

华为认证 HarmonyOS 系列教程

# HCIA-HarmonyOS

## Device Developer

### 实验环境搭建指南

版本：1.0



华为技术有限公司

版权所有 © 华为技术有限公司 2021。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## 注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址：深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址：<http://e.huawei.com>

---

## 华为认证体系介绍

华为认证是华为公司基于“平台+生态”战略，围绕“云-管-端”协同的新ICT技术架构，打造的覆盖ICT（Information and Communications Technology 信息通信技术）全技术领域的认证体系，包含ICT技术架构与应用认证、云服务与平台认证两类认证。

根据ICT从业者的学习和进阶需求，华为认证分为工程师级别、高级工程师级别和专家级别三个认证等级。

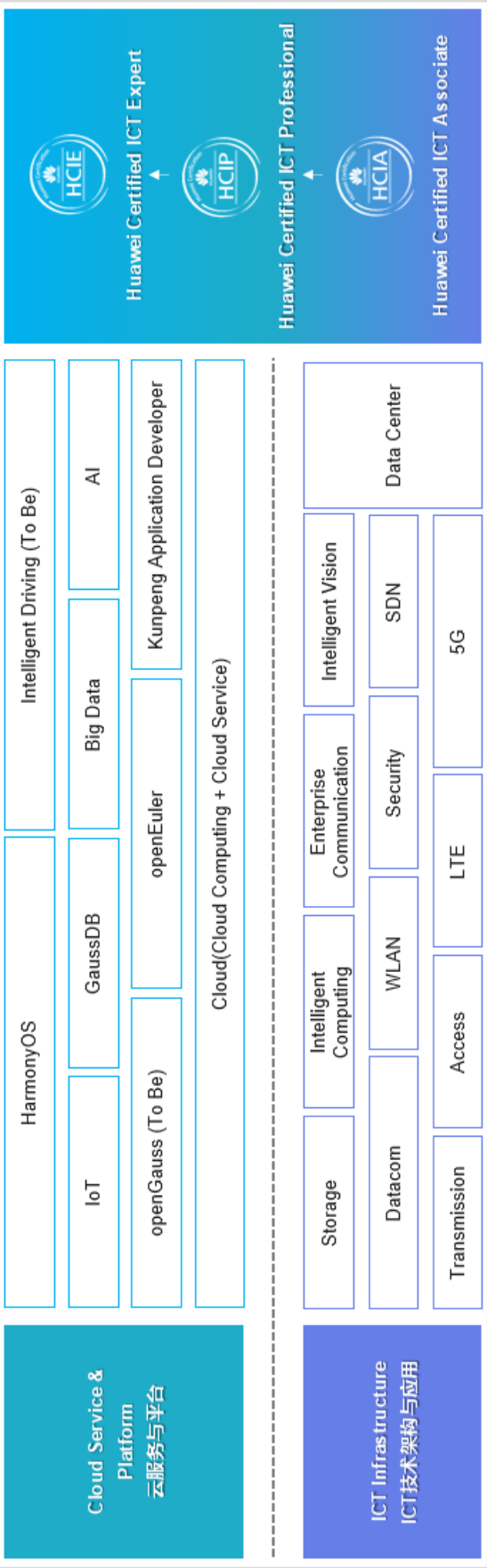
华为认证覆盖ICT全领域，符合ICT融合的技术趋势，致力于提供领先的人才培养体系和认证标准，培养数字化时代新型ICT人才，构建良性ICT人才生态。

HCIA-HarmonyOS Device Developer V1.0定位于培养基于HarmonyOS设备开发场景具备专业知识和技能水平的工程师。

通过HCIA-HarmonyOS Device Developer V1.0认证，您将掌握HarmonyOS基本概念及原理、HarmonyOS技术架构、HarmonyOS应用开发流程、内核与驱动知识，具备HarmonyOS设备子系统开发、移植的能力，能够胜任HarmonyOS设备开发工程师岗位。

华为认证协助您打开行业之窗，开启改变之门，屹立在HarmonyOS行业的潮头浪尖！

# Huawei Certification



# 前言

## 简介

本书为 HCIA-HarmonyOS Device Developer 认证培训教程，适用于搭建 HarmonyOS 设备的开发环境。

## 实验环境说明

### 设备介绍

为了满足 HCIA-Harmony Device Developer 实验需要，建议每套实验环境采用以下配置：

设备名称、型号与版本的对应关系如下：

设备名称	设备型号	软件版本
笔记本电脑	Windows系统	Windows 10 64位
Hi3861开发板	BearPi-HM Nano	HarmonyOS 1.1.0

## 准备实验环境

### 检查设备

实验开始之前请每组学员检查自己的实验设备是否齐全，实验清单如下。

设备名称	数量	备注
笔记本或台式机	每组 1 台	台式机要有无线网卡

每组检查自己的设备列表如下：

- 笔记本或台式机 1 台。

## 参考资料及工具

文档中所列出的命令以及参考文档，请根据实际环境中的不同产品版本使用对应的命令以及文档。

## 参考文档：

1. 《 HarmonyOS 官方文档 》，获取地址：  
<https://developer.harmonyos.com/cn/documentation>
2. 《 HarmonyOS Device 官方文档 》，获取地址：  
<https://device.harmonyos.com/cn/home/>

## 软件工具：

编号	工具名称	版本
1	Visual Studio Code	V1.55.2
2	HiBurn	V2.2
3	DevEco Device Tool	V2.1
4	Oracle VM VirtualBox	V6.1.22



# 目录

<b>前 言</b>	<b>3</b>
简介	3
实验环境说明	3
准备实验环境	3
参考资料及工具	3
<b>1 开发环境搭建</b>	<b>6</b>
1.1 实验介绍	6
1.1.1 关于本实验	6
1.1.2 实验目的	6
1.2 实验任务	6
1.2.1 任务简要	6
1.2.2 安装 Oracle VM VirtualBox	6
1.2.3 安装原生态 Ubuntu20 版本	7
1.2.4 Windows 主机访问 Linux 主机配置	20
1.2.5 获取代码（方法一：HPM 获取）	26
1.2.6 获取代码（方法二：gitee 获取）	27
1.2.7 安装 Linux 编译环境（方法一：Docker 方式）	28
1.2.8 安装 Linux 编译环境（方法二：安装包方式）	29
1.2.9 安装 Window 开发环境	29
1.3 思考题	38

# 开发环境搭建

## 1.1 实验介绍

### 1.1.1 关于本实验

本实验通过在 Windows 电脑上搭建 HarmonyOS 的设备开发环境，辅助后续实验开展。

### 1.1.2 实验目的

- 掌握 HarmonyOS 设备开发的环境搭建。

## 1.2 实验任务

### 1.2.1 任务简要

- 1、安装 Oracle VM VirtualBox。
- 2、安装原生态 Ubuntu20。
- 3、从零开始部署 HarmonyOS 开发环境。
- 4、Windows Visual Studio Code 安装。
- 5、Windows 开发环境安装。

### 1.2.2 安装 Oracle VM VirtualBox

#### 步骤 1 下载 Oracle VM VirtualBox 安装包

打开 VirtualBox 官网网站 <https://www.virtualbox.org/wiki/Downloads>，下载 Windows 64 位对应的软件包。



选择 Windows 64 位版本的软件包，点击链接下载。

#### 步骤 2 安装 Oracle VM VirtualBox

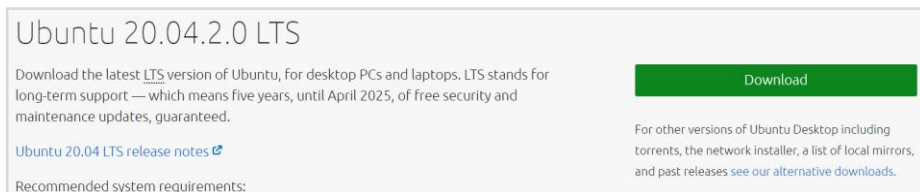
下载完成后，双击安装文件，全部默认安装，直至完成。



## 1.2.3 安装原生态 Ubuntu20 版本

### 步骤 1 下载 Ubuntu 镜像

打开 Ubuntu 官网网站 <https://ubuntu.com/download/desktop>。

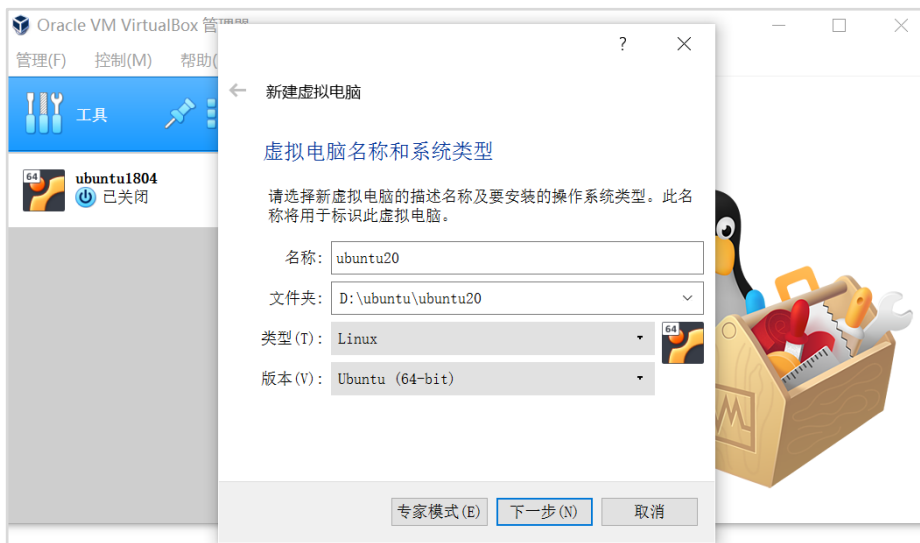


选择 Ubuntu20.04.2.0.LTS 版本的软件包，点击 Download 下载。

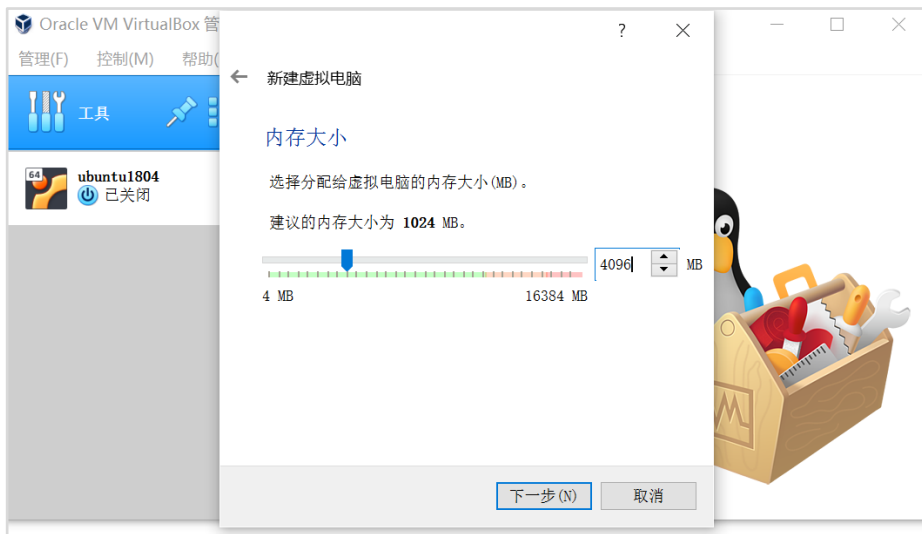
### 步骤 2 打开 Oracle VM VirtualBox，新建虚拟机



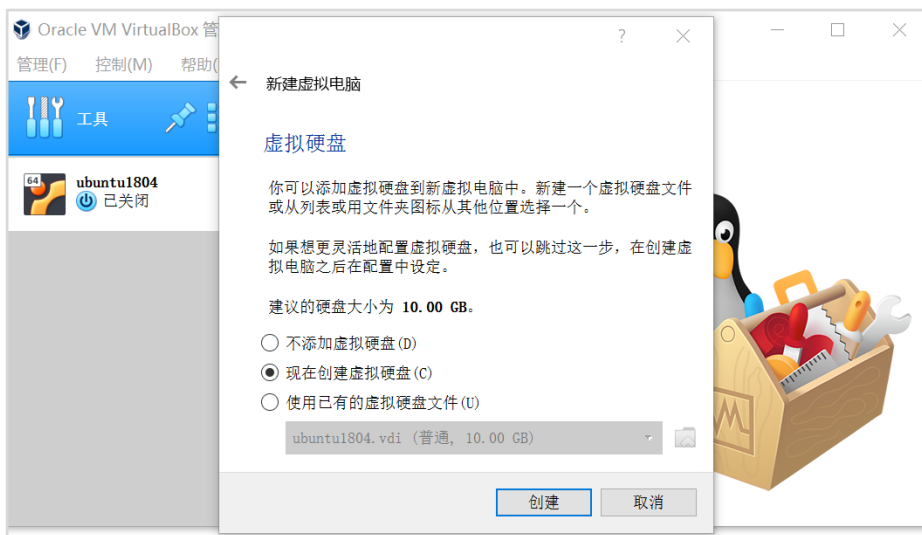
点击新建，新建虚拟机，名称，文件夹可以自定义；



分配内存 4096M；



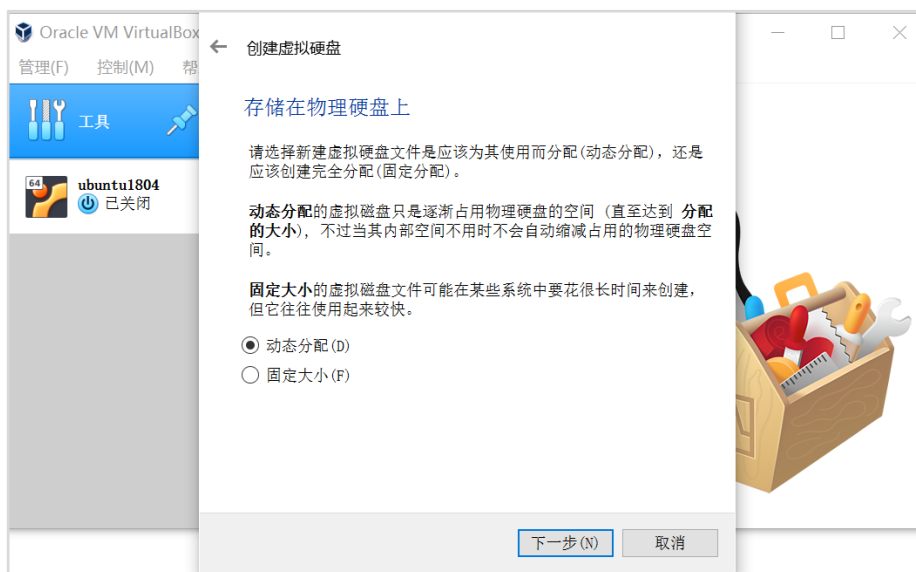
新建虚拟硬盘，点现在创建虚拟硬盘(C)；



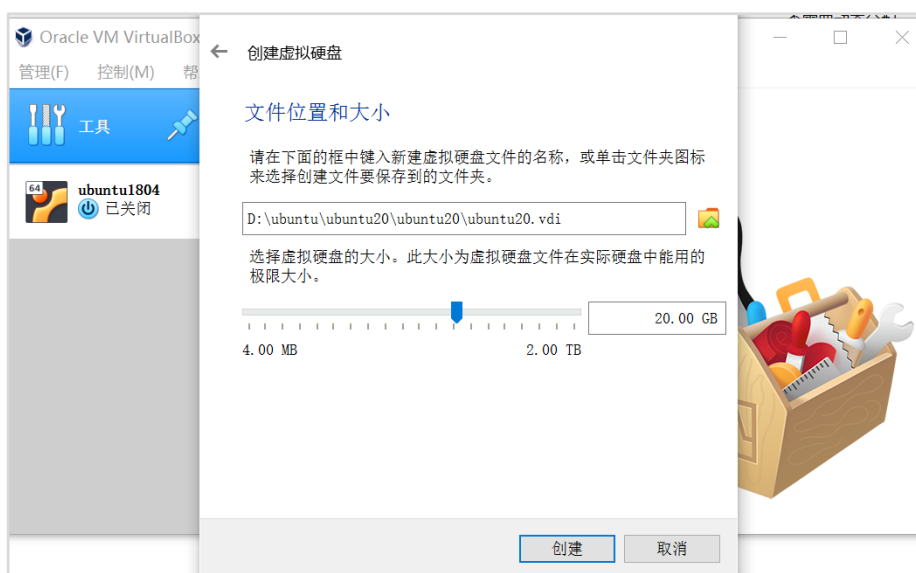
选择虚拟硬盘文件类型为 VDI；



选择动态分配；



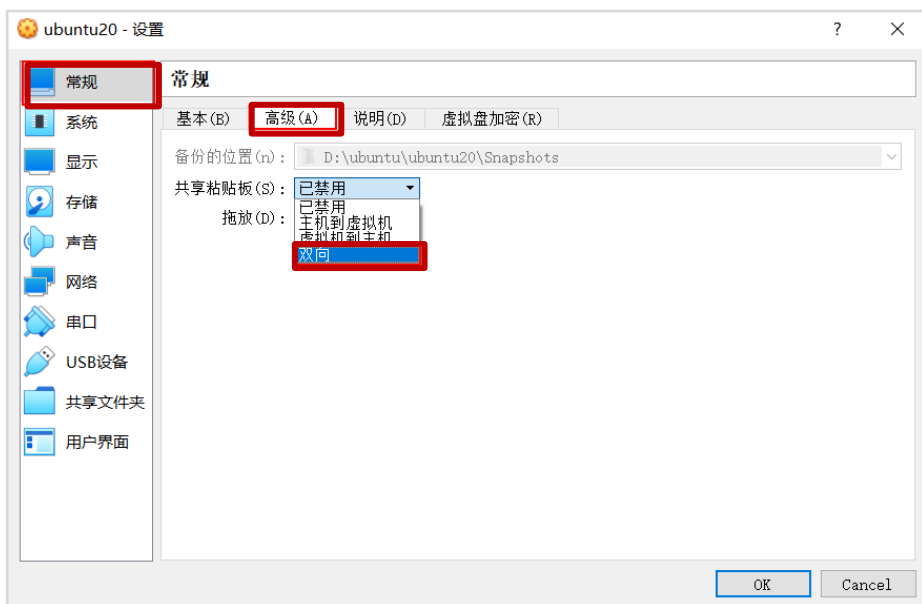
选择虚拟硬盘，文件位置和大小，大小为 20G；

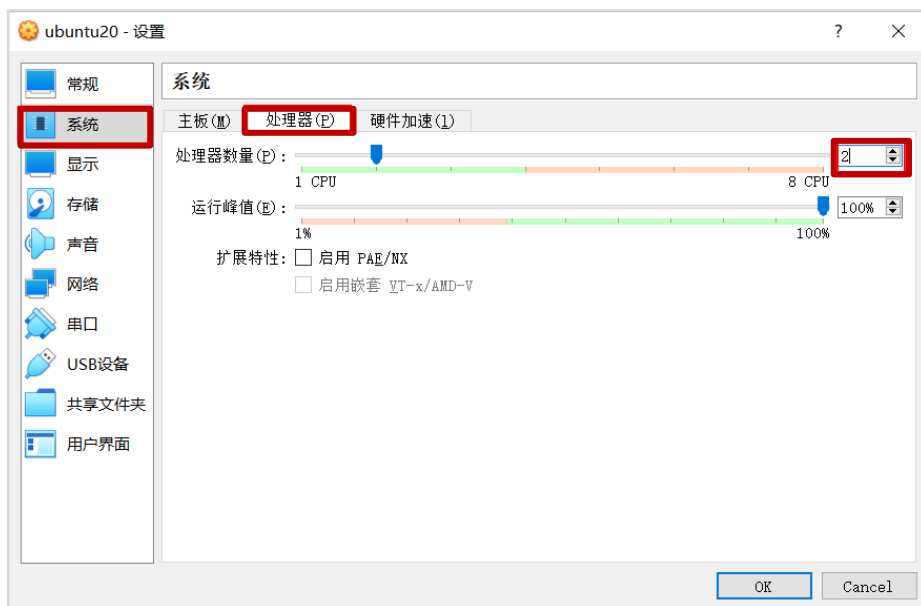


创建虚拟机成功；



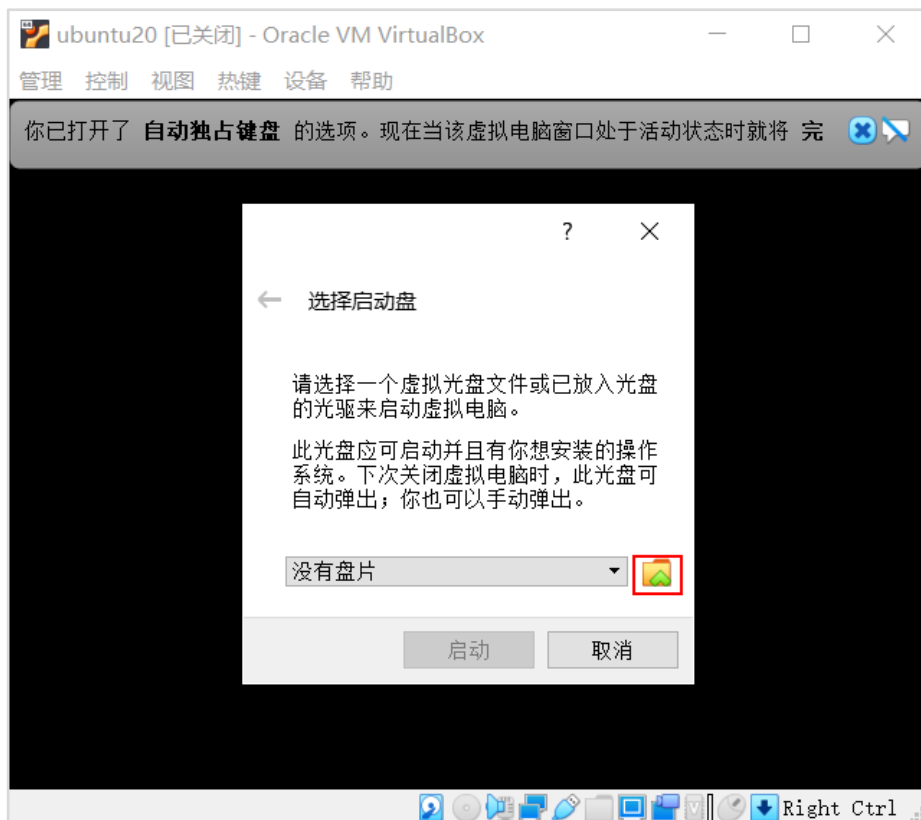
点击设置 Oracle VM VirtualBox。



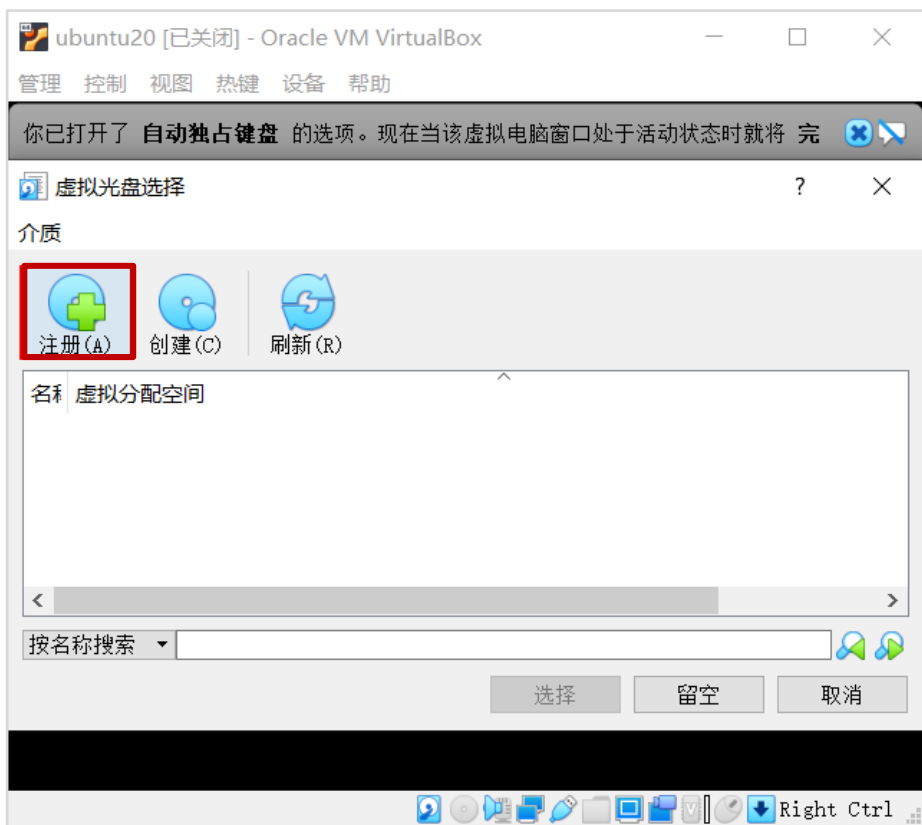


### 步骤 3 安装 ubuntu20.04 镜像

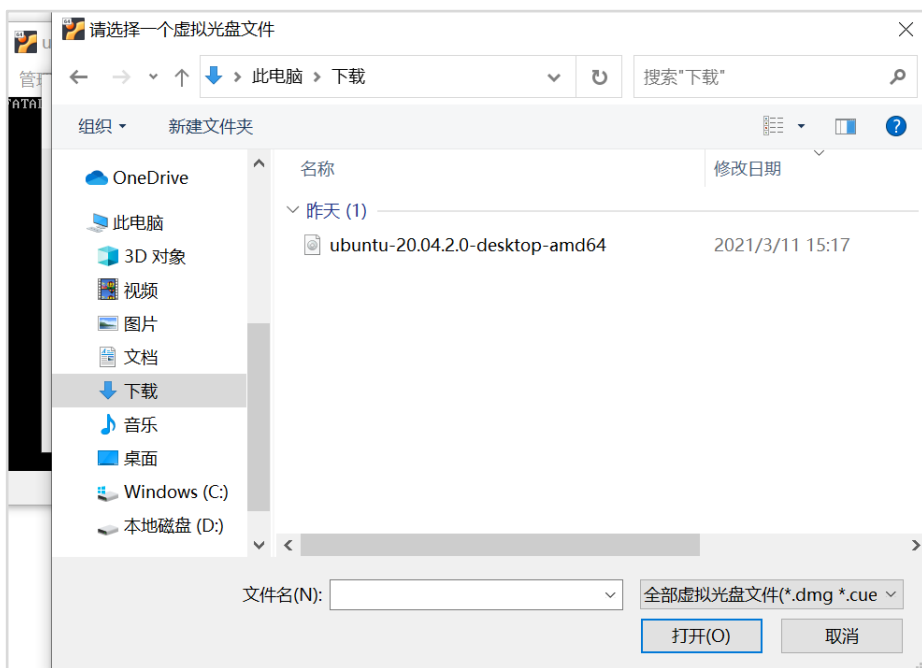
点击启动按钮，启动虚拟机，选择启动盘；



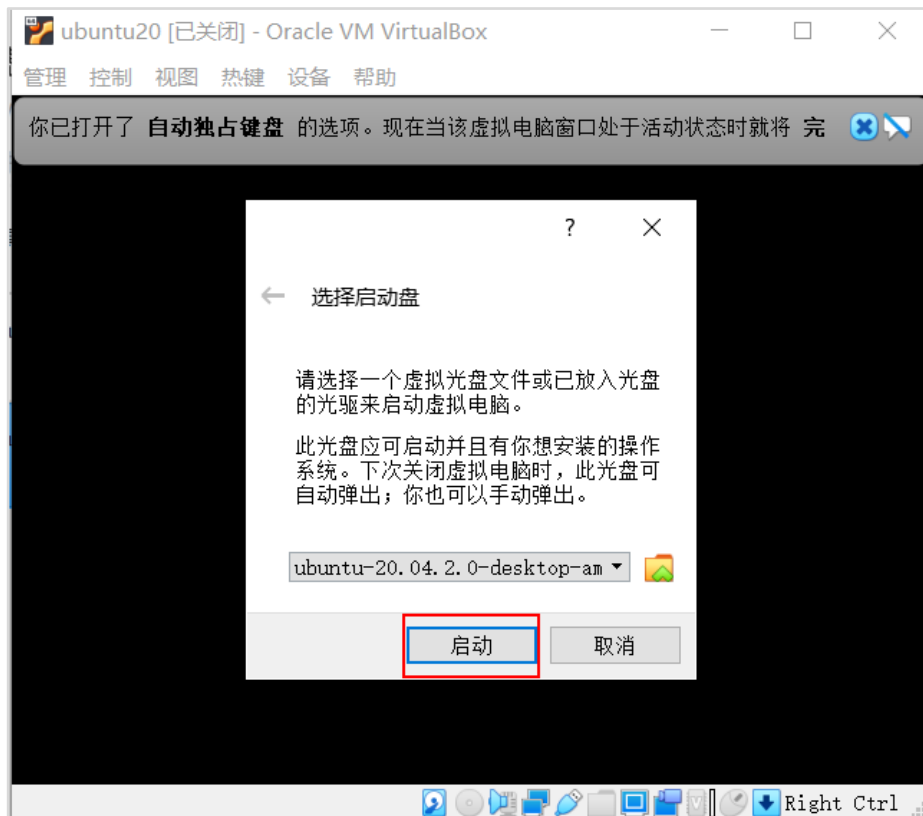
选择注册；



选择一个虚拟光盘文件，选择步骤 2 中已经下载好的 Ubuntu 镜像文件；



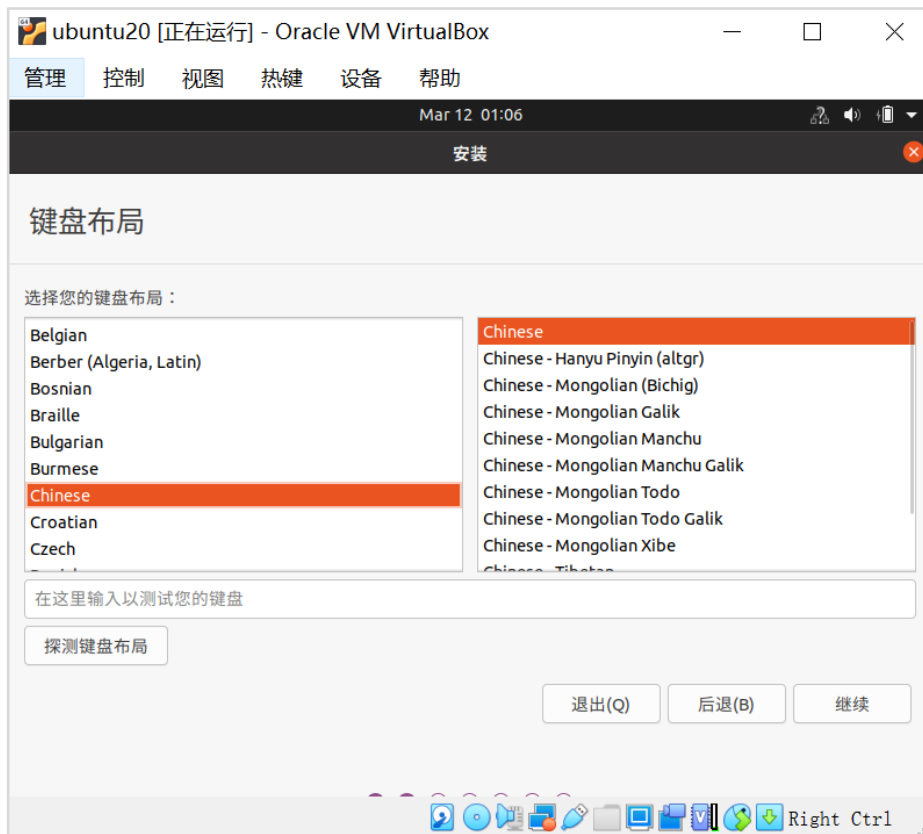
启动，进行后续 ubuntu 安装；



点击安装 Ubuntu;



选择键盘布局，继续；



选择正常安装，继续；



选择清除整个磁盘并安装 ubuntu，点击现在安装；





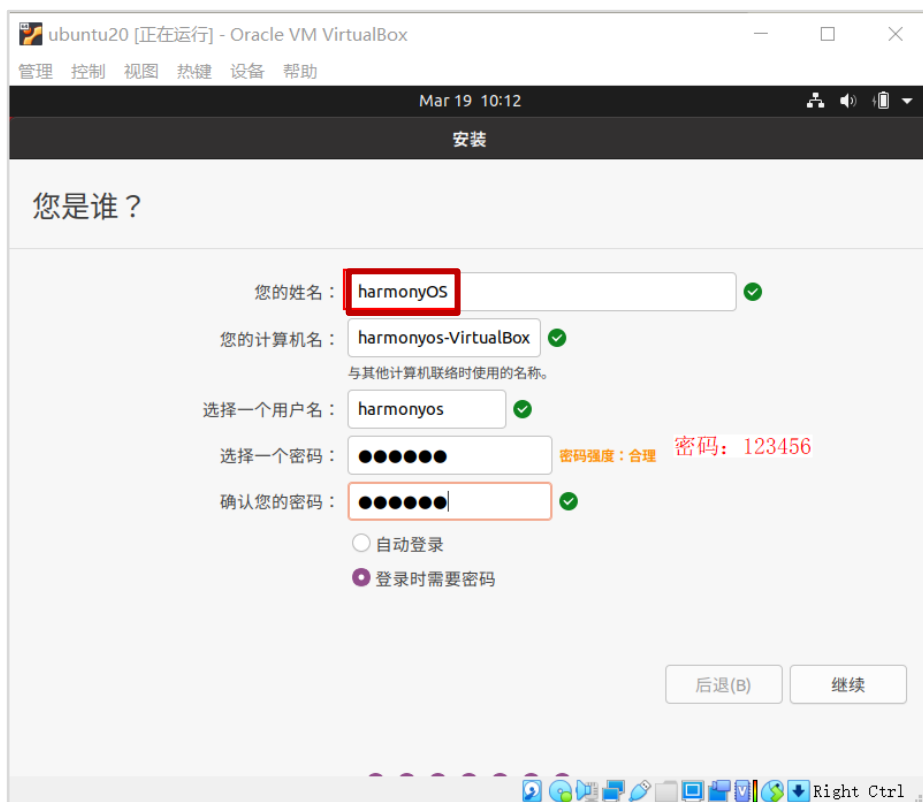
点击继续；



选择位置，继续；



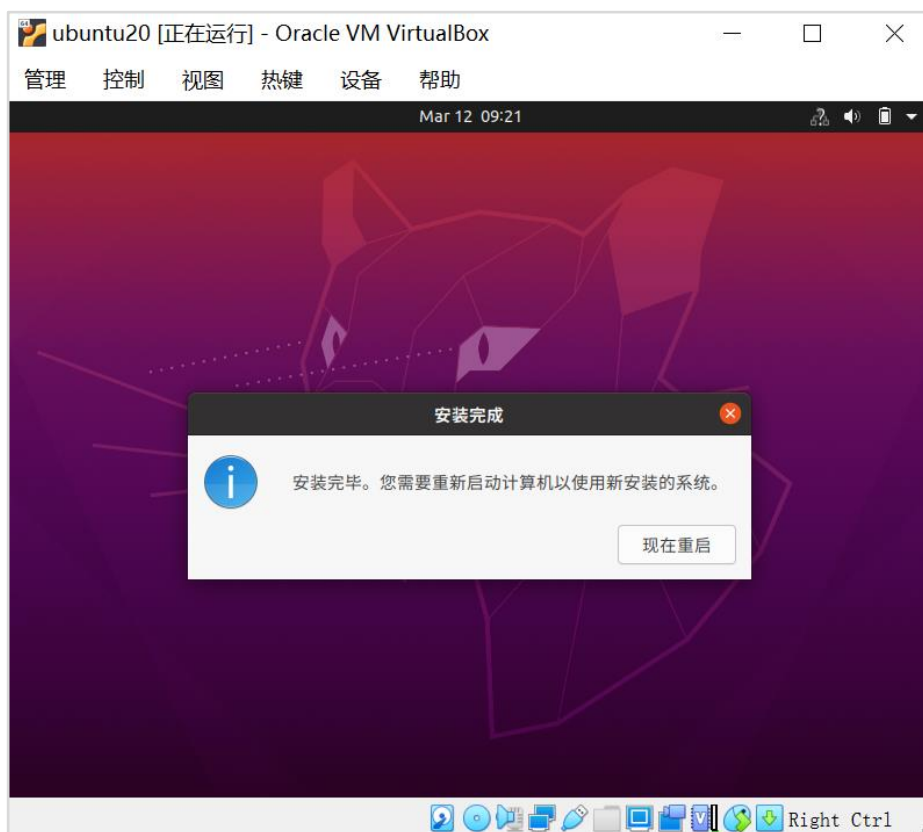
填入姓名，用户名密码，用户名密码可自定义，继续；



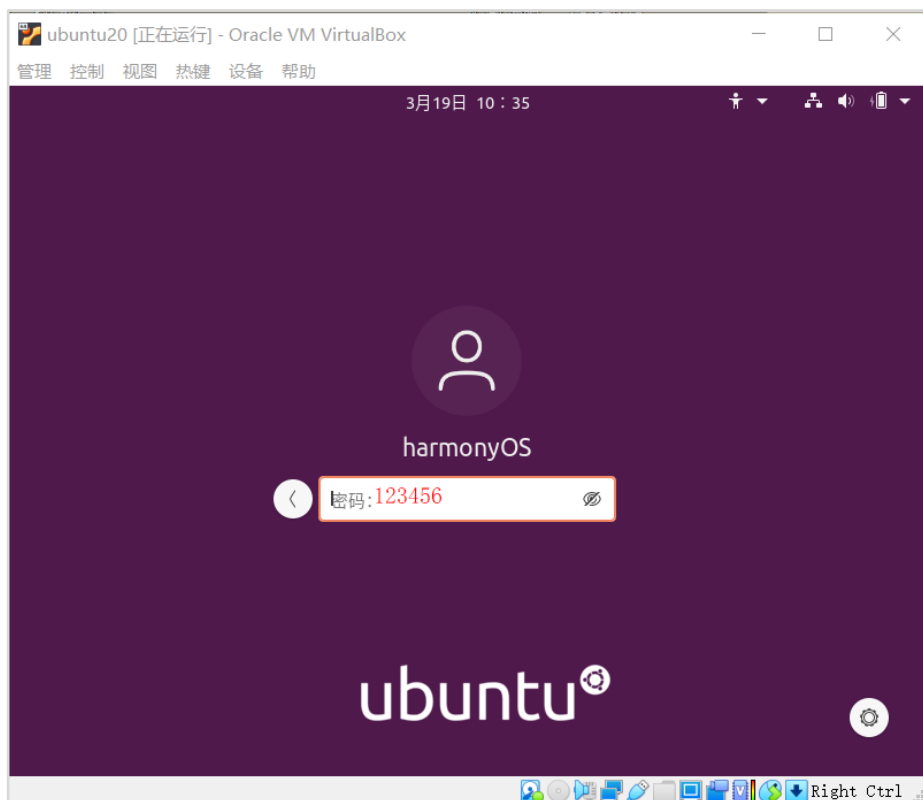
安装需要等待 15 分钟左右；



点击现在重启；



输入登录密码；



更新软件，点击立即安装；



安装增强功能；



如果出现以下错误；



请手动安装增强工具；

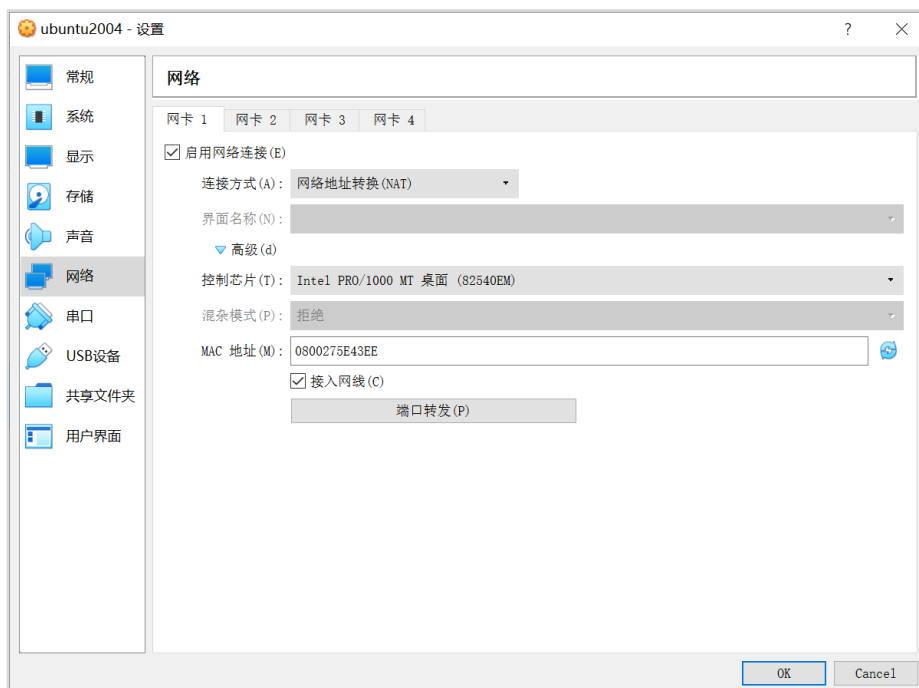
进入命令行模式，在 Ubuntu 系统中桌面点击右键，进入命令行模式。

```
/media/harmonyos/VBox_GAs_6.1.18/autorun.sh
```

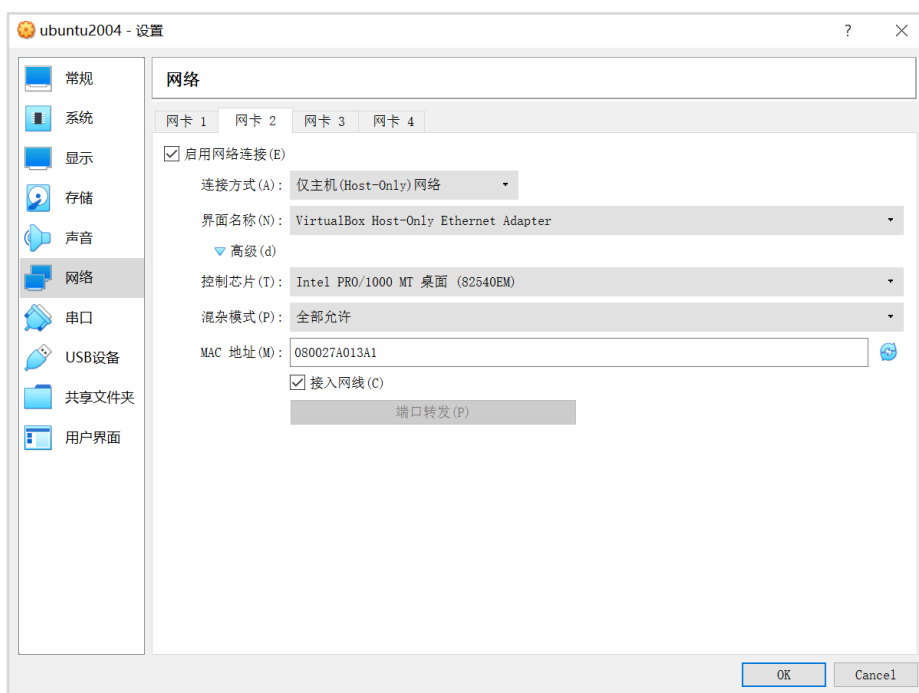
## 1.2.4 Windows 主机访问 Linux 主机配置

### 步骤 1 VirtualBox 设置双网卡

默认已启用网卡 1，连接方式（网络地址转换 NAT）；



添加一个 HostOnly 的虚拟网卡 2，混杂模式：全部允许；



启动虚拟机，进入命令行模式，查看 Linux 虚拟机中的 IP 地址，确保 windows 可以 ping 通 Ubuntu。

如果 ifconfig 命令没有，则运行 `sudo apt install net-tools` 安装。

```

harmonyos@harmonyos-VirtualBox:~/桌面$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::5360:28ba:726f:9050 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:5e:43:ee txqueuelen 1000 (以太网)
    RX packets 152 bytes 31990 (31.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 273 bytes 33330 (33.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.101 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::a00:27ff:fea0:13a1 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:a0:13:a1 txqueuelen 1000 (以太网)
    RX packets 52 bytes 5916 (5.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 141 bytes 18292 (18.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

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 (本地环回)
    RX packets 186 bytes 18221 (18.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 186 bytes 18221 (18.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

在 windows 里面 ping 虚拟机里面的 IP;

```

C:\ 命令提示符
Microsoft Windows [版本 10.0.17763.1577]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\>ping 192.168.56.101

正在 Ping 192.168.56.101 具有 32 字节的数据:
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64

192.168.56.101 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms

```

安装 vim 编辑工具, 用于编辑文件。

```
sudo apt-get install vim
```

```

harmonyos@harmonyos-VirtualBox:~/桌面$ sudo apt-get install vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成

```

## 步骤 2 支持远程终端访问

先安装 openssh-server, 在 Linux 命令行模式下, 输入如下命令。

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

## 步骤 3 在 Linux 环境中安装 samba 服务器



安装 samba 服务器，用于和 windows 共享文件；

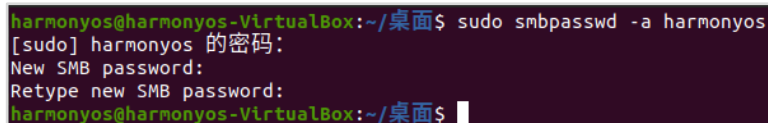
```
sudo apt install samba
```

创建一个 samba 目录，用于文件分享，设置权限为：可读、可写、可执行；

```
sudo mkdir ~/share  
sudo chmod 777 ~/share
```

添加 samba 用户（用户名为：harmonyos，设置 samba 的密码为：123456）；

```
sudo smbpasswd -a harmonyos
```



```
harmonyos@harmonyos-VirtualBox:~/桌面$ sudo smbpasswd -a harmonyos  
[sudo] harmonyos 的密码：  
New SMB password:  
Retype new SMB password:  
harmonyos@harmonyos-VirtualBox:~/桌面$
```

配置 samba 的配置文件；

```
sudo vim /etc/samba/smb.conf
```

在配置文件 smb.conf 的最后添加下面的内容：

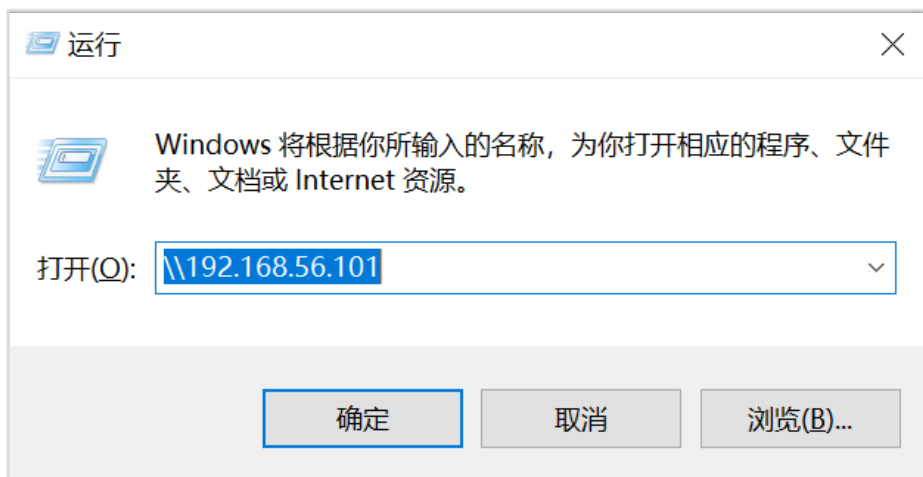
```
[share]  
comment = share folder  
browseable = yes  
path = /home/harmonyos/share  
create mask = 0700  
directory mask = 0700  
valid users = harmonyos  
force user = harmonyos  
force group = harmonyos  
public = yes  
available = yes  
writable = yes
```

重启 samba 服务器。

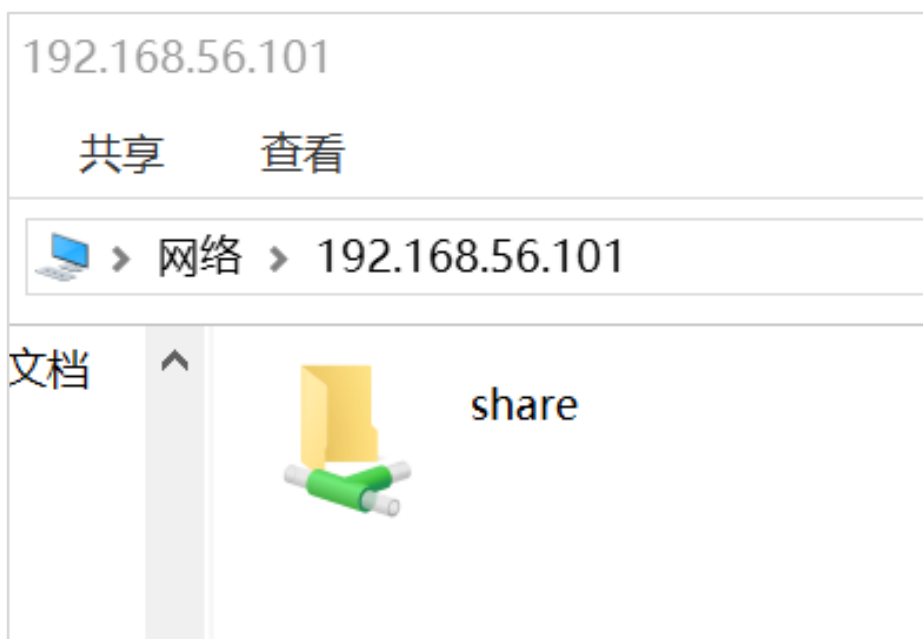
```
sudo service smbd restart
```

#### 步骤 4 Windows 上创建网络硬盘

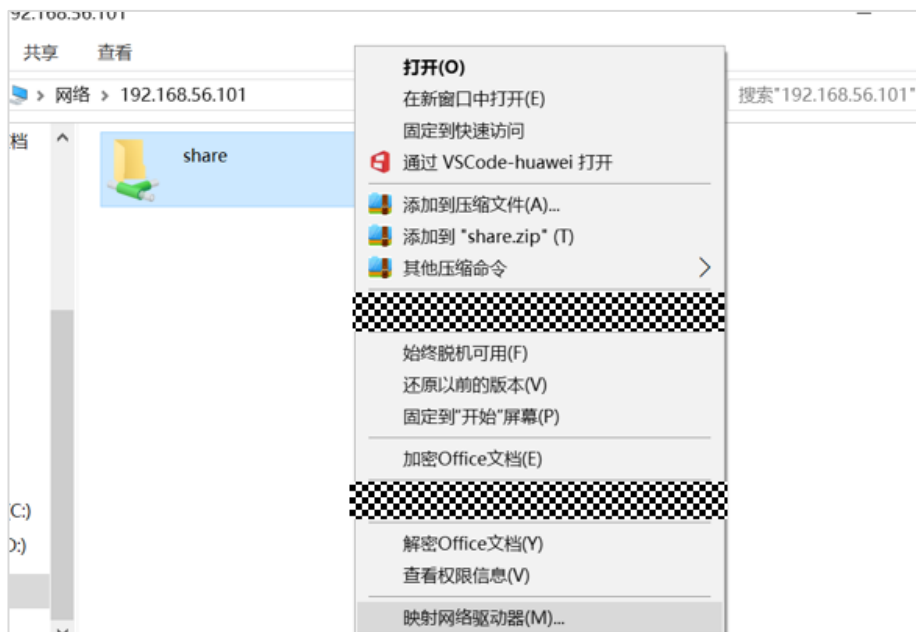
使用快捷键“Win+R”打开“运行”窗口，在弹出的运行窗口中输入虚拟机 IP 地址即可访问；



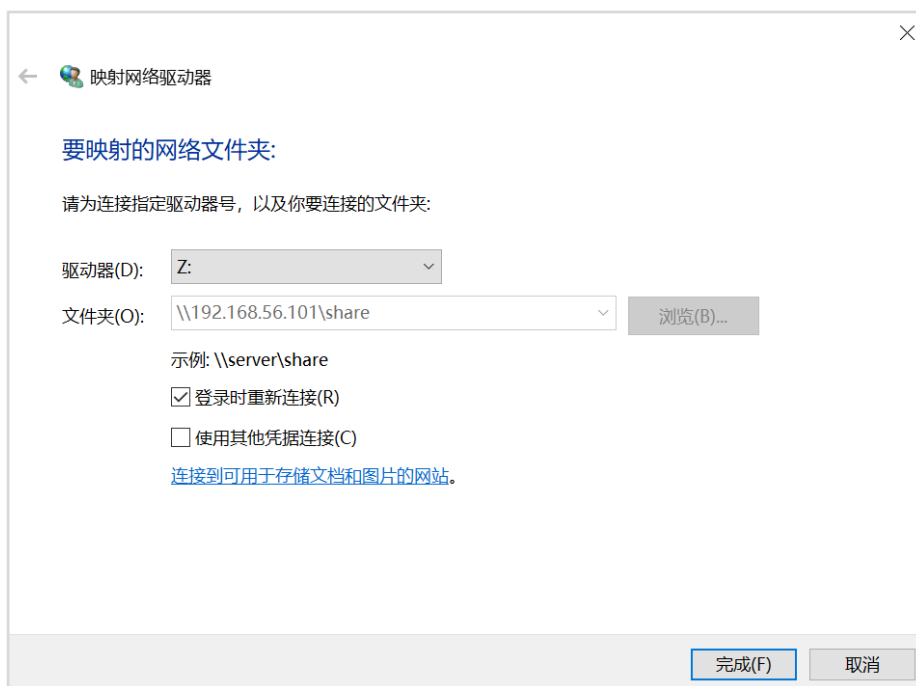
输入 samba 的用户名和密码；



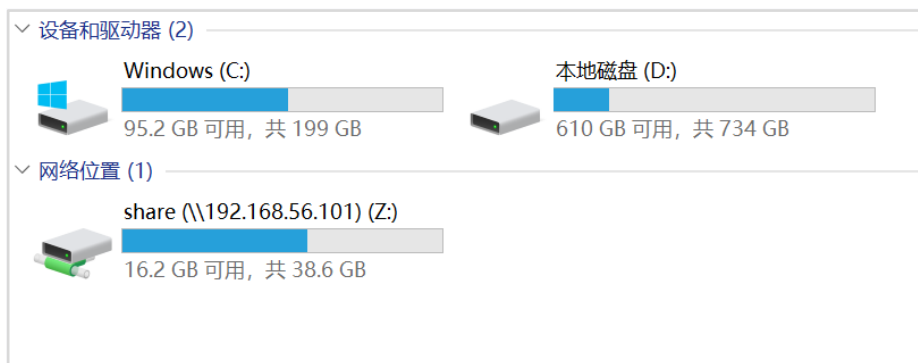
映射网络驱动器；



驱动器 Z 盘，点击完成；



新增网络位置 Z 盘，用于共享 Linux 主机数据，后续将代码存放该磁盘。



## 1.2.5 获取代码（方法一：HPM 获取）

安装 Node.js，推荐安装 Node.js 12.x (包含 npm 6.14.4) 或更高版本 (推荐 12.13.0+)。可先通过 `node -V` 命令先查询是否已经安装 Node.js。

下载 Nodejs 安装包；

```
wget https://nodejs.org/dist/v14.15.1/node-v14.15.1-linux-x64.tar.xz
```

解压安装包；

```
tar -xvf node-v14.15.1-linux-x64.tar.xz -C ~/
```

使用 `ln` 命令来设置软连接；

```
sudo ln -s ~/node-v14.15.1-linux-x64/bin/npm /usr/bin/  
sudo ln -s ~/node-v14.15.1-linux-x64/bin/node /usr/bin/
```

安装 hpm 命令行工具；

```
npm install -g @ohos/hpm-cli  
sudo ln -s ~/node-v14.15.1-linux-x64/bin/hpm /usr/bin
```

在 share 目录下，新建目录 `bearpi-hm_nano`；

```
cd share  
mkdir bearpi-hm_nano  
chmod 777 bearpi-hm_nano
```

进入 `bearpi-hm_nano` 目录下载源码；

```
hpm init -t dist  
hpm i @bearpi/bearpi_hm_nano_hcia
```

耐心等待提示 `successful`，查看当前目录下源码已经下载完成。

如果本地网络不能直接访问 Internet，需要在下载代码前先配置 hpm 代理。

hpm 代理设置参考：[https://device.harmonyos.com/cn/docs/ide/user-guides/hpm\\_proxy-0000001074487706](https://device.harmonyos.com/cn/docs/ide/user-guides/hpm_proxy-0000001074487706)

```
43/50: Installing @ohos/hilog_lite
44/50: Installing @ohos/huks
45/50: Installing @ohos/kv_store
46/50: Installing @ohos/os_dump
47/50: Installing @ohos/utls_base
48/50: Installing @bearpi/hm_nano_paho_mqtt
49/50: Installing @bearpi/hm_nano_iot_link
50/50: Installing @bearpi/bearpi_hm_nano_hcia
sucessful.
harmonyos@harmonyos-VirtualBox:~/share/bearpi-hm_nano$ ll
total 92
drwxrwxrwx 15 harmonyos harmonyos 4096 ./
drwxrwxrwx 20 harmonyos harmonyos 4096 ../
drwxrwxr-x 3 harmonyos harmonyos 4096 applications/
drwxrwxr-x 7 harmonyos harmonyos 4096 base/
drwxrwxr-x 3 harmonyos harmonyos 4096 build/
-rw-rw-r-- 1 harmonyos harmonyos 344 bundle.json
-rw-rw-r-- 1 harmonyos harmonyos 18658 bundle-lock.json
drwxrwxr-x 3 harmonyos harmonyos 4096 device/
drwxrwxr-x 3 harmonyos harmonyos 4096 domains/
drwxrwxr-x 4 harmonyos harmonyos 4096 foundation/
drwxrwxr-x 3 harmonyos harmonyos 4096 kernel/
-rw-rw-r-- 1 harmonyos harmonyos 29 LICENSE
drwxrwxr-x 5 harmonyos harmonyos 4096 ohos_bundles/
drwxrwxr-x 4 harmonyos harmonyos 4096 prebuilts/
-rw-rw-r-- 1 harmonyos harmonyos 150 README.md
drwxrwxr-x 4 harmonyos harmonyos 4096 test/
drwxrwxr-x 16 harmonyos harmonyos 4096 third_party/
drwxrwxr-x 3 harmonyos harmonyos 4096 utls/
drwxrwxr-x 3 harmonyos harmonyos 4096 vendor/
harmonyos@harmonyos-VirtualBox:~/share/bearpi-hm_nano$
```

## 1.2.6 获取代码（方法二：gitee 获取）

安装 git 工具；

```
sudo apt-get install git
```

进入 share 目录下载代码；

```
git clone https://gitee.com/bearpi/bearpi-hm_nano.git -b hcia
```

下载完成；

```
harmonyos@harmonyos-VirtualBox:~/share$ git clone https://gitee.com/bearpi/bearpi-hm_nano.git -b hcia
Cloning into 'bearpi-hm_nano'...
remote: Enumerating objects: 70372, done.
remote: Counting objects: 100% (70372/70372), done.
remote: Compressing objects: 100% (46933/46933), done.
remote: Total 70372 (delta 22332), reused 68080 (delta 20603), pack-reused 0
Receiving objects: 100% (70372/70372), 266.49 MiB | 3.85 MiB/s, done.
Resolving deltas: 100% (22332/22332), done.
Checking out files: 100% (30224/30224), done.
harmonyos@harmonyos-VirtualBox:~/share$ ll
total 12
drwxrwxrwx 3 harmonyos harmonyos 4096 ./
drwxr-xr-x 25 harmonyos harmonyos 4096 ../
drwxrwxr-x 15 harmonyos harmonyos 4096 bearpi-hm_nano/
harmonyos@harmonyos-VirtualBox:~/share$
```

进入 bearpi-hm\_nano，可以查看所有源码。

```

harmonyos@harmonyos-VirtualBox:~/share$ cd bearpi-hm_nano/
harmonyos@harmonyos-VirtualBox:~/share/bearpi-hm_nano$ ll
total 100
drwxrwxr-x 15 harmonyos harmonyos 4096 ./
drwxrwxr-x 3 harmonyos harmonyos 4096 ../
drwxrwxr-x 3 harmonyos harmonyos 4096 applications/
drwxrwxr-x 7 harmonyos harmonyos 4096 base/
drwxrwxr-x 3 harmonyos harmonyos 4096 build/
-rw-rw-r-- 1 harmonyos harmonyos 208 bundle.json
-rw-rw-r-- 1 harmonyos harmonyos 18658 bundle-lock.json
drwxrwxr-x 3 harmonyos harmonyos 4096 device/
drwxrwxr-x 3 harmonyos harmonyos 4096 domains/
drwxrwxr-x 4 harmonyos harmonyos 4096 foundation/
drwxrwxr-x 8 harmonyos harmonyos 4096 .git/
drwxrwxr-x 3 harmonyos harmonyos 4096 kernel/
-rw-rw-r-- 1 harmonyos harmonyos 29 LICENSE
-rw-rw-r-- 1 harmonyos harmonyos 409 ohos_config.json
drwxrwxr-x 4 harmonyos harmonyos 4096 prebuilts/
-rwxrwxr-x 1 harmonyos harmonyos 5820 README.md*
drwxrwxr-x 4 harmonyos harmonyos 4096 test/
drwxrwxr-x 16 harmonyos harmonyos 4096 third_party/
drwxrwxr-x 3 harmonyos harmonyos 4096 utils/
drwxrwxr-x 3 harmonyos harmonyos 4096 vendor/
harmonyos@harmonyos-VirtualBox:~/share/bearpi-hm_nano$

```

## 1.2.7 安装 Linux 编译环境（方法一：Docker 方式）

### 步骤 1 安装 docker

```

harmonyos@harmonyos-VirtualBox:~/share/ohos$ docker

Command 'docker' not found, but can be installed with:

sudo snap install docker      # version 19.03.13, or
sudo apt install docker.io    # version 19.03.8-0ubuntu1.20.04.2

See 'snap info docker' for additional versions.

```

运行如下命令，安装 docker。

```
sudo apt install docker.io
```

```

harmonyos@harmonyos-VirtualBox:~/share/ohos$ sudo apt install docker.io
[sudo] harmonyos 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
下列软件包是自动安装的并且现在不需要了:
  enchant libenchantic2a libffi6 libhunspell-1.6-0 libicu60 libisl19 libncursesw5 libpython3.8-stdlib
  libreadline7 libtinfo5 python python-minimal python2.7 python2.7-minimal
使用 'sudo apt autoremove' 来卸载它(它们)。
将会同时安装下列软件:
  bridge-utils cgroupfs-mount containerd git git-man liberror-perl pigz runc ubuntu-fan
建议安装:
  ifupdown aufs-tools btrfs-progs debootstrap docker-doc rinse zfs-fuse | zfsutils git-c
下列【新】软件包将被安装:
  bridge-utils cgroupfs-mount containerd docker.io git git-man liberror-perl pigz runc
升级了 0 个软件包，新安装了 10 个软件包，要卸载 0 个软件包，有 2 个软件包未被升级。
需要下载 74.8 MB 的归档。
解压后会消耗 372 MB 的额外空间。
您希望继续执行吗？ [Y/n]

```

### 步骤 2 获取 Docker 镜像

从 HuaweiCloud SWR 上直接获取 Docker 镜像进行构建；

```
sudo docker pull swr.cn-south-1.myhuaweicloud.com/openharmony-docker/openharmony-docker:0.0.4
```

在 bearpi-hm\_nano 目录下运行, --name 后面的参数自己命名即可;

```
sudo docker run -it --name ohos -v $(pwd):/home/openharmony swr.cn-south-1.myhuaweicloud.com/openharmony-docker/openharmony-docker:0.0.4
```

编译: 设置编译路径, 选择当前路径, 选择 bearpi\_hm\_nano 并回车;

```
hb set
```

Input code path 选择当前文件夹;

```
[OHOS INFO] Input code path: .
OHOS which product do you need? (Use arrow keys)

bearpi
> bearpi_hm_nano
```

执行编译命令, -f 是全量编译。

```
hb build -f
```

查看编译结果:

```
[OHOS INFO] bearpi_hm_nano build success
```

编译结果文件生成在 out/bearpi\_hm\_nano/bearpi\_hm\_nano/目录下。

若虚拟机被重启过, 需要运行 `sudo docker start ohos -i`, 来启动 docker, 再进行 `hb set`, `hb build -f` 来编译源码。

## 1.2.8 安装 Linux 编译环境 (方法二: 安装包方式)

参考官网文档链接进行安装:

[https://device.harmonyos.com/cn/docs/start/introduce/oem\\_minitinier\\_environment\\_linux-0000001105407498](https://device.harmonyos.com/cn/docs/start/introduce/oem_minitinier_environment_linux-0000001105407498)

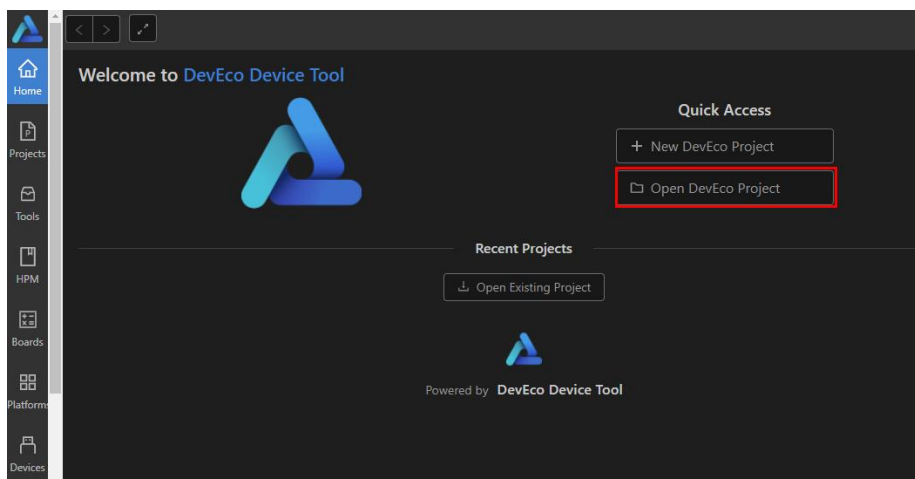
## 1.2.9 安装 Window 开发环境

步骤 1 参考官网文档安装 DevEco Device Tool

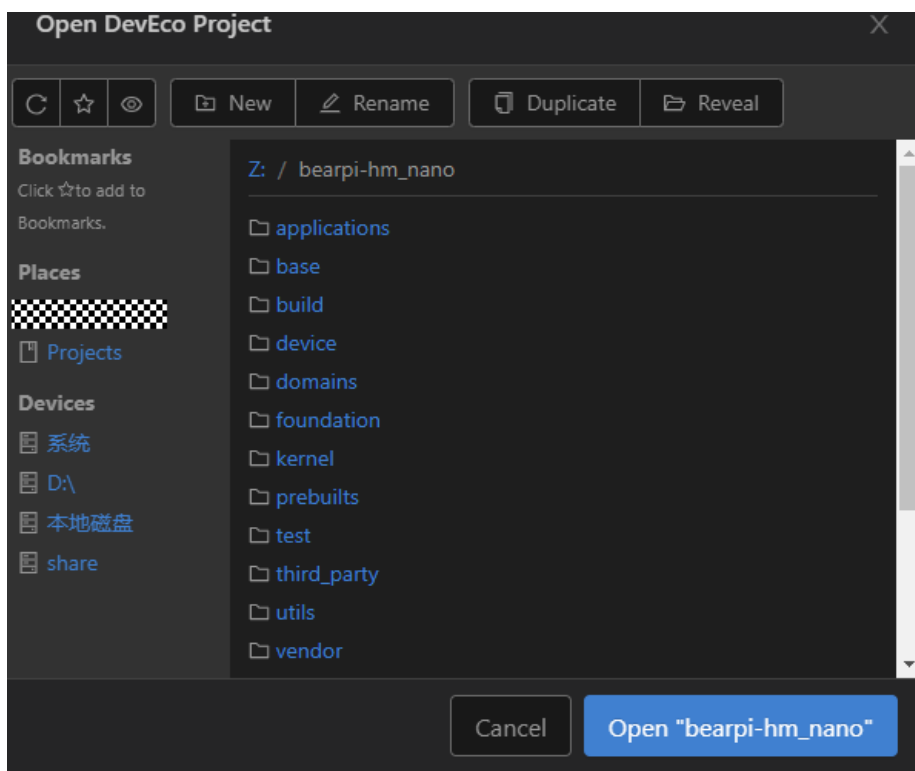
官网文档链接: [https://device.harmonyos.com/cn/docs/ide/user-guides/install\\_windows-0000001050164976](https://device.harmonyos.com/cn/docs/ide/user-guides/install_windows-0000001050164976)

步骤 2 VSCode 打开源码工程

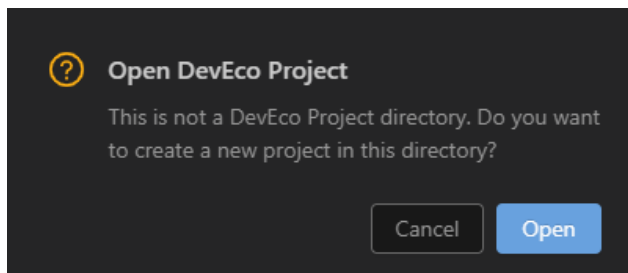
打开 DevEco Device Tool, 进入 Home 页, 点击 Open DevEco Project 打开工程。



选择 bearpi-hm\_nano 目录，点击 Open 打开；



如果打开的 HarmonyOS 源码，则会提示该工程不是 DevEco Device Tool 工程，点击 Open；

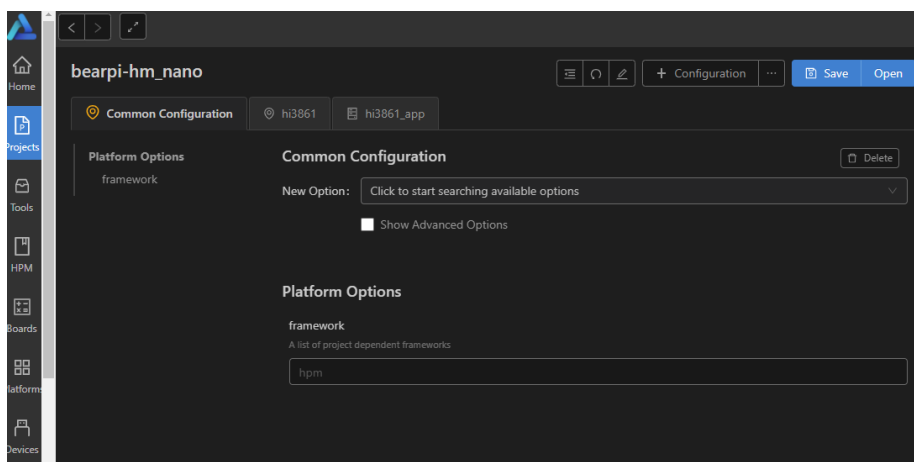


设置开发板类型，Board 选择 “Hi3861”，Framework 选择 “Hpm”，点击 Open；

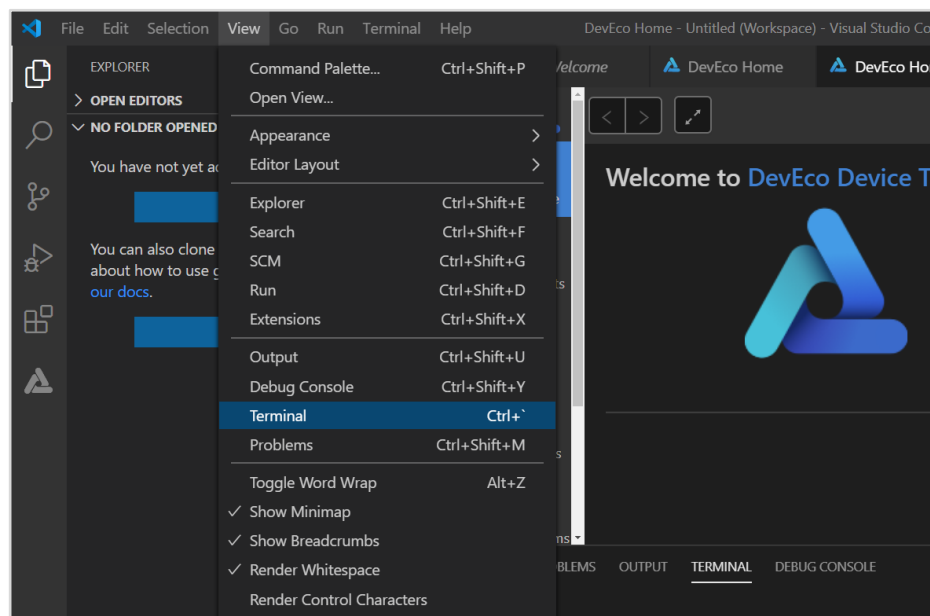




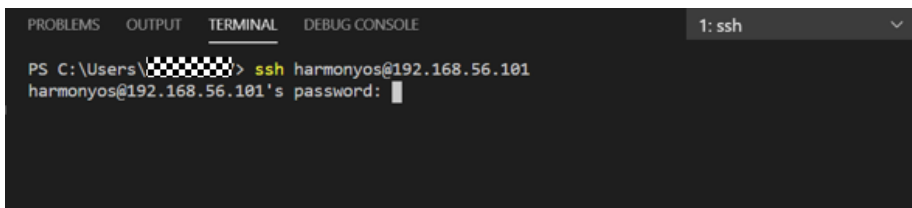
点击右上角 Open，打开工程，即可在 VSCode 中可以查看和编辑源码。



### 步骤 3 SSH 访问 linux 虚拟机



在 terminal 窗口输入 `ssh harmonyos@192.168.56.101`，输入密码。



也可以使用其他工具访问 Linux，可以避免在 Linux 虚拟机界面操作。

在源码目录下执行：

```
hb build -f
```

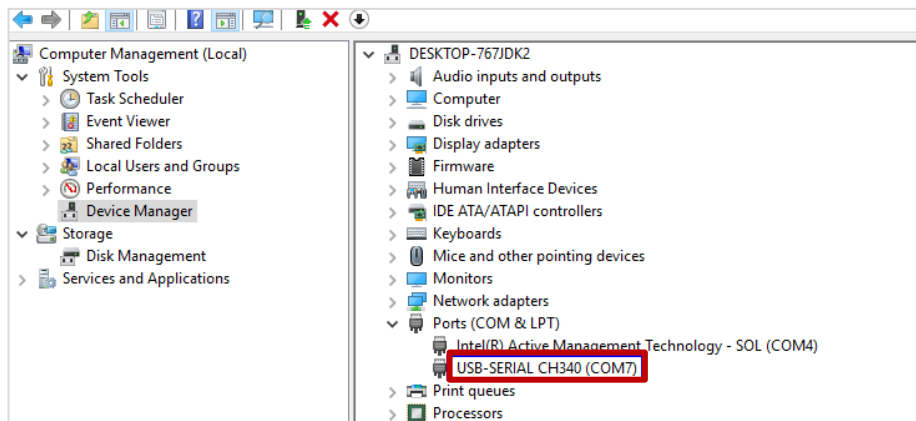
生成 Hi3861\_wifiot\_app\_allinone.bin 烧录文件。

#### 步骤 4 安装 CH340 驱动

安装 CH341SER.EXE USB 转串口驱动。

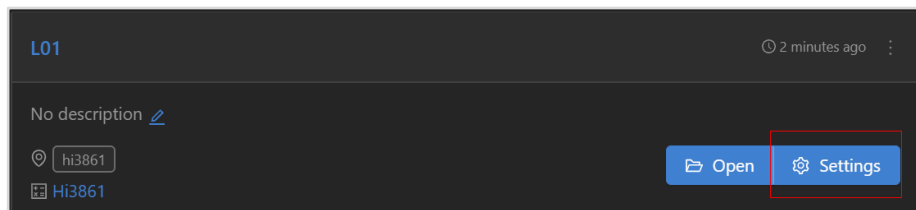
<http://www.wch.cn/search?q=ch340g&t=downloads>

下载安装后，插入开发板 TypeC-USB 线，查看电脑设备管理器，查看并记录对应的串口号。

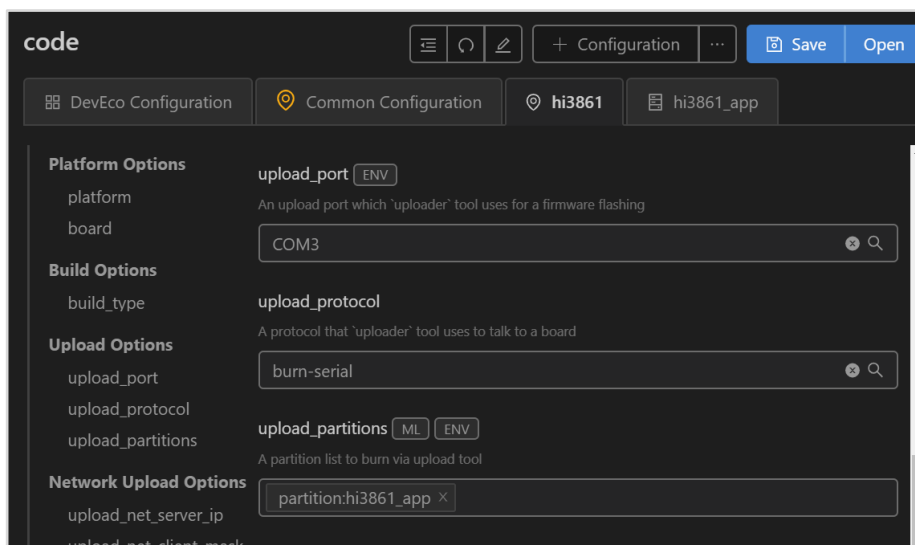


#### 步骤 5 烧录（方法一：DevEco Device Tool）

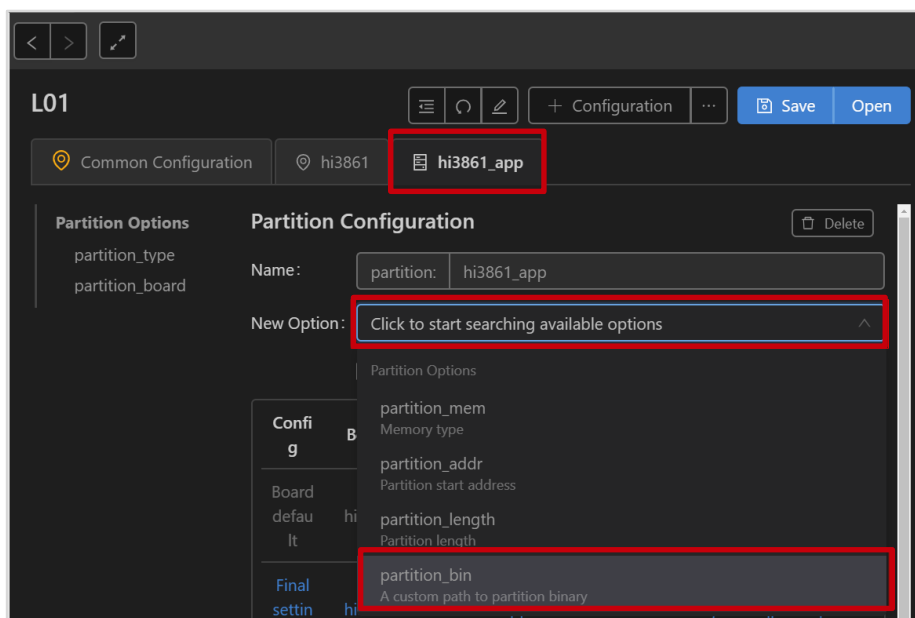
设置工程，在 Projects 中，点击 **Settings** 打开工程配置界面；



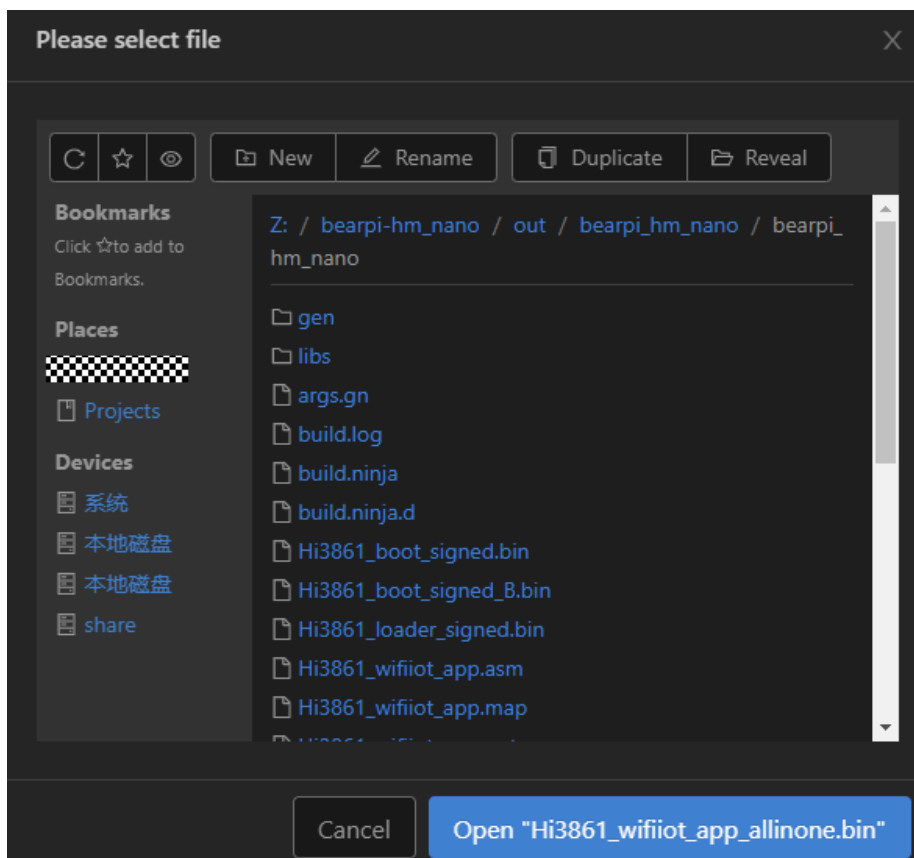
在“hi3861”页签，设置烧录选项，包括 upload\_port、upload\_partitions 和 upload\_protocol；



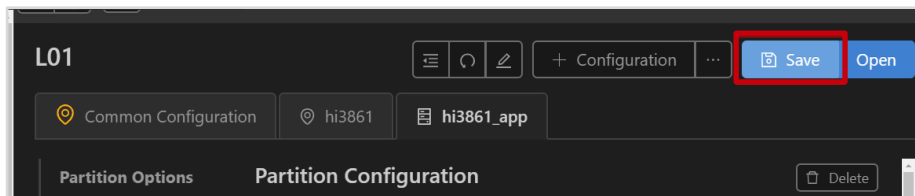
修改待烧录文件 hi3861\_app，请在 New Option 中，选择 partition\_bin 进行更改；



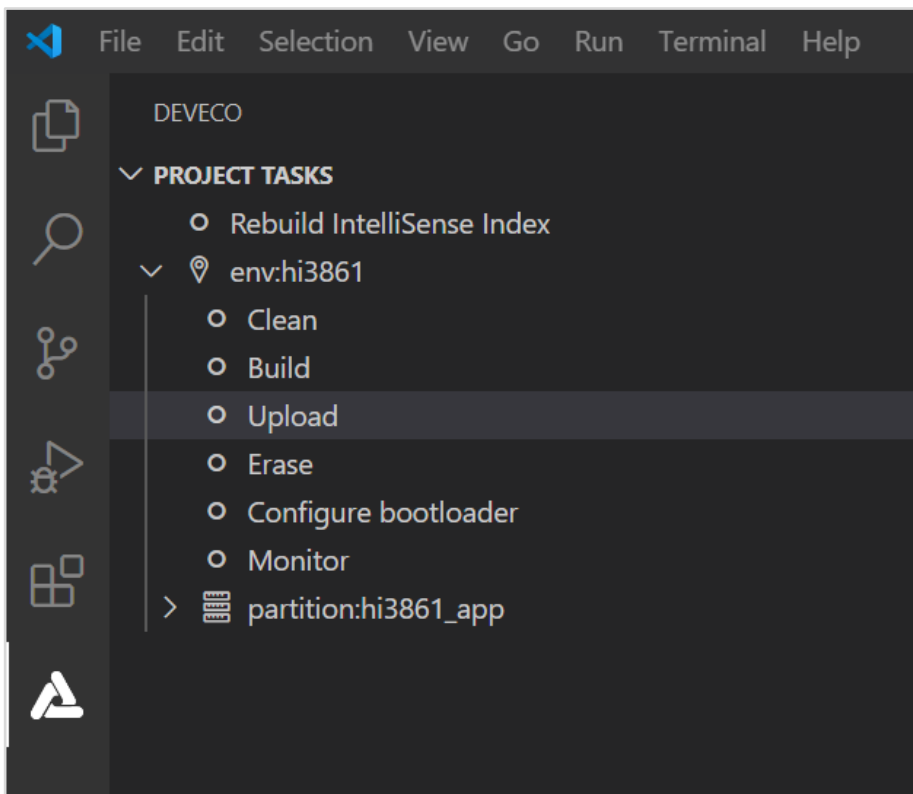
选择待烧录的文件 Hi3861\_wifiot\_app\_allinone.bin；



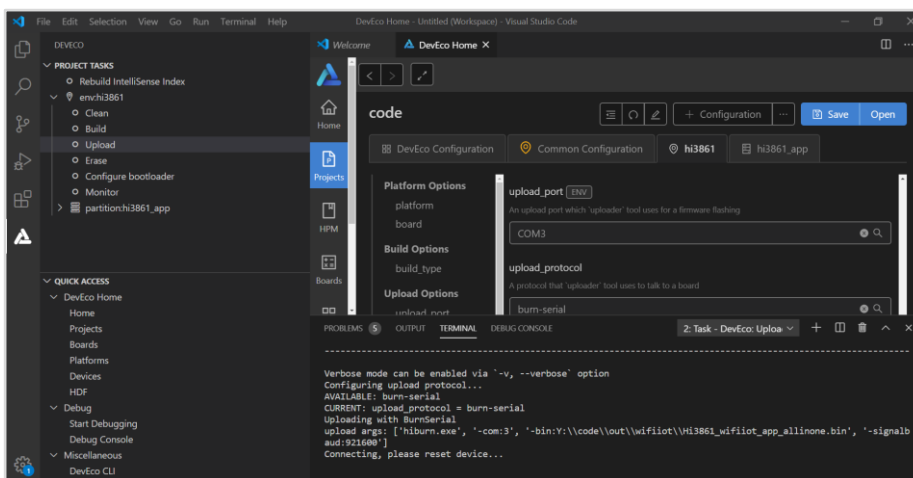
所有的配置都修改完成后，在工程配置页签的顶部，点击 Save 进行保存；



点击 Open 打开工程，点击 DevEco 图标，打开 DevEco Device Tool 界面，在“PROJECT TASKS”中，点击 env:hi3861 下的 Upload 按钮，启动烧录；



启动烧录后，显示如下提示信息时，请按开发板上的 RST 按钮重启开发板；



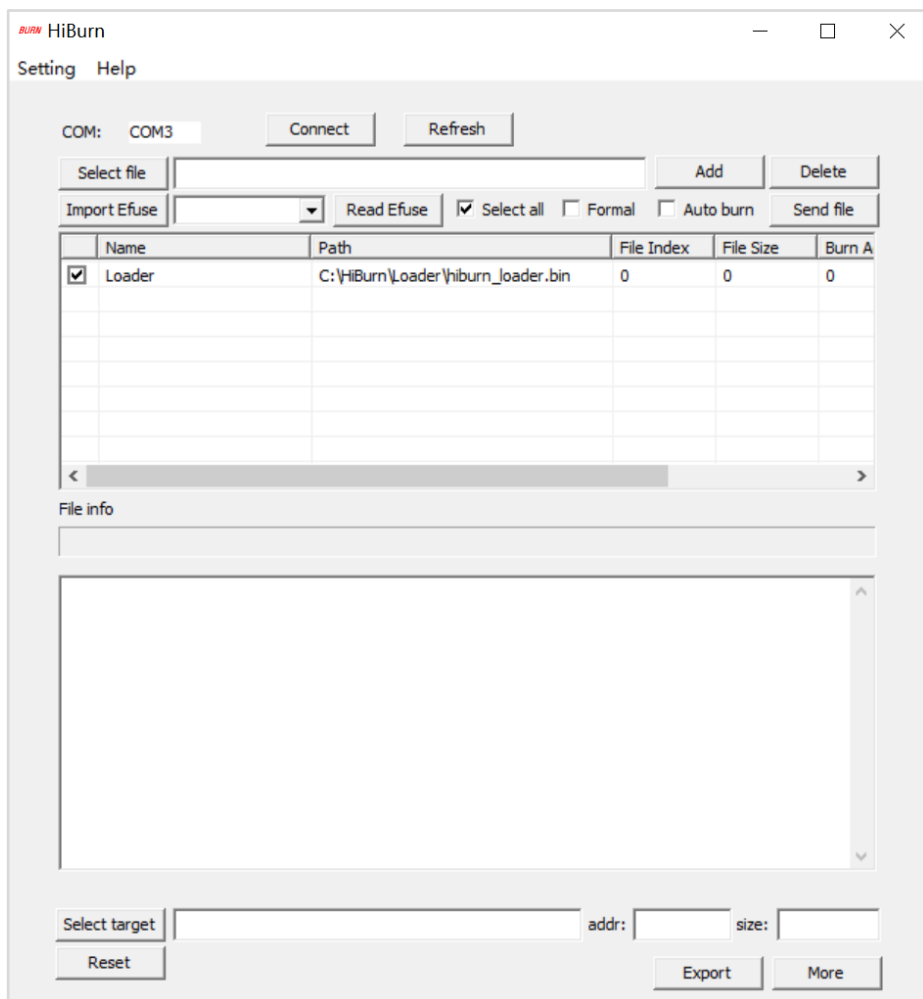
重新上电后，启动烧录，界面提示如下信息时，表示烧录成功。

## 步骤 6 烧录（方法二：Hiburn）

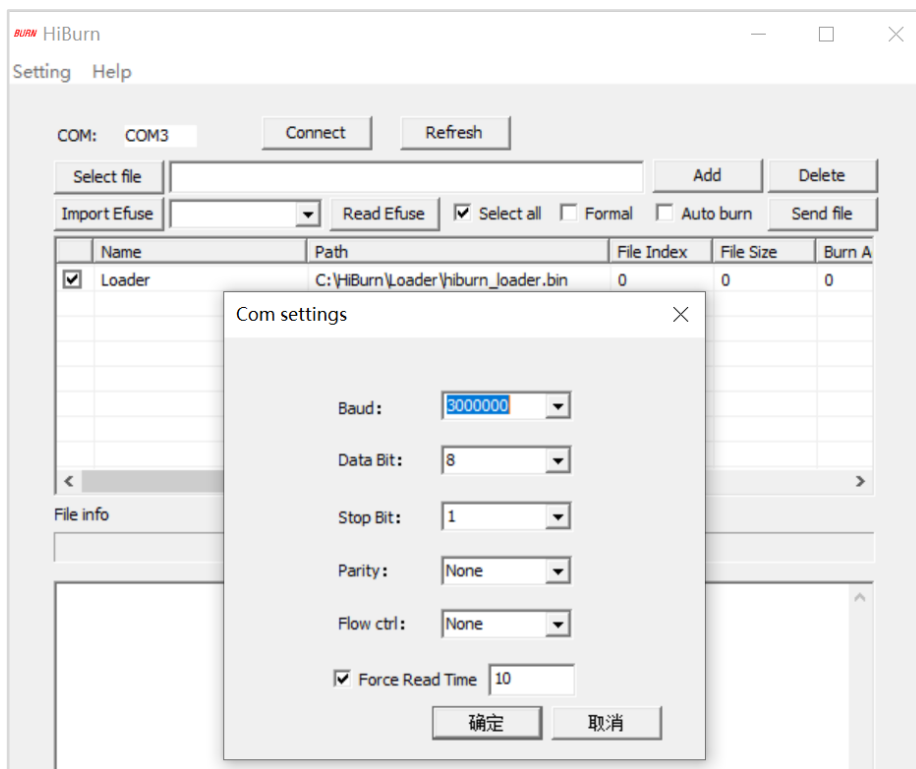
获取 Hiburn 工具。

解压 DevEco Device Tool 提供的.vsix 之后，可以在 extension\deveco\tools 子目录中看到一个文件名为 HiBurn.exe 的文件。

打开 HiBurn；

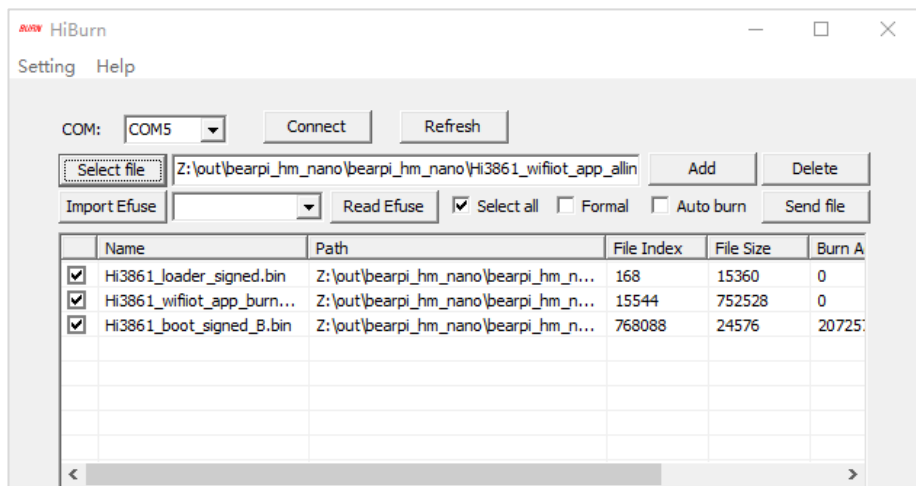


点界面左上角的 Setting->Com settings 进入串口参数设置界面，串口参数设置界面上，Baud 为波特率，默认 115200，可以选择 921600，2000000，或者 3000000（实测最快支持的值），其他参数保持默认，点“确定”保存。

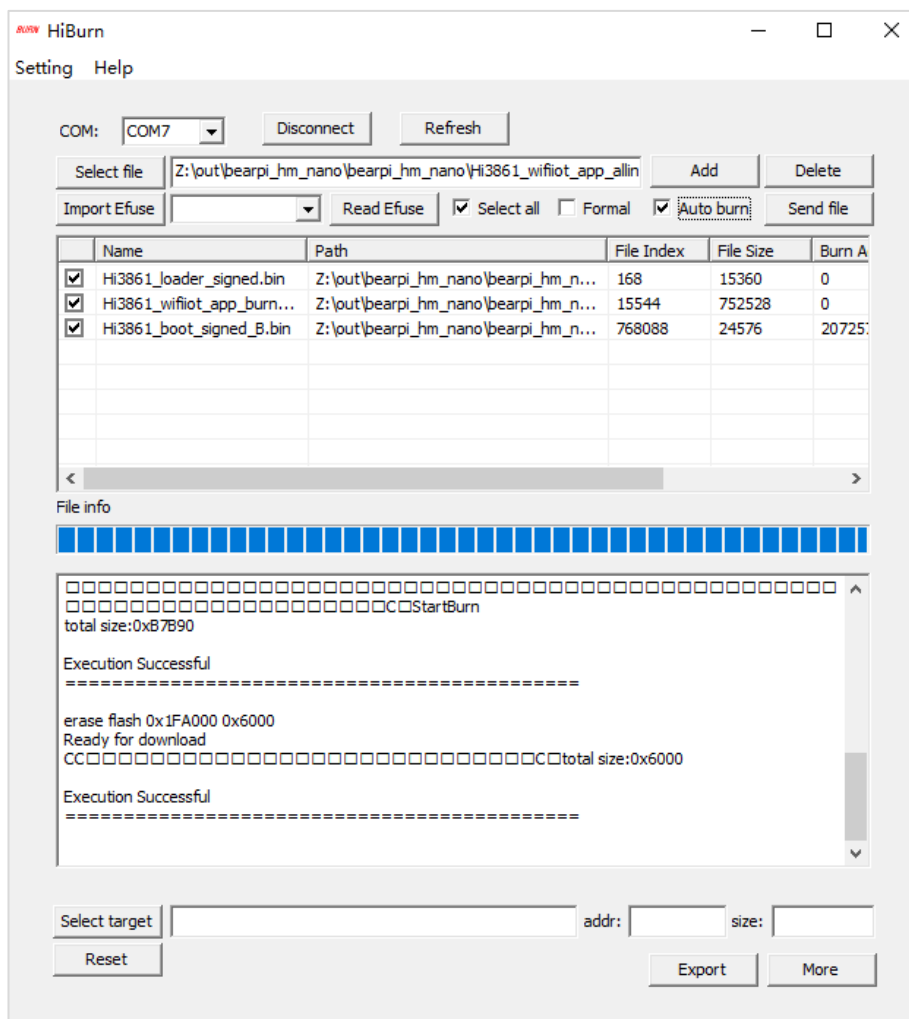


点“Select file”弹出文件选择对话框，选择编译生成的 allinone.bin 文件，这个 bin 其实是多个 bin 合并的文件，从命名上也能看得出来，例如，选择

Z:\harmonyos\openharmony\out\bearpi\_hm\_nano\bearpi\_hm\_nano\Hi3861\_wifiot\_app\_allinone.bin。勾选“Auto burn”，自动下载多个 bin 文件，到这里，配置完毕，应如下图：



点击 Connect，连接串口设备，这时 HiBurn 会打开串口设备，并尝试开始烧写，需要确保没有其他程序占用串口设备（烧写之前可能正在用超级终端或串口助手查看串口日志，需要确保其他软件已经关闭了当前使用的串口），复位设备，按开发板的 RESET 按键；等待输出框出现三个“=====”以及上方均出现 successful，即说明烧录成功。



烧录成功后，需要手动点“Disconnect”断开串口连接，否则会提示“Wait connect success flag (hisilicon) overtime.”。

## 1.3 思考题

1、HarmonyOS 的编译环境是：

Ubuntu，是基于 Linux 的操作系统，在该环境中，进行 HarmonyOS 源码的编译。

可以使用 Docker 的方式，一步到位。也推荐可以使用安装包的方式，一步一步的安装交叉编译工具链，来达到编译源码的目的。

推荐 Docker 方式，安装包的方式感兴趣的可以自行搭建。