

# UBUNTU安装

---

看视频

## Ubuntu 24.04更换清华源

---

### 1.备份原有的源配置文件：

---

打开终端，输入以下命令来备份原有的源配置文件，也可以直接不备份。

```
sudo cp /etc/apt/sources.list.d/ubuntu.sources  
/etc/apt/sources.list.d/ubuntu.sources.bak
```

### 2.编辑源配置文件：

---

使用vi或其他文本编辑器打开源配置文件：

```
sudo vi /etc/apt/sources.list.d/ubuntu.sources
```

### 3. 添加清华源的配置：

---

在打开的文件中，删除原有的内容，并粘贴以下清华源的配置：

```
Types: deb  
URIs: https://mirrors.tuna.tsinghua.edu.cn/ubuntu/  
Suites: noble noble-updates noble-security  
Components: main restricted universe multiverse  
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg
```

### 4. 更新源：

---

更新源配置并升级系统：

```
sudo apt-get update  
sudo apt-get upgrade
```

通过以上步骤，你可以成功将Ubuntu 24.04的源更换为清华源。

## 2K0300板卡联网

---

下载终端软件：

<https://mobaxterm.mobatek.net/download-home-edition.html>

### 1、手动连接网络

---

2K0300需要先插入天线才能更好和更稳定的连接无线网络。

```
wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "账号" "密码")
sleep 3
wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "账号" "密码")
sleep 3
udhcpc -i wlan0
```

// 连接WIFI

```
wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "账号" "密码")
```

// 获取IP

```
udhcpc -i wlan0
```

连接成功后，输入ipconfig，就可以查看设备的ip地址

```
root@LoongOS:~# ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2418 bytes 183956 (179.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2418 bytes 183956 (179.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.124 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::5e8a:aeff:fe55:72c2 prefixlen 64 scopeid 0x20<link>
    ether 5c:8a:ae:55:72:c2 txqueuelen 1000 (Ethernet)
    RX packets 55970 bytes 59199741 (56.4 MiB)
    RX errors 0 dropped 494 overruns 0 frame 0
    TX packets 4567 bytes 269259 (262.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

通过上面我们可以看到IP地址为192.168.2.124

## 2、开机自动连接网络

使用vi /etc/rc.local命令

可以看到这里面有如下的代码。

```
#!/bin/bash -e
```

```
do_start()
```

```
{
```

```
#systemctl disable dhcpcd
```

```
#systemctl stop dhcpcd
```

```
insmod /usr/lib/modules/4.19.190+/aic8800_bsp.ko
```

```
insmod /usr/lib/modules/4.19.190+/aic8800_fdrv.ko
```

```
sleep 3
```

```
hostapd -B /etc/hostapd.conf -f /var/log/hstap.log &
```

```
ifconfig wlan0 192.168.1.3
```

```
udhcpd -f /etc/udhcpd.conf &
```

```
}
```

```
do_stop()
```

```
{
```

```
rmmod aic8800_fdrv.ko
```

```
rmmod aic8800_bsp.ko
```

```
}
```

```
case "$1" in
```

```
start)
```

```
do_start
```

```
- /etc/rc.local 1/30 3%
```

```
#!/bin/bash -e
```

```
do_start()
```

```
{
```

```
#systemctl disable dhcpcd
```

```
#systemctl stop dhcpcd
```

```
insmod /usr/lib/modules/4.19.190+/aic8800_bsp.ko
```

```
insmod /usr/lib/modules/4.19.190+/aic8800_fdrv.ko
```

```
sleep 3
```

```
hostapd -B /etc/hostapd.conf -f /var/log/hstap.log &
```

```
ifconfig wlan0 192.168.1.3
```

```
udhcpd -f /etc/udhcpd.conf &
```

```
}
```

```
do_stop()
```

```
{
```

```
rmmod aic8800_fdrv.ko
```

```
rmmod aic8800_bsp.ko
```

```
}
```

```
case "$1" in
```

```
start)
```

```
do_start
```

```
- /etc/rc.local 1/30 3%
```

1

删除

```

#!/bin/bash -e

do_start()
{
#systemctl disable dhcpcd
#systemctl stop dhcpcd
insmod /usr/lib/modules/4.19.190+/aic8800_bsp.ko
insmod /usr/lib/modules/4.19.190+/aic8800_fdrv.ko
}

do_stop()
{
rmmod aic8800_fdrv.ko
rmmod aic8800_bsp.ko
}

case "$1" in
    start)
        do_start
        ;;
    stop)
        do_stop
        ;;
    *)
        ;;
esac
I /etc/rc.local [Modified] 10/25 40%

```

然后添加如下代码：

```

wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "SEEKFREE_5G" "SEEKFREEV2") &
sleep 3
udhcpc -i wlan0 &
sleep 3

```

```

wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "账号" "密码") &
udhcpc -i wlan0 &

```

这两条命令后面一定需要添加&，否则可能一直卡在开机中。

```
#!/bin/bash -e

do_start()
{
#systemctl disable dhcpcd
#systemctl stop dhcpcd
insmod /usr/lib/modules/4.19.190+/aic8800_bsp.ko
insmod /usr/lib/modules/4.19.190+/aic8800_fdrv.ko

wpa_supplicant -B -i wlan0 -c <(wpa_passphrase "账号" "密码") &
sleep 3
udhcpc -i wlan0 &
sleep 3
}
do_stop()
{
rmmod aic8800_fdrv.ko
rmmod aic8800_bsp.ko
}
case "$1" in
    start)
        do_start
I /etc/rc.local [Modified] 11/30 36%
```

## 使用ssh传输文件

### 1、安装ssh

```
sudo apt install openssh-client openssh-server
```

输入ssh

```
xiaom@ubuntu24:~/桌面$ ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface] [-b bind_address]
          [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
          [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
          [-J destination] [-L address] [-l login_name] [-m mac_spec]
          [-O ctl_cmd] [-o option] [-P tag] [-p port] [-R address]
          [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
          destination [command [argument ...]]
          ssh [-Q query_option]
```

### 2、使用ssh

新建一个SEEKFREE\_APP文件

```
xiaom@ubuntu24:~/桌面$ vi SEEKFREE_APP
```

随便输入个内容，我这里输入125



使用cat SEEKFREE\_APP指令可以查看该文件的内容。

```
root@LoongOS:~# cat SEEKFREE_APP
125
```

## 安装教程交叉编译器和OpenCV库

安装之前需要先安装open-vm-tools，安装后可以直接通过WINDOWS和UBUNTU之间，复制粘贴文件。

```
sudo apt-get install open-vm-tools
sudo apt-get install open-vm-tools-desktop
sudo apt-get install git
```

拉取开源库

```
git clone https://gitee.com/seekfree/LS2K0300_Library.git
```

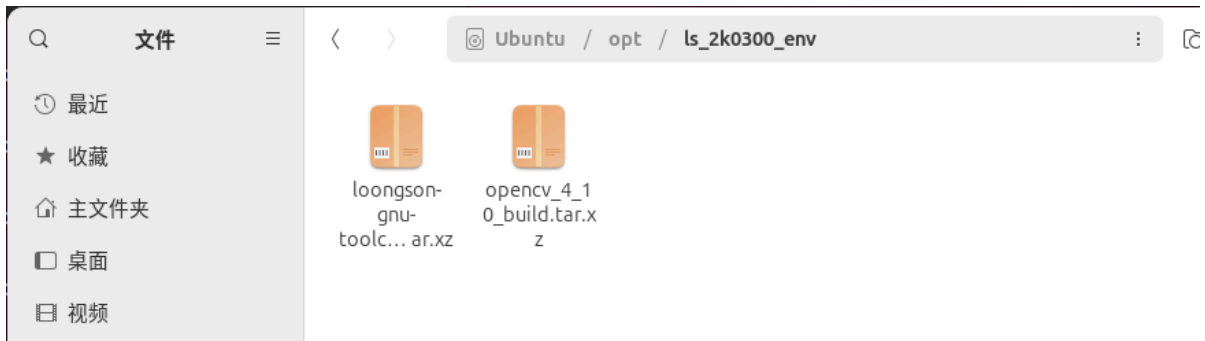
主文件夹 / LS2K0300_Library				大小	修改时间	
名称						
【封装】开源封装				2个项目	今天 14:08	☆
【软件】交叉编译工具 上位机等				3个项目	今天 15:38	☆
【原理图】原理图 丝印图 尺寸图 位号图				5个项目	今天 14:08	☆
Example				4个项目	昨天 18:49	☆
ls2k0300_linux_4.19				34个项目	今天 15:26	☆
Seekfree_LS2K0300_Opensource_Library				3个项目	今天 15:43	☆
LICENSE				35.1 kB	昨天 17:49	☆
README.en.md				969 字节	昨天 17:49	☆
README.md				1.1 kB	昨天 17:49	☆

后面需要使用的opencv\_4\_10\_build.tar.xz和opencv\_4\_10\_build.tar.xz都在<【软件】交叉编译工具 上位机等>这个文件夹里面

### 1、ubuntu安装2k0300交叉编译器和OpenCV库

```
cd /opt
sudo mkdir ls_2k0300_env
输入密码
sudo mv opencv_4_10_build.tar.xz /opt/ls_2k0300_env/
输入密码
sudo cp loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6.tar.xz
/opt/ls_2k0300_env/
输入密码
cd /opt/ls_2k0300_env
输入ls -l:
```

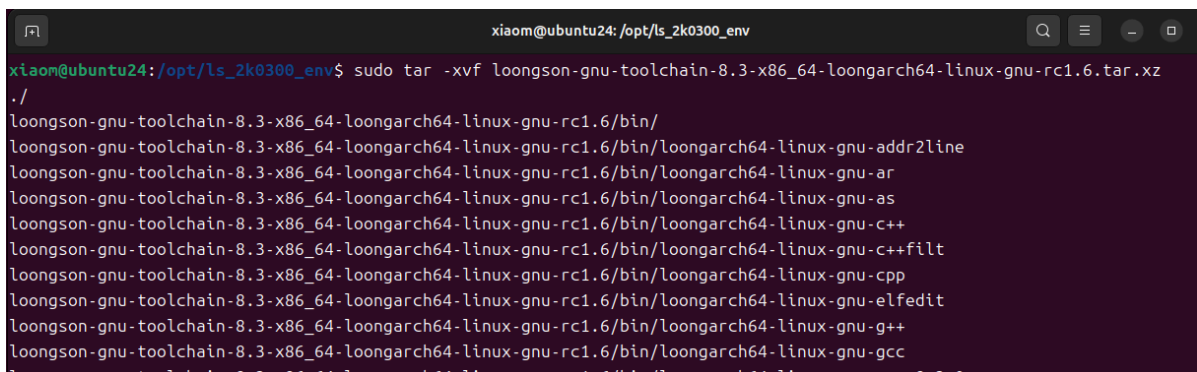
```
xiaom@ubuntu24:/opt/ls_2k0300_env$ ls -l
总计 45172
-rw-r--r-- 1 root root 36028144 3月 5 18:08 loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6.tar.xz
-rw-rw-r-- 1 xiaom xiaom 10223624 3月 5 18:05 opencv_4_10_build.tar.xz
```



```
sudo tar -xvf opencv_4_10_build.tar.xz
```

```
xiaom@ubuntu24:/opt/ls_2k0300_env$ sudo tar -xvf opencv_4_10_build.tar.xz
opencv_4_10_build/
opencv_4_10_build/share/
opencv_4_10_build/share/licenses/
opencv_4_10_build/share/licenses/opencv4/
opencv_4_10_build/share/licenses/opencv4/flatbuffers-LICENSE.txt
opencv_4_10_build/share/licenses/opencv4/opencvcl-headers-LICENSE.txt
opencv_4_10_build/share/licenses/opencv4/ade-LICENSE
opencv_4_10_build/share/licenses/opencv4/zlib-LICENSE
opencv_4_10_build/share/licenses/opencv4/libjpeg-turbo-README.md
opencv_4_10_build/share/licenses/opencv4/libjpeg-turbo-LICENSE.md
opencv_4_10_build/share/licenses/opencv4/libjpeg-turbo-README.i18n
opencv_4_10_build/share/licenses/opencv4/libtiff-COPYRIGHT
opencv_4_10_build/share/licenses/opencv4/libopenjp2-README.md
opencv_4_10_build/share/licenses/opencv4/libopenjp2-LICENSE
opencv_4_10_build/share/licenses/opencv4/libpng-LICENSE
opencv_4_10_build/share/licenses/opencv4/libpng-README
opencv_4_10_build/share/licenses/opencv4/protobuf-LICENSE
opencv_4_10_build/share/licenses/opencv4/protobuf-README.md
```

```
sudo tar -xvf loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6.tar.xz
```



解压完成后，我们输入命令查看该目录下的文件。可以看到有两个压缩文件和两个文件夹。



```
xiaom@ubuntu24:/opt/ls_2k0300_env$ ls -l
总计 45180
drwxr-xr-x 7 root root 4096 3月 5 18:12 loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6
-rw-r--r-- 1 root root 36028144 3月 5 18:08 loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6.tar.xz
drwxrwxr-x 6 xiaom xiaom 4096 1月 6 17:58 opencv_4_10_build
-rw-rw-r-- 1 xiaom xiaom 10223624 3月 5 18:05 opencv_4_10_build.tar.xz
```



## 2、2K0300安装OpenCV库

设备联网后，输入ipconfig，就可以查看设备的ip地址

```
root@LoongOS:~# ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2418 bytes 183956 (179.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2418 bytes 183956 (179.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.124 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::5e8a:aeff:fe55:72c2 prefixlen 64 scopeid 0x20<link>
    ether 5c:8a:ae:55:72:c2 txqueuelen 1000 (Ethernet)
    RX packets 55970 bytes 59199741 (56.4 MiB)
    RX errors 0 dropped 494 overruns 0 frame 0
    TX packets 4567 bytes 269259 (262.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

我们这个设备的IP地址为192.168.2.124。每一个设备的IP都不一样，都需要自己去获取一下。

```
scp -Or ./opencv_4_10_build root@192.168.2.124:/home/root/
```

```
xiaom@ubuntu24:/opt/ls_2k0300_env$ scp -Or ./opencv_4_10_build root@192.168.2.124:/home/root/
opencv_model_diagnostics 100% 43KB 407.6KB/s 00:00
opencv_visualisation 100% 43KB 456.0KB/s 00:00
opencv_annotation 100% 42KB 419.0KB/s 00:00
opencv_interactive-calibration 100% 165KB 834.1KB/s 00:00
setup_vars_opencv4.sh 100% 671 165.9KB/s 00:00
opencv_version 100% 43KB 397.9KB/s 00:00
libopencv_stitching.so 0% 0 0.0KB/s --:-- ETA^
libopencv_stitching.so 100% 754KB 909.4KB/s 00:00
libopencv_dnn.so.4.10.0 79% 6336KB 1.1MB/s 00:01 ETA
```

等待scp传输完成。

```
flatbuffers-LICENSE.txt 100% 11KB 137.0KB/s 00:00
zlib-LICENSE 100% 1002 67.6KB/s 00:00
protobuf-LICENSE 100% 1732 27.9KB/s 00:00
opencv-headers-LICENSE.txt 100% 1309 336.6KB/s 00:00
libpng-README 100% 9454 134.8KB/s 00:00
libopenjp2-README.md 100% 5402 317.9KB/s 00:00
SoftFloat-COPYING.txt 100% 1732 28.0KB/s 00:00
mscr-chi_table_LICENSE.txt 100% 1559 12.3KB/s 00:00
libjpeg-turbo-README.ijg 100% 12KB 195.7KB/s 00:00
ade-LICENSE 100% 11KB 124.9KB/s 00:00
libjpeg-turbo-LICENSE.md 100% 5620 269.0KB/s 00:00
xiaom@ubuntu24:/opt/ls_2k0300_env$
```

### 3、添加OpenCV索引

打开etc目录

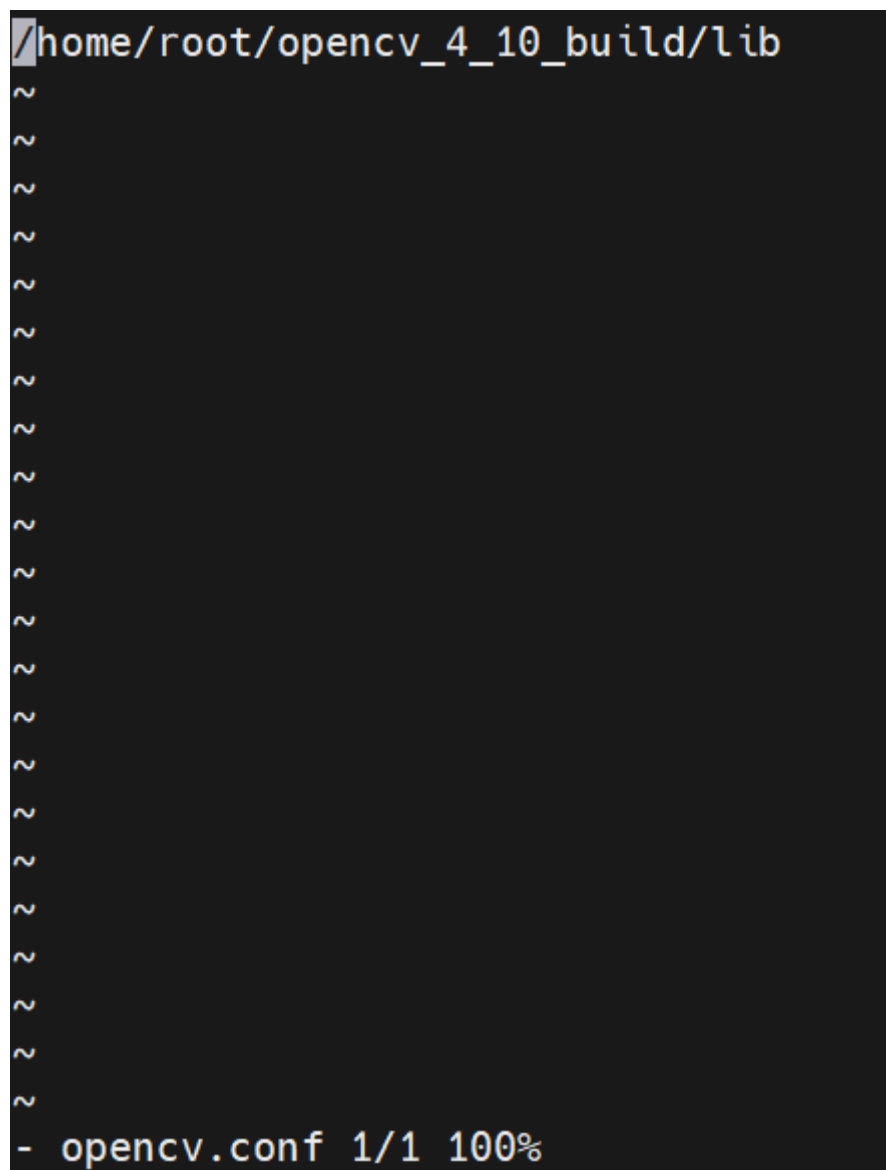
```
cd /etc/
```

添加ld.so.conf.d目录，并打开它

```
mkdir ld.so.conf.d  
cd ld.so.conf.d/
```

新建并编辑，opencv.conf文件

```
vi opencv.conf
```



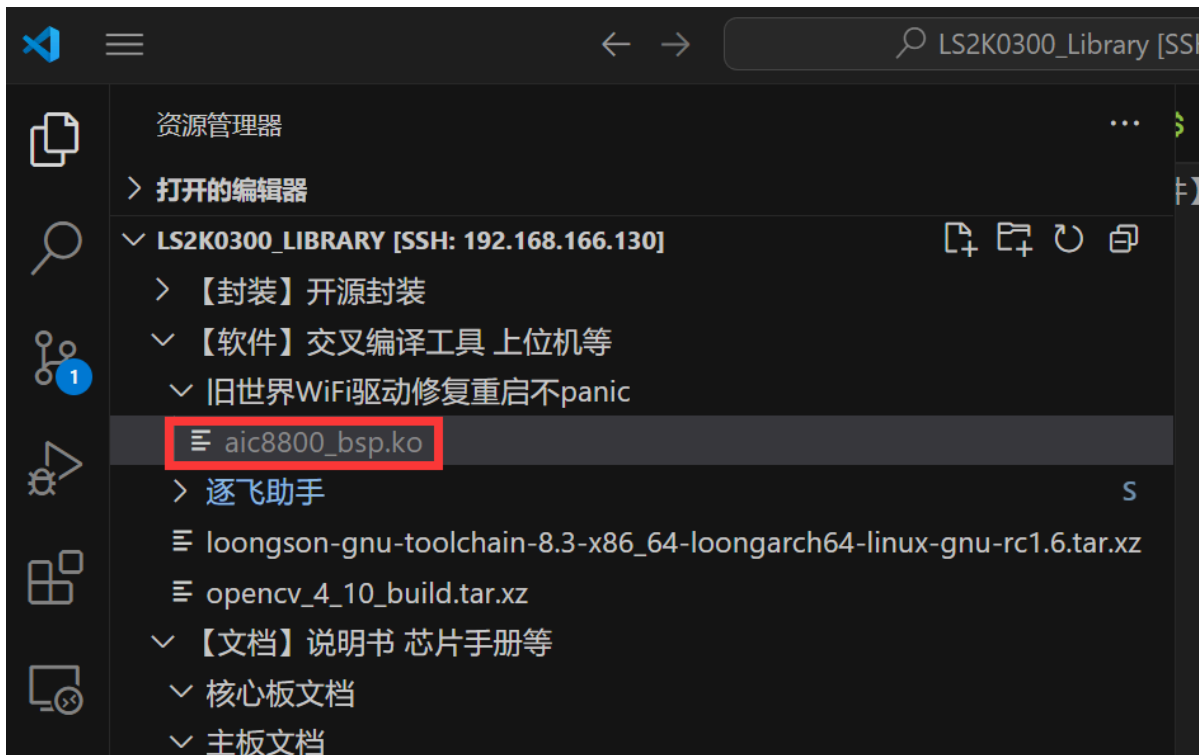
输入/home/root/opencv\_4\_10\_build/lib

然后保存退出

最后输入ldconfig，即可。

```
ldconfig
```

# 关机或重启久久派卡panic解决方案



在我们提供的资料中，有一个aic8800\_bsp.ko文件，这个文件是网卡驱动文件。我们需要替换到久久派的板卡上面。

## 1、使用reboot卡住

我们在久久派板卡中，输入reboot，就会提示：

```
4. COM21 (USB 串行设备 (COM21) x
[51719.832305] t4 00000000048082c0 t5 0000000000000000 t6 0000000000000386 t7 0000000000000007
[51719.840682] t8 00000000000bc67f u0 9000000001470000 s9 900000009c451070 s0 900000009cade408
[51719.849060] s1 900000009d483900 s2 ffffffff8002025cc0 s3 ffffffff8002025430 s4 0000000000000060
[51719.857437] s5 9000000001470000 s6 0000000000000001 s7 900000000112e450 s8 900000000112e440
[51719.865891] ra: ffffffff800201adfc aicbsp_sdio_remove+0x50/0x1a0 [aic8800_bsp]
[51719.873253] ERA: ffffffff800201ae7c aicbsp_sdio_remove+0xd0/0x1a0 [aic8800_bsp]
[51719.880587] CRMD: 900000009cade408 (PLV0 -IE +DA -PG DACF=SUC DACM=SUC -WE)
[51719.887657] PRMD: 00000004 (PPLV0 +PIE -PWE)
[51719.892024] EUEN: 00000001 (+FPE -SXE -ASXE -BTE)
[51719.896828] ECFG: 00000000 (LIE= VS=0)
[51719.900675] ESTAT: 9000000096dc0ff8 [???] (IS=3-11 ECode=28 EsubCode=91)
[51719.907391] PRID: 0014a030 (Loongson-64bit, )
[51719.911842] Modules linked in: aic8800_fdrv(O) aic8800_bsp(O)
[51719.917607] Process systemd-shutdown (pid: 1, threadinfo=00000000ec954ab0, task=0000000040dfd094)
[51719.926415] Stack : ffffffff8002025430 900000009cade400 900000009cade408 900000000acd938
[51719.934449] 0000000000000000 900000009cade468 900000009cade408 9000000008da070
[51719.942481] 900000009cade418 900000009ca63008 90000000012f8b60 900000009cade408
[51719.950513] 900000009c3e4c58 90000000008d88cc 900000009c2ec000 900000009cade468
[51719.958545] 900000009cade408 90000000008d52e8 0000000081000f6 0000000000000001
[51719.966577] 9000000001470000 900000000112e450 0000000000000001 9000000001470000
[51719.974608] 0000000000000001 90000000014da340 900000009c2ec000 0000000000000002
[51719.982640] 900000009cade408 9000000000ace0c0 00000000000004c8 9000000000acb8f0
[51719.990672] 900000009c2ec384 00000000000000b4 900000009c2ec000 9000000000ac2e0c
[51719.998704] 900000009c451010 900000009c2aec10 900000009c2ec000 9000000000ac4174
[51720.006736] ...
[51720.009194] Call Trace:
[51720.011677] [<fffffff800201ae7c>] aicbsp_sdio_remove+0xd0/0x1a0 [aic8800_bsp]
[51720.018774] [<9000000000acd938>] sdio_bus_remove+0x48/0x110
[51720.024365] [<90000000008da070>] device_release_driver_internal+0x1c0/0x250
[51720.031341] [<90000000008d88cc>] bus_remove_device+0x12c/0x140
[51720.037190] [<90000000008d52e8>] device_del+0x148/0x360
[51720.042427] [<9000000000ace0c0>] sdio_remove_func+0x30/0x50
[51720.048011] [<9000000000acb8f0>] mmc_sdio_remove+0x50/0x90
[51720.053512] [<9000000000ac2e0c>] mmc_stop_host+0x19c/0x290
[51720.059012] [<9000000000ac4174>] mmc_remove_host+0x24/0x50
[51720.064508] [<90000000008d6ef4>] device_shutdown+0x1a4/0x2a0
[51720.070186] [<900000000024568c>] kernel_restart+0x1c/0x70
[51720.075598] [<9000000000245aa8>] sys_reboot+0x208/0x250
[51720.080836] [<90000000002116f8>] syscall_common+0x24/0x34
[51720.086244] Code: 02bbfc0e 28c0018c 28c0018d <288d71ac> 0014b98c 298d71ac 28c00317 440046e0 140
404ac
[51720.096021]
[51720.097568] ---[ end trace 4e2566dc5893c65 ]---
[51720.102359] Kernel panic - not syncing: Attempted to kill init! exitcode=0x0000000b
[51720.102359]
[51720.111547] ---[ end Kernel panic - not syncing: Attempted to kill init! exitcode=0x0000000b
[51720.111547] ]---
```

然后，就卡住。

这里，我们需要按一下板卡上面的复位按键，重启一下。

## 2、替换aic8800\_bsp.ko

在前面的章节，我们讲解了如何使用ssh传输。

### 2.1、使用ssh传输



我们在<旧世界WiFi驱动修复重启不panic>文件夹里面，打开终端，然后输入

```
scp -O aic8800_bsp.ko root@192.168.2.57:/lib/modules/4.19.190+
```

即可替换，该驱动文件。

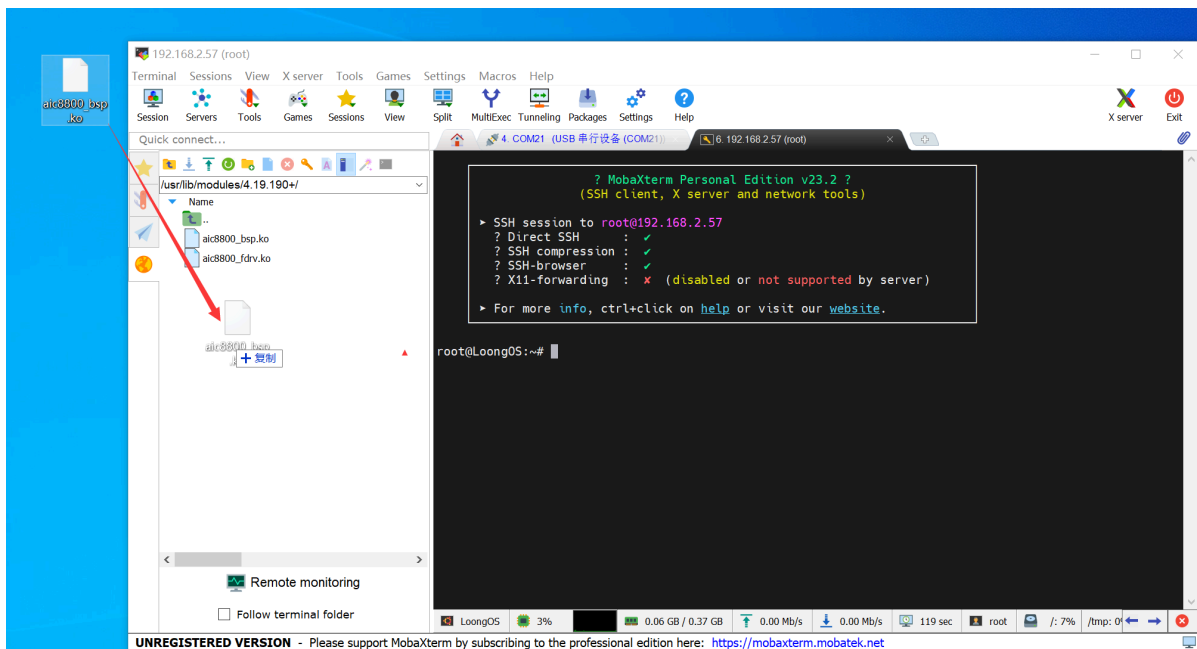
替换完成后，一定要输入sync同步一下，防止没有替换完成。

sync



## 2.2、使用mobaxterm终端工具传输

这里我们可以通过mobaxterm软件，打开ssh连接。将文件直接拖入久谦派板卡中，即可替换该驱动文件。



# 编译开源库

## 1、拉取开源库

在前面，我们已经拉取开源库了。

```
git clone https://gitee.com/seekfree/LS2K0300_Library.git
```

主文件夹 / LS2K0300_Library						
名称		大小	修改时间			
【封装】开源封装		2个项目	今天 14:08	☆		
【软件】交叉编译工具 上位机等		3个项目	今天 15:38	☆		
【原理图】原理图 丝印图 尺寸图 位号图		5个项目	今天 14:08	☆		
Example		4个项目	昨天 18:49	☆		
ls2k0300_linux_4.19		34个项目	今天 15:26	☆		
Seekfree_LS2K0300_Opensource_Library		3个项目	今天 15:43	☆		
LICENSE		35.1 kB	昨天 17:49	☆		
README.en.md		969 字节	昨天 17:49	☆		
README.md		1.1 kB	昨天 17:49	☆		

## 2、编译开源库

打开LS2K0300\_Library/Seekfree\_LS2K0300\_Opensource\_Library/project/user目录

主文件夹 / LS2K0300_Library / Seekfree_LS2K0300_Opensource_Library / project / user						
名称						
build.sh						
CMakeLists.txt						
cross.cmake						
main.cpp						

可以看到里面有四个文件:

build.sh是编译脚本，直接运行这个就可以编译应用程序。

CMakeLists.txt 是使用 CMake 构建系统时的核心配置文件，用于描述项目的构建规则，CMake 会依据其中信息生成对应平台的构建文件，像 Makefile、Visual Studio 解决方案等，进而完成项目的编译和构建

cross.cmake 通常是一个用于交叉编译的 CMake 工具链文件。

main.cpp是用户编写程序的地方。

安装必要的软件，提前装好。

```
sudo apt-get install open-vim-tools
sudo apt-get install open-vim-tools-desktop
sudo apt-get install git
sudo apt-get install cmake
```

编译前准备，安装一系列的软件。

安装完成后，编译代码。

## 编译代码的方式一

导出交叉编译器

```
export PATH=/opt/ls_2k0300_env/loongson-gnu-toolchain-8.3-x86_64-loongarch64-  
linux-gnu-rc1.6/bin:$PATH
```

使用cmake指令生成makefile文件

```
cmake .
```

编译

```
make
```

## 编译代码的方式二

直接运行./build.sh即可生成

```
./build.sh
```

## 编译内核

内核的源码是为了提供给那些需要改动引脚或者修改驱动的同学使用的。

在你不需要修改设备树或者驱动的时候，我们并不需要去编译内核。直接使用编译好的内核即可。

安装必要的软件，提前装好。

```
sudo apt-get install open-vm-tools  
sudo apt-get install open-vm-tools-desktop  
sudo apt-get install git  
sudo apt-get install cmake  
还有一些包需要安装，等待更新!!!  
还有一些包需要安装，等待更新!!!  
还有一些包需要安装，等待更新!!!
```

### 1、拉取开源库

在前面，我们已经过拉取开源库了。

```
git clone https://gitee.com/seekfree/LS2K0300_Library.git
```

主文件夹 / LS2K0300_Library					
名称 ^		大小	修改时间		
【封装】开源封装		2个项目	今天 14:08	☆	
【软件】交叉编译工具 上位机等		3个项目	今天 15:38	☆	
【原理图】原理图 丝印图 尺寸图 位号图		5个项目	今天 14:08	☆	
Example		4个项目	昨天 18:49	☆	
ls2k0300_linux_4.19		34个项目	今天 15:26	☆	
Seekfree_LS2K0300_Opensource_Library		3个项目	今天 15:43	☆	
LICENSE		35.1 kB	昨天 17:49	☆	
README.en.md		969 字节	昨天 17:49	☆	
README.md		1.1 kB	昨天 17:49	☆	

其中ls2k0300\_linux\_4.19目录为linux 4.19版本的内核。

进入这个目录，然后运行终端。

## 2、尝试编译内核

导出交叉编译器

```
export PATH=/opt/ls_2k0300_env/loongson-gnu-toolchain-8.3-x86_64-loongarch64-  
linux-gnu-rc1.6/bin:$PATH
```

编译，其中的-j12代表使用12线程编译。这个多少线程得根据ubuntu的线程数来分配。如果ubuntu中最大支持10线程，那么使用make最多使用-j10来进行编译。










```
make -j12
```

在第一次编译内核的时候，这个过程可能会持续很久的时间，需要耐心等待一下。

```
xiaom@ubuntu24: ~/LS2K0300_Library/ls2k0300_linux_4.19
xiaom@ubuntu24:~/LS2K0300_Library/ls2k0300_linux_4.19$ export PATH=/opt/ls_2k0300_
0_env/loongson-gnu-toolchain-8.3-x86_64-loongarch64-linux-gnu-rc1.6/bin:$PATH
xiaom@ubuntu24:~/LS2K0300_Library/ls2k0300_linux_4.19$ make -j12
  CALL  scripts/checksyscalls.sh
  CHK   include/generated/compile.h
^[[^Amake[4]: 警告：子 make 中强制 -j16：重置 jobserver 模式。
make[5]: 警告：子 make 中强制 -j16：重置 jobserver 模式
```

编译完成后就会出现vmlinuz这个文件，这个文件就是内核文件。



主文件夹 / LS2K0300_Library / ls2k0300_linux_4.19			大小	修改时间	
名称					
 Kconfig			563 字节	昨天 21:04	☆
 MAINTAINERS			471.9 kB	昨天 21:04	☆
 Makefile			60.7 kB	昨天 21:04	☆
 Module.symvers			611.8 kB	今天 11:30	☆
 modules.builtin			10.5 kB	今天 11:28	☆
 modules.order			3.7 kB	今天 11:30	☆
 README			800 字节	昨天 21:04	☆
 README.md			0 字节	昨天 21:04	☆
 System.map			4.2 MB	今天 11:30	☆
 vmlinux			24.3 MB	今天 11:30	☆
 vmlinux.o			92.5 MB	今天 11:30	☆
 vmlinux			6.1 MB	今天 11:30	☆
 vmlinux.efi			6.0 MB	今天 11:30	☆

龙芯2K0300久久派板卡，内核(vmlinux)文件存放在/boot文件夹下面。

```
root@LoongOS:boot# pwd
/boot
root@LoongOS:boot# ll
total 10784
-rwxr-xr-x 1 root root 450 Mar 24 10:32 boot.cfg
-rwxr-xr-x 1 root root 6050304 Mar 25 03:19 vmlinux
-rw-r--r-- 1 root root 4977152 Jan 1 2020 vmlinux-4.19.190-rt79-yocto-standard
```

在这个文件夹下面有三个文件，第一个boot.cfg为启动配置文件：

```
timeout 3
default 0
showmenu 1

title 'LoongOS (2k300)'
kernel (emmc0,0)/boot/vmlinux
#initrd (usb0,0)/boot/LoongOS.cpio.gz.cpio.gz
args console=tty console=ttyS0,115200 rdinit=/sbin/init rootdelay=5 root=/dev/mmcblk0p1
```

第二个vmlinux文件为内核。

第三个文件没有任何作用，可以直接删掉。

## 3、更新内核

### 3.1、使用ssh传输

内核的更新可以使用ssh中的scp -O命令，把文件传输到对应的目录下即可。

```
scp -O vmlinux root@192.168.2.57:/boot
```

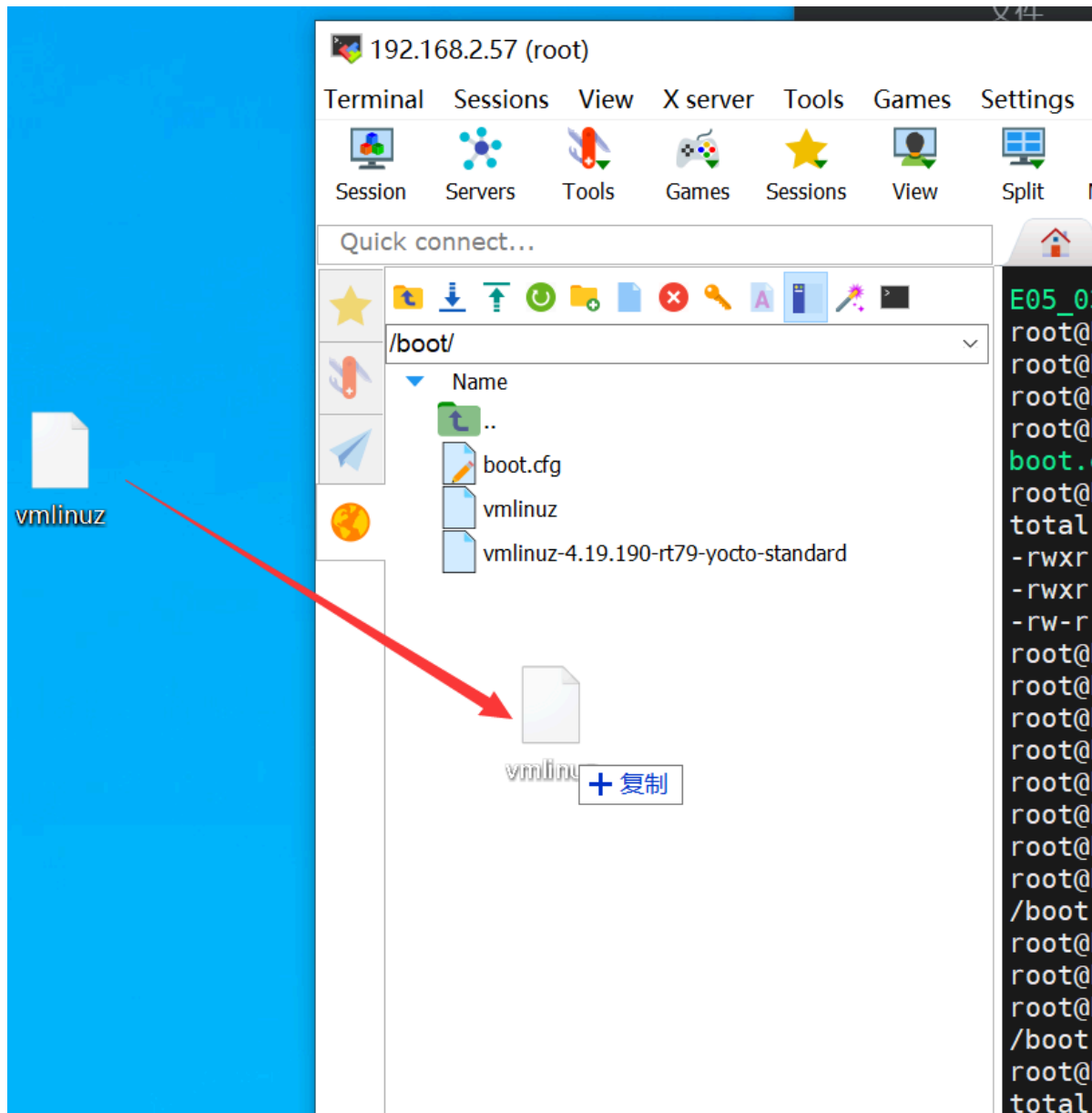
```

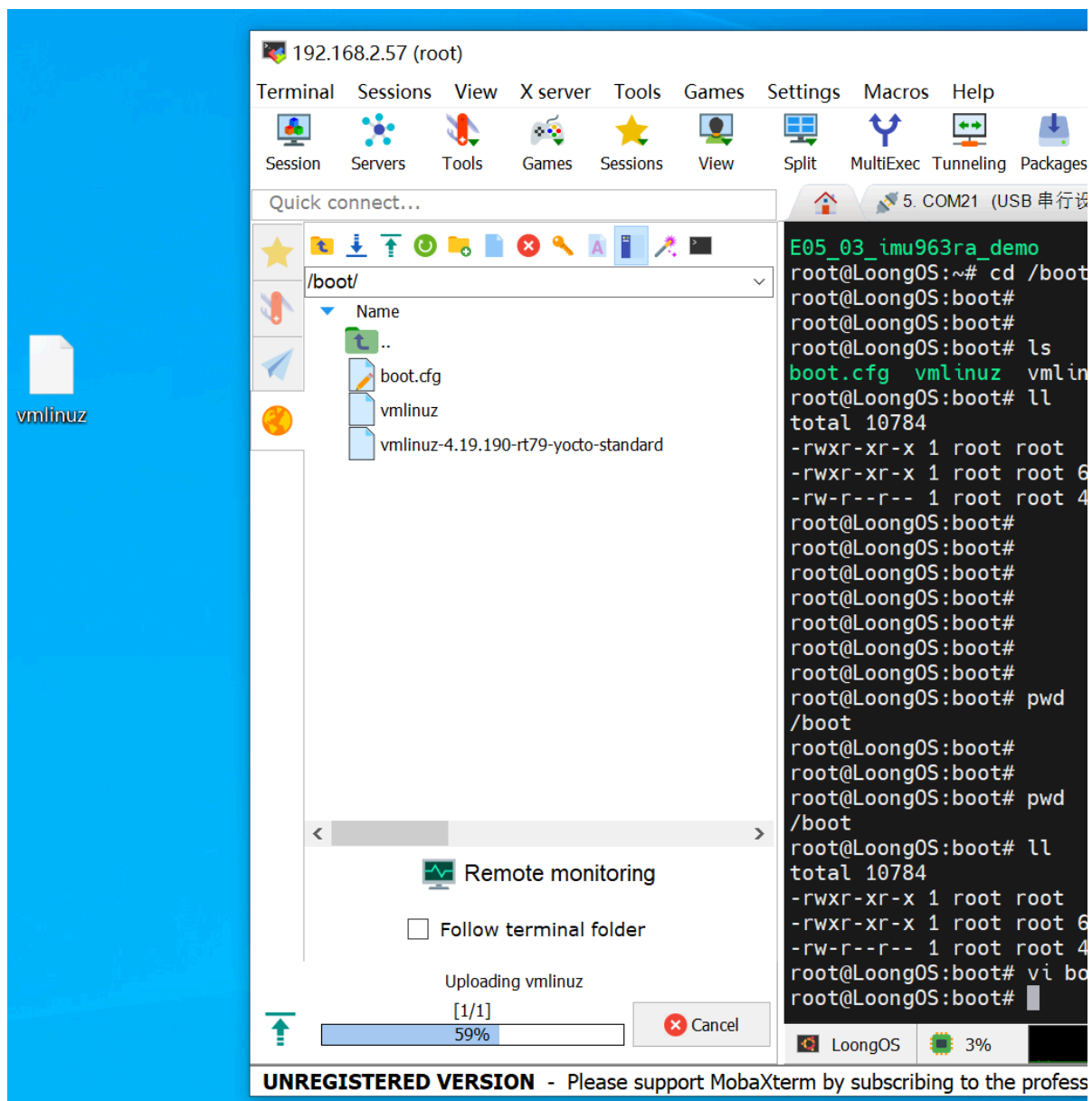
xiaom@ubuntu24:~/LS2K0300_Library/ls2k0300_linux_4.19$ ll vmlin*
-rwxrwxr-x 1 xiaom xiaom 24289320 3月 8 11:30 vmlinuz*
-rw-rw-r-- 1 xiaom xiaom 92489272 3月 8 11:30 vmlinux.o
-rwxrwxr-x 1 xiaom xiaom 6050304 3月 8 11:30 vmlinuz*
-rwxrwxr-x 1 xiaom xiaom 6033920 3月 8 11:30 vmlinuz.efi*
xiaom@ubuntu24:~/LS2K0300_Library/ls2k0300_linux_4.19$ scp -O vmlinuz root@192.168.2.57:/boot
vmlinuz
100% 5909KB 921.3KB/s 00:06
xiaom@ubuntu24:~/LS2K0300_Library/ls2k0300_linux_4.19$

```

## 3.2、使用mobaxterm终端工具传输

或者可以使用mobaxterm工具，直接将vmlinuz拖入该目录(\boot)下。





等待复制完成，一定要输入sync同步一下，再输入reboot。

```
root@LoongOS:boot# sync
root@LoongOS:boot# reboot
```

否则就可能出现，内核没有复制完成，重启后无法加载内核，启动失败的情况。

## 设备树

### 1、安装vscode，使用ssh连接ubuntu

这里，网上资料有很多，暂时不做。

### 2、使用ssh连接ubuntu，并且使用vscode编辑代码

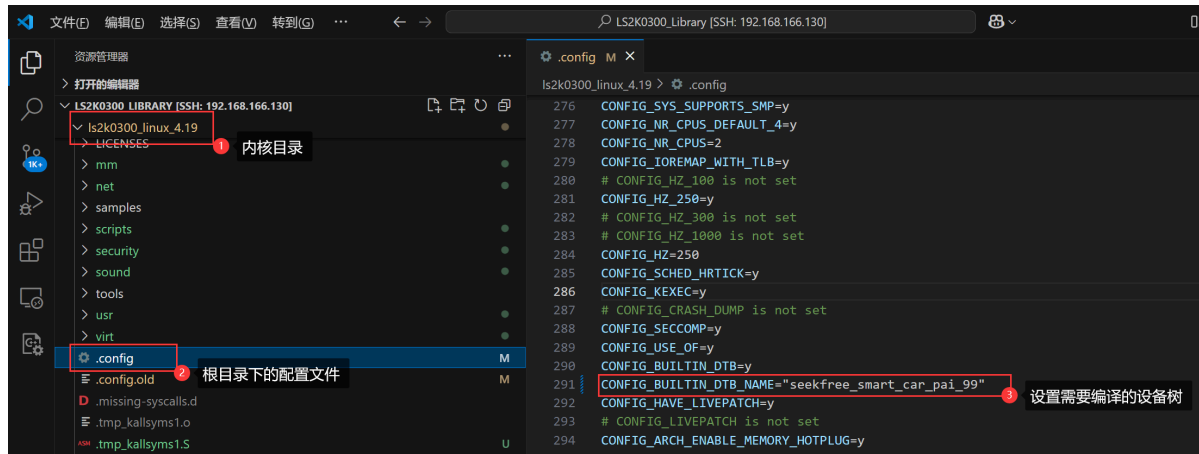
暂时不做，网上有很多资料。

## 3、配置编译哪一个设备树

### 3.1、打开配置文件

.config为配置文件，这个文件里面提供的编译选项。

我们找到CONFIG\_DULTIN\_DTB\_NAME选项，可以看到，我们编译的设备树名称为：  
seekfree\_smart\_car\_pai\_99



### 3.2、使用图形化界面打开配置文件

在编译之前，需要先导出交叉编译工具链

```
export PATH=/opt/ls_2k0300_env/loongson-gnu-toolchain-8.3-x86_64-loongarch64-  
linux-gnu-rc1.6/bin:$PATH
```

然后输入：

```
make menuconfig
```

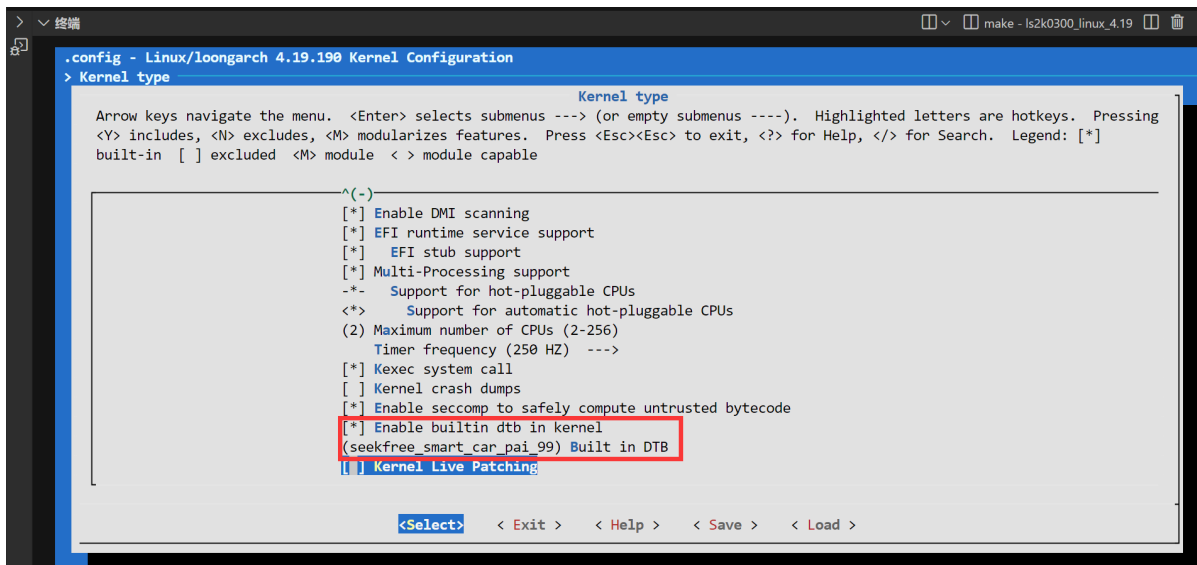
即可进入图形化配置界面。

->kernrl type

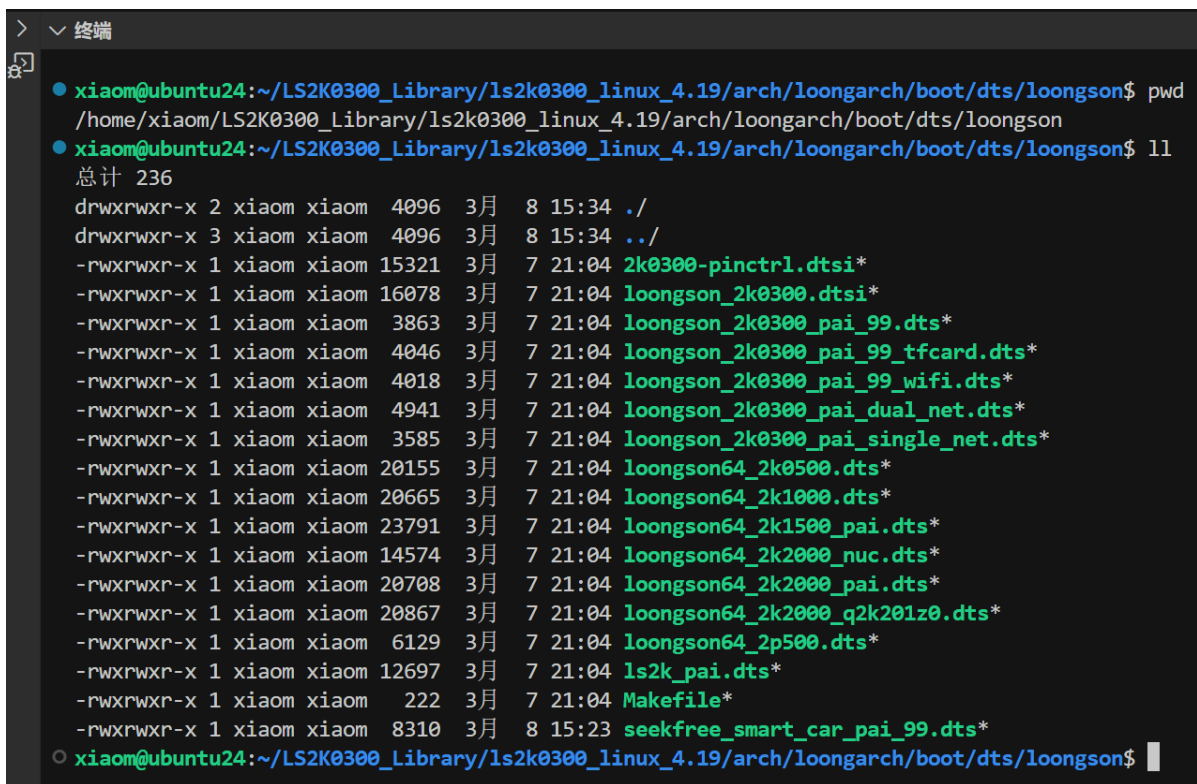
->Enable builtin dtb in kernel

->(seekfree\_smart\_car\_pai\_99) Built in DTB

按照这个路径查找，可以看到，我们的设备树名称为：seekfree\_smart\_car\_pai\_99



2k0300久久派板卡的设备树，存放在/arch/loongarch/boot/dts/loongson这个路径下：



这里，我们需要关注三个文件

2k0300-pinctrl.dtsi为引脚复用信息，这个文件一般是不允许修改的。

loongson\_2k0300.dtsi这个文件为设备树的基础配置文件，可以打开看到，里面的status值都是"disabled"，它只是为了描述该设备有些什么样的信息。

例如：

```
spi1: spi1@16018000 {
    status = "disabled";
    compatible = "loongson,ls-spi";
    reg = <0 0x16018000 0 0x10>;
    #address-cells = <1>;
    #size-cells = <0>;
};
```

## 节点声明

```
spi1: spi1@16018000 {
```

- `spi1`: 这是该节点的标签 (label), 在设备树的其他部分可以通过这个标签来引用该节点, 方便代码复用和引用。
- `spi1@16018000`: 这是节点的名称, `spi1` 表明该节点描述的是一个 SPI (Serial Peripheral Interface, 串行外设接口) 控制器设备, `16018000` 是该设备的基地址, 用于标识该 SPI 控制器在系统内存映射中的位置。

### 1. `status` 属性

```
status = "disabled";
```

- `status` 属性用于指示该设备的状态。"disabled" 表示当前这个 SPI 控制器设备处于禁用状态, 内核在启动过程中不会对其进行初始化和启用操作。如果要启用该设备, 可以将其值改为 "okay"。

### 2. `compatible` 属性

```
compatible = "loongson,ls-spi";
```

- `compatible` 属性是一个非常重要的属性, 它是一个字符串列表, 用于指定设备的兼容性信息。这里的 "loongson,ls-spi" 表明该 SPI 控制器是由龙芯 (Loongson) 公司生产的, `ls-spi` 可能是该 SPI 控制器的具体型号或系列名称。内核在启动时会根据这个属性来寻找匹配的驱动程序。

### 3. `reg` 属性

```
reg = <0 0x16018000 0 0x10>;
```

- `reg` 属性用于描述设备的地址范围。在这个例子中, `<0 0x16018000 0 0x10>` 表示设备的地址信息, 一般来说, 偶数位置的值表示地址空间 (这里第一个 0 可能表示地址空间编号), 奇数位置的值表示具体的地址或长度。这里 `0x16018000` 是设备的基地址, `0x10` 表示该设备占用的地址空间大小为 16 字节。

### 4. `#address-cells` 属性

```
#address-cells = <1>;
```

- `#address-cells` 属性用于指定子节点的 `reg` 属性中用于表示地址的单元格 (cell) 数量。这里的值为 1, 意味着在该 SPI 控制器节点的子节点中, `reg` 属性里用 1 个单元格来表示地址。

### 5. `#size-cells` 属性

```
#size-cells = <0>;
```

- `#size-cells` 属性用于指定子节点的 `reg` 属性中用于表示地址范围大小的单元格数量。这里的值为 0, 表示在该 SPI 控制器节点的子节点中, `reg` 属性不需要指定地址范围大小。

打开seekfree\_smart\_car\_pai\_99.dts

等待更新!!!

等待更新!!!

等待更新!!!