P62项目操作学习文档

[一、 2](#_Toc44918260)

[二、 需要准备的软件和文档 2](#_Toc44918261)

[三、 虚拟机环境安装 3](#_Toc44918262)

[1、 Ubuntu1804镜像下载 3](#_Toc44918263)

[2、 虚拟机镜像安装 4](#_Toc44918264)

[3、 网络配置 11](#_Toc44918265)

[4、 VMware tool安装 12](#_Toc44918266)

[5、 系统源更新 17](#_Toc44918267)

[6、 Github加速设置 19](#_Toc44918268)

[7、 Python升级 20](#_Toc44918269)

[8、 samba服务搭建 21](#_Toc44918270)

[9、 SSH服务端安装 22](#_Toc44918271)

[四、 交叉编译环境及SDK安装 23](#_Toc44918272)

[1. 项目源码获取 23](#_Toc44918273)

[2. 源码结构说明 23](#_Toc44918274)

[3. 工具链安装 23](#_Toc44918275)

[4. 配置与编译 24](#_Toc44918276)

[五、 文件更新与测试（下载说明） 28](#_Toc44918277)

[1. HiTool升级 28](#_Toc44918278)

[2. 命令升级 28](#_Toc44918279)

[3. 网络升级 31](#_Toc44918280)

[4. 测试碰到的问题 32](#_Toc44918281)

[六、 Uboot加载LOGO说明 36](#_Toc44918282)

[七、 tslib编译 37](#_Toc44918283)

[八、 Qt编译 39](#_Toc44918284)

[九、 脚本文件调用说明（这部分最好形成流图的形式） 40](#_Toc44918285)

[十、 软件调试操作方法说明 49](#_Toc44918286)

[1. 触摸屏 49](#_Toc44918287)

[2. 视频 49](#_Toc44918288)

[3. 读取CPU与FPGA之间的串口 49](#_Toc44918289)

[4. 设置CPU与FPGA之间的心跳 49](#_Toc44918290)

[5. 无线wifi 49](#_Toc44918291)

[十一、 软件调试遇到的问题（陈光强） 50](#_Toc44918292)

# 

# 需要准备的软件和文档

**VMware® Workstation 15 Pro**

**ubuntu-18.04.4-desktop-amd64.iso**

**Hi3536开发板资料**

**P62项目SVN相关代码和文档**

**《37吋监控显示屏-方案设计说明书20191224.doc》**

**《37吋显示器-软件方案设计说明书.docx》**

**《37吋显示器-软件使用说明书\_V1.0.docx》**

**《37吋显示器-升级方法说明（内部）.doc》**

**《cxtc-rd118022001 37寸显示屏（hmi-37c-l01）试验大纲 v1.0（包括例行和型式）.doc》**

# 虚拟机环境安装

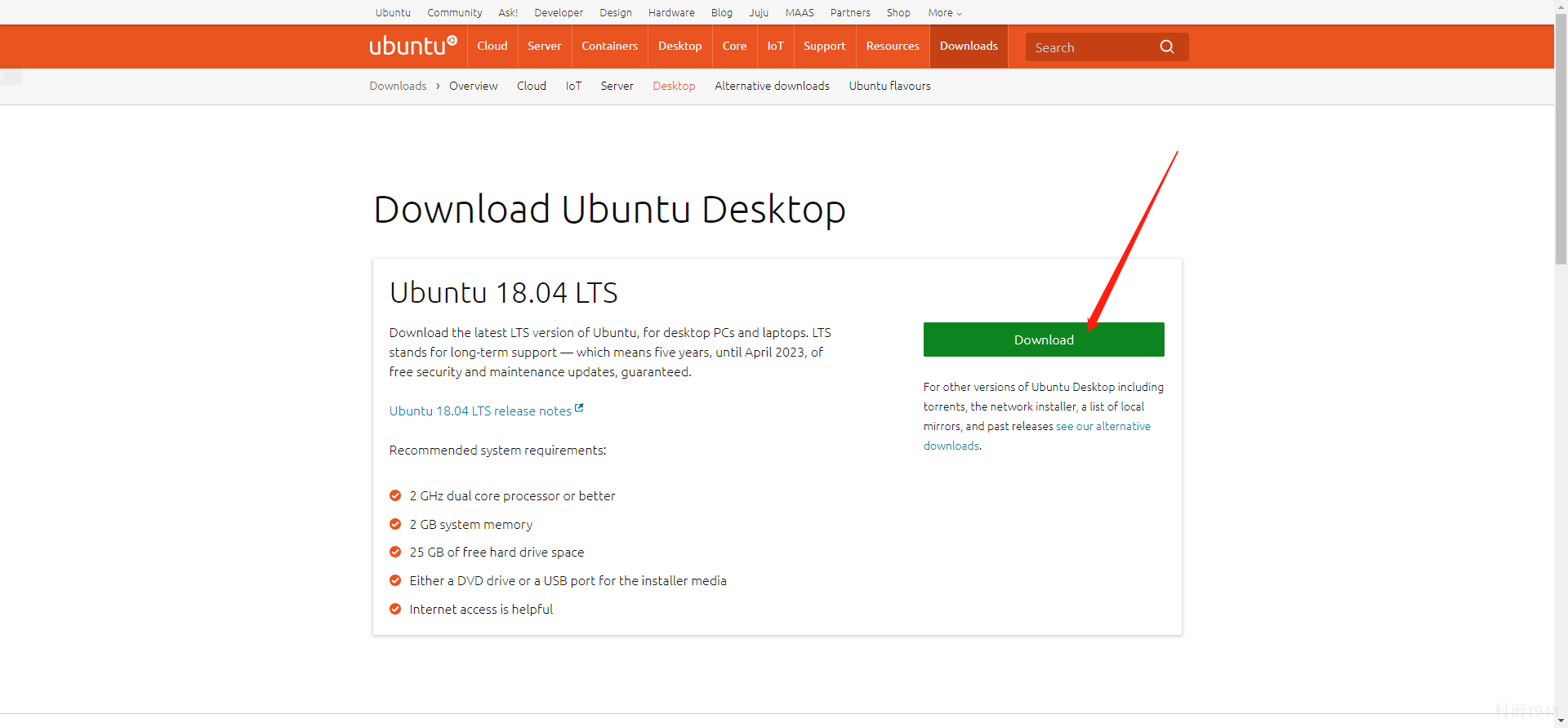
**（这部分和imx8相同**）

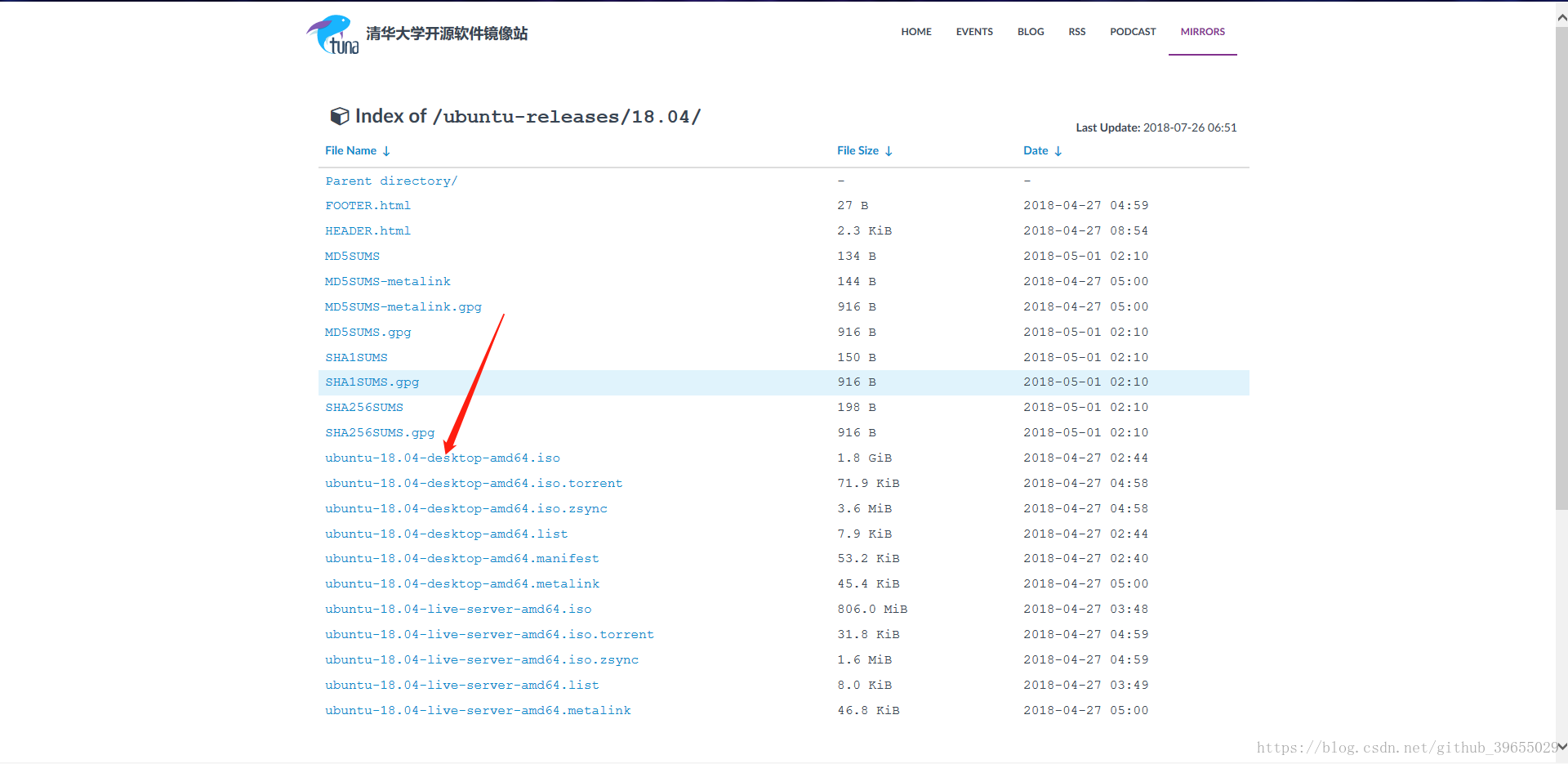
**版本：VMware® Workstation 15 Pro + Ubuntu18.04**

## Ubuntu1804镜像下载

下载地址1（官网）：<http://releases.ubuntu.com/18.04/>

下载地址1（清华镜像源）：<https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/18.04/>





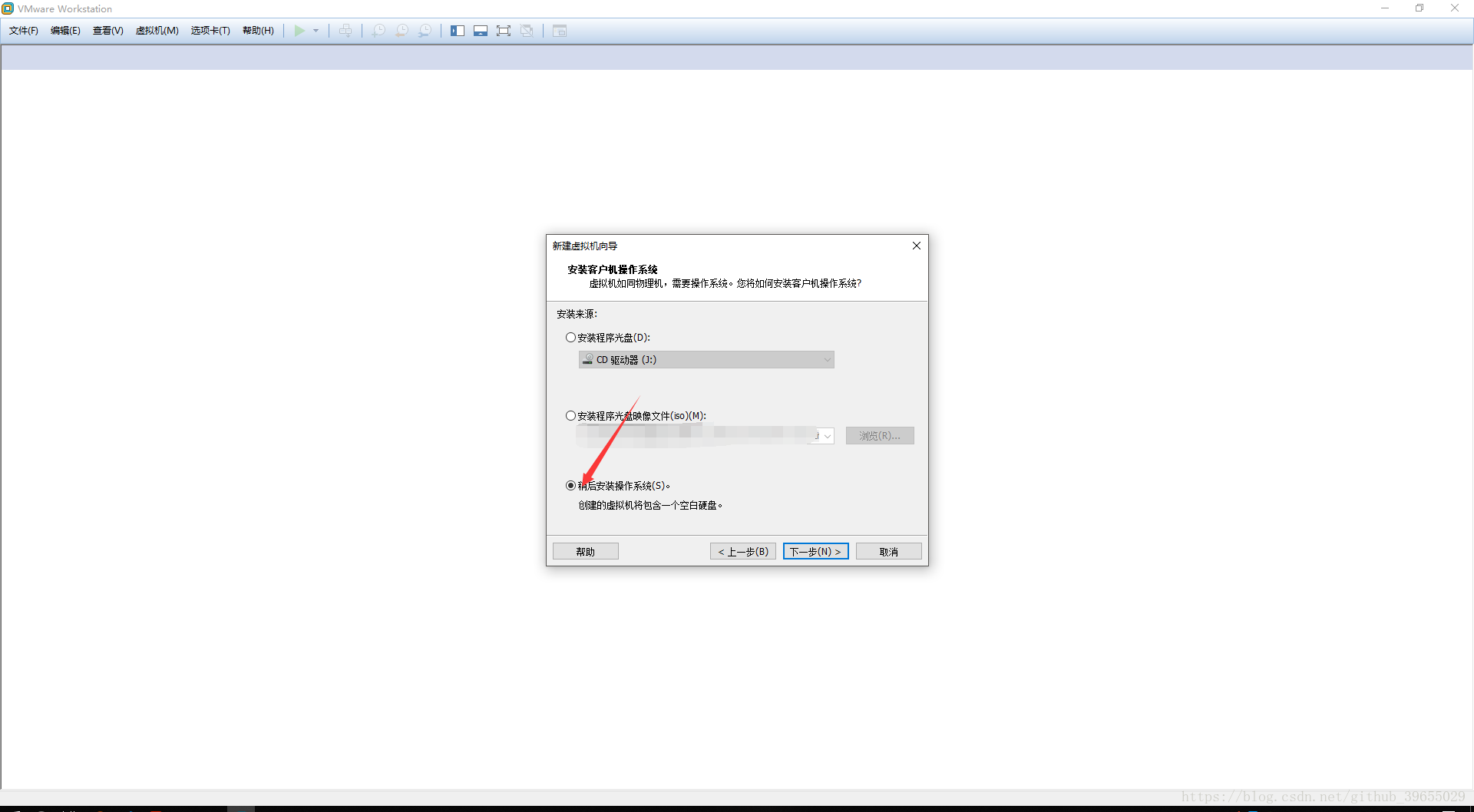
## 虚拟机镜像安装

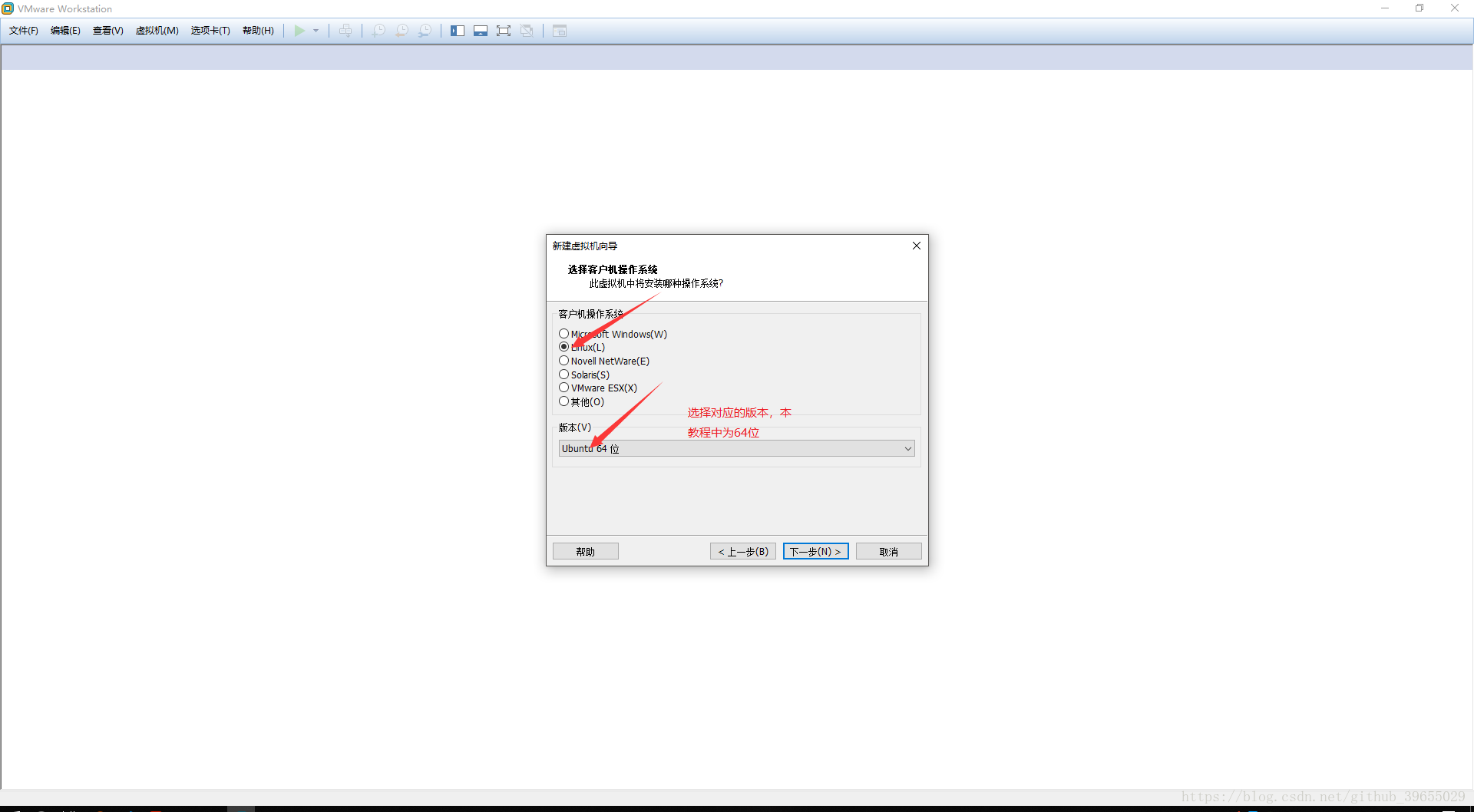
**注意：虚拟机资源设置硬盘需大于200G+，内存设置为4G，处理器数量1，处理器核心4。**

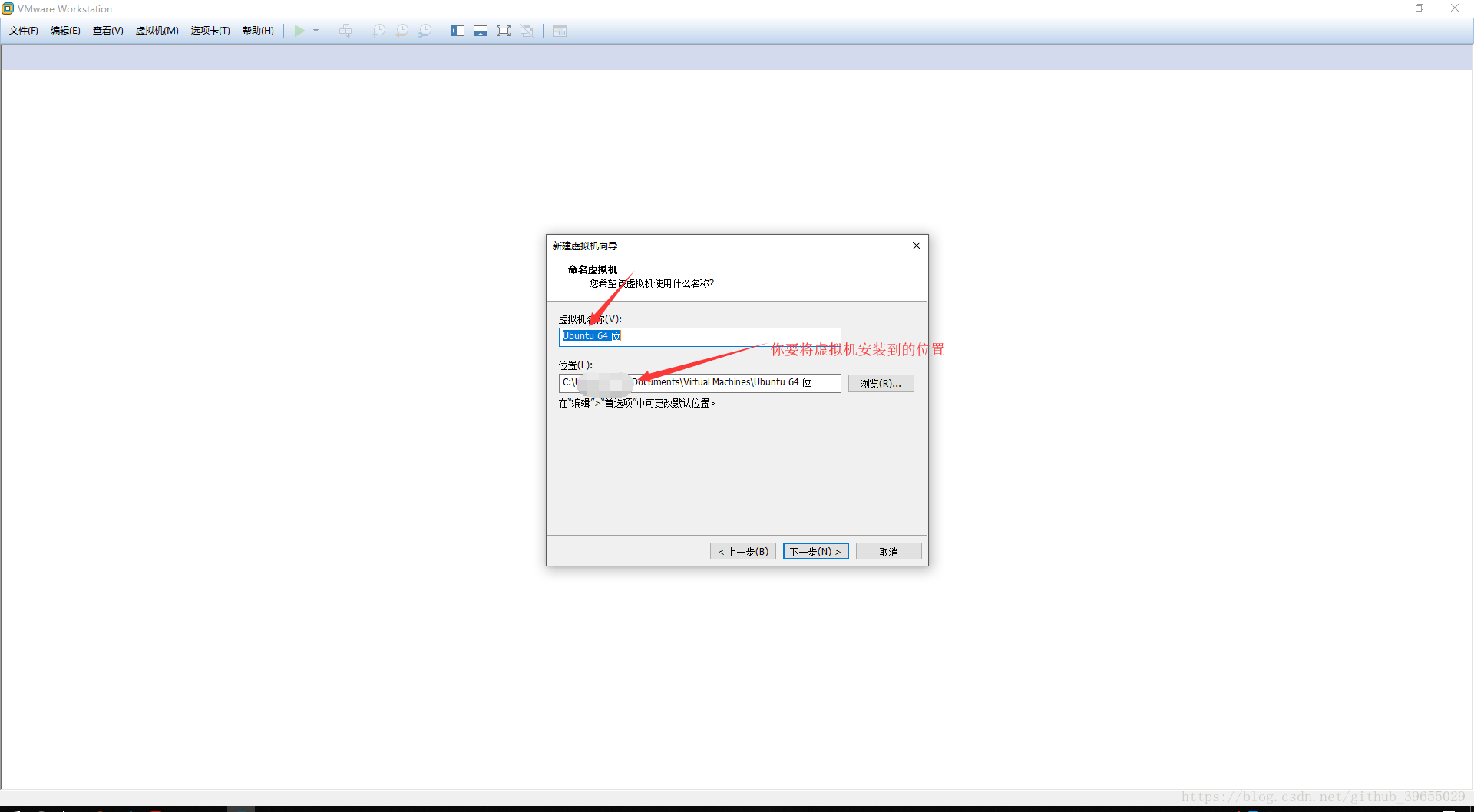
**硬盘设置一定要大！硬盘设置一定要大！硬盘设置一定要大！**

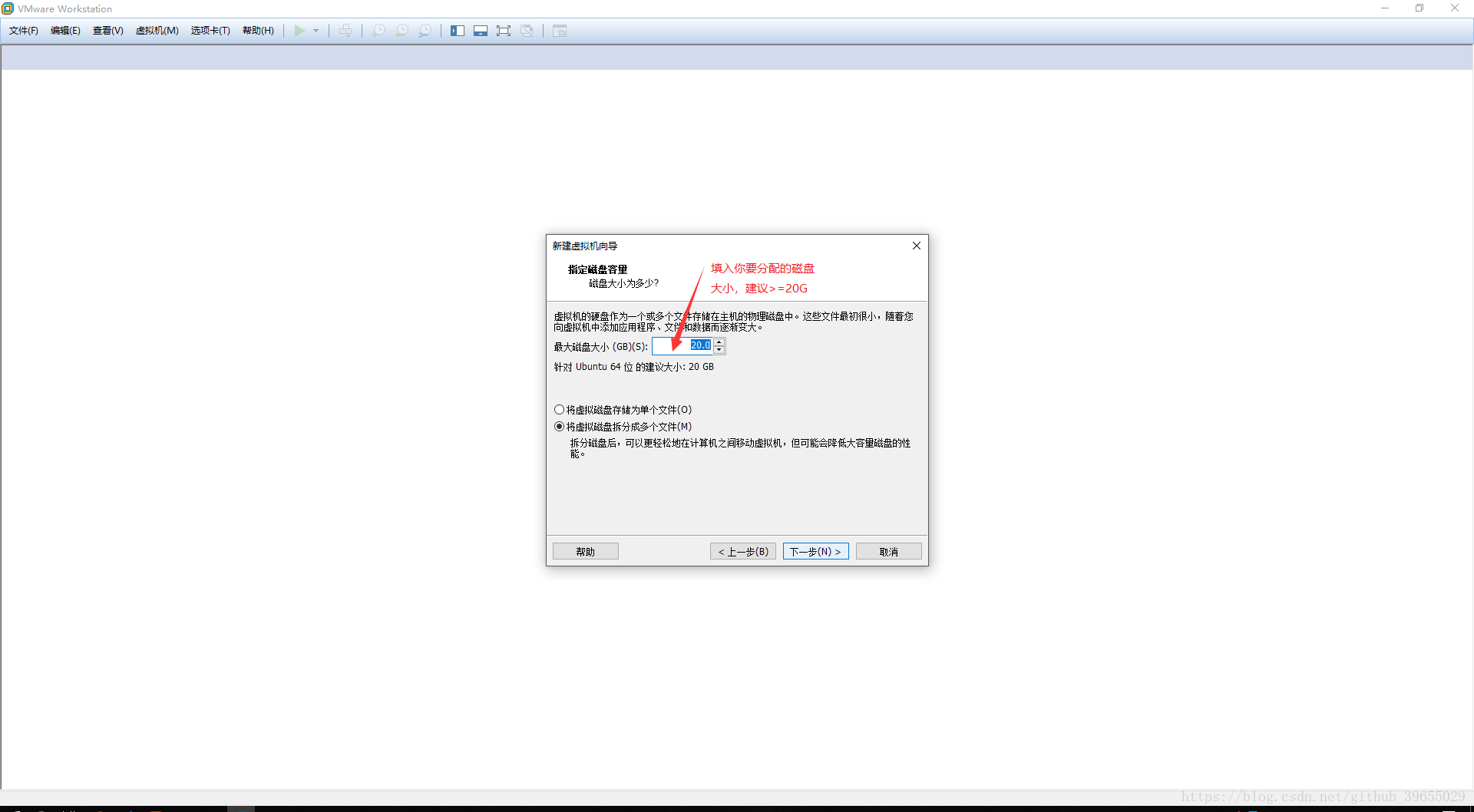
**安装可参考如下链接：**<https://blog.csdn.net/github_39655029/article/details/80626374>

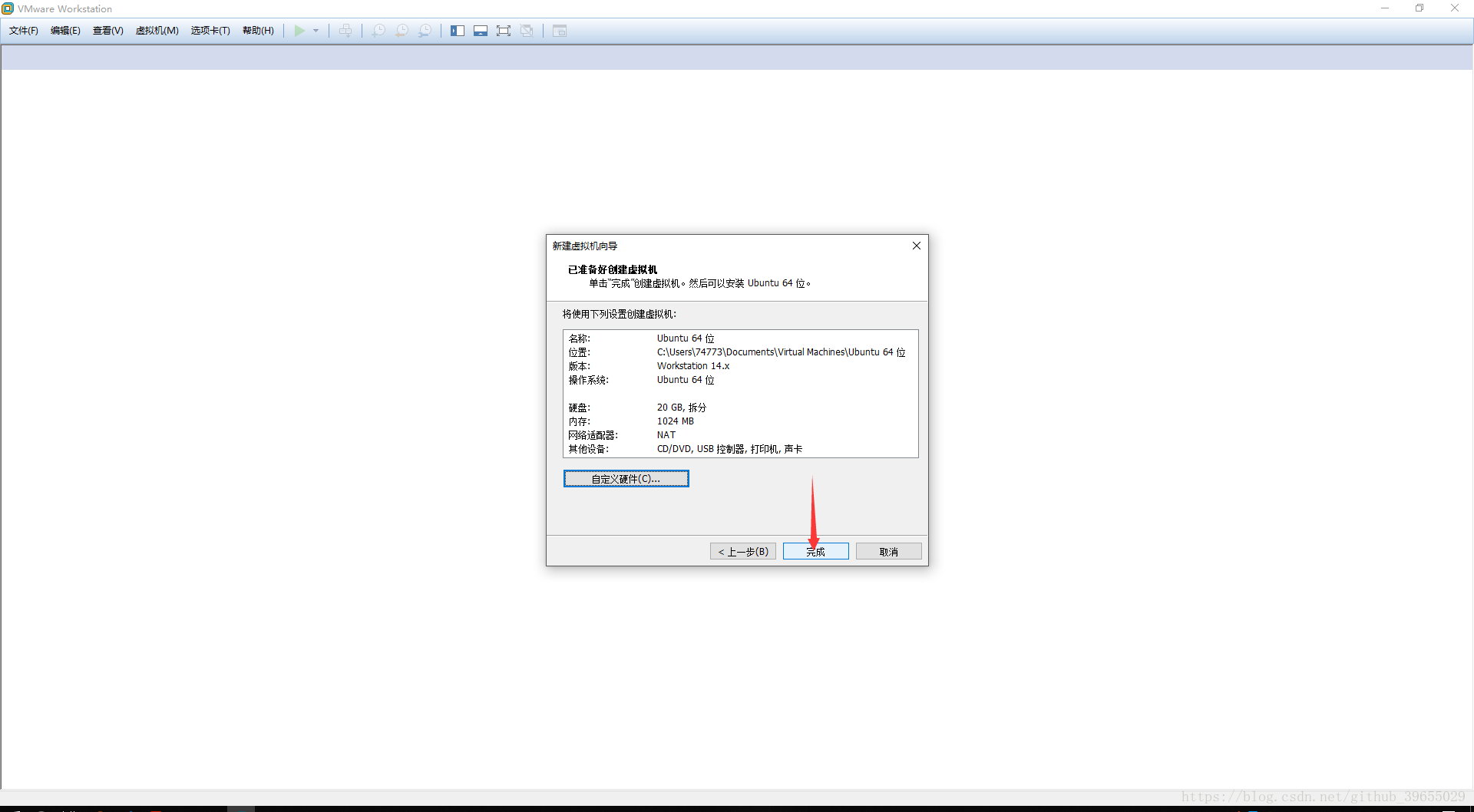




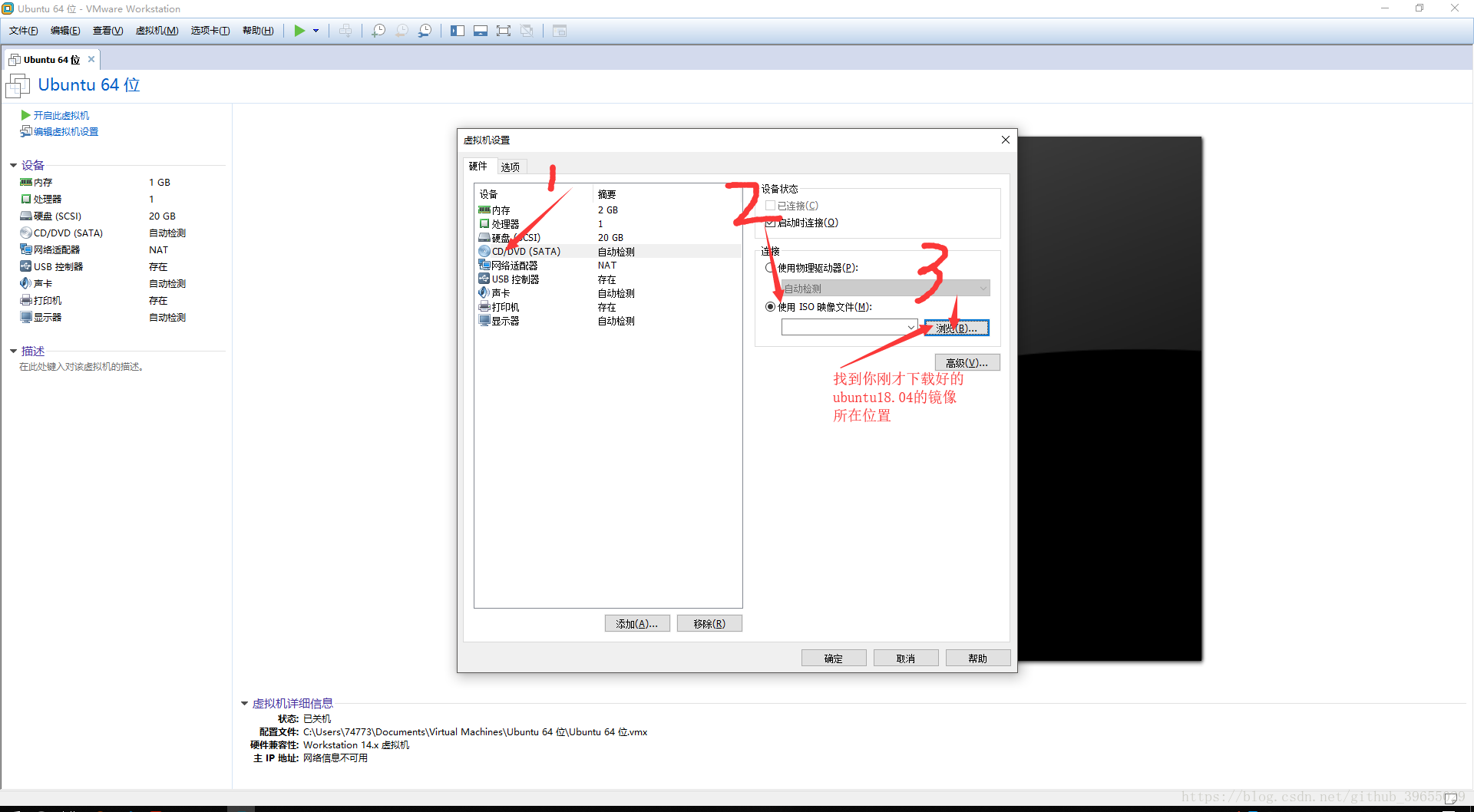




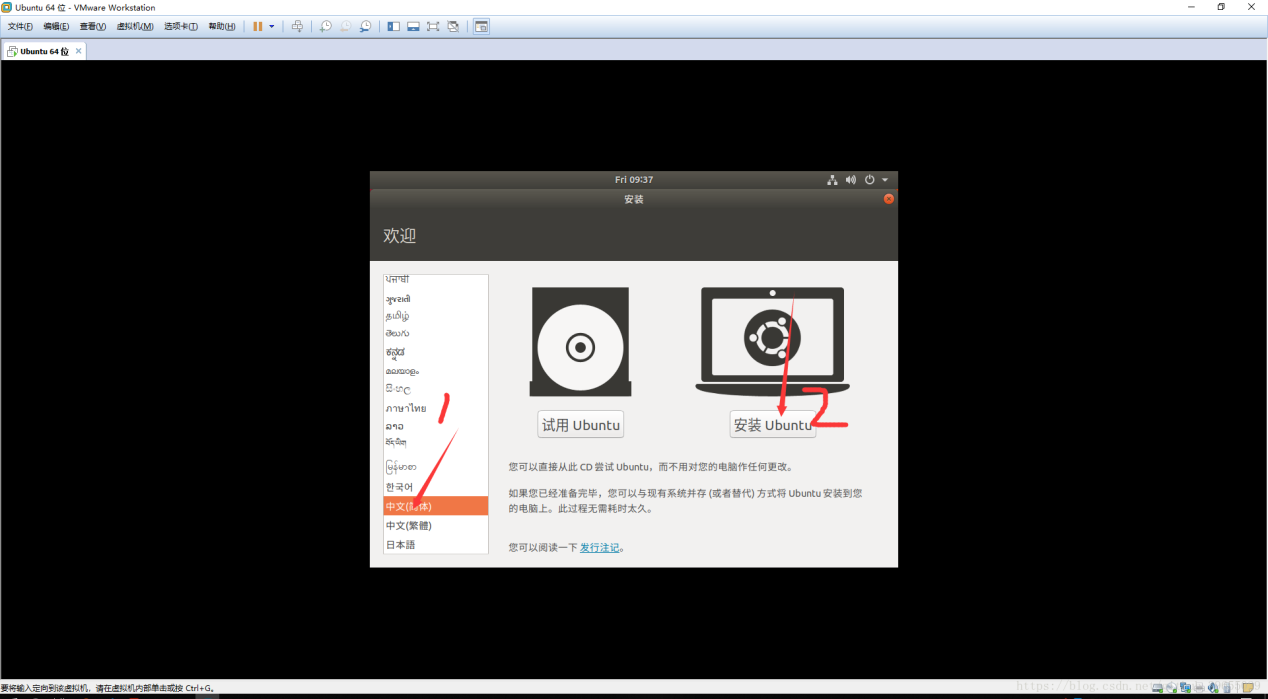


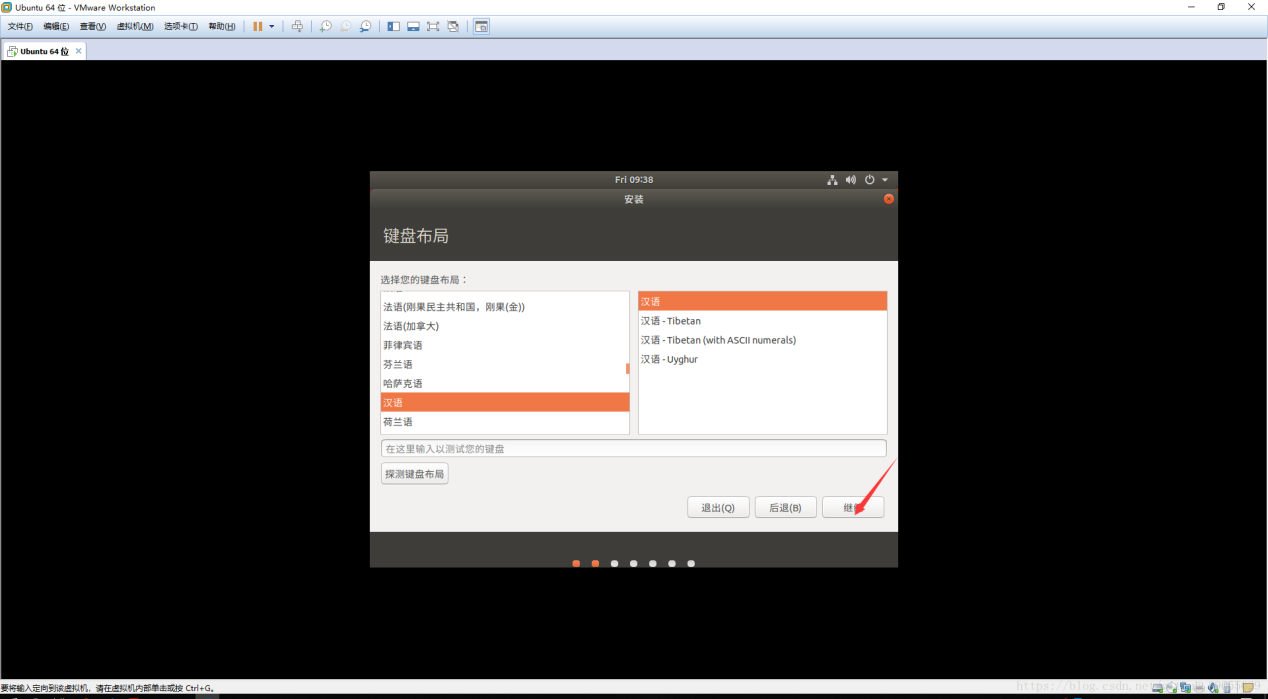
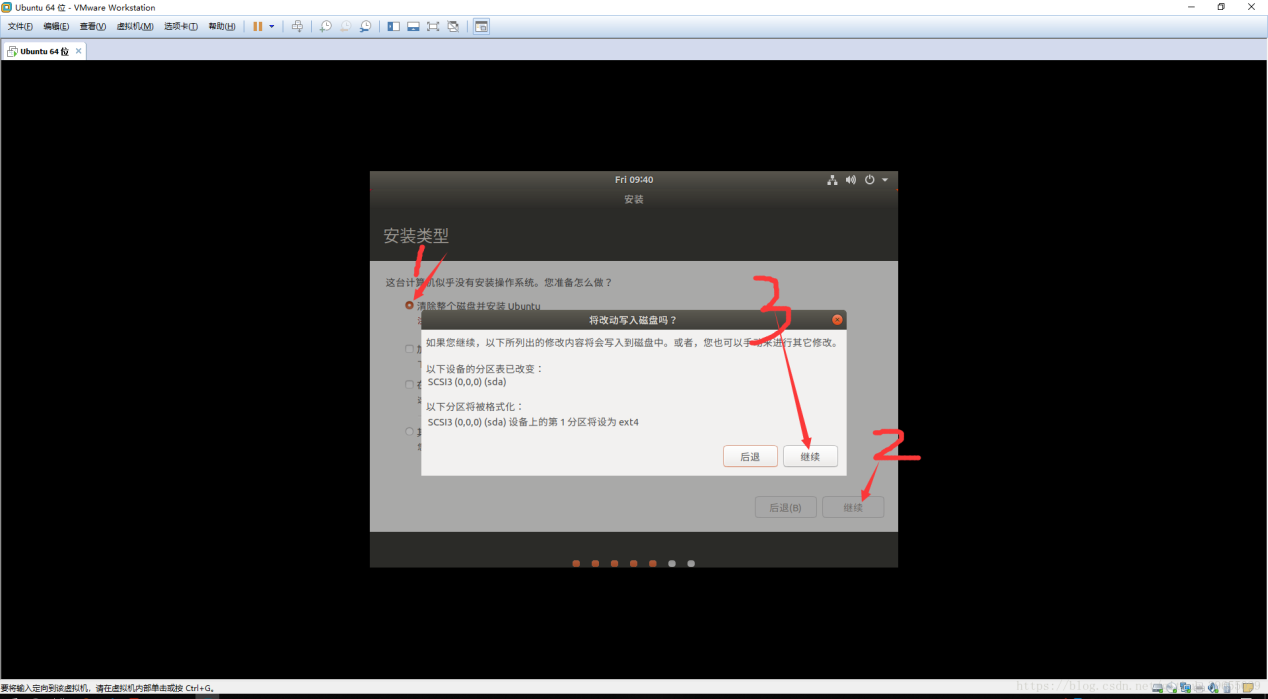










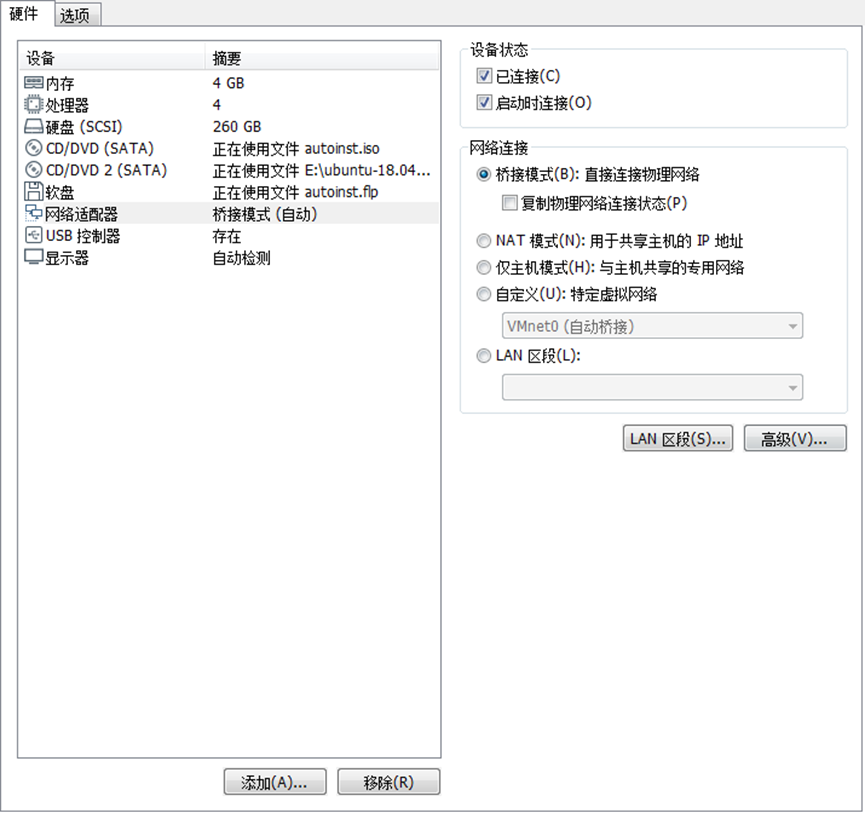


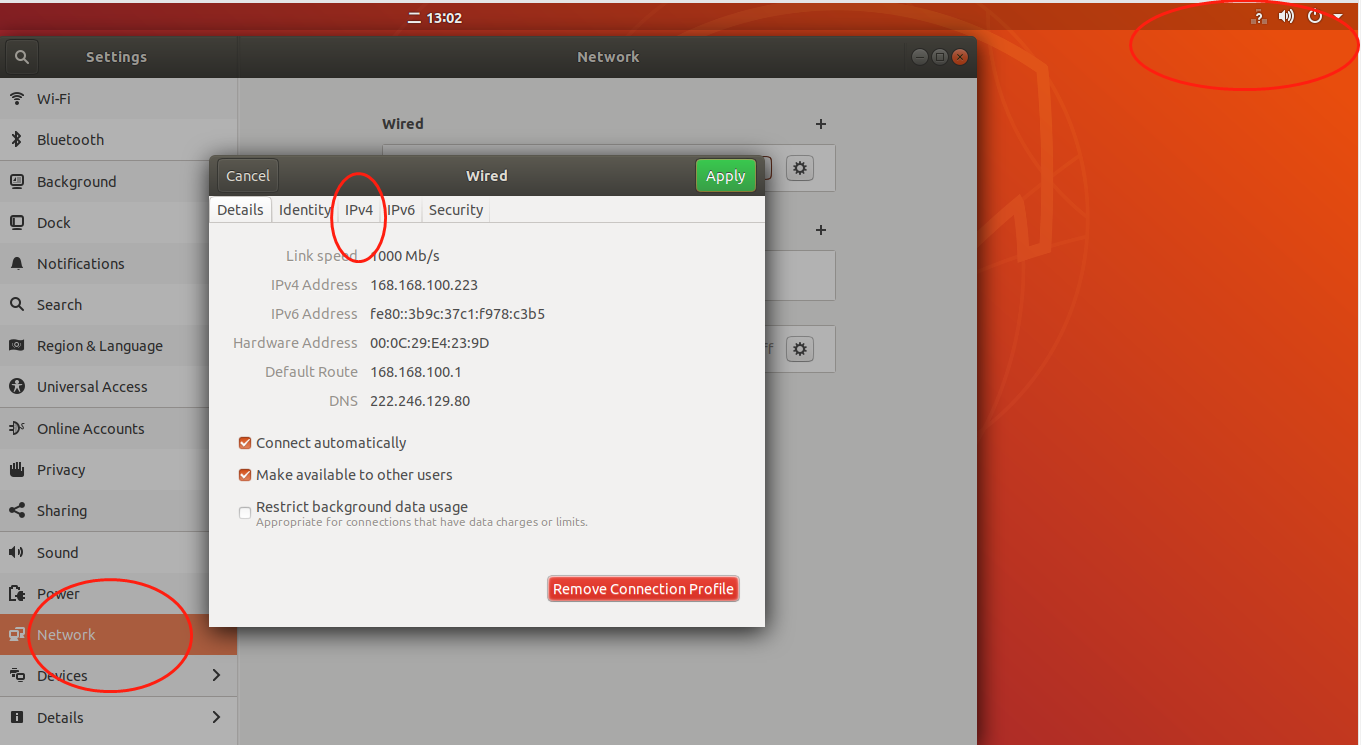
如果系统未设置root用户密码，可参考以下链接进行设置

<https://www.cnblogs.com/cursorhu/p/5803072.html>

## 网络配置

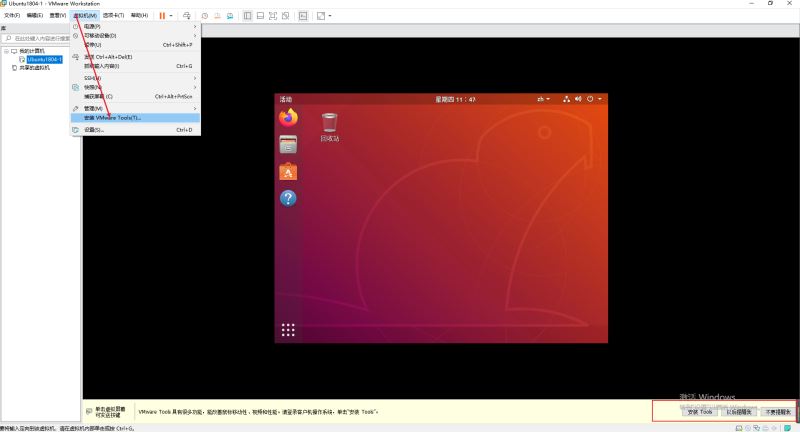
虚拟机网络配置如下，采用桥接模式，并设置虚拟机固定IP地址和DNS，确保外网能正常PING通。

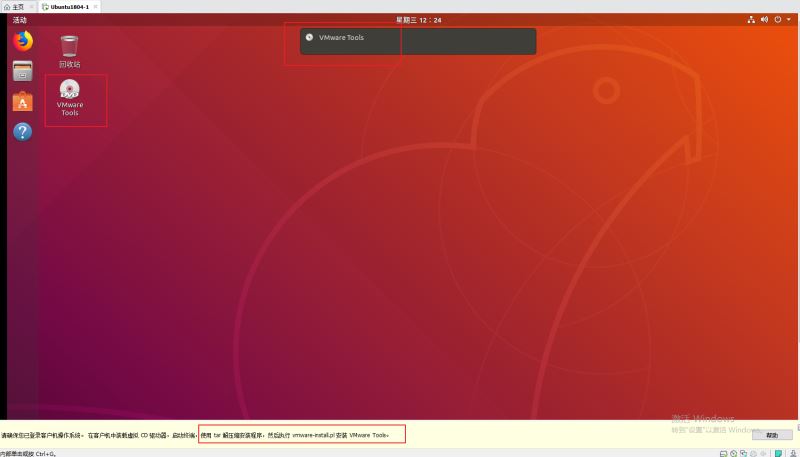


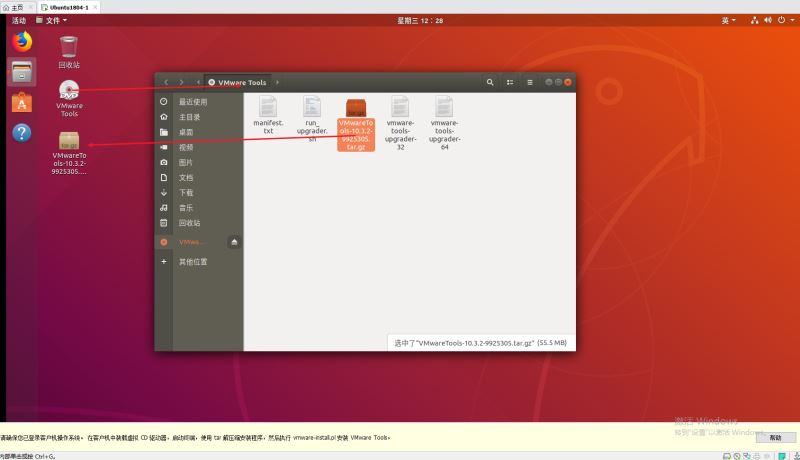


## VMware tool安装

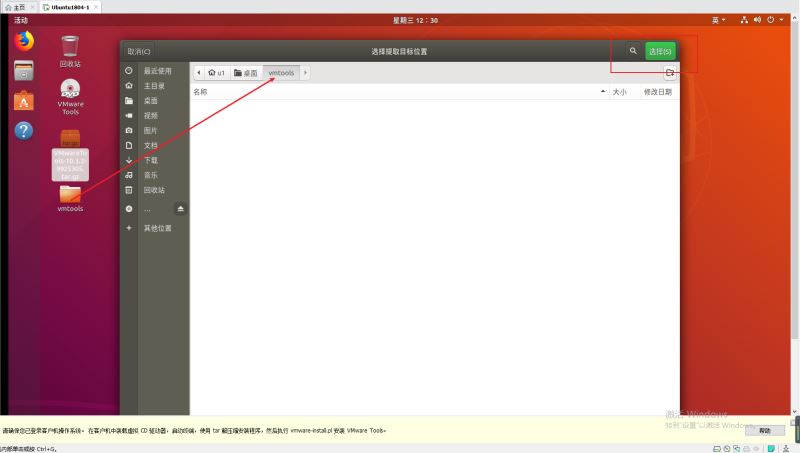
**解决复制粘贴、适应屏幕问题**

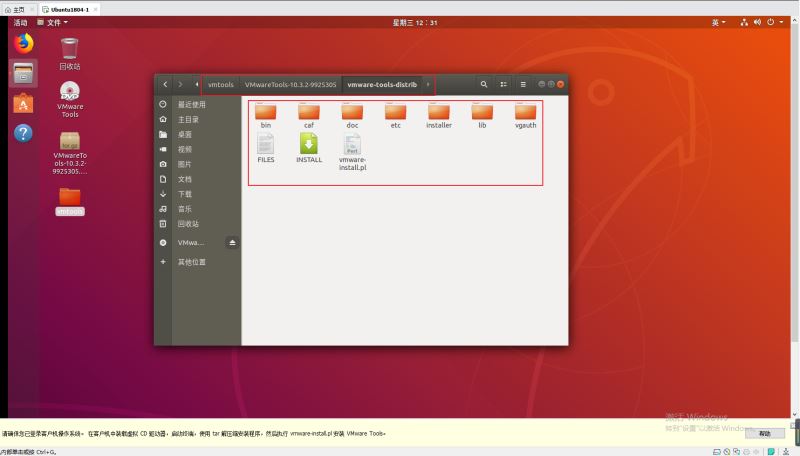


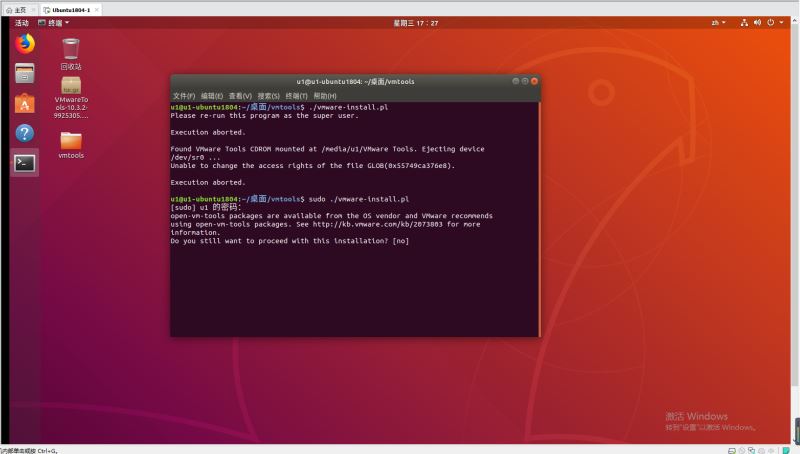


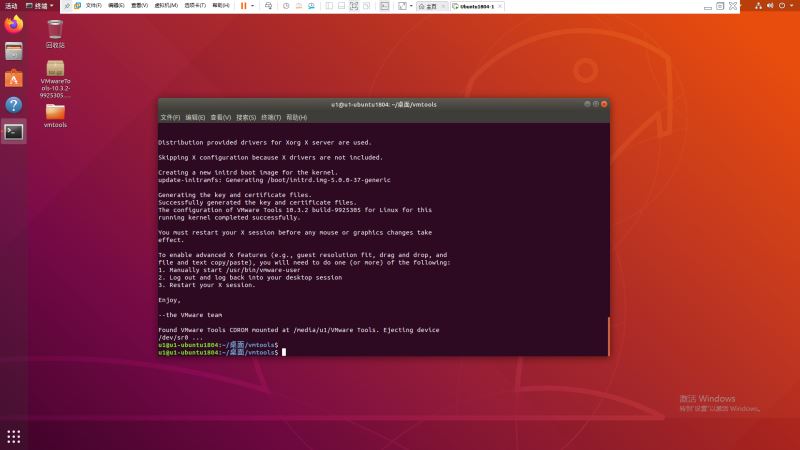












## 系统源更新

Ubuntu系统自带的源都是国外的网址，国内用户在使用的时候网速比较慢。一般建议更换成国内的阿里云源（阿里云、清华源、网易源）。可通过如下脚本进行快速更新。参考链接：<https://blog.csdn.net/kaikai136412162/article/details/91155439?utm_medium=distribute.pc_relevant.none-task-blog-baidujs-2>

目前虚拟机在/home/cftc/tools目录新建source-update.sh脚本，内容如下

#!/bin/sh

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo "当前系统版本："

lsb\_release -a

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

sudo echo "选择当前安装系统的版本："

echo " Ubuntu 14.04 请输入 1"

echo " Ubuntu 16.04 请输入 2"

echo " Ubuntu 18.04 请输入 3"

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

read SystemIndex

if [ -z $SystemIndex ];then

echo "输入格式错误"

exit

else

index=`echo "$SystemIndex\*1" | bc `

if [ $index -eq 1 ];then

System=14.04

elif [ $index -eq 2 ];then

System=16.04

elif [ $index -eq 3 ];then

System=18.04

fi

fi

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo "选择需要更新源："

echo " 阿里源请输入 1"

echo " 清华源请输入 2"

echo " 网易源请输入 3"

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

read SourceIndex

if [ -z $SystemIndex ];then

echo "输入格式错误"

exit

else

index=`echo "$SourceIndex\*1" | bc `

if [ $index -eq 1 ];then

echo 1111

Source=ali

elif [ $index -eq 2 ];then

echo 222

Source=qinghua

elif [ $index -eq 3 ];then

Source=163

fi

fi

echo $Source

filename="source\_${Source}\_ubuntu${System}.list"

echo "开始下载安装git..."

sudo apt-get install git

sudo apt-get -f install

sudo apt-get install git

echo "安装完成"

echo "开始下载库文件..."

git clone https://github.com/kaikai136/Ubuntu-China-Source.git

cd Ubuntu-China-Source

ls

echo "备份原链接..."

sudo cp /etc/apt/sources.list /etc/apt/sources\_init.list

echo "设置链接..."

sudo cp $filename /etc/apt/sources.list

echo "更新源..."

sudo apt-get update

echo "修复软件..."

sudo apt-get -f install

echo "更新软件..."

sudo apt-get upgrade

echo "清理文件..."

cd ..

sudo rm -rf Ubuntu-China-Source

echo "国内源清理完毕"

#rm -rf $(basename $0)

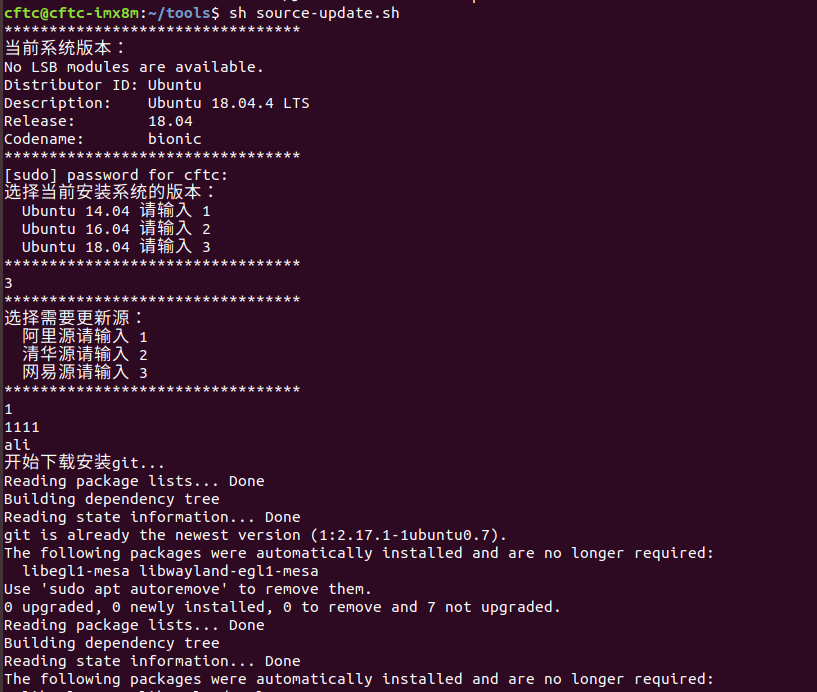
执行脚本:

cd ~/tools

chmod +x source-update.sh

sh source-update-sh

执行情况如下：选择ubuntu1804，阿里源。



## Github加速设置

Github可以手动通过修改hosts文件加速，也可以使用以下脚本进行加速，参考以下链接：

<https://github.com/dbarobin/github>

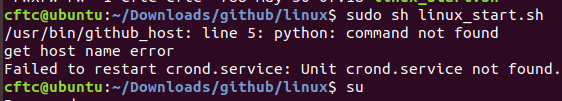
cd /home/cftc/tools

git clone https://github.com/dbarobin/github.git

cd github/linux

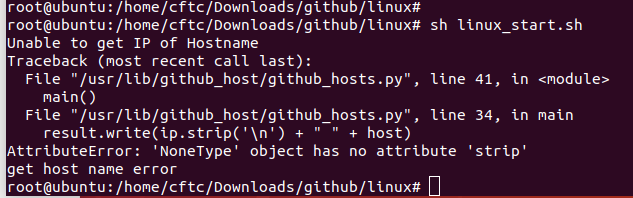
sudo sh [linux\_start.sh](https://github.com/dbarobin/github/blob/master/linux/linux_start.sh)

* 问题1：安装python。



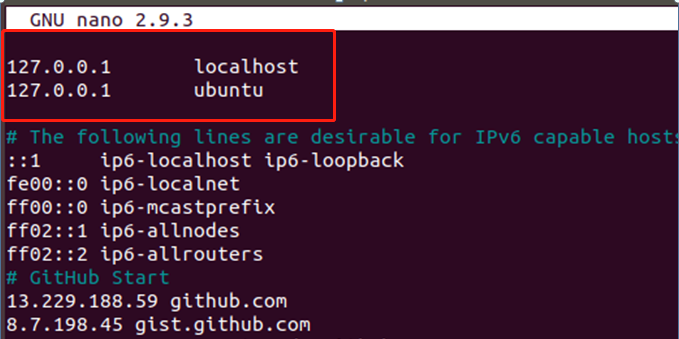
sudo apt-get install python

* 问题2：修改Hosts文件，配置hostname



sudo nano /etc/hosts

修改文件如下：



修改后，保存文件，系统会每隔一个小时更新github对应的IP地址到/etc/hosts文件中。

## Python升级

在ubuntu1404上编译yocto存在python源代码编译升级问题，此处折腾1d+，在ubuntu1804上无该需求。后续有需要再补充该部分内容。

## samba服务搭建

* 安装samba服务

sudo apt-get install samba samba-common

sudo apt-get install cifs-utils

sudo apt-get install smbclient

* 配置samba账户密码，密码设置cx125

sudo smbpasswd -a cftc

* 配置samba的配置文件

sudo cp /etc/samba/smb.conf /etc/samba/smb.conf.bak

sudo nano /etc/samba/smb.conf

在配置文件smb.conf的最后添加下面的内容：

[share]

comment = share folder

browseable = yes

path = /home/cftc/imx-yocot-bsp

create mask = 0700

directory mask = 0700

valid users = cftc

force user = cftc

force group = cftc

public = yes

available = yes

writable = yes

* 重启samba服务器。

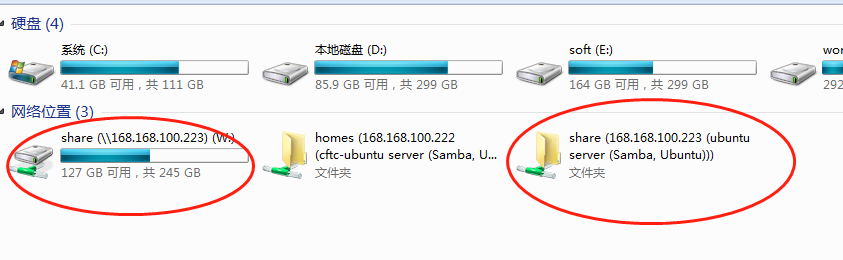
sudo service smbd restart

* ubuntu samba配置测试

smbclient -L //localhost/share

* ubuntu samba配置测试

Windows键 + R 在弹出的运行窗口中输入 \\ip即可访问。如[\\168.168.100.223](file:///\\\\168.168.100.223)，输入samba用户名及密码访问即可看到共享，然后就可以在Linux系统与Windows系统直接进行文件共享了。



## SSH服务端安装

sudo apt-get install openssh-server –y #安装openssh-server

sudo /etc/init.d/ssh start #启动ssh

sudo ufw disable #关闭防火墙

# 交叉编译环境及SDK安装

## 项目源码获取

#通过SVN获取项目源代码及SDK

$ cd ~/

$ svn checkout https://168.168.168.191:8443/svn/softgroup/projects/P62

## 源码结构说明

主目录下目录说明

├── drv ===>海思自带的外置驱动

├── mpp\_master ===>海思主模式视频音频内核模块及库

├── mpp\_single ===>海思Single模式视频音频内核模块及库

├── mpp\_slave ===>海思从模式视频音频内核模块及库

├── osdrv ===>存放操作系统及相关驱动的目录 ===>内核，uboot、busybox，应用，根文件系统

├── package ===>压缩包目录 ===>海思原始代码包

├── scripts ===>脚本目录 ===>脚本

├── sdk.cleanup ===>清除解压出的包脚本

└── sdk.unpack ===>解压最原始包脚本

osdrv下文件目录说明

├── drv ==>PCIE驱动

│   └── pcie\_mcc

├── kernel\_config ==>用户自定义内核配置

├── Makefile ==>跟Makefile

├── opensource ==>所有源资源存放目录

│   ├── applications ==>应用软件存放目录

│   ├── busybox

│   ├── kernel ==>内核

│   ├── logo ==>用户logo存放目录

│   ├── toolchain ==>工具链存放目录

│   ├── uboot ==>引导程序（uboot）

│   └── userdrv ==>外置驱动存放目录

├── pub ==>编译好的镜像、工具、drv驱动等 ==>编译输出文件存放目录

├── readme\_cn.txt

├── readme\_en.txt

├── rootfs\_scripts ==>源文件系统存放目录

│   ├── pack.sh

│   ├── rootfs.tgz ==>跟文件系统

│   └── userfs.tgz ==>应用文件系

├── Rule.mk ==>编译顶层配置：编译链，版本控制，

├── Makefile ==>顶层编译脚本 ==>编

└── tools ==>外置工具

## 工具链安装

$ cd ~/P62/trunk/Hi3536\_SDK\_V2.0.6.0/osdrv/opensource/toolchain/arm-hisiv400-linux

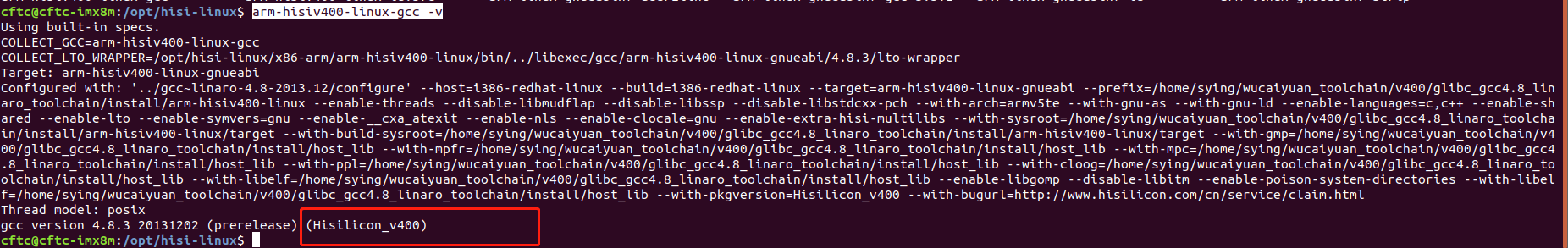
$ chmod +x cross.v400.install

$ sudo ./cross.v400.install

安装目录位于/opthisi-linux/目录下。安装完毕后测试是否安装成功

$ source /etc/profile

$ arm-hisiv400-linux-gcc -v



问题1：执行脚本用./test.sh的方式，如果采用source test.sh，则环境变量会生效。

./XXX.sh的时候，脚本里面打印PATH是改了，但是在终端echo $PATH却没有看到变化，因为这样执行等于说不在当前进程直接执行一个脚本文件是在一个子shell中运行的，而source则是在当前shell环境中运行的

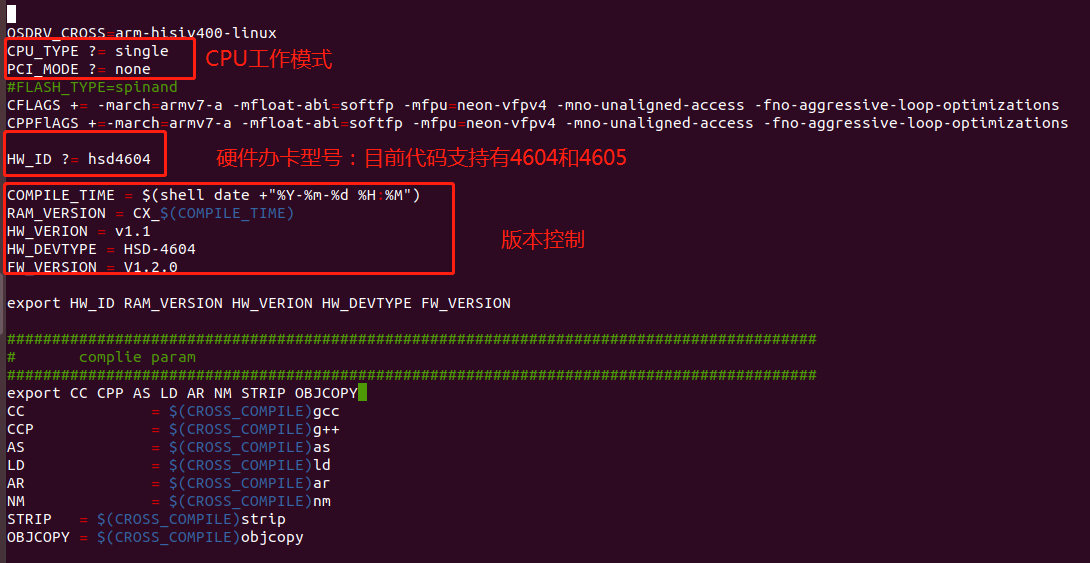
## 配置与编译

* 配置（在osdrv目录下）。修改Rule.mk文件，选择编译工具链及CPU是主从模式或single模式，还有软硬件版本。

CPU\_TYPE=master PCI\_MODE=master 表示主从模式

CPU\_TYPE=slave PCI\_MODE=slav 表示从模式

CPU\_TYPE=single PCI\_MODE=none 表示single模式



(2)、编译

$ cd ~/P62/trunk/Hi3536\_SDK\_V2.0.6.0/osdrv/

$ cp kernel\_config ./ opensource/kernel/linux-3.10.y/.config #

$ make #编译所有模块（uboot,busybox,kernel,rootfs）

**注意：在opensource/kernel/linux-3.10.y下存在.config, 不会重新从arch/arm/configs下拷贝配置，用户自定义的配置放在osdrv/kernel\_config下**

如果需要单独编译，可以使用以下指令

make hikernel 编译内核

make hikernel\_menuconfig 配置内核

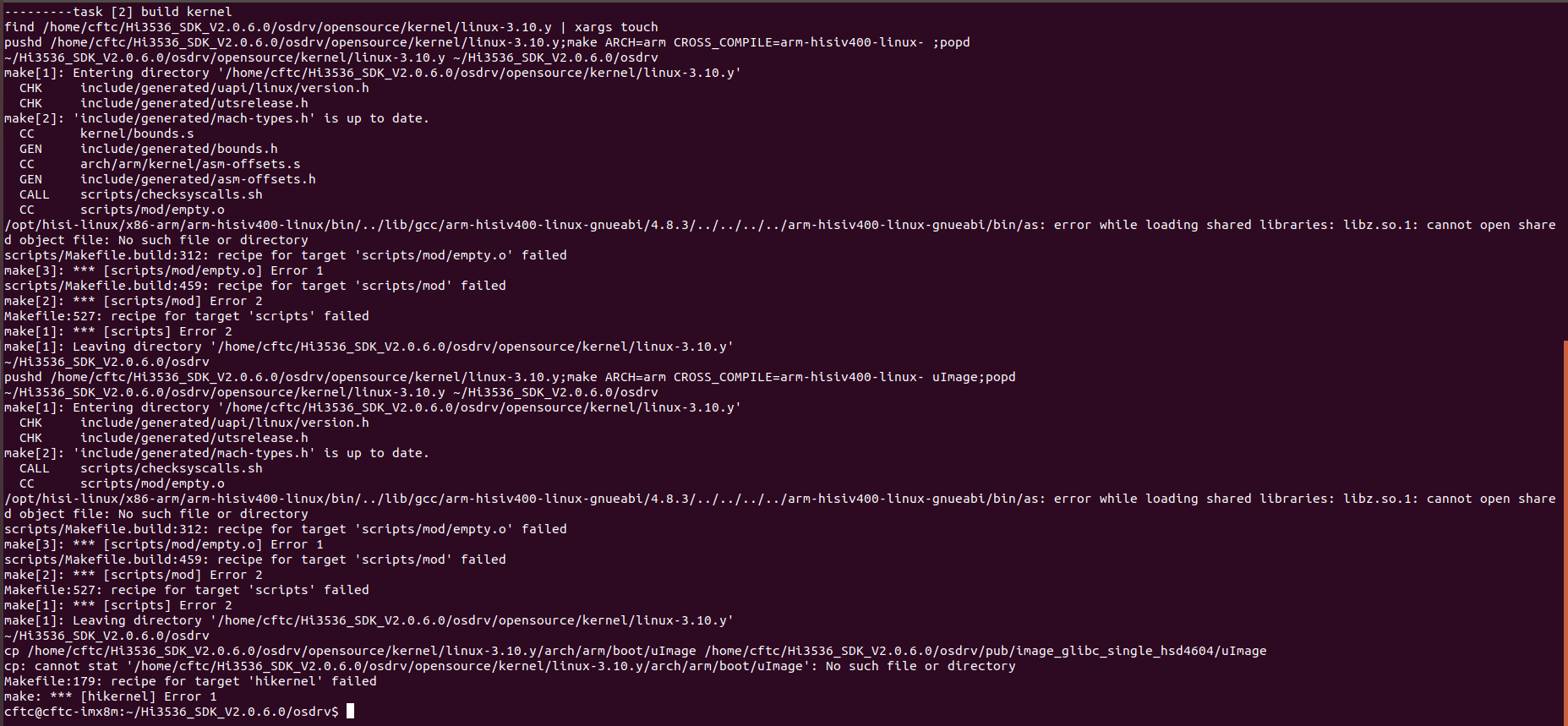
make hiboot 编译uboot

make hirootfs 编译文件系统

make hiuserfs 编译应用文件系统

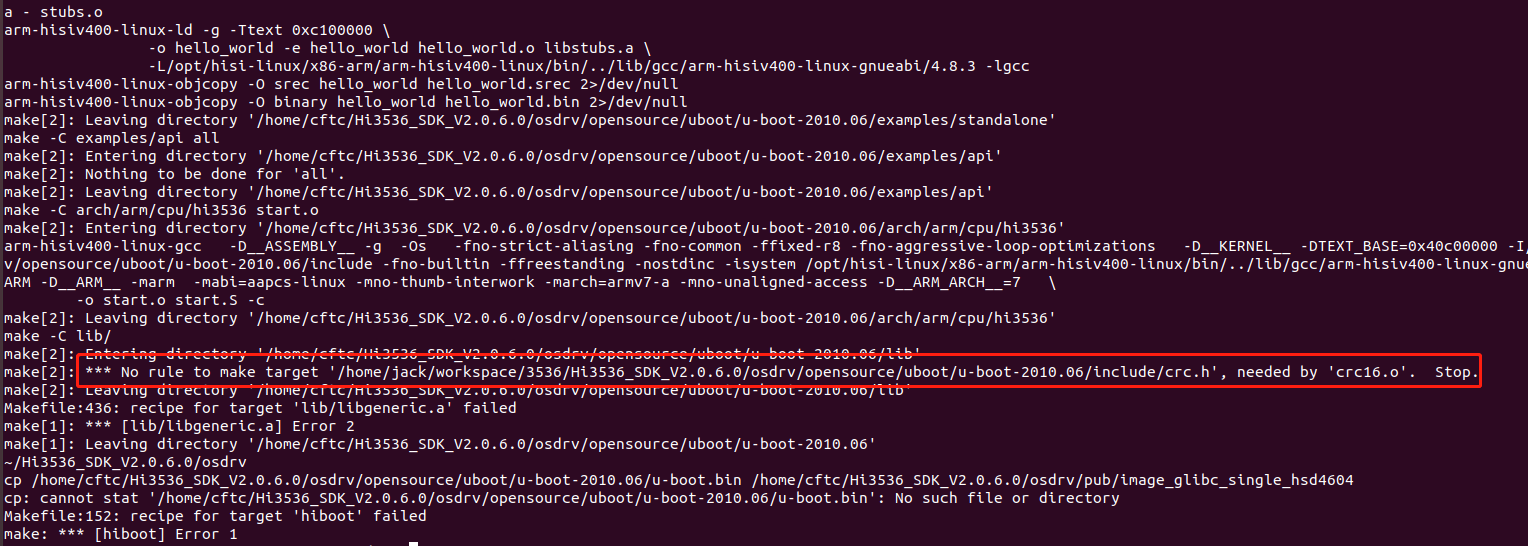
make hiuserdrv 编译外置驱动

问题1：缺少libz.so.1库文件

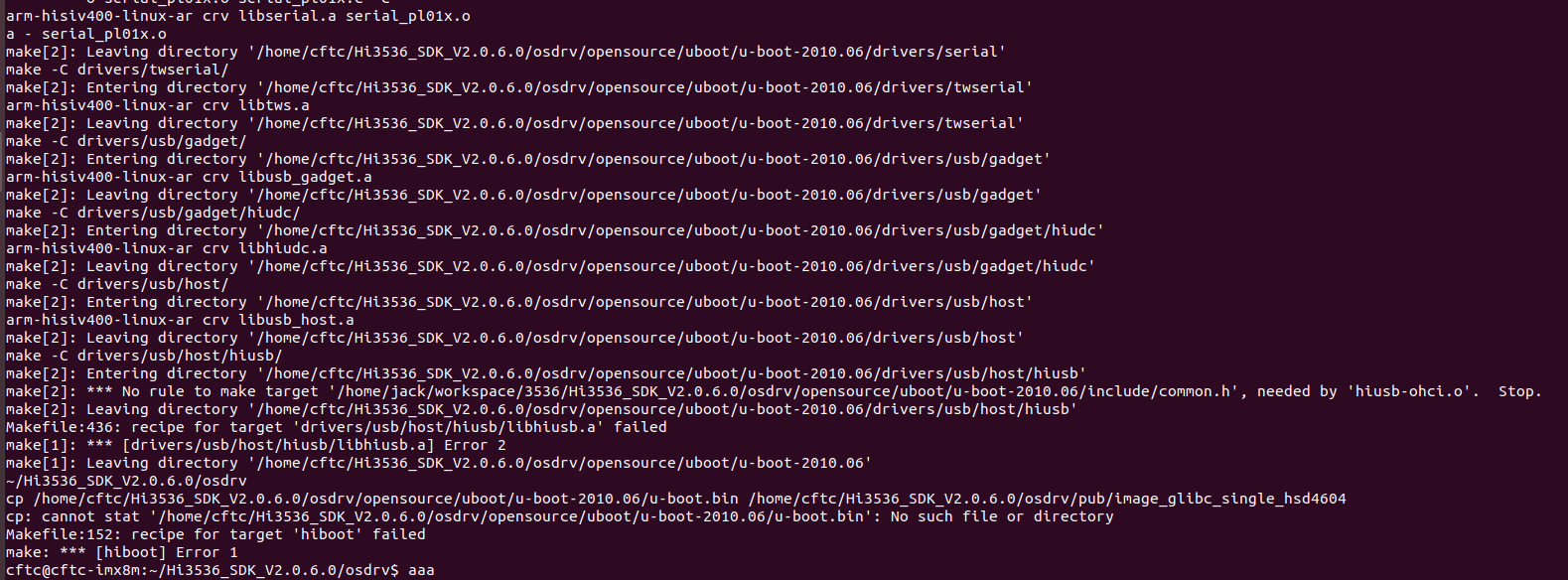


sudo apt-get install lib32z1

问题2：引用存在问题







**此问题可以通过删除depend文件解决。最终分析是由于代码是通过window端下载，然后压缩后拷贝到Linux下进行操作，可能存在文件格式问题，建议直接在Linux下通过SVN获取项目源码**

问题3：jffs2工具编译问题: 报错：serve\_image.c:32:18: error: storage size of ‘hints’ isn’t known



原因：ubuntu版本过高，SDK使用的mtd-utils版本过低导致。解决方法：  
修改SDK版本下的mtd-utils-1.5.0为高版本的mtd-utils-2.0.0.

$ cd ~/Download

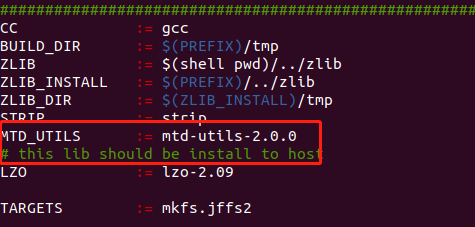
$ wget ftp://ftp.infradead.org/pub/mtd-utils/mtd-utils-2.0.0.tar.bz2

$ cp mtd-utils-2.0.0.tar.bz2 ~/~/P62/trunk/Hi3536\_SDK\_V2.0.6.0/osdrv/ \

$ opensource/tools/pc/jffs2\_tool

修改makefile文件替换mtd版本

$ nano Makefile



解压mtd-utils-2.0.0 这里面是没有Makefile文件的，但是有Makefile.am和Makefile.in文件，所以需要手动生成make文件

$ cd tmp/mtd-utils-2.0.0

$ ./configure

问题4：jffs2工具安装报错



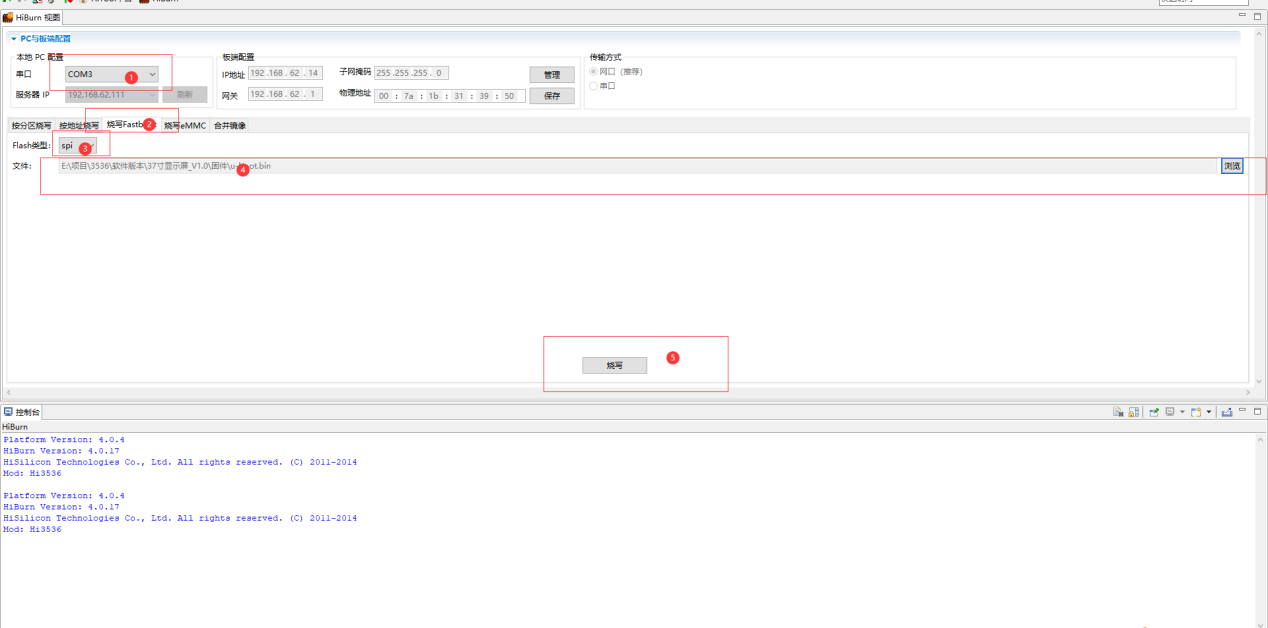
<https://blog.csdn.net/KerasCat/article/details/85163046>

# 文件更新与测试（下载说明）

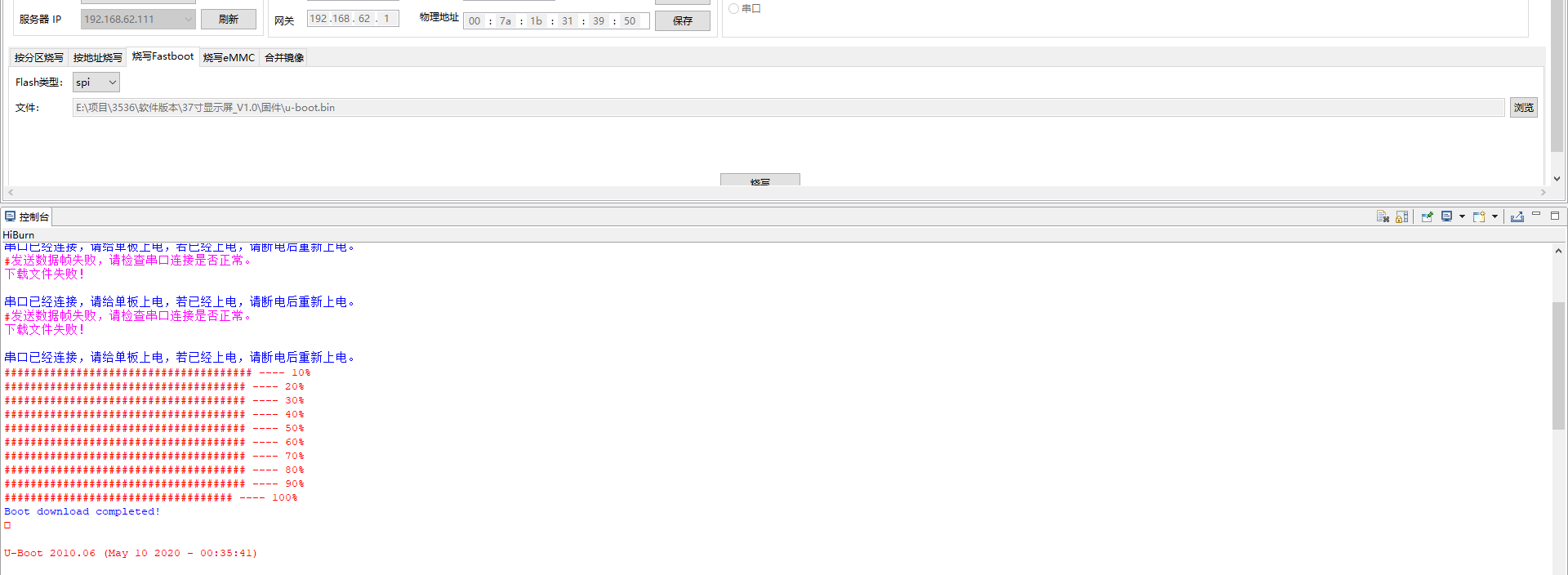
## HiTool升级

**（主要用于硬件裸板情况下，进行uboot烧写）**

* 使用串口连接pc和板卡调试串口（CPU1=>X9, CPU2=>X10）。
* 打开HiTool工具，选择串口，点击烧写fastboot。如下图：



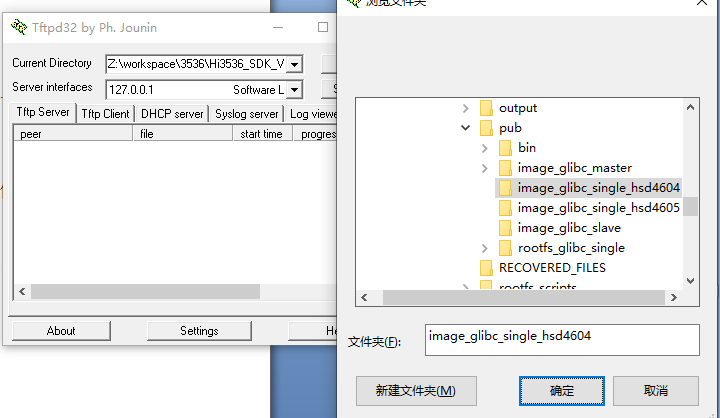
* 短接（CPU1🡺J3，CPU2🡺J57）
* 点击烧录。
* 板卡上电。如下图：



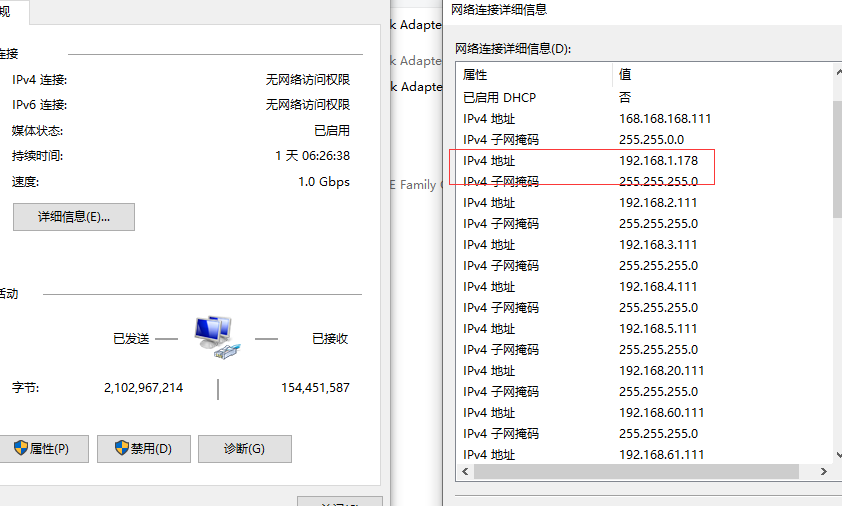
## 命令升级

（主要用于研发人员系统调试）

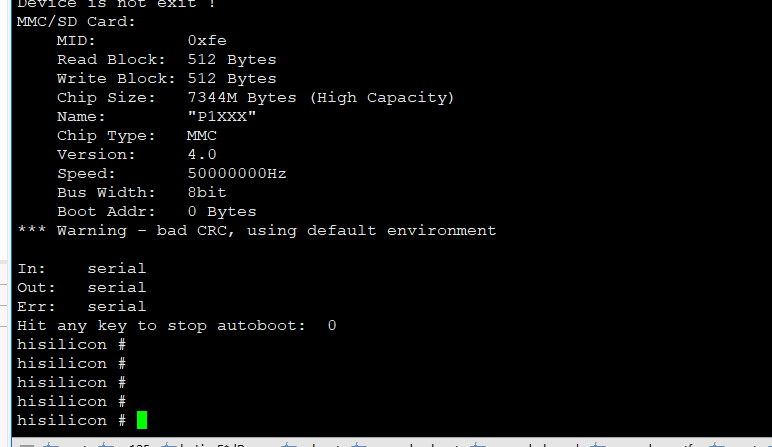
* + 准备工作：
* PC安装tftpd server服务器。
* 使用串口连接pc和板卡调试串口（CPU1=>X9, CPU2=>X10）。
* RJ45网线连接pc和板卡。
  + 升级步骤：
* 在PC机上打开tftpd server，并且选择升



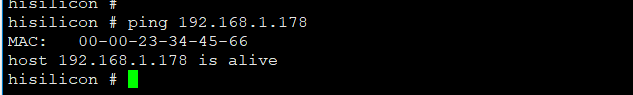
* 在PC端添加连接板卡的网口IP地址192.168.1.178。如下图：



* 在控制台（串口下），使系统停止在uboot模式下（在读秒的时候，按下回车键）。如下图：



* 在控制台上ping 电脑的IP地址，ping 192.168.1.178。如下图：如果出现is active，表示网络已通。否则需要检查环境（是否关闭防火墙、地址是否正确）。如下图：

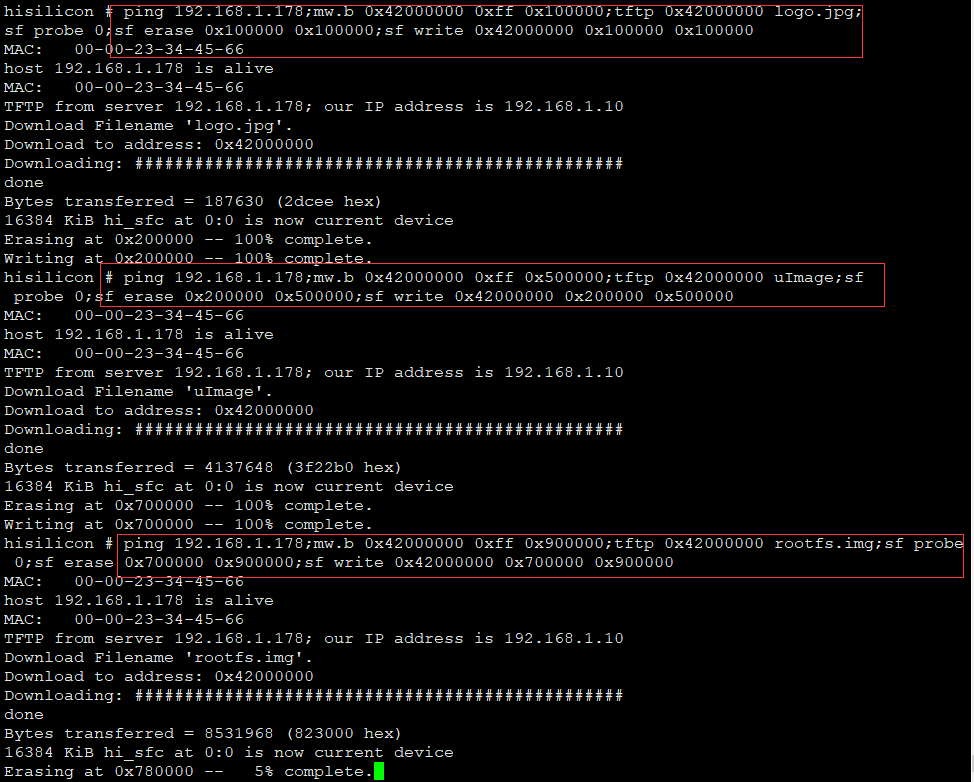


* 命令行升级内核和根文件系统，如下图：

升级uboot：ping 192.168.1.178;mw.b 0x42000000 0xff 0x100000;tftp 0x42000000 u-boot.bin;sf probe 0;sf erase 0x0 0x100000;sf write 0x42000000 0x0 0x100000

升级logo：ping 192.168.1.178;mw.b 0x42000000 0xff 0x100000;tftp 0x42000000 logo.jpg;sf probe 0;sf erase 0x100000 0x100000;sf write 0x42000000 0x100000 0x100000

升级内核：ping 192.168.1.178;mw.b 0x42000000 0xff 0x500000;tftp 0x42000000 uImage;sf probe 0;sf erase 0x200000 0x500000;sf write 0x42000000 0x200000 0x500000

升级根文件系统：ping 192.168.1.178;mw.b 0x42000000 0xff 0x900000;tftp 0x42000000 rootfs.img;sf probe 0;sf erase 0x700000 0x900000;sf write 0x42000000 0x700000 0x900000

* 重启之后，需要网络升级方式升级应用分区（userfs.img和conffs.img）。

Uboot环境变量修改说明：单板上电后，敲任意键进入u-boot。设置serverip（即tftp服务器的ip）、ipaddr（单板ip）和ethaddr（单板的MAC地址）。

setenv serverip xx.xx.xx.xx

setenv ipaddr xx.xx.xx.xx

setenv ethaddr xx:xx:xx:xx:xx:xx

setenv netmask xx.xx.xx.xx

setenv gatewayip xx.xx.xx.xx

ping serverip，确保网络畅通。

系统分区及烧录说明

1）16M FLASH地址空间说明

| 1M | 1M | 5M | 9M |

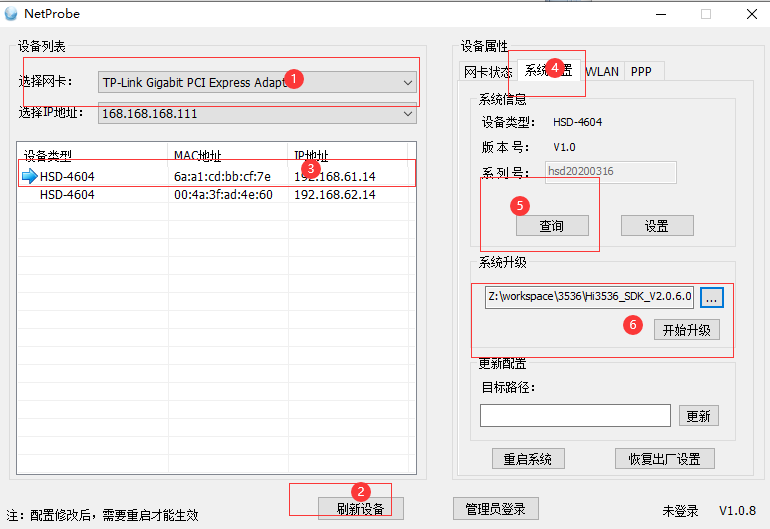
|---------------|---------------|---------------|-------------------|

| boot | logo | kernel | rootfs |

## 网络升级

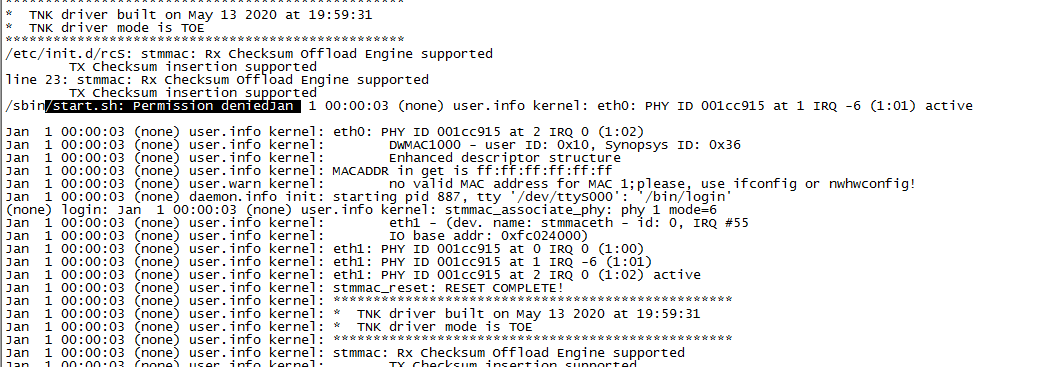
（主要用于客户升级）

* RJ45网线连接pc和板卡。
* 打开nettool.exe工具。选项连接的网卡刷新，找到对应的设备型号，然后点击系统配置，最后选择升级文件升级即可。如下图



## 测试碰到的问题

问题1：**Application目录脚本权限不够，启动报错**



$ cd ~/P62/trunk/Hi3536\_SDK\_V2.0.6.0/osdrv/opensource/applications/scripts

$ chmod 777 user\_hsd4605\_start.sh

$ cd ../../../

$ make hirootfs

**问题2：emmc无法挂载，**用户文件系统无法访问。（未截图）观察内核编译日志，发现mmc确实未编译进内核，但是.config中已配置mmc相关宏定义。通过查找发现，在内核include/config目下，auto.conf文件里面没有mmc相关配置宏定义，此问题困扰很久。和陈光强分析是由于一直未执行scripts/kconfig/conf --silentoldconfig Kconfig，导致MMC配置未更新，编译错误，原理暂时不清楚。目前解决方案是我在内核目录下去执行make menuconfig此时会更新include目录下的auto.conf文件，这样整个个编译过程才正确。

问题3：内核完毕后调用脚本启动系统应用报错（不影响使用）

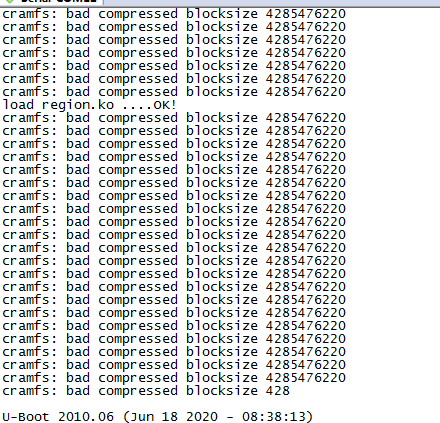
（此处nettools只是用于无用户文件系统加载的情况下，对用户文件系统进行更新，系统启动完毕后，会关闭nettools，然后从用户文件系统处启动nettools）



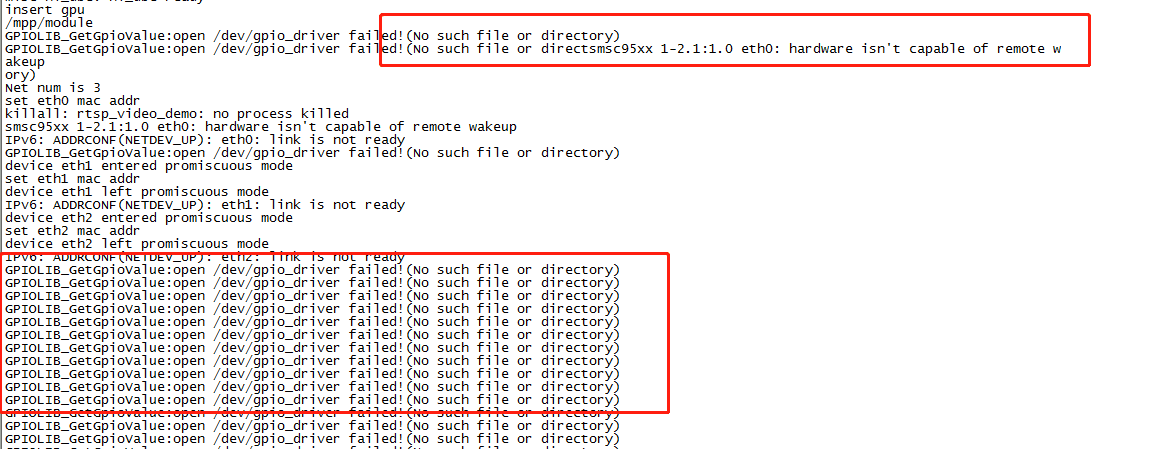
由于陈光强修改代码，导致nettool会依赖libsysapp.so，而libsysapp.so位于用户文件系统中，因此导致内核启动根文件系统的nettool报错。修改nettools，删除libsysapp.so相关依赖。参照如下修改代码并去掉Makefile中对libsysapp.so的依赖即可。



问题4：nettools静态编译，rootfs超过9M，导致系统启动报错。



问题5：升级userfs的时候，是在重启之后才拷贝到/home/user下，导致gpio驱动没有加没有加载导致。重启即可



# Uboot加载LOGO说明

海思3536芯片uboot下使用hdmi显示1920x1080的logo图片。（详细请参考《HiMPP 开机画面使用指南.pdf》）

* + 要求logo图片必须是1920x1080的jpeg格式图片
  + 添加环境变量（以下变量值是针对芯片3536的）

jpeg\_addr=0x43000000 # 用于存放源图片的内存地址

jpeg\_size=0x10000 # 图像的大小

vobuf=0x43800000 # 解码输出地址

* + 将准备好的图片，上传保存在flash里面；例如

ping 192.168.1.178;mw.b 0x42000000 0xff 0x100000;tftp 0x42000000 logo.jpg;sf probe 0;sf erase 0x100000 0x100000;sf write 0x42000000 0x100000 0x100000

* + 启动显示logo图片，如下命令：

sf probe 0; # 打开flash读写保护

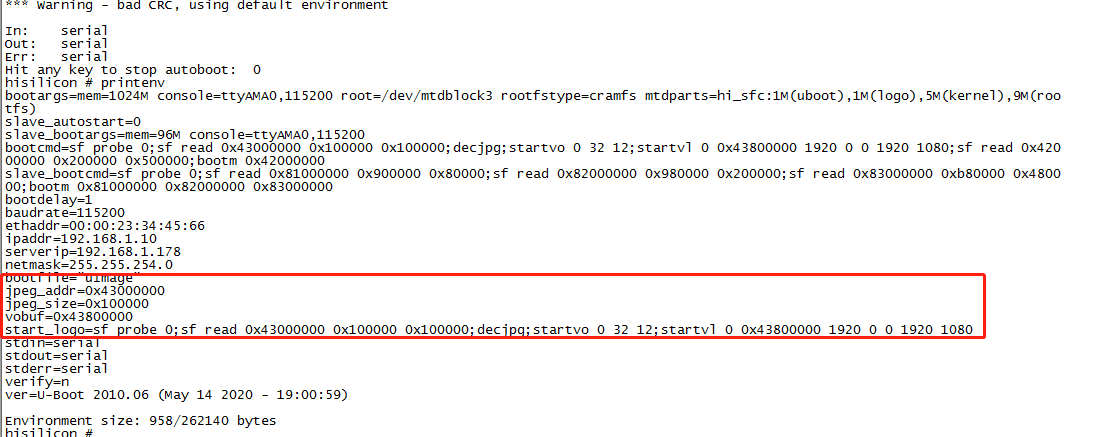
sf read 0x43000000 0x100000 0x100000; # 将图片从flash 0x100000位置读到内存地址0x43000000上

decjpg; # 图像JPEG解码，会解码到环境变量vobuf位置上

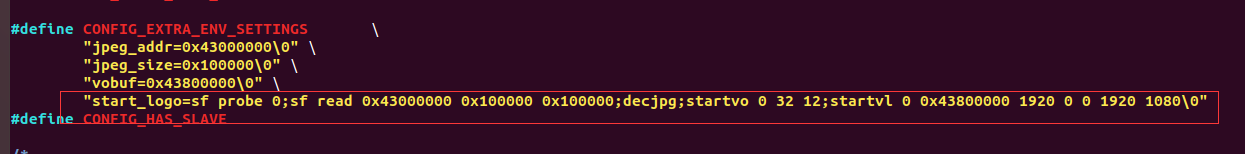
# 解码如果报错（1、请检查地址是否正确，2、图片是否正确）

startvo 0 32 12; # 启动VO设备（0==>设备号，32(HDMI)==>接口类型，12(1080p@60)==>时序类型）

startvl 0 0x43800000 1920 0 0 1920 1080; # 启动视频层 # 1920==>图像行宽，0 0 1920 1080==>显示位置(x,y,w,h)）

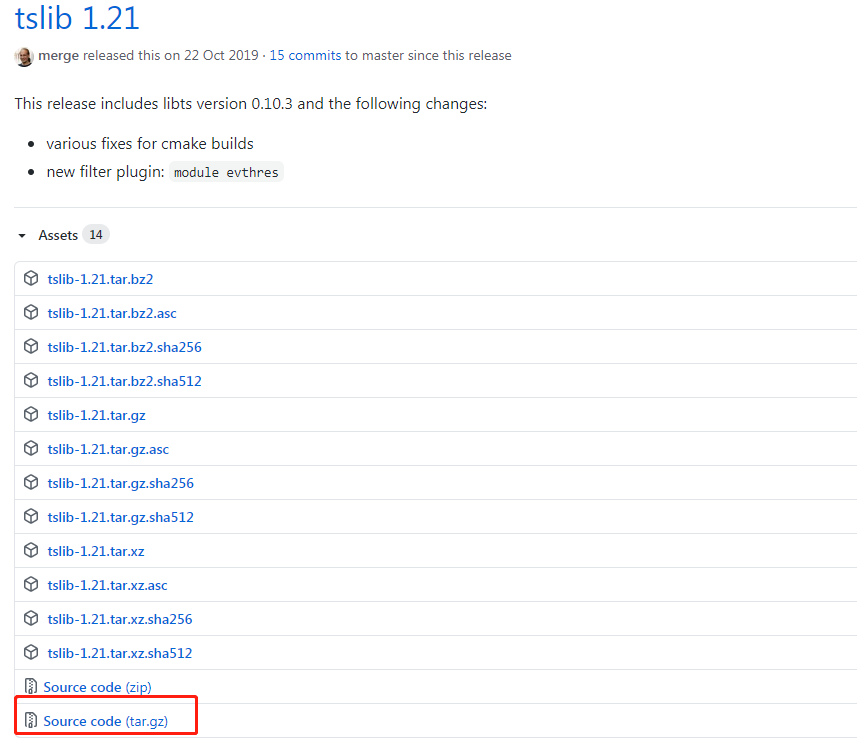


在uboot中修改了如下代码：include/configs/hi3536.h文件修改如下



# tslib编译

1. 下载tslib源代码包（<https://github.com/libts/tslib/releases/tag/1.21>）



2、解压配置编译

$ cd ~/

$ tar –xvf tslib-1.21.tar.gz

$ cd tslib-1.21

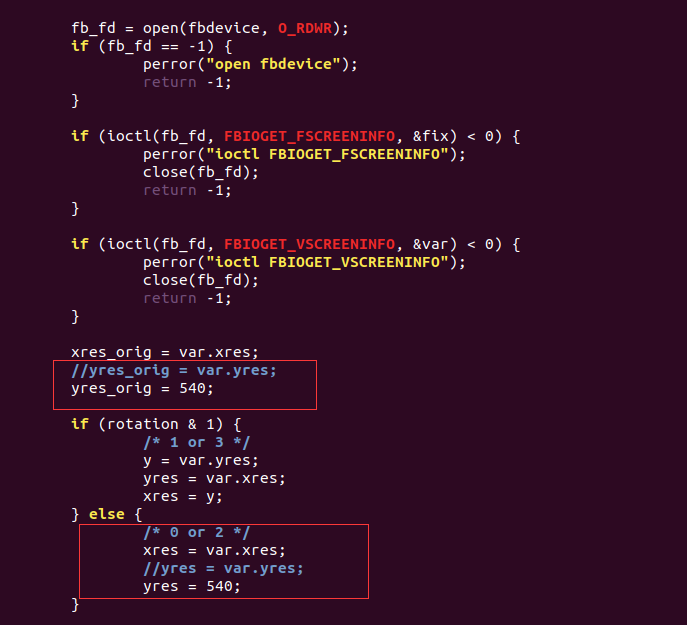
$ . autogen.sh

$./configure --prefix=/opt/tslib1.21 --host=arm-linux ac\_cv\_func\_malloc\_0\_nonnull=yes CC=/opt/hisi-linux/x86-arm/arm-hisiv400-linux/bin/arm-hisiv400-linux-gnueabi-gcc

$ make

$ make install

对于该项目，屏幕分辩率是1920 \* 1080，但是屏幕尺寸为1920 \* 540，所以在编译tslib时，需要对X，Y的分辨率进行修改，否则系统默认从linuxfb中进行读取信息，导致校准失败。修改文件/tests/fbutils-linux.c如下：



安装完成需编译进入用户文件系统

$ cd ~/P62/trunk/Hi3536\_SDK\_V2.0.6.0/osdrv/opensource/applications/ts-utils/

$ cp /opt/tslib1.21 ./tslib-1.21

$ cd ../../../

$ make hiuserfs

# Qt编译

$ cd ~/

$ tar -xvf qt-everywhere-opensource-src-4.8.1.tar.gz

$ mv qt-everywhere-opensource-src-4.8.1 Qt4.8.1

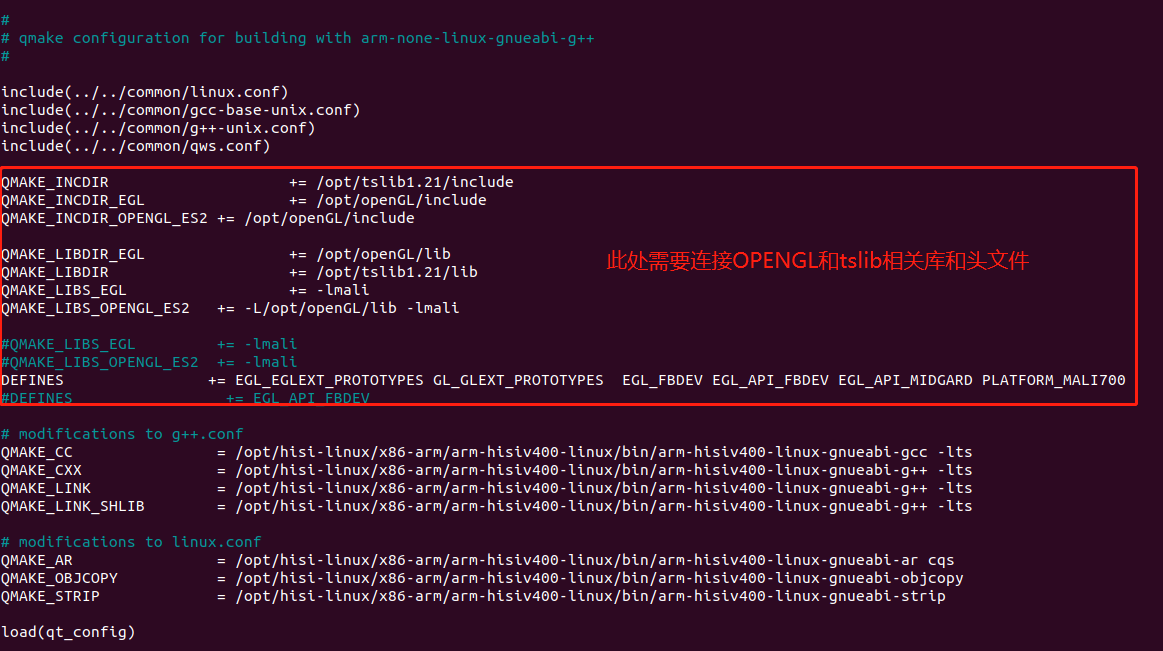
#增加qmake配置

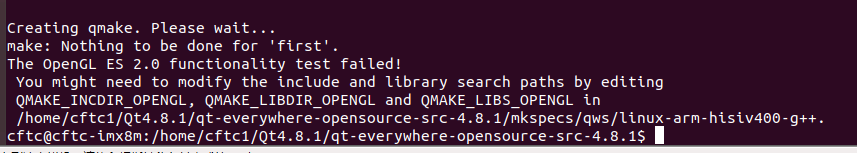
$ cd Qt4.8.1/mkspecs/qws

$ cp -r linux-arm-gnueabi-g++/ linux-arm-hisiv400-g++/

$ nano qmake.conf

修改makefile如下



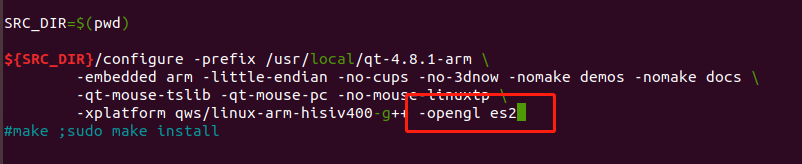


保存并返回Qt源代码根目录，新建配置脚本

$ cd ../../../

$ nano autoconfig.sh

配置脚本内容如下：



$ chmod +x autoconfig.sh

$ . autoconfig.sh

等待配置完毕后，在编译安装

$ make

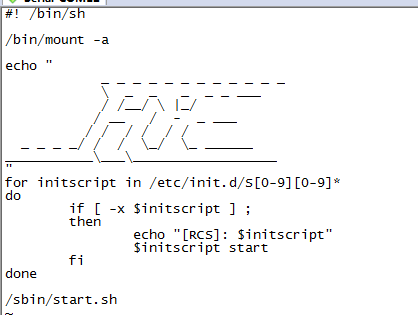
$ make install

默认安装路径在/usr/local下面文件夹qt-4.8.1-arm。拷贝至SDK中，编译进文件系统中。

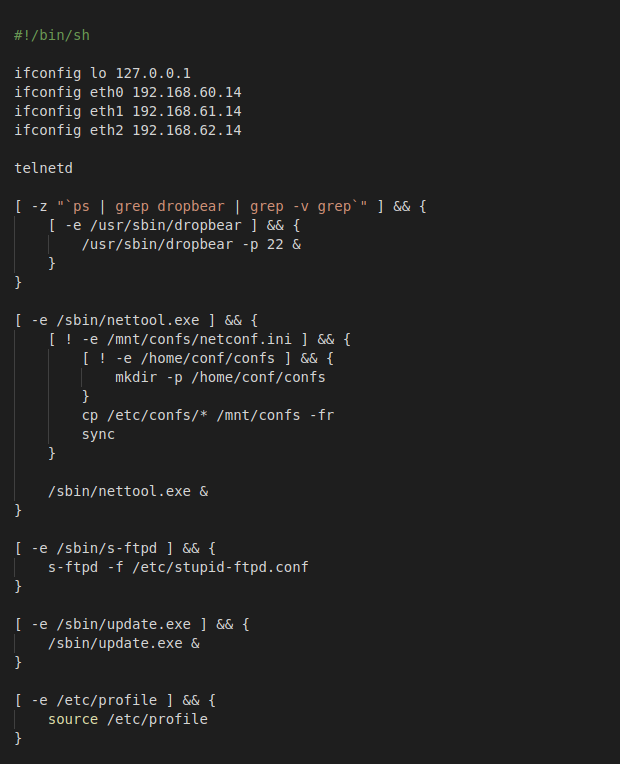
**此处浪费了两天时间**，其中OpenGL库有海思提供，从application目录下面拷贝的文件导致Qt配置OPENGLES2时不通过，通过网上的各种方法都不成功，怀疑为虚拟机系统环境问题导致，从而走了很多弯路。最后从海思的mpp\_single/component/gpu/release目录下拷贝完整的头文件和库文件，然后在配置，问题解决。（此处体现文档的重要性！！！）后续分析发现拷贝至application下的文件为陈光强用于文件系统生成，但是编译的时候需要用到KHR文件夹下的头文件信息！！此处浪费了太多的时间！！！

# 脚本文件调用说明（这部分最好形成流图的形式）

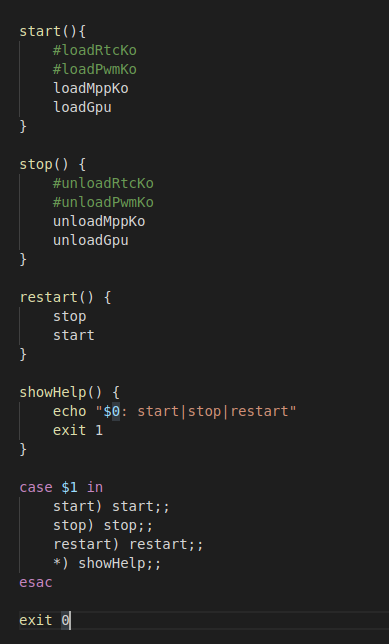
内核启动完毕后调用start.sh脚本（/etc/init.d/ rcS）



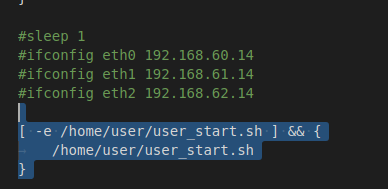
脚本启动执行顺序：/sbin/start.sh



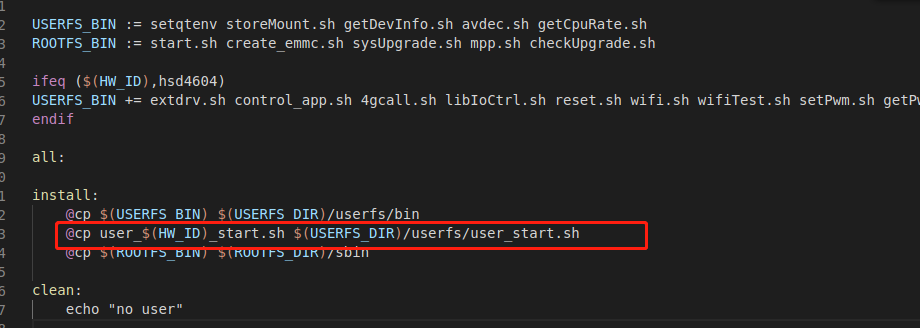
启动ssh（dropbear），加载mpp.sh，启动视频/图像处理库（海思的东西，内核模块加载以及一些参数设置），需要注意/mpp/module/load3536脚本，**里面有关于内存分配相关的参数（loadgpu +** load3536**）**



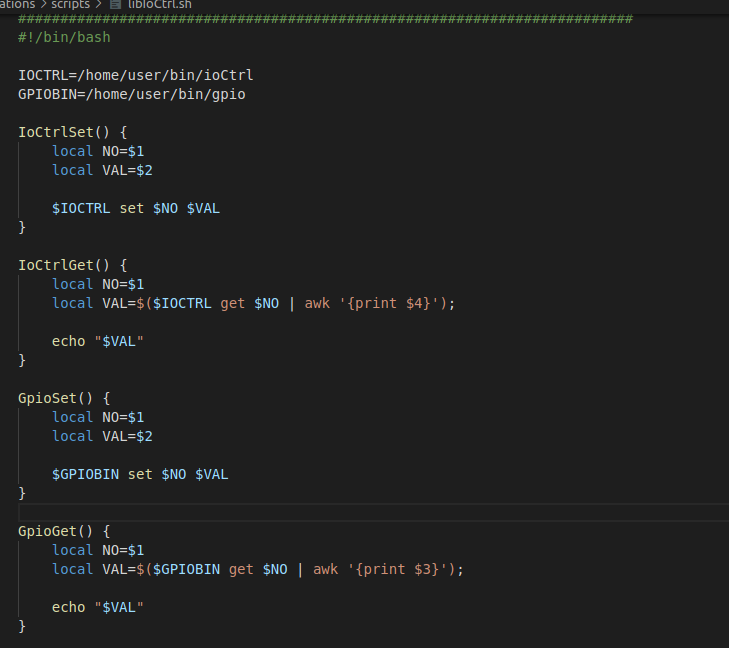
加载user\_start.sh脚本，执行用户应用相关初始化工作



根据HW\_ID选择不同的用户启动脚本，目前支持HSD4604和4605



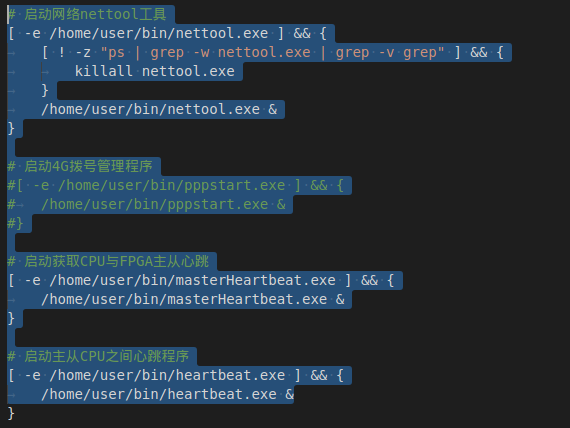
加载io操作脚本：/home/user/bin/libIoCtrl.sh。好像没有干什么



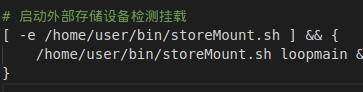
加载外部驱动：/home/user/bin/extdrv.sh

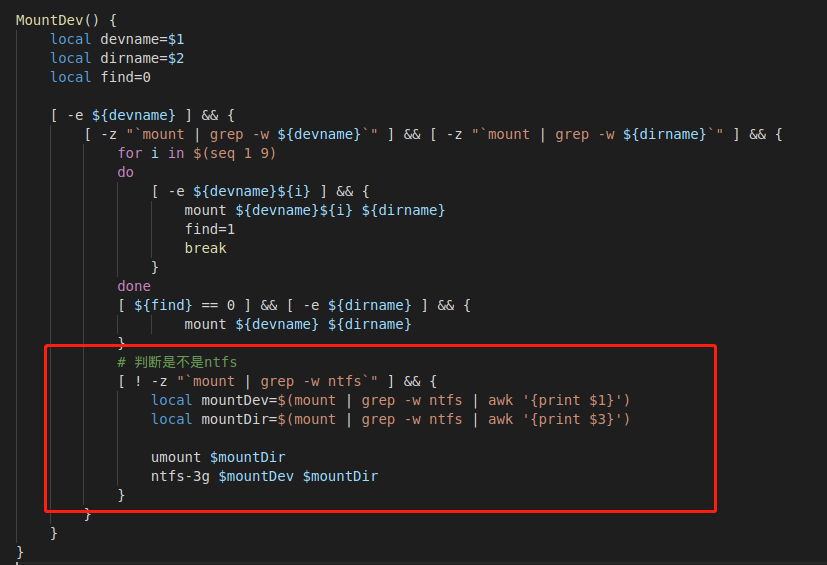


启动系统应用程序

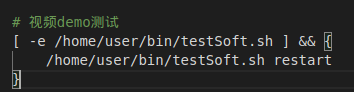


通过/home/user/bin/storeMount.sh脚本，挂载外部文件系统。如果U盘系统为ntfs格式，需要umount后采用ntfs-3g进行挂载（否则会出现无权限读写或者无法加载）





加载视频测试demo脚本：/home/user/bin/testSoft.sh

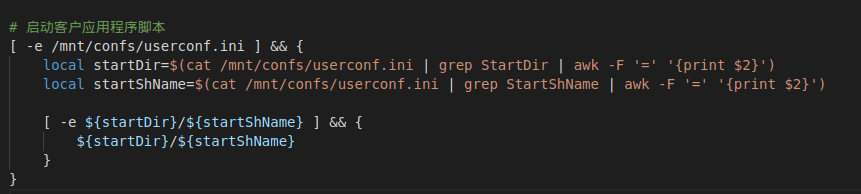


删除校准程序，删除测试程序，**fbopen？？主从程序加载不同的QSS文件，启动测试程序？**

**rtsp\_video\_demo是否可以放到TestStop去执行。？**



启动客户应用程序脚本：可以通过/mnt/confs/userconf.ini文件配置应用程序的启动脚本。默认为StartDir，StartShName。至此整个系统启动完毕。



4gcall.sh：4G拨号测试脚本

avdec.sh：音视频测试脚本

checkUpgrade.sh：检测是否需要进行文件更新脚本

connectWifi.sh：wifi连接测试脚本

control\_app.sh：heartbeat.exe应用程序调用，开启应用程序？

create\_emmc.sh：EMMC分区相关操作

getCpuRate.sh：获取CPU使用率？？

getDevInfo.sh：获取设备主从信息

getPwm.sh：获取屏幕亮度设定值

manageWlanStart.sh:wlan相关控制，供control\_app.sh脚本调用，

mmcsfdisk.sh：》》》？？？

wakeUpTouch：

setPwm.sh：设置屏幕亮度

setqtenv.sh：设置QT环境变量

sysUpgrade.sh：系统自动升级脚本

wakeUpTouch.sh：唤醒触摸屏，有heartbeat调用

reset.sh：wifi，4G复位脚本

wifi.sh：

wifiTest.sh：wifi测试脚本，调用reset.sh和wifi.sh脚本文件，

# 软件调试操作方法说明

## 触摸屏

* + 查看是否加载触摸屏驱动

lsmod | grep egalax\_i2c // 如果有输出。表示已经加载

* + 加载触摸屏驱动

insmod /lib/modules/3.10.y/kernel/egalax\_i2c.ko

rwTouch write 0

* + 触摸屏打开内核调试信息

echo 8 4 1 7 > /proc/sys/kernel/printk

echo 8 > /proc/egalax\_dbg

* + 查看触摸屏是否有效

cat /dev/input/event0 // 点击触摸屏是否有打印，表示有效

* + 卸载触摸屏驱动

rmmod egalax\_i2c

## 视频

* + 查看是否运行视频软件

ps | grep videoSample.exe | grep -v grep //如果有输出，表示已运行，反之

如果后加了-n 1表示CPU2，如果没有加，则表示CPU1

* + 执行视频软件

/home/user/sample/videoSample.exe 或者 /home/user/sample/videoSample.exe -n 1

* + 关闭视频软件

killall videoSample.exe

## 读取CPU与FPGA之间的串口

* + 读取寄存器的值

fpgaSerial read 0x0 0x01 // 读取寄存器0x0的值

* + 设置寄存器的值

fpgaSerial write 0x0 0x1 // 设置寄存器0x0的值为0x1

## 设置CPU与FPGA之间的心跳

* + 开启心跳

gpio addTimer 354 2 // 354表示GPIO的编号对应GPIO11\_2

// 2表示时间间隔2ms

* + 关闭心跳

gpio delTimer 354 // 354表示GPIO的编号对应GPIO11\_2

## 无线wifi

* + 查看是否找到模块ID（lsusb），若是有9113，表示已经找到。
  + 若是没有找ID，可以reset.sh wifi 复位无线。再查看ID

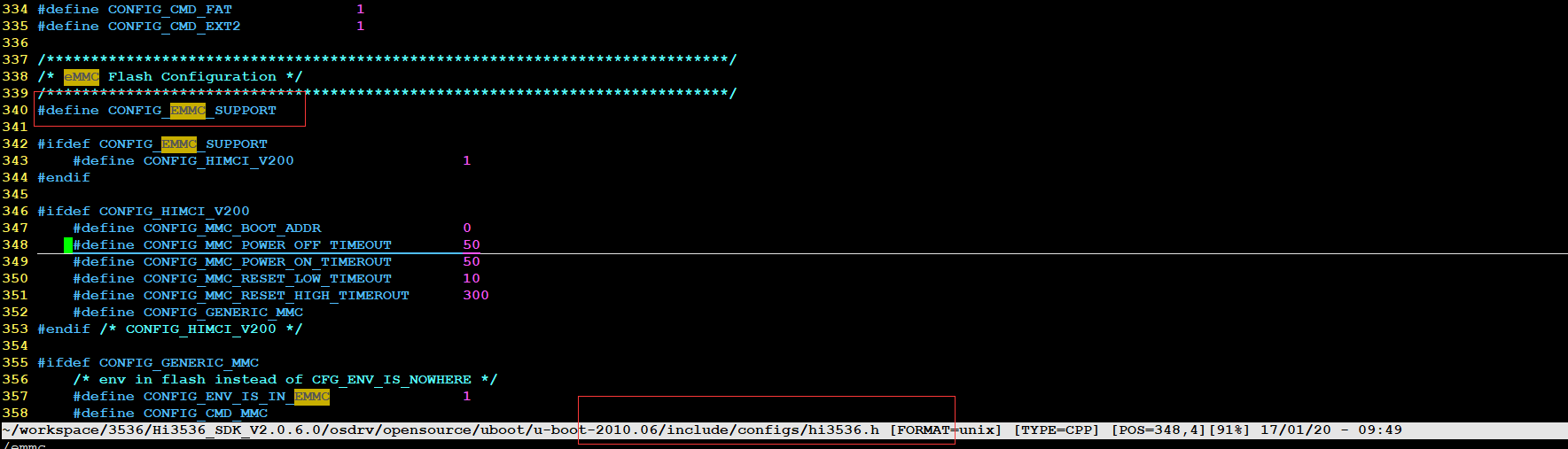
# 软件调试遇到的问题（陈光强）

* 1. emmc无法识别问题

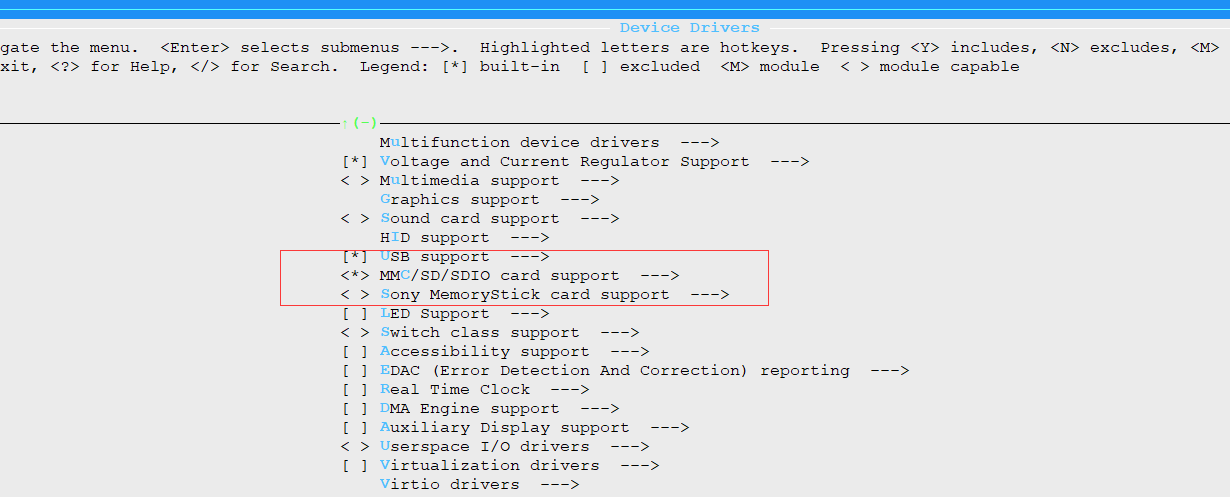
原因：（1）、uboot和kernel没有打开关于emmc配置

（2）、emmc的gpiio管脚复用，初始化错误

解决：打开uboot配置，如下图



打开内核配置，如下图



配置复用管脚

* 1. 双CPU模式，内核出现“slave boot addr is error”报错

原因：启动从uboot的时候，没有找到启动地址，传入的地址是0，使用全局指针变量保存地址值，退出函数之后值就掉了。

解决：uboot代码arch/arm/lib/bootm.c 将slave\_boot\_start指针变量改成数组即可。

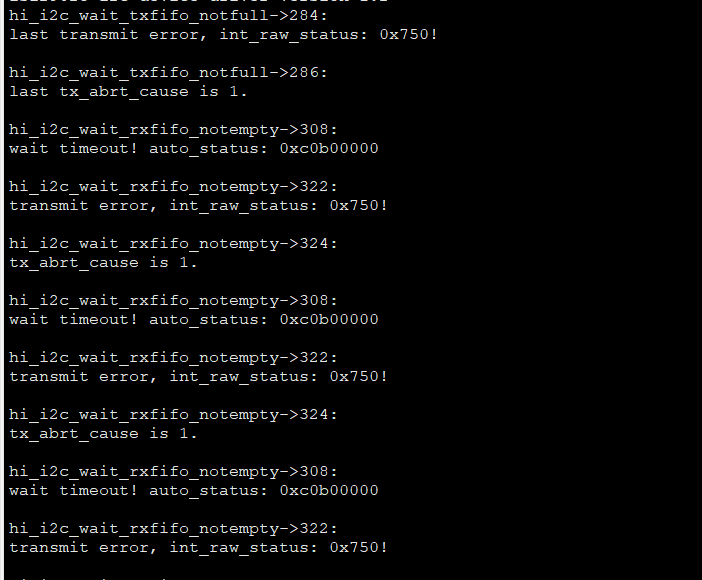
* 1. 加载mpp驱动驱动时，出现IPCM死循环。如下图：



原因：在主从模式下。如果从CPU没有起来，没有加载mpp驱动，就会导致它们之间无法通信交互，就会出现一端没有回复，一直等待，出现死循环此现象。

解决：加载从CPU端mpp驱动。

4、加载mpp驱动驱动时，出现i2c读写错误， 死循环现象。如下图：



原因：加载某一驱动是通过I2c通信的，由于找不到设备或者从设备地址错误，而驱动里面一直进行I2C方式读写，导致死循环。

解决：启动的时候，如果硬件上不支持，就不要加载该驱动即可。例如注释掉tlv\_320aic31.ko和sil9024.ko驱动加载

开启用户图形界面。

sudo systemctl set-default graphical.target  
sudo reboot



