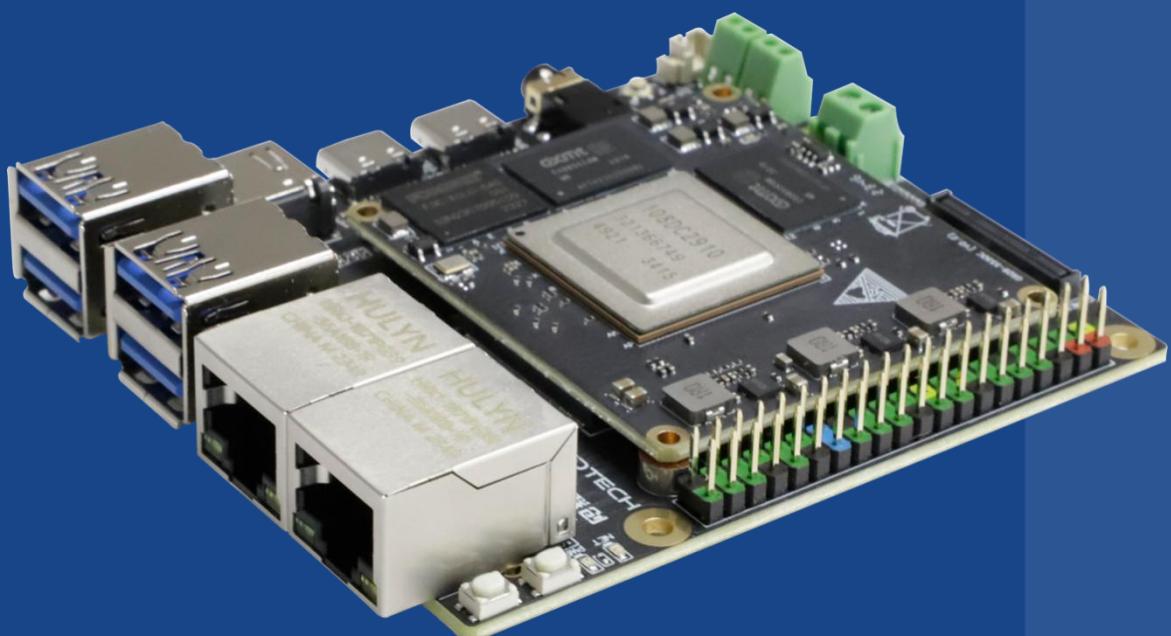


# 快速体验手册



海鸥派 Euler Pi

[www.ebaina.com](http://www.ebaina.com)

让AI触手可及，让连接无处不在



易百纳技术社区

## 免责声明

本文档提供有关南京启诺信息技术有限公司产品的信息，未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。本文档所陈述的产品文本及相关软件版权均属南京启诺信息技术有限公司所有，其产权受国家法律绝对保护，未经本公司授权，其它公司、单位、代理商及个人不得非法使用和拷贝，否则将受到国家法律的严厉制裁。南京启诺信息技术有限公司保留在任何时候修订本用户手册且不需通知的权利。

在订购产品之前，请您与南京启诺信息技术有限公司联系，以获取最新的规格说明。

南京启诺信息技术有限公司保留所有权利。

## 版本历史

版本	日期	原因
V1.0	2024/3/20	创建文档

# 目录

1. VMWARE 下载及安装 .....	5
1.1 Vmware 下载 .....	5
1.2 Vmware 安装 .....	6
2. UBUNTU 镜像下载及安装 .....	9
2.1 Ubuntu 镜像下载 .....	9
2.2 Ubuntu 镜像安装 .....	10
3. UBUNTU 基础配置 .....	16
3.1 安装 vmware tools .....	16
3.2 更换软件下载源 .....	17
4. LINUX&UBUNTU 系统 .....	19
4.1 首次安装 SDK .....	19
4.2 SDK 编译 .....	20
5. OPENEULER 系统 .....	22
5.1 oebuild 环境搭建 .....	22
5.2 创建工作目录并构建 .....	22
5.3 u-boot 的构建 .....	23
6. 固件烧写 .....	24
6.1 接线示意图 .....	24
6.2 烧写工具以及固件 .....	24
6.3 PC 与板端配置 .....	24
6.4 分区配置 .....	26
6.5 烧写 .....	26
7. 功能验证 .....	27
7.1 登录账户和密码 .....	27
7.2 ADC .....	27
7.3 AUDIO .....	28
7.4 CAN .....	28
7.5 sample_hdmi .....	28

7.6 sample_vio.....	29
7.7 sample_uvc.....	31
7.8 sample_mipi_vdec.....	35
7.9 I2C.....	37
7.10 PWM.....	38
7.11 RS485 .....	39
7.12 RTC.....	39
7.13 USB.....	40
7.14 PCIE.....	41
7.15 TF .....	42

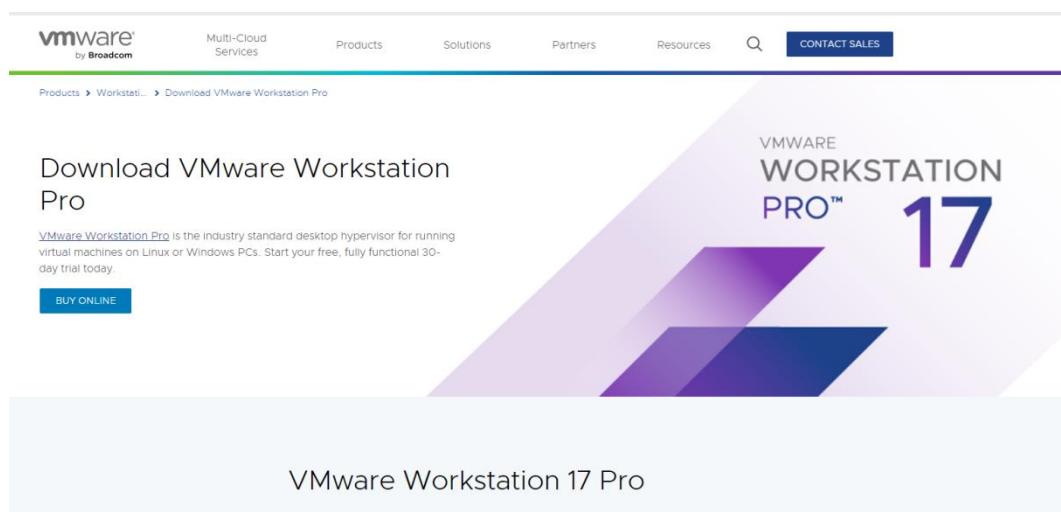
# 1. Vmware 下载及安装

## 1.1 Vmware 下载

嵌入式 Linux 开发大都需要 Linux 的开发环境，VMware 是一款功能强大的虚拟机软件，能够使用该软件在 windows pc 上创建虚拟机，并在虚拟机运行 linux 操作系统。

目前流行的虚拟机软件有 VMware (Vmware Workstation)、Virtual Box 和 Virtual PC 等，本章节将向用户介绍如何通过 VMware 软件创建虚拟机并安装 Ubuntu 操作系统。

VMware 下载链接：[Download VMware Workstation Pro](#)



上述链接地址只能下载当前最新版本的安装包文件，如果用户需要下载旧版本的安装包文件，可以通过链接地址：[https://customerconnect.vmware.com/cn/downloads/#all\\_products](https://customerconnect.vmware.com/cn/downloads/#all_products) 进行下载。

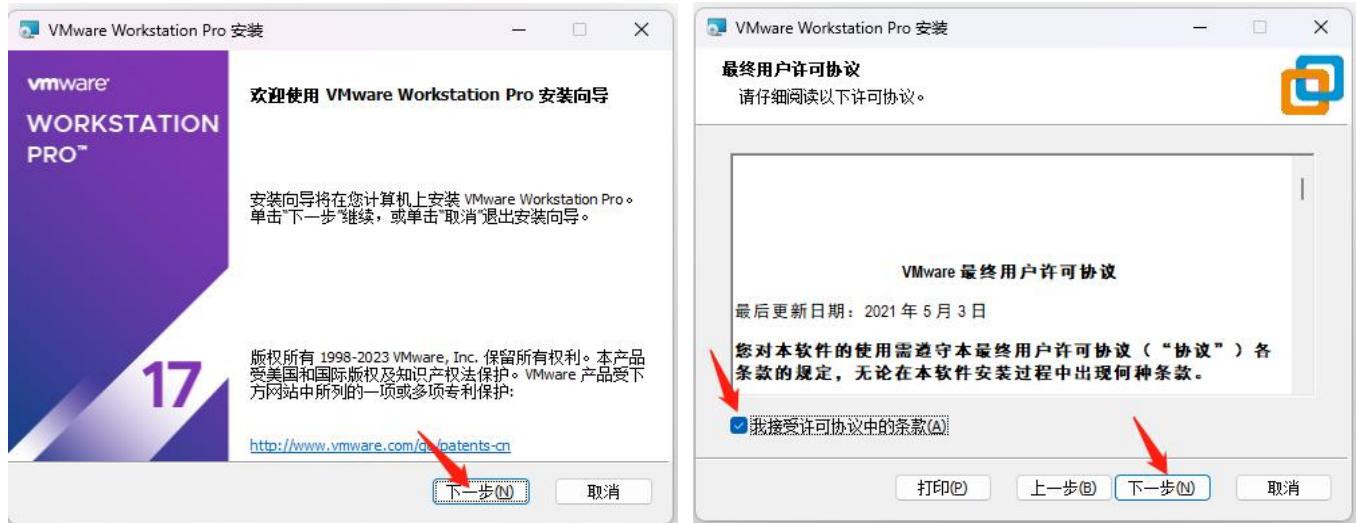
Desktop & End-User Computing	
产品	
VMware Horizon	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">下载试用版</a>   <a href="#">接受培训</a>
VMware Horizon (with View)	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">接受培训</a>
VMware Horizon Apps	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>
VMware Horizon DaaS	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">接受培训</a>
VMware Horizon Clients	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">接受培训</a>
VMware Dynamic Environment Manager	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">下载试用版</a>   <a href="#">接受培训</a>
VMware Workspace	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">接受培训</a>
VMware ThinApp	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>   <a href="#">下载试用版</a>   <a href="#">接受培训</a>
VMware Workstation Pro	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>
VMware Workstation Player	<a href="#">下载产品</a>
VMware Fusion	<a href="#">下载产品</a>   <a href="#">驱动程序和工具</a>

The screenshot shows the VMware Workstation Pro download page. A red arrow points to the '选择版本' dropdown menu, which is set to '17.0'. Below it, there's a link to '获取您的许可证密钥' (Get your license key) and a '了解更多信息' (Learn more) link. On the right, there's a sidebar with links like '产品资源', '查看我的下载历史记录', '产品信息', '文档', '社区', '下载免费试用版: Windows | Linux', and 'Workstation Pro 升级'. At the bottom, there's a table titled '产品下载' (Product Downloads) listing 'VMware Workstation Pro 17.5.0 for Windows' and 'VMware Workstation Pro 17.5.0 for Linux'.

下载过程需要用户登录 VMware 账号，如果没有 VMware 账号、可以通过 VMware 官网 <https://www.vmware.com/cn.html> 进行注册。对 VMware 虚拟机软件的版本没有什么要求，最新版本也行、旧的版本（譬如 15.X.X 或 14.X.X）也可以，本文档将以 17.x.x 版本为例。

## 1.2 Vmware 安装

Windows 下安装 VMware 软件非常简单，直接双击运行安装包文件，按照下图所示操作步骤按照 VMware 虚拟机软件：





点击“完成”按钮，至此，VMware 虚拟机软件安装完成。

安装完成后，会在 Windows 系统桌面生成 VMware 虚拟机软件快捷方式图标



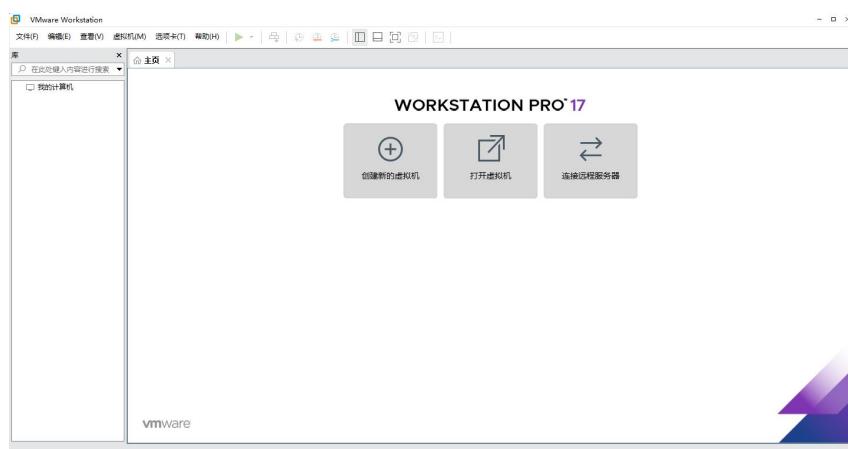
双击图标打开 VMware 虚拟机软件，首次打开软件会提示用户输入许可证密钥。

VMware 虚拟机软件是付费软件，需要用户购买才能使用；如果用户购买了 VMware 软件使用许可，那么将会得到一串许可证密钥，将许可证密钥填写至“我有 VMware Workstation 17 的许可证密钥(H)”所对应的输入框中进行激活，激活成功后便可以正常使用 VMware 软件了。

如果用户没有购买软件使用许可，那么可以选择“我希望使用 VMware Workstation 17 30 天(W)”选项，这样便可获得 VMware 软件的 30 天试用期，点击“继续”按钮：



点击“完成”按钮，接着进入到 VMware 虚拟机软件主界面



至此 vmware 安装已完成，用户可以打开现有的虚拟机或创建新的虚拟机。

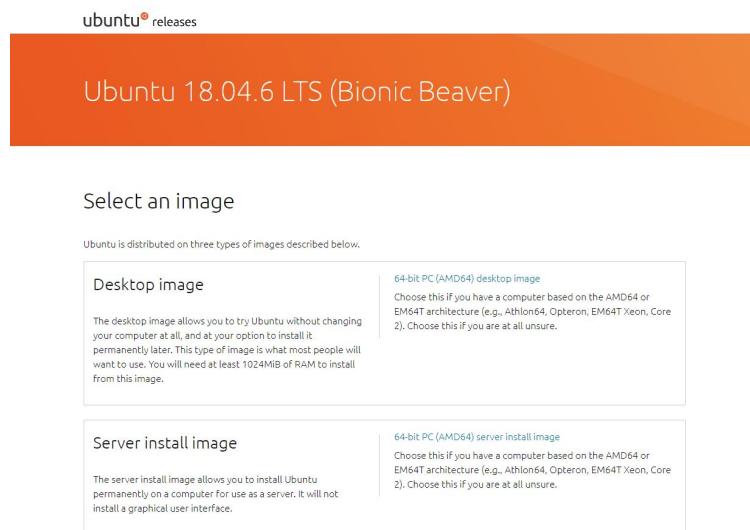
## 2. Ubuntu 镜像下载及安装

### 2.1 Ubuntu 镜像下载

本小节向用户介绍如何在 VMware 虚拟机中安装 Ubuntu 操作系统。Linux 与 ubuntu 系统开发人员使用 Ubuntu 18.04 版本（22.04 版本暂未验证）。opeueuler 系统开发推荐用户使用 Ubuntu22.04（对于 openeuler 系统若版本过低可能会导致部分工具无法正常安装。本文以 ubuntu 18.04 为例介绍系统安装。）

下载链接: [Ubuntu 22.04](#)

下载链接: [Ubuntu 18.04](#)



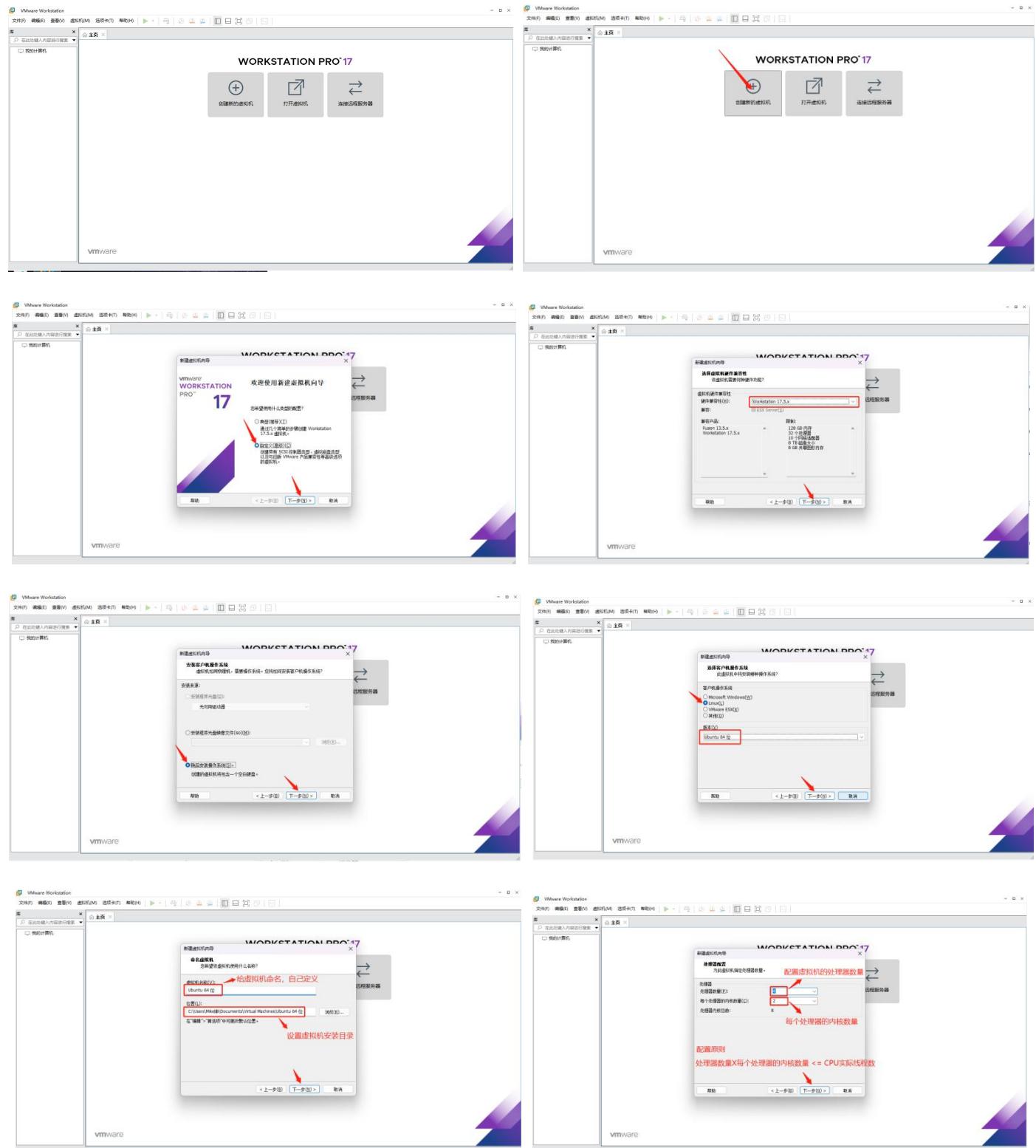
下载选择“ubuntu-18.04.4-desktop-amd64.iso”，如下图所示，选择红色框的镜像

Name	Last modified	Size	Description
Parent Directory		-	
SHA256SUMS	2021-09-16 21:58	202	
SHA256SUMS.gpg	2021-09-16 21:58	833	
ubuntu-18.04.6-desktop-amd64.iso	2021-09-15 20:42	2.3G	Desktop image for 64-bit PC (AMD64) computers (standard download)
ubuntu-18.04.6-desktop-amd64.iso.torrent	2021-09-16 21:46	188K	Desktop image for 64-bit PC (AMD64) computers (BitTorrent download)
ubuntu-18.04.6-desktop-amd64.iso.zsync	2021-09-16 21:46	4.7M	Desktop image for 64-bit PC (AMD64) computers (zsync metafile)
ubuntu-18.04.6-desktop-amd64.list	2021-09-15 20:42	7.8K	Desktop image for 64-bit PC (AMD64) computers (file listing)
ubuntu-18.04.6-desktop-amd64.manifest	2021-09-15 20:36	59K	Desktop image for 64-bit PC (AMD64) computers (contents of live filesystem)
ubuntu-18.04.6-live-server-amd64.iso	2021-09-15 20:42	969M	Server install image for 64-bit PC (AMD64) computers (standard download)

## 2.2 Ubuntu 镜像安装

### 2.2.1 创建虚拟机

打开 VMWareWorkstation17 虚拟机软件。

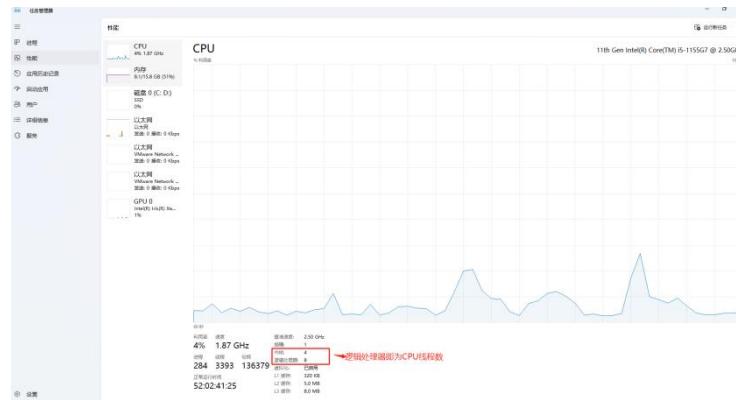


为虚拟机配置处理器数量以及每个处理器的内核数量：

处理器内核总数 = 处理器数量 X 每个处理器的内核数量；

用户需根据自己的电脑配置(CPU 配置)情况来为虚拟机分配处理器数量以及每个处理器的内核数量，只需满足如下要求：

虚拟机处理器内核总数小于或等于 (<=) CPU 实际线程数；

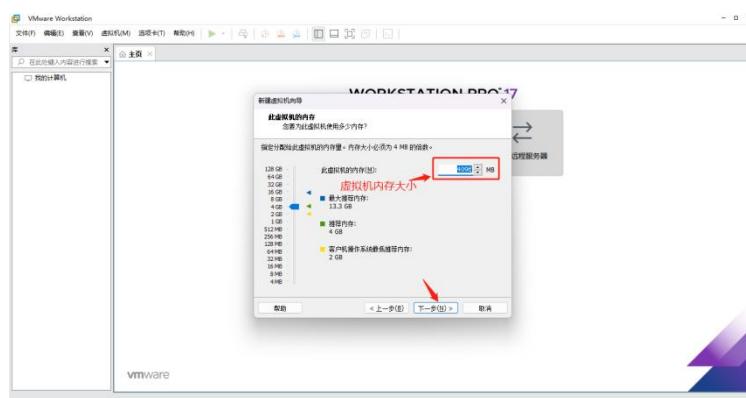


图中逻辑处理器数量即为 CPU 实际线程数，所以我的电脑 CPU 实际线程数为 8，按照配置要求，虚拟机的处理器内核总数需小于或等于 (<=) CPU 实际线程数 8，那么我们可以有多种不同的配置方式，譬如：

处理器数量=1、每个处理器的内核数量=8；

处理器数量=4、每个处理器的内核数量=2；

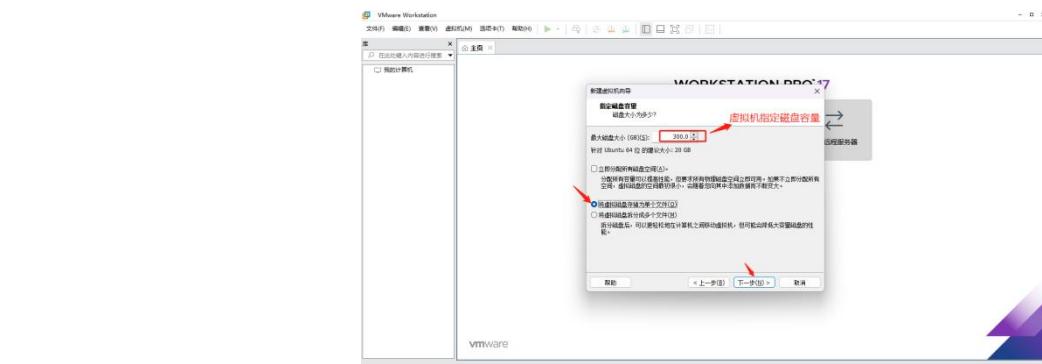
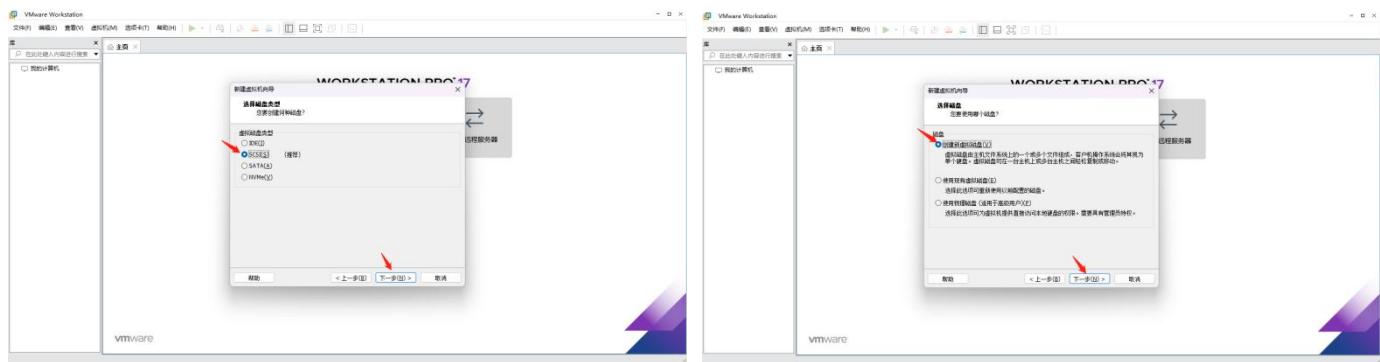
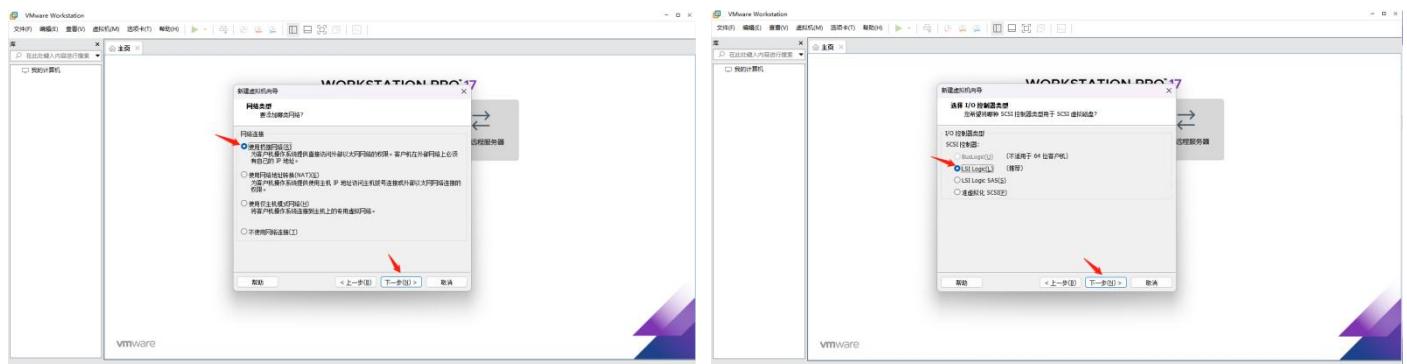
等等……



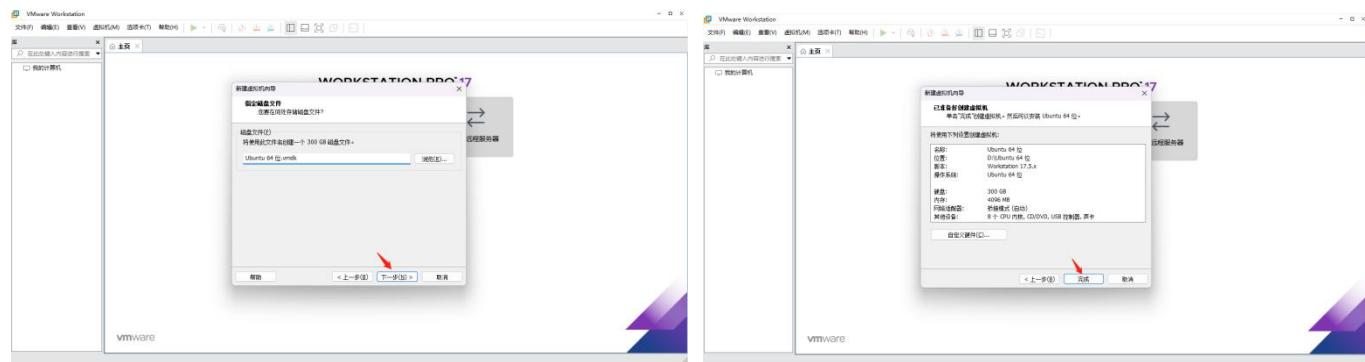
可以通过上下拖动左边的蓝色滑块来调整虚拟机的内存大小，也可以直接在右边的输入框中填写数字来指定虚拟机的内存大小，单位是 MB。同样，虚拟机的内存大小也需要根据用户的电脑配置（内存配置）情况来设置。

# 海鸥派 Euler Pi 快速体验手册

www.ebaina.com



虚拟机磁盘容量也需要根据用户的电脑配置（磁盘配置）情况来设置，建议至少 100GB（若容量不足可后续扩容）。

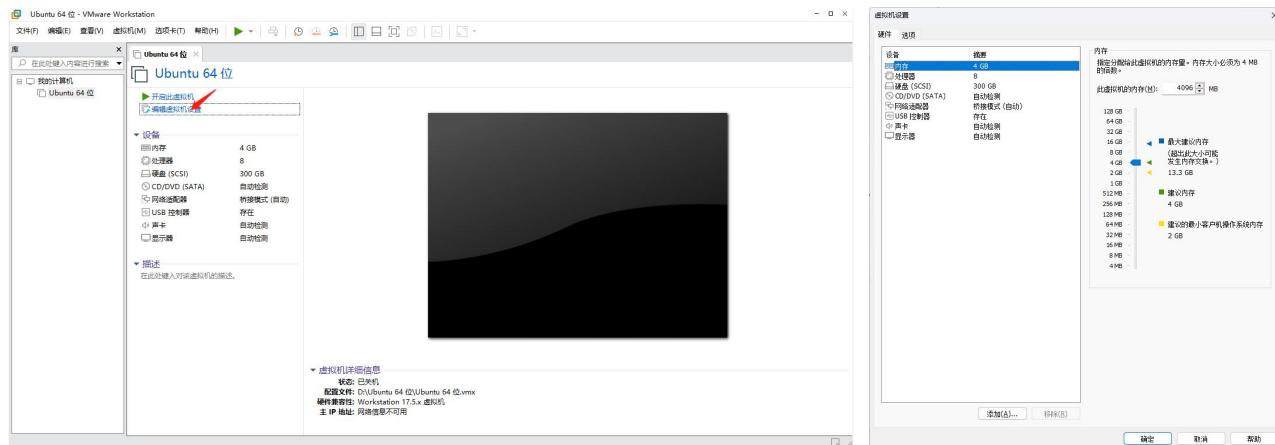




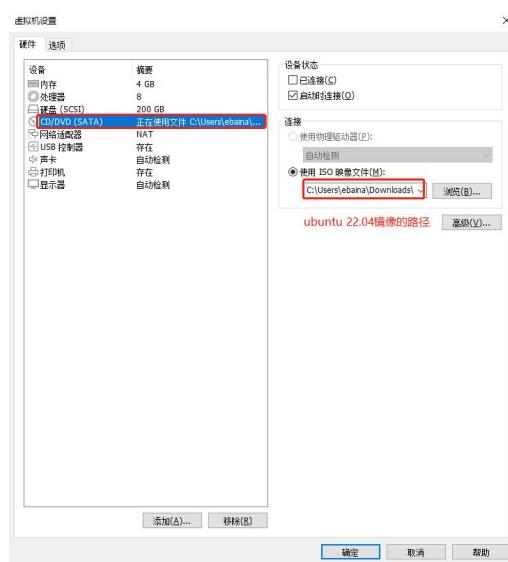
至此，虚拟机便创建完成了。

## 2.2.2 镜像安装

在创建好的虚拟机上点击“编辑虚拟机设置”打开虚拟机设置界面：



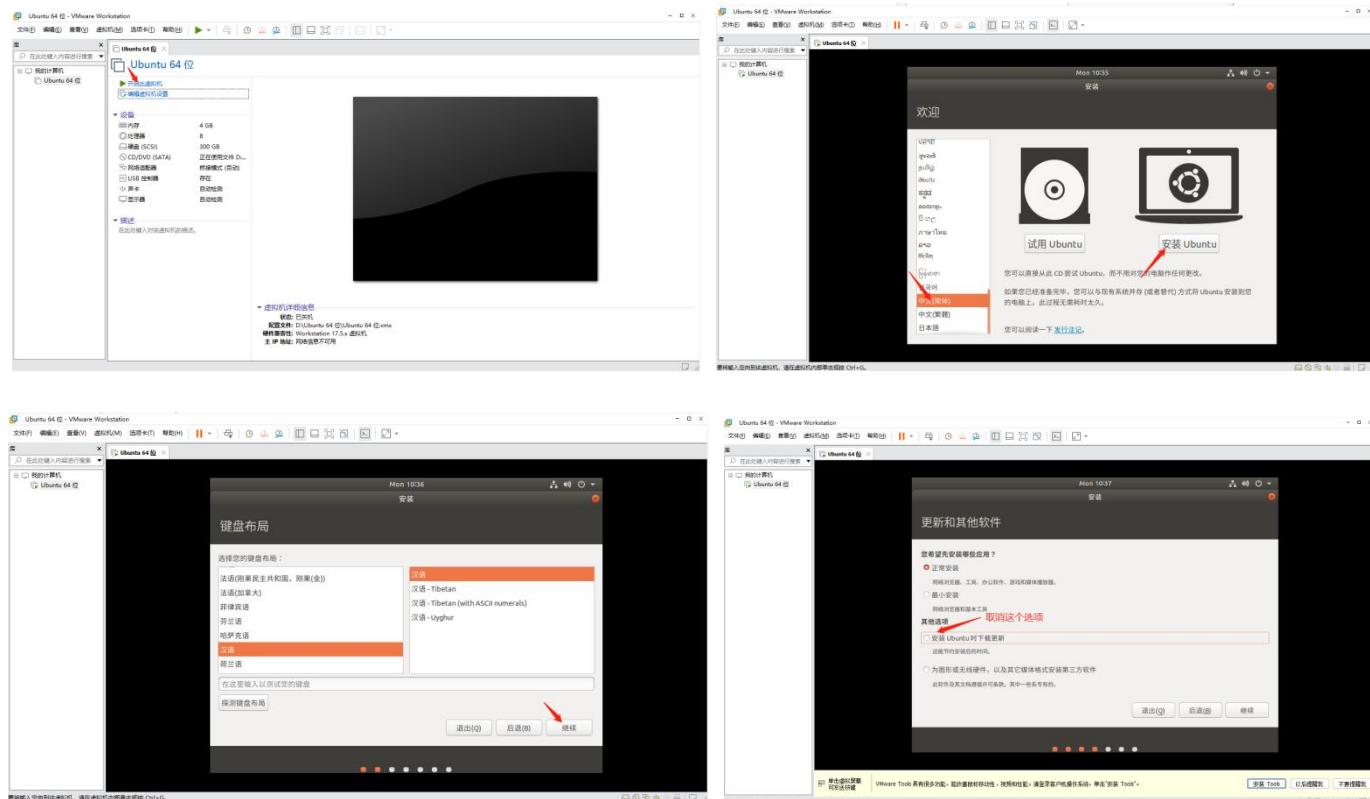
首先指定 Ubuntu 22.04 系统镜像文件 ubuntu-22.04.4-desktop-amd64.iso 所在路径：



点击“浏览(B)”按钮，选择 Ubuntu 22.04 系统镜像文件 ubuntu-22.04.6-desktop-amd64.iso。接下来对虚拟机 USB 控制器进行设置：

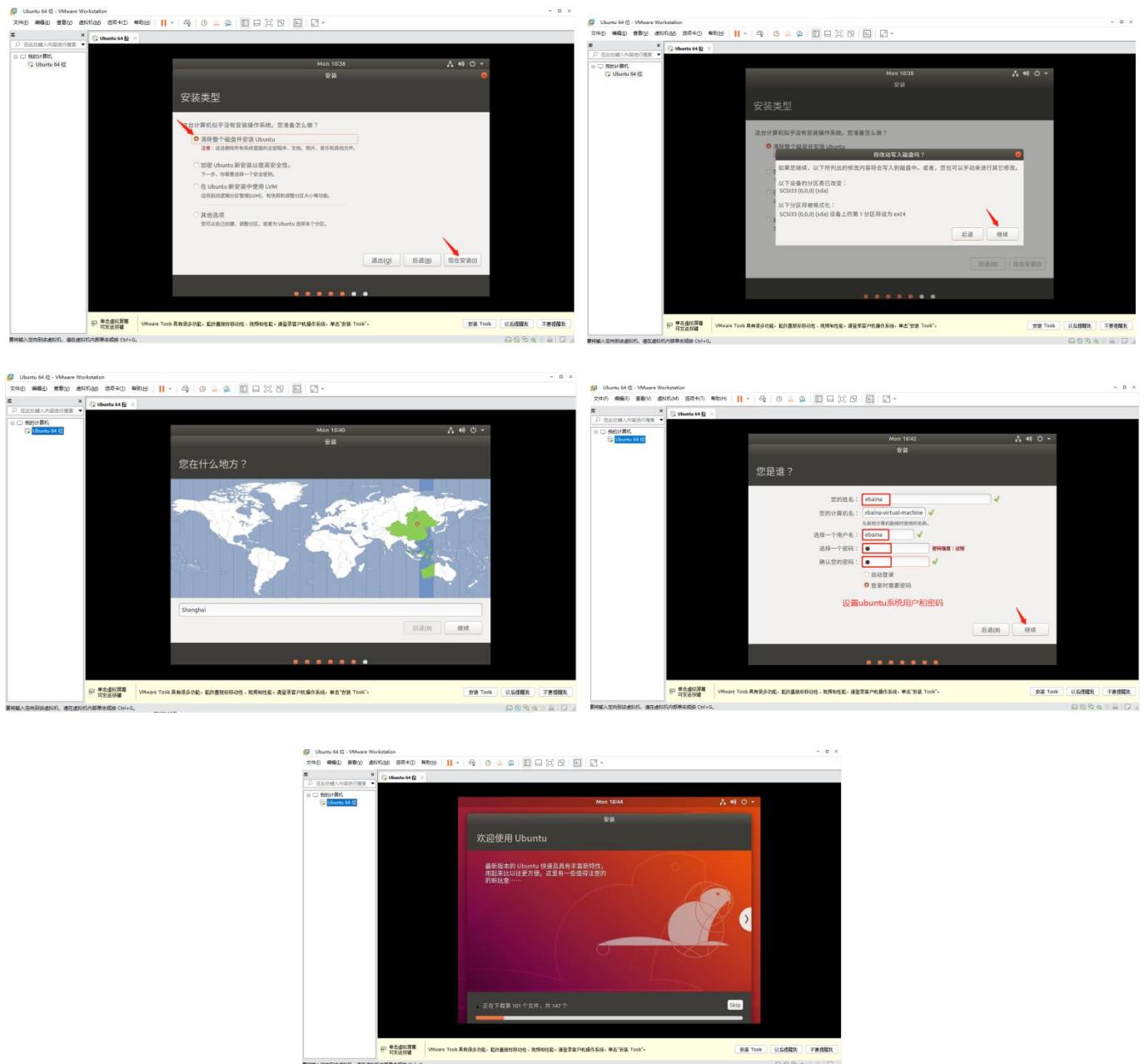


USB 控制器的 USB 兼容性默认为 USB 2.0，当我们使用 USB 3.0 设备时，Ubuntu 系统可能识别不出来，因此我们需要调整 USB 兼容性为 USB 3.1，最后点击“确定”按钮退出窗口。



# 海鸥派 Euler Pi 快速体验手册

www.ebaina.com

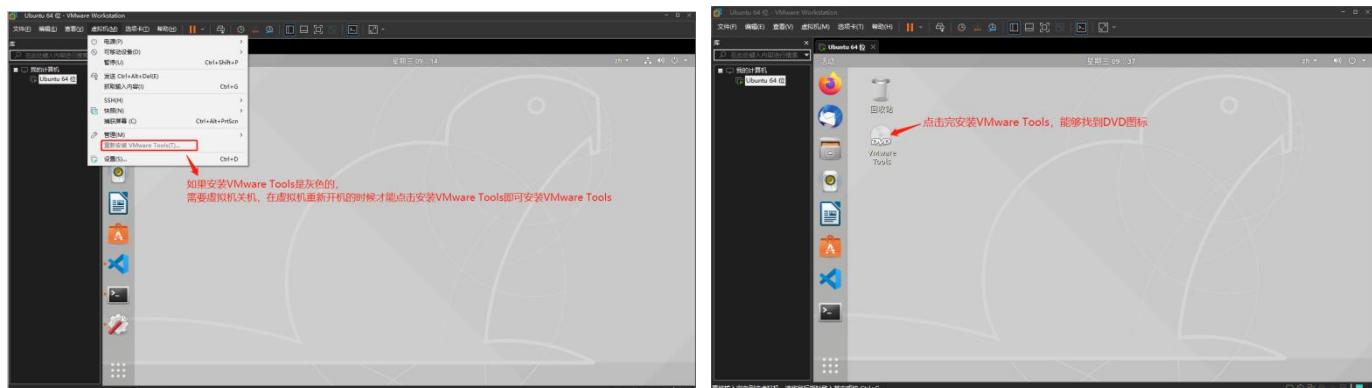


等待系统安装完成，安装完成后将会提示用户重启系统，等待重启后 ubuntu18.04 操作系统安装完成。

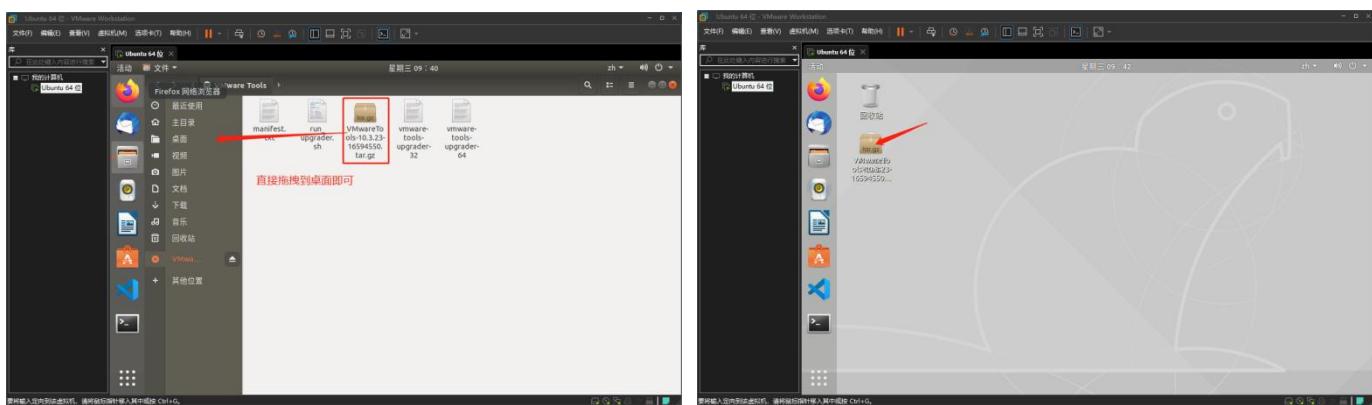
## 3. Ubuntu 基础配置

### 3.1 安装 vmware tools

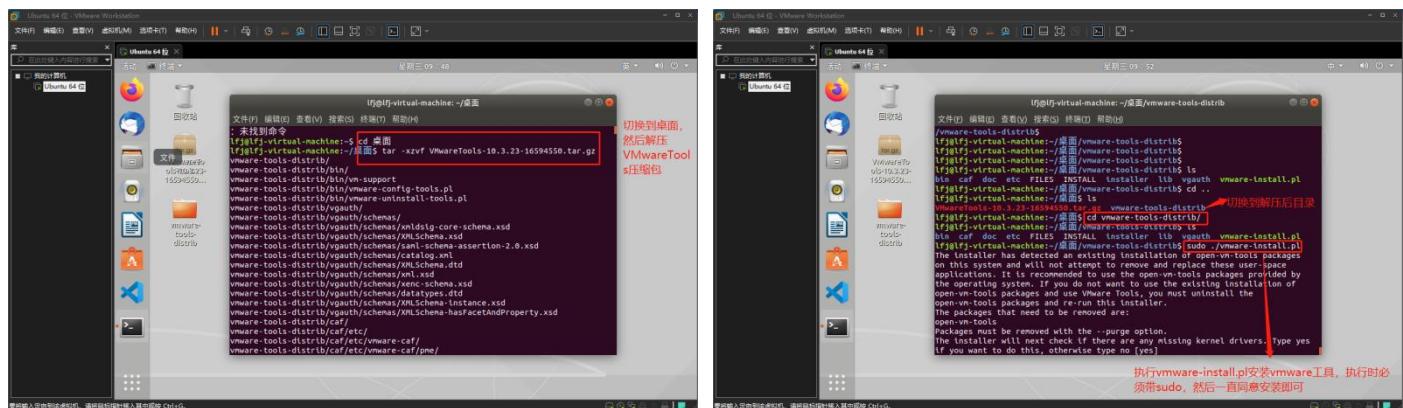
在创建好的虚拟机上点击“虚拟机(M)”打开虚拟机设置界面：

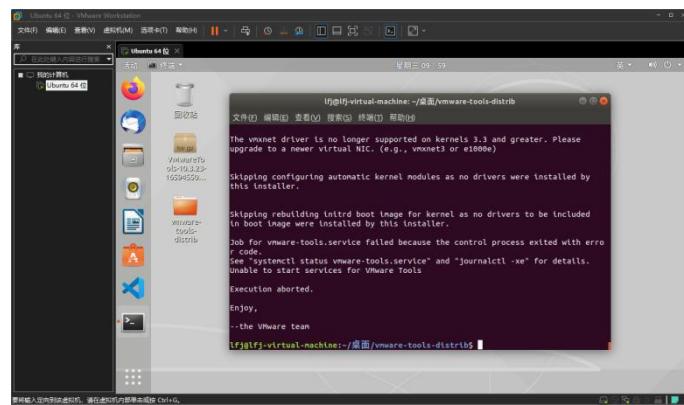


点击 DVD 图标，打开后找到 VMware Tools 压缩文件，并将其复制到桌面



使用快捷键 **ctrl+alt+t** 打开终端





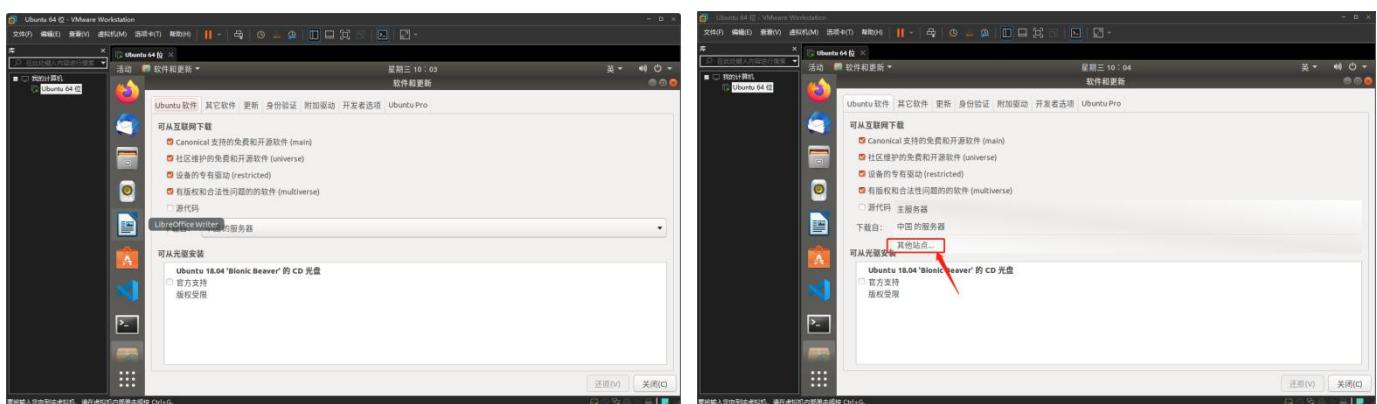
## 3.2 更换软件下载源

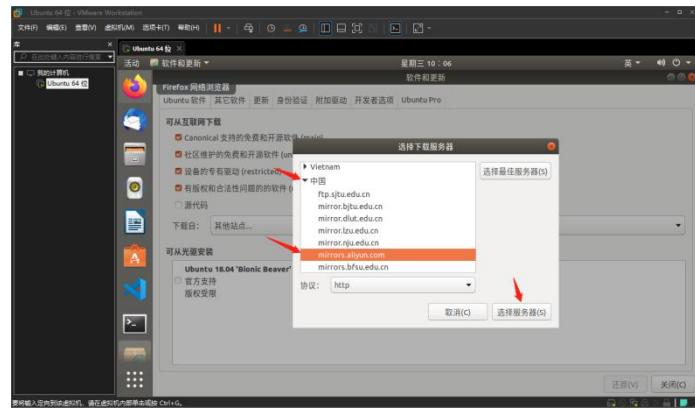
ubuntu 系统默认的软件下载源因为服务器的缘故，在国内的下载速度是比较慢的，因此我们需要将 ubuntu 系统的软件下载源改为国内软件源，例如清华源等等，更改后软件下载速度会提高很多！可根据下图操作进去更改软件源。

1. 首先点击左下角的九宫格，弹出工具界面，点击软件与更新。



2. 鼠标左击红框中的主服务器，更换为阿里源。

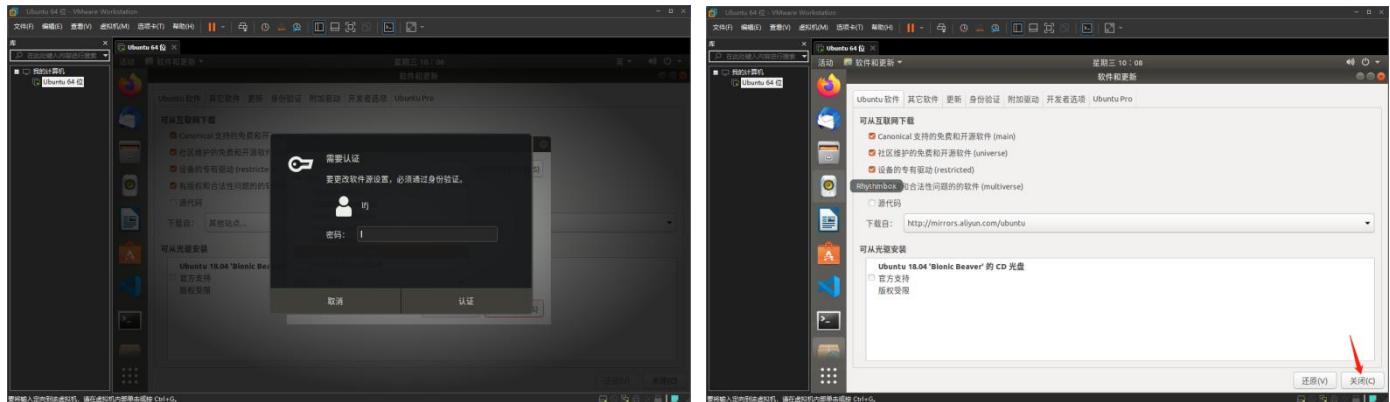




从列表里面可以得知，其实有很多国内的下载源可以供我们使用，这里我们以阿里源作为参考而已，想选择其他的国内源也行。

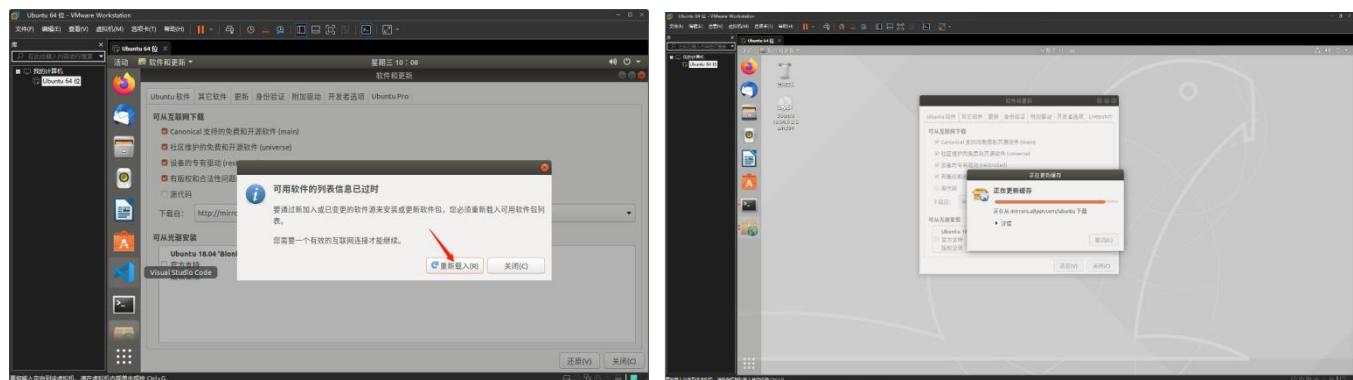
## 3. 软件源更改为阿里源

更改软件源后需要进行认证，密码为 root 密码。



## 4. 更新软件

当我们成功更改软件源后，会弹出提示重新载入软件列表，等待载入完成即可。



## 4. Linux&Ubuntu 系统

### 4.1 首次安装 SDK

#### 4.1.1 下载安装 SDK

网盘目录 04 SDK 资料包/03 开发板软件资料/03 易百纳适配 SDK 中下载 hieuler\_sdk\_v1.0.0.tgz



将下载好的压缩包放进虚拟机，执行 `tar xzf hieuler_sdk_v1.0.0.tgz` 解压即可。

#### 4.1.2 安装交叉编译器

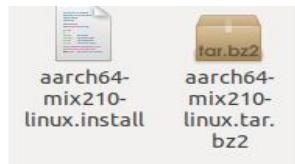
##### 1. aarch64-mix210-linux-

网盘目录 04 原厂资料目录下



将该交叉编译器压缩包拷贝到虚拟机中，家目录下即可

执行 `tar xzf aarch64-mix210-linux.tgz` 解压，一共有两个文件



执行 `sudo ./aarch64-mix210-linux.install`，等待安装完成即可

执行 `aarch64-mix210-linux-gcc -v` 查看版本

```
(base) lewiss@ubuntu:~$ aarch64-mix210-linux-gcc -v
Using built-in specs.
COLLECT_GCC=aarch64-mix210-linux-gcc
COLLECT_LTO_WRAPPER=/opt/linux/x86-arm/aarch64-mix210-linux/host_bin/../libexec/gcc/aarch64-linux-gnu/7.3.0/lto-wrapper
Target: aarch64-linux-gnu
Configured with: /home/pub/software/toolchains/build/aarch64-mix210-linux/../../open_source/aarch64-mix210-linux_src/gcc-7.3.0/configure --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=aarch64-linux-gnu --with-arch=armv8-a --prefix=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/aarch64_mlx210 --enable-threads --disable-libmudflap --enable-libssp --enable-gold=yes --disable-libstdcxx-pch --disable-multilib --enable-multiarch --with-gnu-as --with-gnu-ld --enable-libgomp --enable-gnu-indirect-function --enable-checking=yes --enable-lto --enable-c99 --enable-long-long --disable-nls --enable-fpu=cortex-a53-835769 --enable-fpu=cortex-a53-843419 --enable-shared --with-pkgversion=20220321 --enable-languages=c,c++ --with-headers=/home/pub/software/toolchains/build/aarch64_mlx210/linux/aarch64_mlx210_build_dir/aarch64_mlx210/target/usr/include --with-sysroot=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/aarch64_mlx210/target --with-build-sysroot=/home/pub/software/toolchains/build/aarch64_mlx210-linux/aarch64_mlx210_build_dir/aarch64_mlx210/target --with-gmp=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/obj/host-lts/usr --with-mpfr=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/obj/host-lts/usr --with-mpc=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/obj/host-lts/usr --with-build-time-tools=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/aarch64_mlx210/aarch64-linux-gnu/bin --libdir=/home/pub/software/toolchains/build/aarch64-mix210-linux/aarch64_mlx210_build_dir/aarch64_mlx210/lib --disable-bootstrap --with-system-zlib
Thread model: posix
gcc version 7.3.0 (20220321)
```

## 4.2 SDK 编译

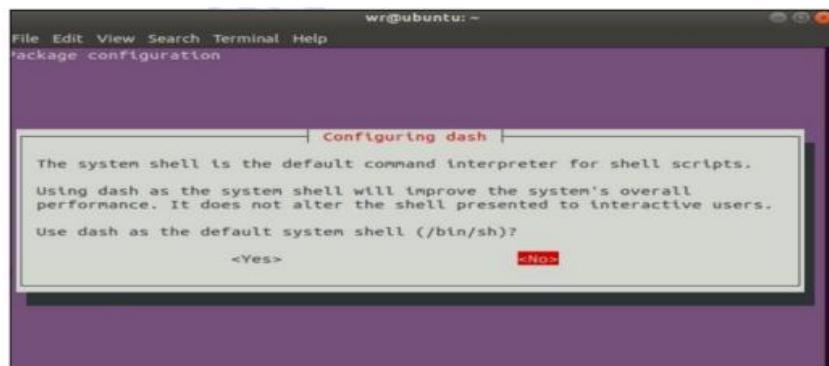
### 4.2.1 软件包安装

操作系统安装好后，且用户已自行配置好网络环境，则可继续如下步骤完成相关软件包的安装：

#### 步骤 1 配置默认使用 bash

执行 `sudo dpkg-reconfigure dash` 选择 no

如图所示：



#### 步骤 2 安装软件包

执行：`sudo apt-get install make libc6-i386 lib32z1 lib32stdc++6 zlib1g-dev libncurses5-dev ncurses-term libncursesw5-dev g++ u-boot-tools texinfo gawk libssl-dev openssl bc p7zip-full gperf bison flex diffutils git unzip libffi-dev libtool libfreetype6`

#### 步骤 3 创建/etc/ld.so.preload 文件，并执行 `echo "" > /etc/ld.so.preload`，以解决 64bit linux server 上某些第三方库编译失败的问题。



#### 步骤 4 mtd-utils 依赖以下几个库，以 ubuntu 为例，安装方式请参考下面命令：

`sudo apt-get install zlib1g-dev liblzo2-dev uuid-dev pkg-config automake`

由于 mtd-utils 通过 `pkg_config` 工具检查各个库是否正常安装，因此请参考如下方式设置 `pkg_config` 工具搜索路径：

`export PKG_CONFIG_PATH="$PKG_CONFIG_PATH:/usr/lib/x86_64-linux-gnu/pkgconfig"`

#### 步骤 5 e2fsprogs 依赖 texlive 库，安装方式请参考如下命令：

`sudo apt-get install texlive`

步骤 6 optee 模块依赖 python3.x.x、pip3、wheel、pycryptodome、pyelftools 库

python3.x.x 安装推荐 python3.7.6 版本

[www.python.org](http://www.python.org)

\*下载安装\*并执行如下指令：

```
tar -zxvf Python-3.7.6.tgz
```

```
cd Python-3.7.6
```

```
./configure
```

```
make
```

```
make install
```

可能需要配置下载源：创建并编辑:  
`vim ~/.pip/pip.conf`

```
sudo pip3 install wheel==0.36.2
```

```
sudo pip3 install pycryptodome==3.9.8
```

```
sudo pip3 install pyelftools==0.27
```

```
sudo pip3 install kconfiglib
```

创建 python 软链接

```
ln -s /usr/local/bin/python3 /usr/local/bin/python
```

备注： pycryptodome 依赖于 python3.x.x 环境请先升级至 python3.x.x 版本再安装 pycryptodome 库。

## 4.2.2 编译 SDK

进入 hieuler\_sdk\_v1.0.0 目录

执行 make 进行总体编译

总体编译生成的镜像文件为出厂固件，都存放在 hieuler\_sdk\_v1.0.0/output 下

详细操作指南参考：Hieuler SDK 操作指南(Linux).pdf

## 5. openeuler 系统

### 5.1 oebuild 环境搭建

ubuntu 中环境搭建，此处参考 [openEuler Embedded 在线文档](#)

【注】：编译过程须保持网络畅通，第一次编译需要较长的时间（大概 2 小时）

```
# 建议使用 ubuntu 22.04
# 安装必要的软件包
# 注意此处与官网不同，官网安装的是 python3，使用 pip 命令安装 oebuild
# 此处使用 ubuntu 22.04 默认的 python3.10，使用 pip3 命令安装 oebuild
sudo apt-get install python3-pip docker docker.io
pip3 install oebuild

# 配置 docker 环境
sudo usermod -a -G docker $(whoami)
sudo systemctl daemon-reload && sudo systemctl restart docker
sudo chmod o+rwx /var/run/docker.sock
```

### 5.2 创建工作目录并构建

```
# 初始化一个名为 build_3403 的工作目录，构建的版本为 openEuler-23.09
oebuild init build_3403 -b master
cd build_3403
# 更新 docker 构建镜像
oebuild update

oebuild generate -p hieulerpi -f openeuler-ros
```

```
# 随后按提示进入构建目录
# 进入构建环境
oebuild bitbake
# 开始构建镜像
bitbake openeuler-image
```

构建完成后会在 output 目录输出固件

```
.
├── Image -> Image-5.10.0
├── Image-5.10.0
└── kernel-demo # 海思机器人 demo 板内核
    └── kernel-pi # EulerPI 使用内核
```

```
└── openeuler-image-ros-sd3403-20240106121030.rootfs.ext4 # 根文件系统
└── vmlinuz-5.10.0
```

在烧录欧拉派时将使用 kernel-pi 和 openeuler-image-ros-sd3403-20240106121030.rootfs.ext4 作为部署件

```
# 在编译过程中若出现问题可执行一下命令清除缓冲，尝试重新编译
# 仅清除缓冲和标识，并不会完全重新编译
bitbake openeuler-image-ros -c cleanall
```

## 5.3 u-boot 的构建

目前，openeuler 系统构建不包括 u-boot 的构建，若需要构建 u-boot 可下载 u-boot 源码单独编译或者直接获取编译完成的 u-boot 部署件

u-boot 源码仓库：<https://gitee.com/HiEuler/u-boot>

编译完成的 u-boot 部署件：<https://gitee.com/HiEuler/u-boot/releases>

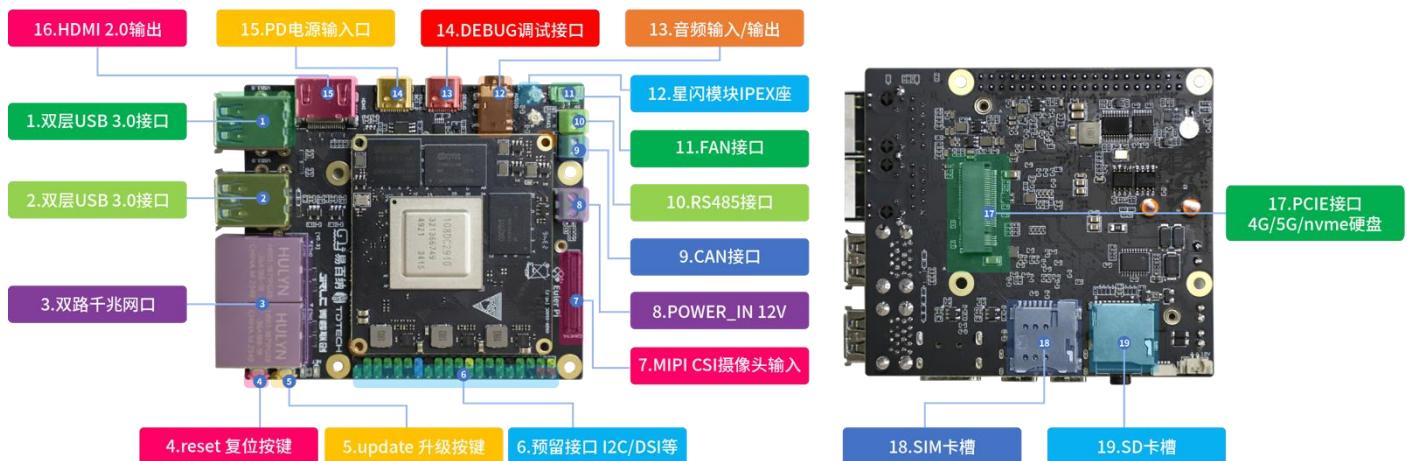
编译 u-boot 需要安装 aarch64-mix210-linux 交叉编译工具链，具体安装方法参考 aarch64-mix210-linux

```
# 编译 u-boot
make
# 查看帮助信息
make help
# 编译 4G 内存版本（默认编译为 8G）
make MEM_SIZE=4G
# 配置 u-boot
make ubootcfg
```

在编译完成后会在当前目录生成 boot\_image\_8G.bin 和 boot\_env\_8G.bin

## 6. 固件烧写

### 6.1 接线示意图



连接 PD 供电口或 12V 供电口, TypeC 串口以及 eth0 网口

### 6.2 烧写工具以及固件

烧写工具在 03 开发板软件资料/02 工具包/01 烧录工具目录下



烧写固件在 03 开发板软件资料/01 出厂固件目录下



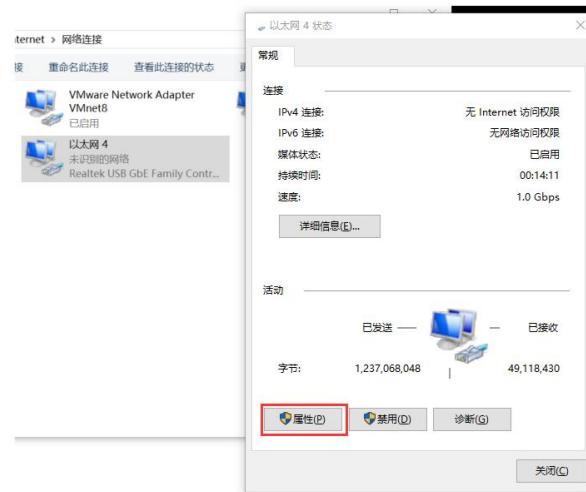
### 6.3 PC 与板端配置

串口选择：插上串口工具后，点击刷新即可自动识别(无法识别可以手动配置)

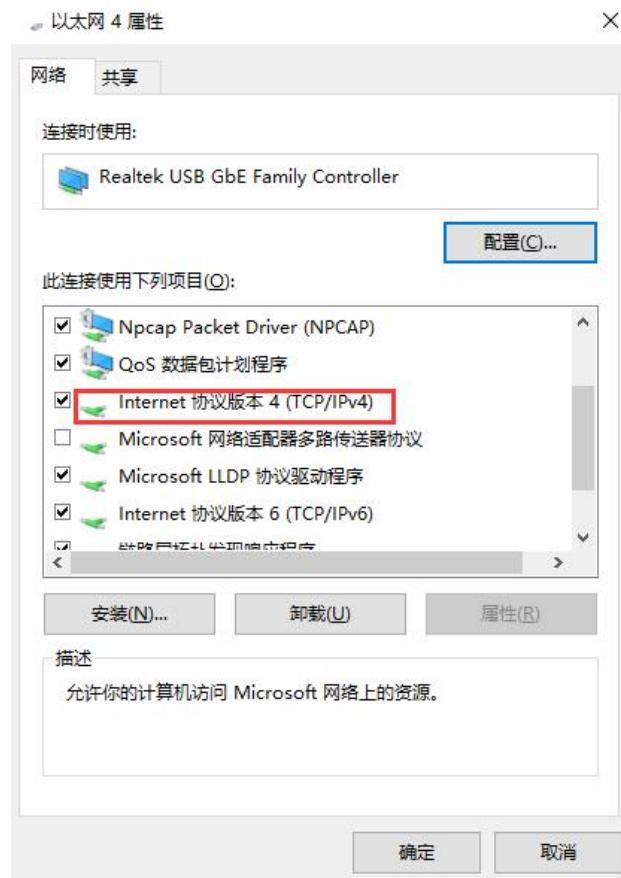
### 6.3.1 网络设置

如果开发板直连 PC，设置 PC 有线网卡的静态 IP

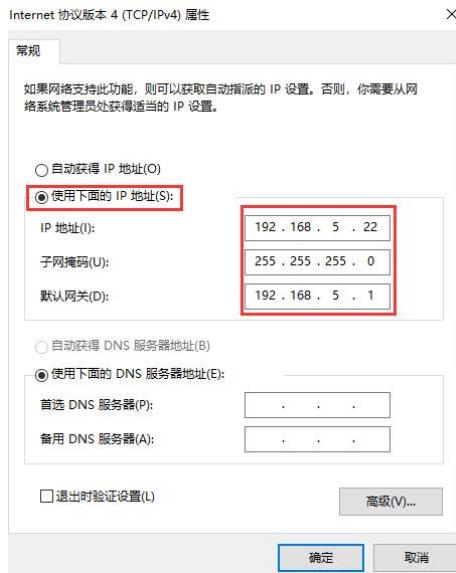
#### 1. 点击属性



#### 2. 双击 IPv4



### 3. 选择使用静态 IP，设置静态 IP



### 4. 设置此 IP 和开发板在同一网段

服务器 IP：选择有线网卡的静态 ip，如果开发板通过交换机与 PC 连接，选择交换机网络所在网段

传输方式：网口下载



## 6.4 分区配置

选择烧写 emmc

点击浏览，选择 partTable.xml



## 6.5 烧写

点击烧写，控制台提示重启单板，按下 RST 重启按键，固件即开始烧写。

注：欧拉系统重新烧写后要重新配置账号、密码

## 7. 功能验证

将出厂固件烧录进开发板，所有的及基础例程都在/root/device\_sample 目录下

例程线上源码仓库：[HiEuler/externed\\_device\\_sample](#)

说明：例程线上源码仓库对每个例程都有详细的 README.md

### 7.1 登录账户和密码

欧拉系统：

账户：root

密码：ebaina@2024

linux：

无密码直接回车 telnet

登录：用户：root 无密码直接回车

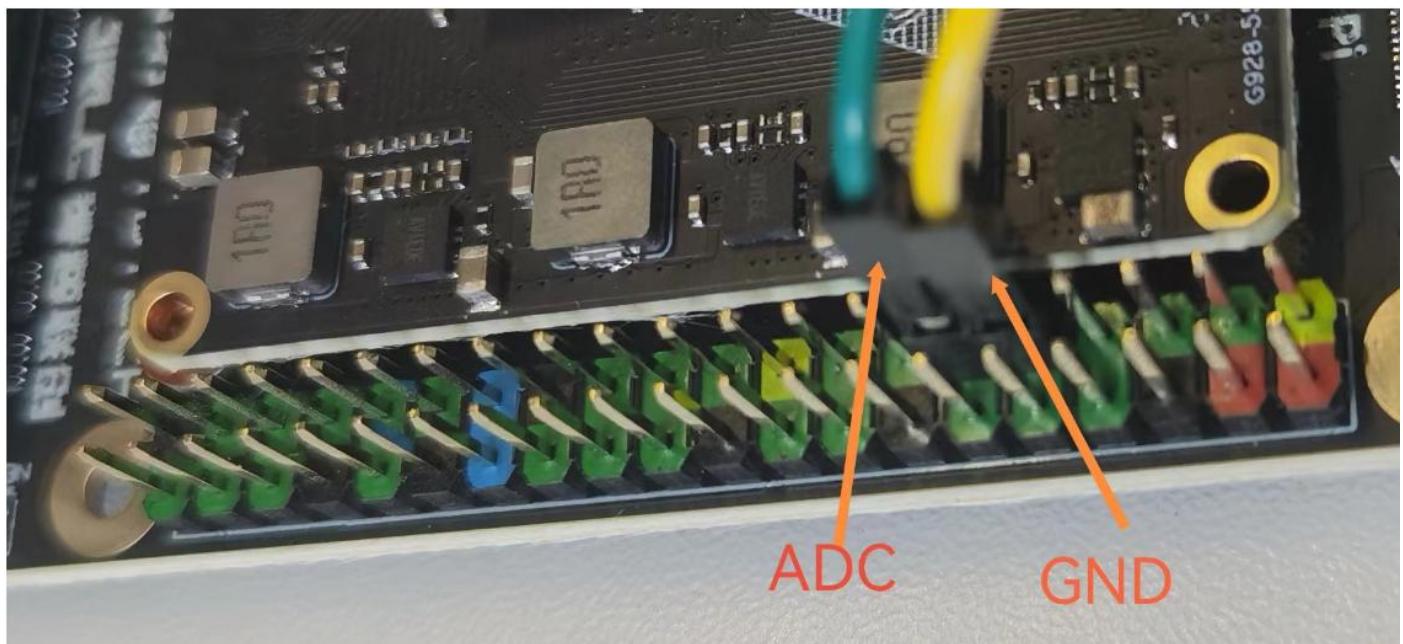
ubuntu：

账号：root

密码：ebaina

### 7.2 ADC

打开欧拉派的终端，并执行 adc 可执行程序，如下图连接被测电源与 GND



```
cd root/device_sample/adc
```

```
./adc #开始接收 ADC 引脚转换过来的数据，并打印在终端上显示
```

默认 ADC 引脚为 1.8V，可通过杜邦线将 ADC 引脚与 GND 连接，连接后打印信息显示为 0V 左右  
使用 Ctrl+C 结束接收

注意：引脚耐压值 1.8V，禁止施加超过 1.8V 的电压

## 7.3 AUDIO

测试前将带麦克风的耳机接入开发板音频输入输出接口

```
export LD_LIBRARY_PATH=/usr/lib ld_library_path
cd /root/device_sample/audio
./audio_sample <index>
index and its function list below
0: start AI to AO loop
1: send audio frame to AENC channel from AI, save them      # 对麦克风讲话，将声音录下,
保存为 audio_chn0.aac 文件
2: read audio stream from file, decode and send AO          # 播放 audio_chn0.aac 文件
3: read audio stream from Mydream44100.aac file, decode and send AO
4: start AI(VQE process), then send to AO
```

## 7.4 CAN

在开始测试前需要将 can 网卡节点开启，在欧拉派的终端上执行如下命令

```
ip link set can0 type can bitrate 500000
ip link set can0 up
```

TOF 数据接收测试

使用 can\_tof 命令进行接收测试，在欧拉派的终端上执行

```
cd /root/device_sample/can
./can_tof
```

应用开启后持续接收 TOF 发送的距离数据并打印到欧拉派终端，使用 ctrl+c 结束接收

## 7.5 sample\_hdmi

测试前将显示器和开发板用 HDMI 线连接好

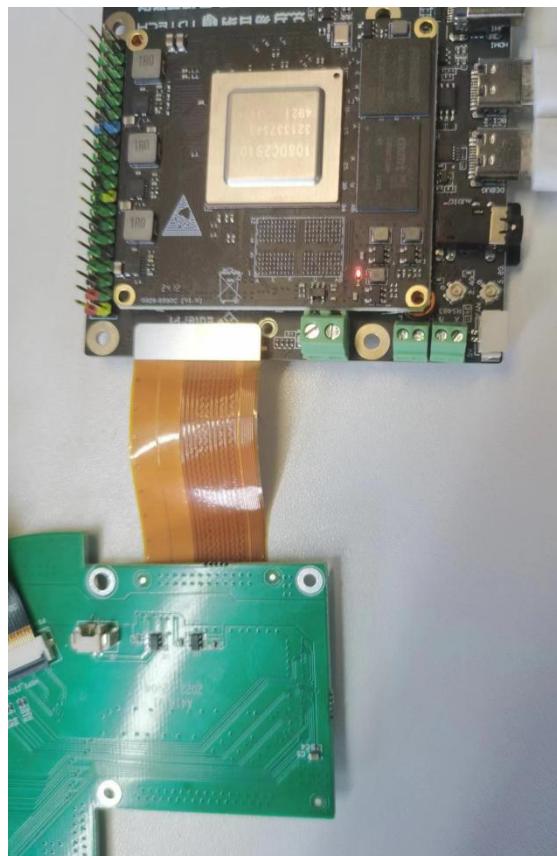
```
export LD_LIBRARY_PATH=/usr/lib:/usr/lib64
cd /root/device_sample/hdmi
./sample_hdmi
hdmi_cmd:
```

help	list all command we provide
q	quit sample test
hdmi_hdmi_force	force to hdmi output
hdmi_dvi_force	force to enter dvi output mode
hdmi_deeppixel	set video deeppixel mode
hdmi_video_timing	set video output timing format
hdmi_color_mode	set video color output(RGB/ycbcr)
hdmi_aspectratio	set video aspectratio
hdmi_a_freq	set audio output frequency
hdmi_authmode	authmode enable or disable

## 7.6 sample\_vio

vio 中有其它许多选项，本次使用选项 7 测试 4 路 IMX347 摄像头

接线如下：



当前例程在板端/root/device\_sample/mipi\_vi 目录下，以下命令需切换到该目录下执行。

```
usage : ./sample_vio <index>
index:
os08a20 24M 1080P60:
(0) one sensor(i2c-5)      :vi one sensor (offline) -> vpss -> venc && vo.
(1) one sensor(i2c-7)      :vi one sensor (offline) -> vpss -> venc && vo.
(2) two sensor             :vi two sensor (offline) -> vpss -> venc && vo.
```

```

os08a20 24M 4K30:
(3) one sensor(i2c-5)      :vi one sensor (offline) -> vpss -> venc && vo.
(4) one sensor(i2c-7)      :vi one sensor (offline) -> vpss -> venc && vo.
(5) two sensor             :vi two sensor (offline) -> vpss -> venc && vo.

imx347 37M 1080P30:
(6) one sensor(i2c-7)      :vi one sensor (offline) -> vpss -> venc && vo.
(7) four sensor            :vi one sensor (offline) -> vpss -> venc && vo.

imx485 37M 1080P60:
(8) one sensor(i2c-5)      :vi one sensor (offline) -> vpss -> venc && vo.
(9) one sensor(i2c-7)      :vi one sensor (offline) -> vpss -> venc && vo.

imx485 37M 4K30:
(10) one sensor(i2c-5)     :vi one sensor (offline) -> vpss -> venc && vo.
(11) one sensor(i2c-7)     :vi one sensor (offline) -> vpss -> venc && vo.

```

## 1. 初始化

sensor 运行需要进行复位和启动时钟等操作。执行下面的脚本完成摄像头的初始化

```
./scripts/init_imx347_4x2lan.sh
```

## 2. 例程运行

```
./sample_vio 7
```

## 3. 执行完成后系统打印如下输出

```

=====> i2c:7 sensor:0
=====> i2c:5 sensor:1
=====> i2c:4 sensor:2
=====> i2c:6 sensor:3
linear mode
==> IMX347 LANE_891MBPS_37MHZ
==> Set imx347 slave ad_bit_mode: 1 (0:10bit 1:l2bit)
==> Set imx347 slave md_bit_mode: 1 (0:rawl0 1:rawl2)
==> Set imx347 slave lane_mode: 1 (1:2lane 3:4lane)
=====
==== IMX347 Slave LINE Init OK! ===
=====

ISP Dev 0 running !
linear mode
==> IMX347 LANE_891MBPS_37MHZ
==> Set imx347 slave ad_bit_mode: 1 (0:10bit 1:l2bit)
==> Set imx347 slave md_bit_mode: 1 (0:rawl0 1:rawl2)
==> Set imx347 slave lane_mode: 1 (1:2lane 3:4lane)
=====
==== IMX347 Slave LINE Init OK! ===
=====

ISP Dev 1 running !
linear mode
==> IMX347 LANE_891MBPS_37MHZ
==> Set imx347 slave ad_bit_mode: 1 (0:10bit 1:l2bit)
==> Set imx347 slave md_bit_mode: 1 (0:rawl0 1:rawl2)
==> Set imx347 slave lane_mode: 1 (1:2lane 3:4lane)
=====
==== IMX347 Slave LINE Init OK! ===
=====

ISP Dev 2 running !
linear mode
==> IMX347 LANE_891MBPS_37MHZ
==> Set imx347 slave ad_bit_mode: 1 (0:10bit 1:l2bit)
==> Set imx347 slave md_bit_mode: 1 (0:rawl0 1:rawl2)
==> Set imx347 slave lane_mode: 1 (1:2lane 3:4lane)
=====
==== IMX347 Slave LINE Init OK! ===
=====

ISP Dev 3 running !
-----press enter key to exit!-----

```

4. HDMI 屏幕显示如下画面



## 7.7 sample\_uvc

接线图如下：



1. 编译文件系统后可在/root/device\_sample/路径下找到此文件

```
sd3403 ~/device_sample # ll
total 10464
-rwxr-xr-x 1 root root 5235120 Mar  9 12:34 sample_audio
-rwxr-xr-x 1 root root 2709528 Mar  9 12:34 sample_hdmi
-rwx----- 1 root root 2747096 Mar  9 12:54 sample_uvc
drwxr-xr-x 2 root root    4096 Mar  9 12:34 source_file
-rwxr-xr-x 1 root root   10224 Mar  9 12:34 test
sd3403 ~/device_sample #
```

2. 执行

```
sd3403 ~/device_sample # ./sample_uvc
sample_uvc_usage: ./sample_uvc device [options]
supported options:
-f, --format format          set the video format
-F, --file[=name]             write file
-h, --help                     show help info
-s, --size WxH                set the frame size (eg. 1920x1080)

inquire USB device format: ./sample_uvc /dev/video0 --enum-formats

example of setting USB device format:
./sample_uvc /dev/video0 -fH264 -s1920x1080 -Ftest.h264
./sample_uvc /dev/video0 -fH265 -s1920x1080 -Ftest.h265
./sample_uvc /dev/video0 -fMJPEG -s1920x1080 -Ftest.mjpg
./sample_uvc /dev/video0 -fYUYV -s1920x1080 -Ftest.yuv
./sample_uvc /dev/video0 -fNV21 -s640x360 -Ftest.yuv

note: set macro MEDIA_WORK to 0 to write file on disk.
```

```
sd3403 ~/device_sample #
```

3. 查看当前有几个 USB 摄像头设备

```
sd3403 ~/device_sample # ll /dev/video
video0  video1
```

4. 查询当前 USB 摄像头支持的视频格式

```
sd3403 ~/device_sample # ./sample_uvc /dev/video0 --enum-formats
usb 3-1.1: reset high-speed USB device number 3 using xhci-hcd
Device /dev/video0 opened.

Device `HD Pro Webcam C920' on `usb-10320000.xhci_1-1.1' (driver 'uvccvideo') supports
video, capture, without mplanes.

- Available formats:
  Format 0: YUYV (56595559)
  Type: Video capture (1)
  Name: YUYV 4:2:2
  Frame size: 640x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x90 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x120 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
```

```

Frame size: 176x144 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 320x180 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 320x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 352x288 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 432x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 640x360 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 800x448 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 800x600 (1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 864x480 (1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 960x720 (1/15, 1/10, 2/15, 1/5)
Frame size: 1024x576 (1/15, 1/10, 2/15, 1/5)
Frame size: 1280x720 (1/10, 2/15, 1/5)
Frame size: 1600x896 (2/15, 1/5)
Frame size: 1920x1080 (1/5)
Frame size: 2560x1472 (1/2)

```

Format 1: MJPEG (47504a4d)

Type: Video capture (1)

Name: Motion-JPEG

```

Frame size: 640x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 160x90 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 160x120 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 176x144 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 320x180 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 320x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 352x288 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 432x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 640x360 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 800x448 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 800x600 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 864x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 960x720 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1024x576 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1280x720 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1600x896 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1920x1080 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)

```

- Available inputs:

[sample\_uvc\_video\_enum\_inputs]-1811: Input 0: Input 1.

[sample\_uvc\_video\_enum\_inputs]-1814:

video format: YUYV (56595559) 640x480 (stride 1280) field filed\_none buffer size 0

Current frame rate: 1/30

Setting frame rate to: 1/25

```
Frame rate set: 1/24
sd3403 ~/device_sample #
```

5. 插上 HDMI 显示器后根据当前摄像头支持的视频格式进行传参，可手动 `ctrl+c` 结束进程

```
sd3403 ~/device_sample # ./sample_uvc /dev/video0 -fMJPEG -s1280x720 -Ftest.jpg
Device /dev/video0 opened.
Device `HD Pro Webcam C920' on `usb-10320000.xhci_1-1.1' (driver 'uvccvideo') supports video,
capture, without mplanes.
- Available formats:
  Format 0: YUYV (56595559)
  Type: Video capture (1)
  Name: YUYV 4:2:2
  Frame size: 640x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x90 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x120 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 176x144 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 320x180 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 320x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 352x288 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 432x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 640x360 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 800x448 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 800x600 (1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 864x480 (1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 960x720 (1/15, 1/10, 2/15, 1/5)
  Frame size: 1024x576 (1/15, 1/10, 2/15, 1/5)
  Frame size: 1280x720 (1/10, 2/15, 1/5)
  Frame size: 1600x896 (2/15, 1/5)
  Frame size: 1920x1080 (1/5)
  Frame size: 2560x1472 (1/2)

  Format 1: MJPEG (47504a4d)
  Type: Video capture (1)
  Name: Motion-JPEG
  Frame size: 640x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x90 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 160x120 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 176x144 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 320x180 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 320x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 352x288 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 432x240 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 640x360 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
  Frame size: 800x448 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
```

```
Frame size: 800x600 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 864x480 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 960x720 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1024x576 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1280x720 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1600x896 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
Frame size: 1920x1080 (1/30, 1/24, 1/20, 1/15, 1/10, 2/15, 1/5)
```

- Available inputs:

```
[sample_uvc_video_enum_inputs]-1811: Input 0: Input 1.
```

```
[sample_uvc_video_enum_inputs]-1814:
```

```
video format set: MJPEG (47504a4d) 1280x720 (stride 0) field filed_none buffer size 1843200
```

```
video format: MJPEG (47504a4d) 1280x720 (stride 0) field filed_none buffer size 1843200
```

```
Current frame rate: 1/30
```

```
Setting frame rate to: 1/25
```

```
Frame rate set: 1/24
```

```
8 buffers requested.
```

```
length: 1843200 offset: 0 timestamp type/source: monotonic/soe
```

```
Buffer 0/0 mapped at address 0x7f97e1e000.
```

```
length: 1843200 offset: 1843200 timestamp type/source: monotonic/soe
```

```
Buffer 1/0 mapped at address 0x7f97c5c000.
```

```
length: 1843200 offset: 3686400 timestamp type/source: monotonic/soe
```

```
Buffer 2/0 mapped at address 0x7f97a9a000.
```

```
length: 1843200 offset: 5529600 timestamp type/source: monotonic/soe
```

```
Buffer 3/0 mapped at address 0x7f978d8000.
```

```
length: 1843200 offset: 7372800 timestamp type/source: monotonic/soe
```

```
Buffer 4/0 mapped at address 0x7f97716000.
```

```
length: 1843200 offset: 9216000 timestamp type/source: monotonic/soe
```

```
Buffer 5/0 mapped at address 0x7f97554000.
```

```
length: 1843200 offset: 11059200 timestamp type/source: monotonic/soe
```

```
Buffer 6/0 mapped at address 0x7f97392000.
```

```
length: 1843200 offset: 12902400 timestamp type/source: monotonic/soe
```

```
Buffer 7/0 mapped at address 0x7f971d0000.
```

```
^C
```

```
media exit...
```

```
sd3403 ~/device_sample #
```

## 7.8 sample\_mipi\_vdec

1. 测试前将显示器和开发板用 mipi 排线连接好

```
cd /root/device_sample/mipi_vdec
sd3403 ~ # ./sample_vdec 1920_1200.h265
start vo dhd0.
```

```
mipi intf sync = 48

sample_test:press 'e' to exit; 'q' to query!;

chn 0, stream file:1920_1200.h265, userbufsize: 3456000

chn 1, stream file:1920_1200.h265, userbufsize: 3456000

chn 2, stream file:1920_1200.h265, userbufsize: 3456000

chn 3, stream file:1920_1200.h265, userbufsize: 3456000
q
-----
chn:0, type:265, start:1, decode_frames:17874, left_pics:0, left_bytes:0, left_frames:0,
recv_frames:17874
format_err:0, pic_size_err_set:0, stream_unsprt:0, pack_err:0, set_pic_size_err:0,
ref_err_set:0, pic_buf_size_err_set:0
-----
-----
chn:1, type:265, start:1, decode_frames:17874, left_pics:0, left_bytes:0, left_frames:0,
recv_frames:17874
format_err:0, pic_size_err_set:0, stream_unsprt:0, pack_err:0, set_pic_size_err:0,
ref_err_set:0, pic_buf_size_err_set:0
-----
-----
chn:2, type:265, start:1, decode_frames:17875, left_pics:0, left_bytes:0, left_frames:0,
recv_frames:17875
format_err:0, pic_size_err_set:0, stream_unsprt:0, pack_err:0, set_pic_size_err:0,
ref_err_set:0, pic_buf_size_err_set:0
-----
-----
chn:3, type:265, start:1, decode_frames:17873, left_pics:0, left_bytes:0, left_frames:0,
recv_frames:17873
format_err:0, pic_size_err_set:0, stream_unsprt:0, pack_err:0, set_pic_size_err:0,
ref_err_set:0, pic_buf_size_err_set:0
-----
```

## 2. mipi 屏播放视频文件内容



## 注意事项

如果开发板上电后如果 MIPI 屏幕背光不亮, 请先检查接线, 接线没问题可以设置 gpio4

```
bspmm 0x0102F00E0 0x1200
echo 4 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio4/direction
echo 1 > /sys/class/gpio/gpio4/value
```

## 7.9 I2C

1. 在开始测试前需检查 oled 驱动节点是否存在

```
ls /dev/oled*
```

2. oled 显示屏显示 ETH0 网卡 IP

```
ifconfig eth0 | grep -oP 'inet addr:\K\S+' | awk '{print $1}' >> "/dev/oled-0"
```

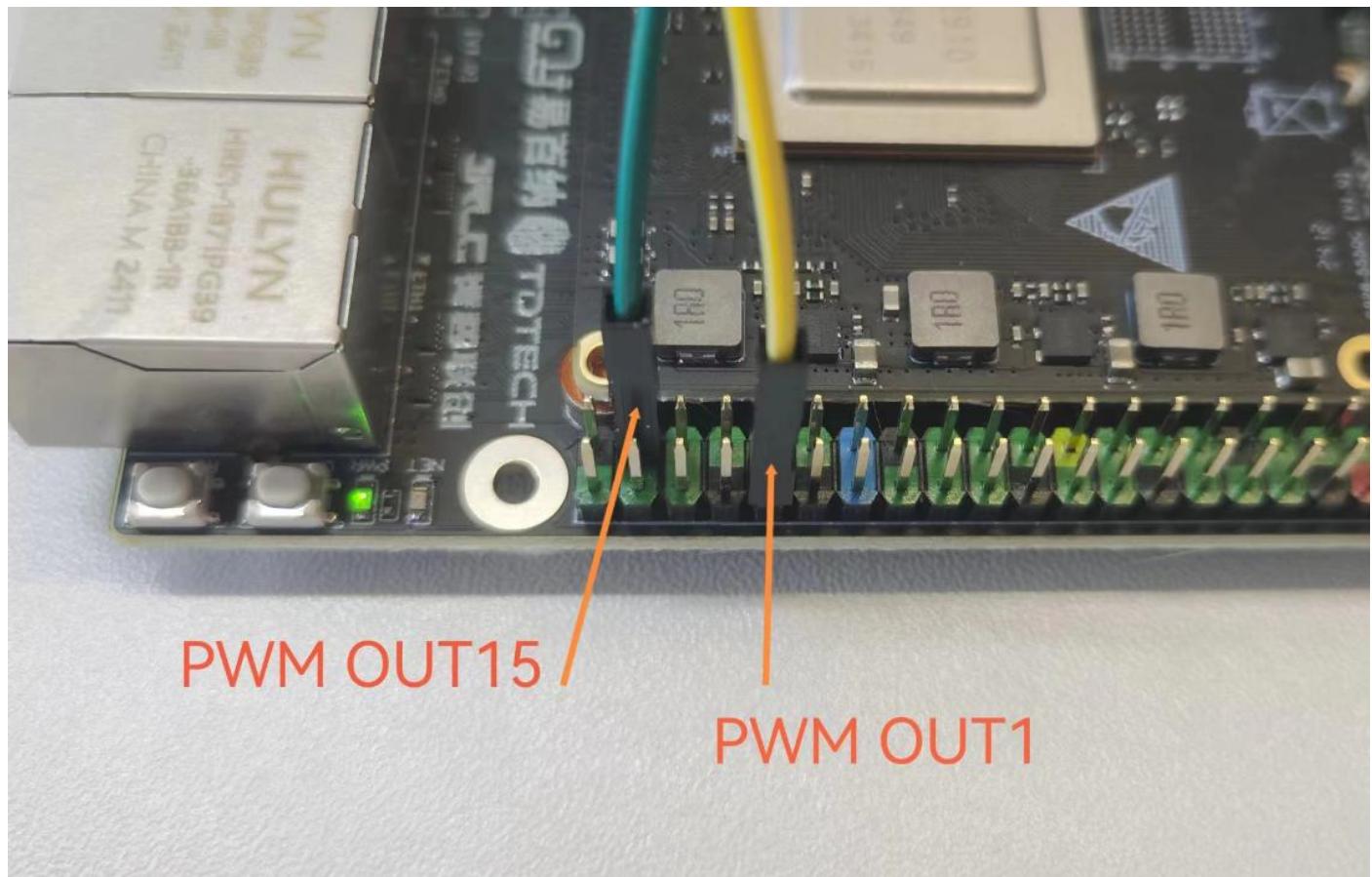
3. oled 显示屏显示易百纳鲸鱼 logo

```
./oled "/dev/oled-0" 1
```



## 7.10 PWM

PWM 接线图



1. 进入欧拉派的终端，执行 pwm 程序

```
PWM_help:  
cd /root/device_sample/pwm  
./pwm <1> <2> <3> <4>  
<1> be open or close to enable/disable PWM  
<2> be 1 or 15 to chose PWM0_1 or PWM0_15  
<3> be value for period  
<3> be value for duty_cycle
```

示例一 开启舵机

将舵机与拓展板连接，并执行如下指令

```
./pwm open 1 20000000 2500000
```

舵机开始转动

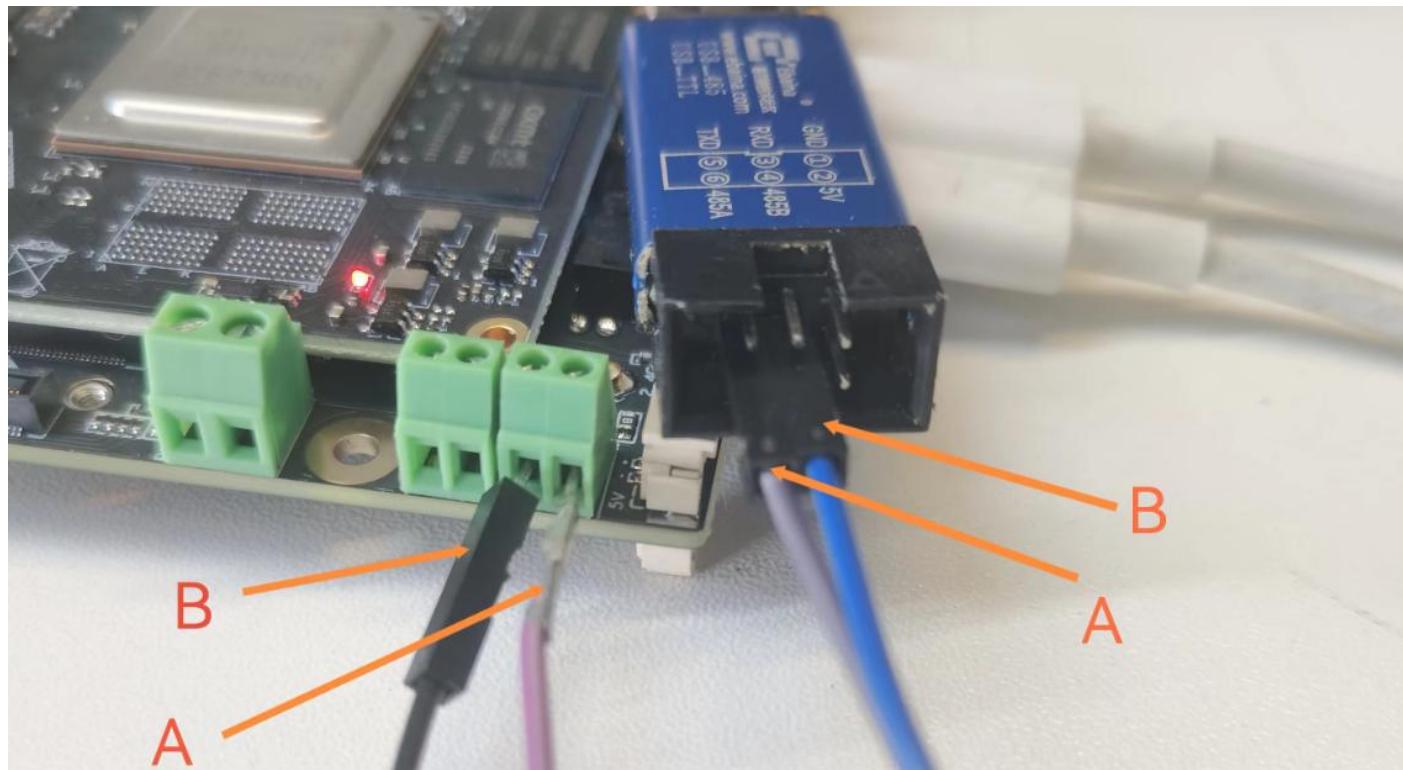
示例二 关闭舵机

```
./pwm close 1
```

舵机停止转动

## 7.11 RS485

RS485 接线图如下



在开始测试前将 RS485 调试工具接到 PC 端，并在 PC 端打开串口调试工具

### 1. 发送测试

使用 `rs485_senddata` 命令进行发送测试，在欧拉派的终端执行，通过 RS485 发送 Hello World

```
cd /root/device_sample/rs485  
. ./rs485_senddata /dev/ttyAMA3 115200 "Hello World"
```

发送后能在 PC 端的串口调试工具的软件中收到 Hello World 消息

### 2. 接收测试

使用 `rs485_recvdata` 命令进行接收测试，在欧拉派的终端执行

```
./rs485_recvdata /dev/ttyAMA3 115200
```

应用开启后持续接收 RS485 总线上的消息并打印到欧拉派终端，在 PC 端发送数据可在欧拉派的终端收到。使用 `Ctrl+C` 结束接收

## 7.12 RTC

欧拉派上电后，打开串口终端，并执行如下命令

```
hwclock --set --date="2024-02-01 14:30:00" #该命令是用于设置 RTC 时钟  
hwclock --hctosys #该命令是将 RTC 时钟同步到系统时钟
```

## 7.13 USB

U 盘插入后将会打印相关信息

```
usb 4-1.3: new SuperSpeed Gen 1 USB device number 5 using xhci-hcd
usb-storage 4-1.3:1.0: USB Mass Storage device detected
scsi host0: usb-storage 4-1.3:1.0
scsi 0:0:0:0: Direct-Access Kingston DataTraveler 3.0 0000 PQ: 0 ANSI: 6
sd 0:0:0:0: [sda] 60538881 512-byte logical blocks: (31.0 GB/28.9 GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
  sda:
sd 0:0:0:0: [sda] Attached SCSI removable disk
```

执行 U 盘分区并格式化为 ext4 文件系统格式

```
fdisk /dev/sda
```

注意：需要通过打印信息查看你自己的 u 盘节点是哪个，一般会在 sd[a-z] 之间，如果你的 u 盘节点是 /dev/sdb，那么就执行 fdisk /dev/sdb

按下图红框执行即可做出一个 500M 的分区 /dev/sda1

```
sd3403 ~ # fdisk /dev/sda
Welcome to fdisk (util-linux 2.39.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-60538880, default 2048): 2048
Last sector, +/sectors or +/-size{K,M,G,T,P} (2048-60538880, default 60538880): +500M
Created a new partition 1 of type 'Linux' and of size 500 MiB.
Partition #1 contains a ext4 signature.

Do you want to remove the signature? [Y]es/[N]o yes
The signature will be removed by a write command.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
  sda: sda1
Syncing disks.

sd3403 ~ #
```

```
mkfs.ext4 /dev/sda1 # 将刚刚做好的分区格式化为 ext4 格式
```

```
sd3403 ~ # mkfs.ext4 /dev/sda1
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 512000 1k blocks and 128016 inodes
Filesystem UUID: 3a858a36-7957-49e7-8e96-7129f9bb6be9
Superblock backups stored on blocks:
          8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

sd3403 ~ #
```

test\_usb.sh 脚本用法：

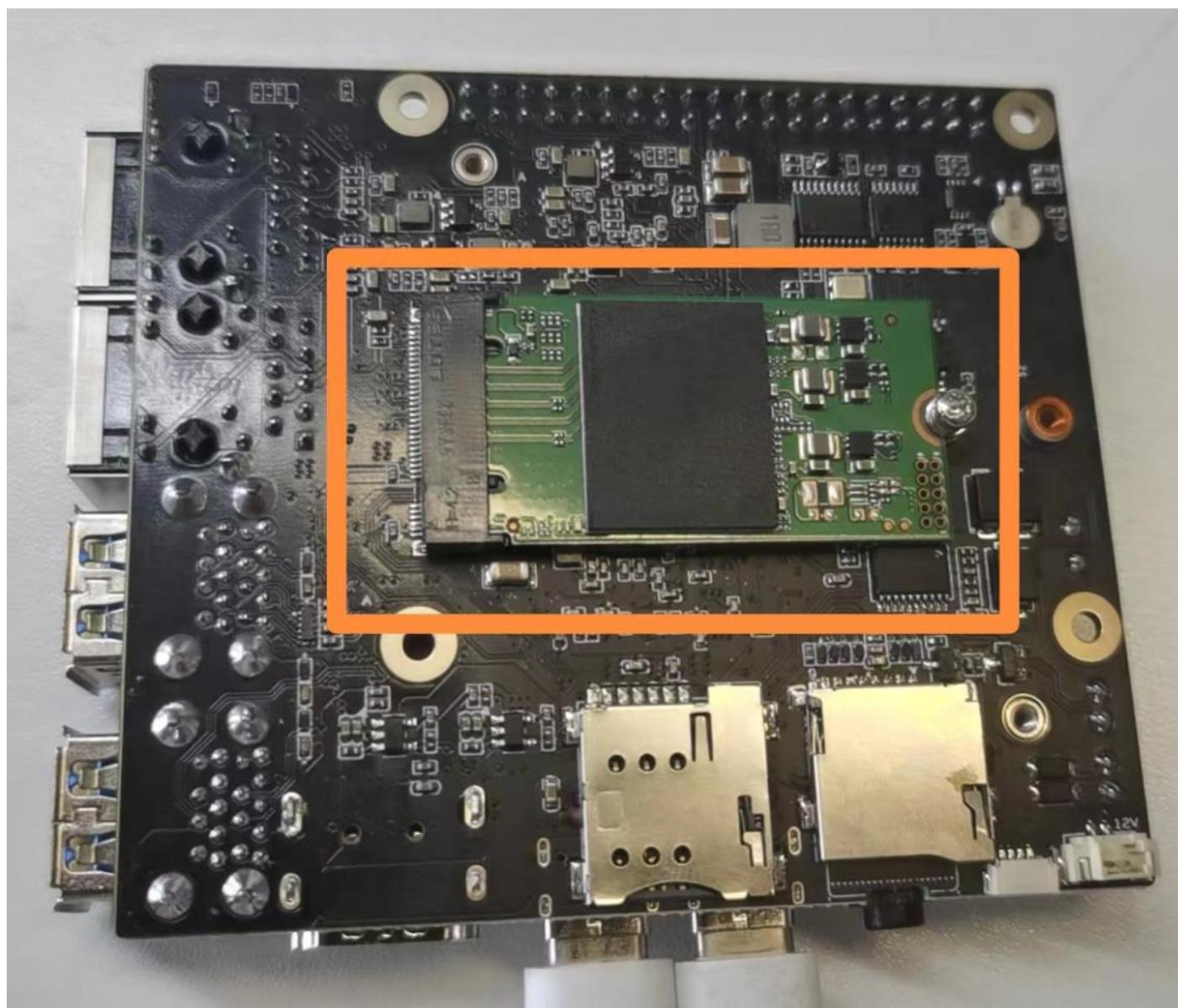
```
./test_usb.sh <device_node> #参数为存储设备分区节点  
Eg:  
./test_usb.sh /dev/sda1
```

## 7.14 PCIE

出厂固件里的内核已将 NVME 硬盘驱动打开

PCIE 硬盘不是热插拔设备，将 PCIE 硬盘接上后需要重启才能识别到设备

PCIE 接线图：



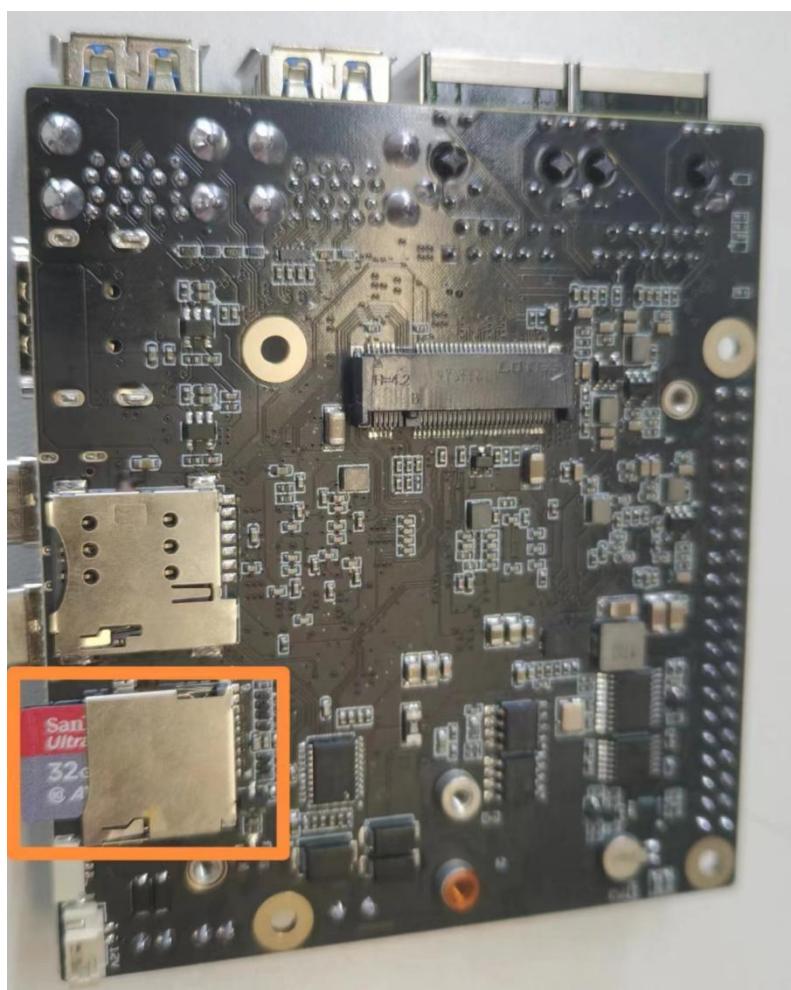
执行操作：

```
fdisk -l # 查看PCIE 硬盘是否被正确识别
mount /dev/nvme0n1p1 /mnt # PCIE 硬盘分区挂载到/mnt 目录
dd if=/dev/zero of=/mnt/testfile bs=1M count=200 # 测试写入速度
dd if=/mnt/testfile of=/dev/null bs=1M      # 测试读取速度
umount /mnt      # 解挂载
```

## 7.15 TF

出厂固件里的内核已将 TF 卡自动打开，插上会打印信息

TF 接线图



执行操作：

```
fdisk -l # 查看TF 卡是否被正确识别
mount /dev/mmcblk1p1 /mnt # TF 卡分区挂载到/mnt 目录
dd if=/dev/zero of=/mnt/testfile bs=1M count=200 # 测试写入速度
dd if=/mnt/testfile of=/dev/null bs=1M      # 测试读取速度
umount /mnt      # 解挂载
```



易百纳公司名称：南京启诺信息技术有限公司

易百纳技术社区：[www.ebaina.com](http://www.ebaina.com)

技术咨询电话：18013825199

技术服务微信：david089968

