connected to the network cable, the yellow light is always on, the Lan port is unplugged, the yellow light is off

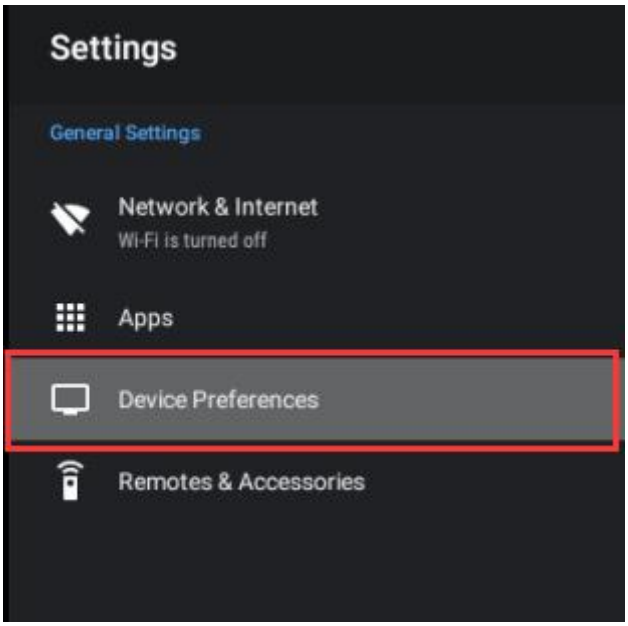
7. 4. How to use ADB

7. 4. 1. Open USB debugging option

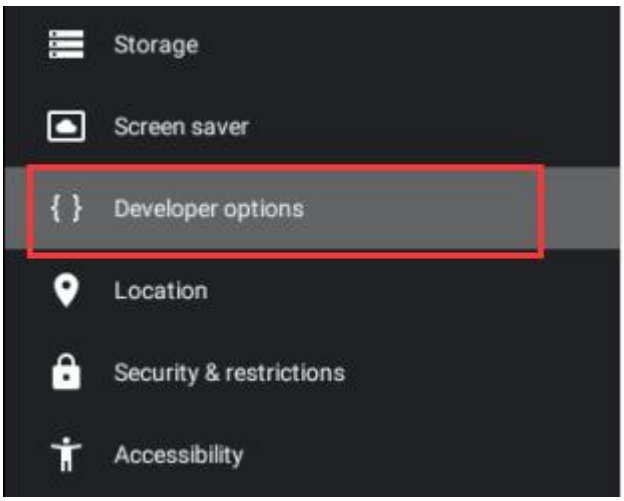
1) The USB debugging mode is turned on by default, and the ADB debugging can be used directly by default. If there is a problem with the use of ADB, you can use the following method to turn it on, first select settings



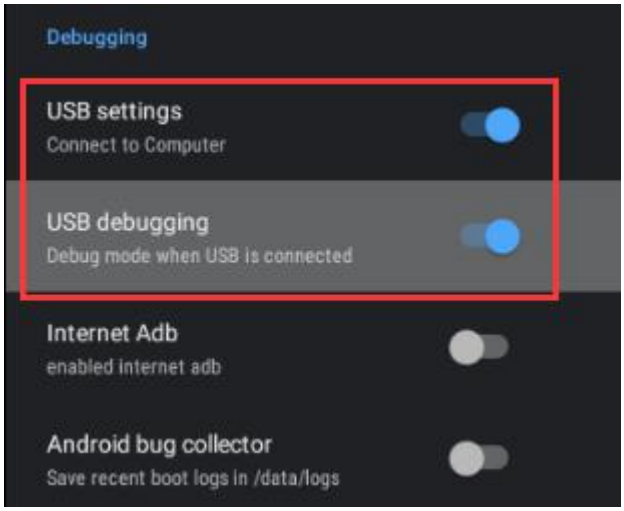
2) Then select device preferences



3) Then **open the developer options**



4) Then **confirm the USB connection status and USB debugging**



7. 4. 2. Use network connection adb debugging

1) To use the network adb, there is no need to use the Type-C interface data cable to connect the computer and the development board, but to communicate through the network, so first make sure that the wired 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 OS is set to port number 5555

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

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

```
console:/ # setprop service.adb.tcp.port 5555
console:/ # stop adbd
console:/ # start adbd
```

5) Install adb tool on Ubuntu PC

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

6) 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
```

7) Then you can log in to the Android OS through adb shell on the Ubuntu PC

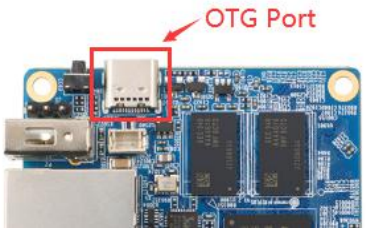
```
test@test:~$ adb shell
rk3328_box:/ #
```

7. 4. 3. Use data cable to connect adb for debugging

1) The USB debugging mode is turned on by default, and ADB debugging can be used directly by default. If there is a problem with the use of ADB, you need to turn on the USB debugging function in the system settings first

2) Then you need to use the Type-C interface data cable to connect the development board to the USB interface of the computer (make sure that the computer power supply is

sufficient for the development board to work normally)



3) Install adb tool on Ubuntu PC

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

4) View the identified ADB device

```
test@test:~$ adb devices
List of devices attached
20080411      device
```

5) Then you can log in to the Android OS through adb shell on the Ubuntu PC

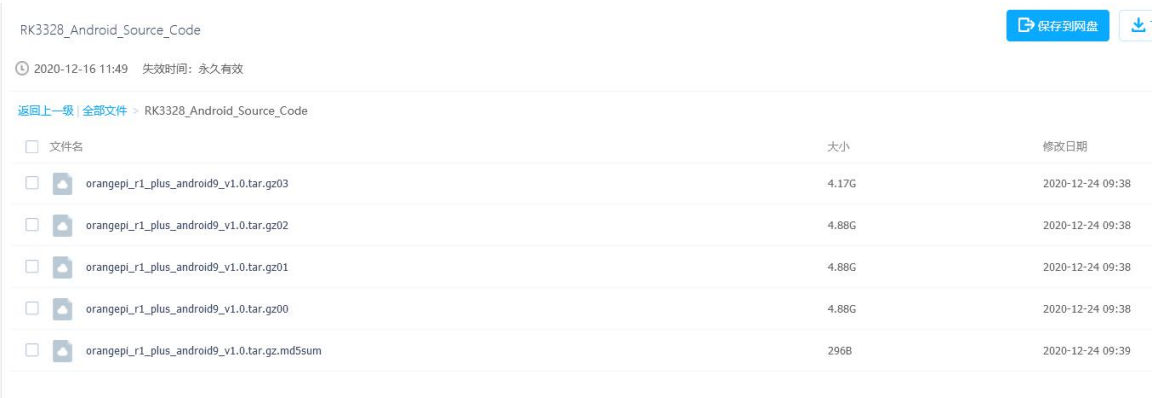
```
test@test:~$ adb shell
Rk3328-box:/ #
```

8. Android SDK instructions

The Android SDK is compiled on a PC with **Ubuntu 18.04** installed. Other versions of Ubuntu OS may have some differences

8. 1. Download the source code of Android SDK

1) First download the sub-volume compressed package of Android SDK from Google Drive



2) After downloading the sub-volume compressed package, you can first use the md5sum -c *.md5sum command to calculate whether the checksum is correct. If the prompt is successful, it means that the downloaded image is correct, and you can decompress it without worry. If the **checksum does not match**, it means it was downloaded. There is a problem with the sub-volume compression package, please try to download again

```
test@test:~$ md5sum -c *.md5sum
OrangePi_R1Plus_RK3328_Android9_v1.1.tar.gz00: success
OrangePi_R1Plus_RK3328_Android9_v1.1.tar.gz01: success
OrangePi_R1Plus_RK3328_Android9_v1.1.tar.gz02: success
OrangePi_R1Plus_RK3328_Android9_v1.1.tar.gz03: success
```

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

```
test@test:~$ mkdir OrangePiR1Plus
test@test:~$ cat OrangePi_R1Plus_RK3328_Android9_v1.1.tar.gz* \
 > OrangePiR1Plus.tar.gz
test@test:~$ tar xf OrangePiR1Plus.tar.gz -C OrangePiR1Plus
```

8.2. Build Android compilation environment

4) Because the Android 9.0 source code is too large, in order to avoid unnecessary errors in the compilation and development process, it is recommended that the local environment hardware and software configuration:

a. Operating system: 64-bit Ubuntu 18.04 and above

b. Hard disk space: minimum 150GB or more

5) 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
```

6) 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
```

7) Install platform support software

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

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

8. 3. Compile Android image

8. 3. 1. Compile u-boot

1) Enter the u-boot directory, select the configuration of rk3328 and start compiling u-boot

```
test@ubuntu:~/OrangePi_R1Plus_RK3328/$ cd u-boot
test@ubuntu:~/OrangePi_R1Plus_RK3328/u-boot$ ./make.sh rk3328
make for rk3328_defconfig by -j24
```

2) After compiling, three files of trust.img, rk3328_loader_vx.xx.xxx.bin and uboot.img will be generated

```
uboot version: U-Boot 2017.09-gdc89501087 (Dec 17 2020 - 19:43:15)
pack uboot.img success!
merge success(rk3328_loader_v1.16.250.bin)
merge success(trust.img)

Platform RK3328 is build OK, with new .config(make rk3328_defconfig)
```

8.3.2. Compile the kernel

1) Enter the kenel directory, specify the kernel configuration rockchip_defconfig, and then start to compile the kernel image, rk3328-orangepi-r1-plus-lts corresponds to the file name of the dts used

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$ cd kernel
test@ubuntu:~/OrangePi_R1Plus_RK3328/kernel$
make ARCH=arm64 rockchip_defconfig
test@ubuntu:~/OrangePi_R1Plus_RK3328/kernel$
make ARCH=arm64 rk3328-orangepi-r1-plus-lts.img -j8
```

2) After compiling, generate three files resource.img, boot.img and zboot.img

```
Pack to resource.img successed!
  Image: resource.img (with rk3328-orangepi-r1-plus-lts.dtb logo.bmp logo_kernel.bmp) is ready
  Image: boot.img (with Image resource.img) is ready
  Image: zboot.img (with Image.lz4 resource.img) is ready
```

8.3.3. Compile Android

After configuring the JDK environment variables according to the actual compilation environment, follow the steps below and execute make

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$ source build/envsetup.sh
test@ubuntu:~/OrangePi_R1Plus_RK3328$ lunch rk3328_box-eng
test@ubuntu:~/OrangePi_R1Plus_RK3328$ make -j8
```

8.3.4. Firmware packaging

After completing the above compilation, execute the mkimage.sh script in the root directory of the SDK to generate the firmware, and all the images required for programming will be copied to the rockdev/Image-rk3328_box directory.

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$ ./mkimage.sh
```

8.3.5. Generate upgrade image

1)After completing the above operations, run the ./make.sh -u script in the SDK root directory to generate the update.img image

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$ ./make.sh -u
```

2)The path where the generated upgrade image is stored is

```
OrangePi_R1Plus_RK3328/rockdev/Image-rk3328_box/update.img
```

8.3.6. Automatically compile scripts

In order to improve the efficiency of compilation and reduce the possible misoperation of manual compilation, a fully automated compilation script is integrated in the SDK to facilitate firmware compilation and backup.

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$    ./make.sh -h
Usage: ./make.sh [ARGS]
Options:
   -A                build Android
   -B                build U-Boot
   -K                build Linux kernel
   -F, --all    build all (U-Boot, kernel, Android)
   -M                 make rockdev image
   -u                generate update.img
   -h                show this help message and exit
```

If you need to compile all functions and package them into a image, you can execute the following commands to complete the entire compilation process

```
test@ubuntu:~/OrangePi_R1Plus_RK3328$    ./make.sh -F -M -u
```