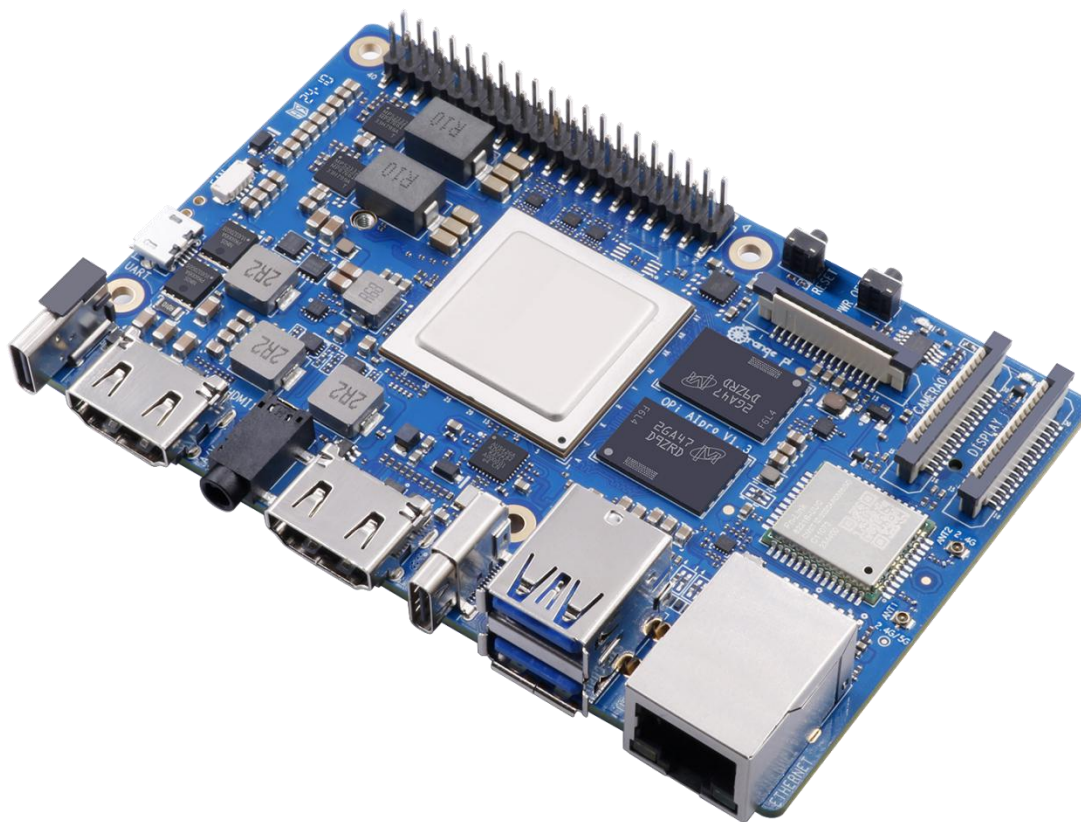


# Orange Pi AI Pro 用户手册





# 目录

1. 开发板参数介绍 .....	1
1.1. 开发板简介 .....	1
1.2. 开发板的硬件规格 .....	1
1.3. 开发板的顶层视图和底层视图 .....	3
1.4. 开发板的接口详情图 .....	4
2. 开发板使用介绍 .....	5
2.1. 准备需要的配件 .....	5
2.2. 下载开发板的镜像和相关的资料 .....	9
2.3. 控制启动设备的两个拨码开关的使用说明 .....	9
2.4. 烧写 Linux 镜像到 TF 卡中的方法 .....	10
2.4.1. 基于 Windows PC 将 Linux 镜像烧写到 TF 卡的方法 .....	10
2.4.2. 基于 Ubuntu PC 将 Linux 镜像烧写到 TF 卡的方法 .....	14
2.5. 烧写 Linux 镜像到 eMMC 中的方法 .....	18
2.6. 烧写 Linux 镜像到 NVMe SSD 中的方法 .....	23
2.7. 烧写 Linux 镜像到 SATA SSD 中的方法 .....	29
2.8. 启动开发板的步骤 .....	35
2.9. 调试串口的使用方法 .....	36
2.9.1. 通过 Micro USB 接口来使用调试串口的连接说明 .....	37
2.9.2. 通过 40 pin 接口中的 uart0 来使用调试串口的连接说明 .....	37
2.9.3. Ubuntu 平台调试串口的使用方法 .....	38
2.9.4. Windows 平台调试串口的使用方法 .....	42
3. Ubuntu Xfce 桌面系统使用说明 .....	45
3.1. 已支持的 Ubuntu 镜像类型和内核版本 .....	45
3.2. Linux 系统功能适配情况 .....	45
3.3. Linux 系统登录说明 .....	46
3.3.1. 登录 Linux 系统桌面的方法 .....	46



3.3.2. Linux 系统默认登录账号和密码 .....	48
3.4. 板载 LED 灯测试说明 .....	48
3.5. 网络连接测试 .....	49
3.5.1. 以太网口测试 .....	49
3.5.2. WIFI 连接测试 .....	50
3.5.3. 设置静态 IP 地址的方法 .....	56
3.6. SSH 远程登录开发板 .....	63
3.6.1. Ubuntu 下 SSH 远程登录开发板 .....	63
3.6.2. Windows 下 SSH 远程登录开发板 .....	64
3.7. HDMI 接口的使用说明 .....	65
3.7.1. HDMI 显示 Linux 桌面的说明 .....	65
3.7.2. 使用 HDMI 接口显示图片和播放音频的方法 .....	66
3.8. 蓝牙使用方法 .....	67
3.9. USB 接口测试 .....	70
3.9.1. Type-C USB 接口使用注意事项 .....	70
3.9.2. 连接 USB 鼠标或键盘测试 .....	70
3.9.3. USB 摄像头测试 .....	71
3.9.4. USB 音频测试 .....	72
3.10. 音频测试 .....	74
3.10.1. 耳机接口播放音频测试 .....	74
3.10.2. HDMI 音频播放测试 .....	75
3.10.3. 耳机 MIC 录音测试 .....	75
3.11. MIPI LCD 屏幕的测试说明 .....	76
3.11.1. 树莓派 5 寸屏幕的组装方法 .....	76
3.11.2. 输出图片到 LCD 屏幕的方法 .....	77
3.12. MIPI 摄像头的测试说明 .....	78
3.13. 40 Pin 接口引脚功能说明 .....	80
3.14. 40 pin 接口 GPIO、I2C、UART、SPI 和 PWM 测试 .....	81
3.14.1. 40 pin GPIO 口的测试方法 .....	81
3.14.2. 40 pin SPI 回环测试 .....	84
3.14.3. 40 pin I2C 测试 .....	86



3. 14. 4. 40 pin UART 测试 .....	87
3. 14. 5. 40 pin PWM 测试 .....	89
3. 15. 上传文件到开发板 Linux 系统中的方法 .....	91
3. 15. 1. 在 Ubuntu PC 中上传文件到开发板 Linux 系统中的方法 .....	91
3. 15. 2. 在 Windows PC 中上传文件到开发板 Linux 系统中的方法 .....	95
3. 16. 散热风扇的使用方法 .....	99
3. 17. AI CPU 和 control CPU 的设置方法 .....	100
3. 18. 设置 Swap 内存的方法 .....	101
3. 19. 测试 MindSpore 的方法 .....	102
3. 20. 关机和重启开发板的方法 .....	103
4. 体验 AI 应用样例 .....	104
4. 1. 登录 jupyter lab .....	104
4. 2. 运行目标检测样例 .....	106
4. 3. 运行文字识别样例 .....	115
4. 4. 运行目标分类样例 .....	118
4. 5. 运行图像曝光增强样例 .....	123
4. 6. 运行卡通图像生成样例 .....	127
4. 7. 运行蛋白质分类评估样例 .....	131
4. 8. 运行细胞图像分割样例 .....	136
4. 9. 运行人像分割与背景替换样例 .....	140
4. 10. 运行语音识别样例 .....	144
5. Linux 内核源码包的使用说明 .....	149
5. 1. 编译主机系统的需求 .....	149
5. 2. 安装交叉编译工具链和依赖包 .....	150
5. 3. 下载解压 Linux 内核源码包 .....	152
5. 4. 编译并生效内核 Image 文件的方法 .....	153
5. 5. 编译并生效内核 DTB 文件的方法 .....	155

---

6. Linux 镜像编译脚本的使用说明 .....	157
6.1. 编译主机系统的需求 .....	157
6.2. 制作 Linux 镜像需要准备的东西 .....	158
6.3. 下载 Linux 镜像编译脚本的源码压缩包 .....	159
6.4. 制作最小镜像的方法 .....	160
6.5. 制作完整镜像的方法 .....	163
6.6. 制作压缩扩容镜像的方法 .....	165
7. 附录 .....	168
7.1. 用户手册更新历史 .....	168
7.2. 镜像更新历史 .....	168

# 1. 开发板参数介绍

## 1.1. 开发板简介

Orange Pi AI Pro 开发板是香橙派联合华为精心打造的高性能 AI 开发板，其搭载了昇腾 AI 处理器，可提供 8TOPS INT8 的计算能力，内存提供了 8GB 和 16GB 两种版本。可以实现图像、视频等多种数据分析与推理计算，可广泛用于教育、机器人、无人机等场景。

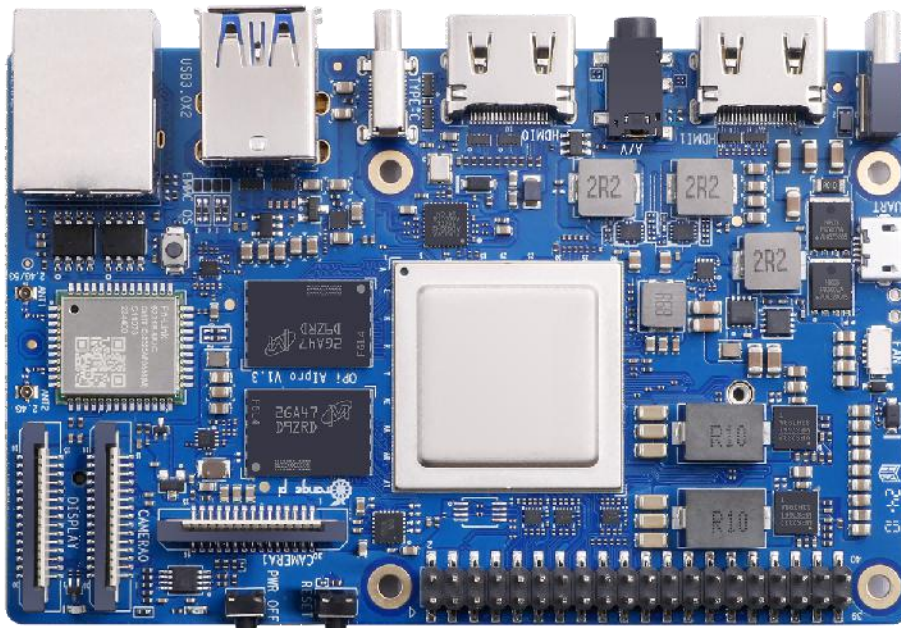
## 1.2. 开发板的硬件规格

Orange AI Pro 开发板硬件规格	
昇腾 AI 处理器	4 核 64 位 Arm 处理器 + AI 处理器
AI 算力	<ul style="list-style-type: none"> <li>半精度 (FP16) : 4 TFLOPS</li> <li>整数精度 (INT8) : 8 TOPS</li> </ul>
内存	<ul style="list-style-type: none"> <li>类型: LPDDR4X</li> <li>容量: 8GB 或 16GB</li> </ul>
存储	<ul style="list-style-type: none"> <li>板载 32MB 的 SPI Flash</li> <li>Micro SD 卡插槽</li> <li>eMMC 插座: 可外接 eMMC 模块</li> <li>M.2 M-Key 接口: 可接 2280 规格的 NVMe SSD 或 SATA SSD</li> </ul>
以太网	<ul style="list-style-type: none"> <li>支持 10/100/1000Mbps</li> <li>板载 PHY 芯片: RTL8211F</li> </ul>
Wi-Fi+蓝牙	<ul style="list-style-type: none"> <li>支持 2.4G 和 5G 双频 WIFI</li> <li>BT4.2</li> <li>模组: 欧智通 6221BUUC</li> </ul>
USB	<ul style="list-style-type: none"> <li>2 个 USB3.0 Host 接口</li> <li>1 个 Type-C 接口 (只支持 USB3.0, 不支持 USB2.0)</li> </ul>
摄像头	2 个 MIPI CSI 2 Lane 接口
显示	<ul style="list-style-type: none"> <li>2 个 HDMI 接口</li> <li>1 个 MIPI DSI 2 Lane 接口</li> </ul>

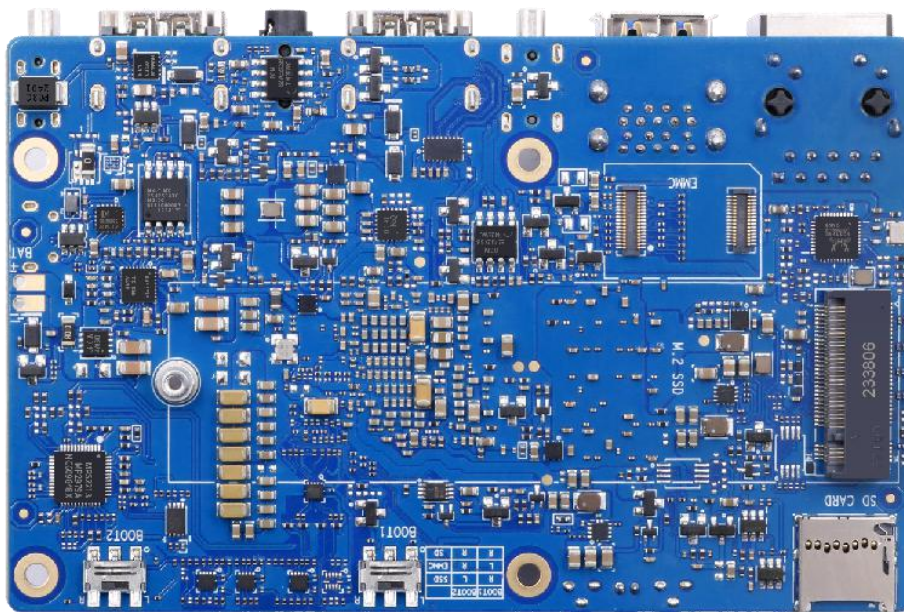
音频	<ul style="list-style-type: none"> <li>• 1 个 3.5mm 耳机孔，支持音频输入输出</li> <li>• 2 个 HDMI 音频输出</li> </ul>
40 pin 扩展口	用于扩展 UART、I2C、SPI、PWM 和 GPIO 等接口
按键	1 个复位键，1 个关机键，1 个升级按键
拨码开关	2 个拨码开关：用于控制 SD 卡、eMMC 和 SSD 启动选项
电源	支持 Type-C 供电，20V PD-65W 适配器
LED 灯	1 个电源指示灯和 1 个软件可控指示灯
风扇接口	4pin, 0.8mm 间距，用于接 12V 风扇，支持 PWM 控制
电池接口	2pin, 2.54mm 间距，用于接 3 串电池，支持快充
调试串口	Micro USB 接口的调试串口
支持的操作系统	Ubuntu 22.04 和 openEuler 22.03
外观规格介绍	
产品尺寸	107*68mm
重量	82g
 rangePi™是深圳市迅龙软件有限公司的注册商标	

### 1.3. 开发板的顶层视图和底层视图

顶层视图:



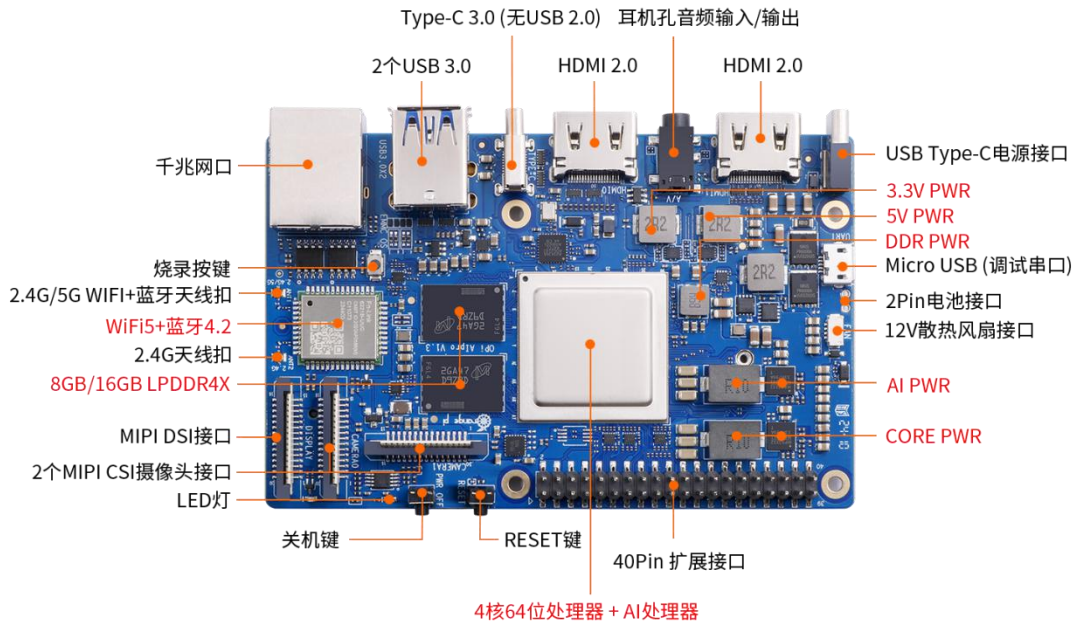
底层视图:



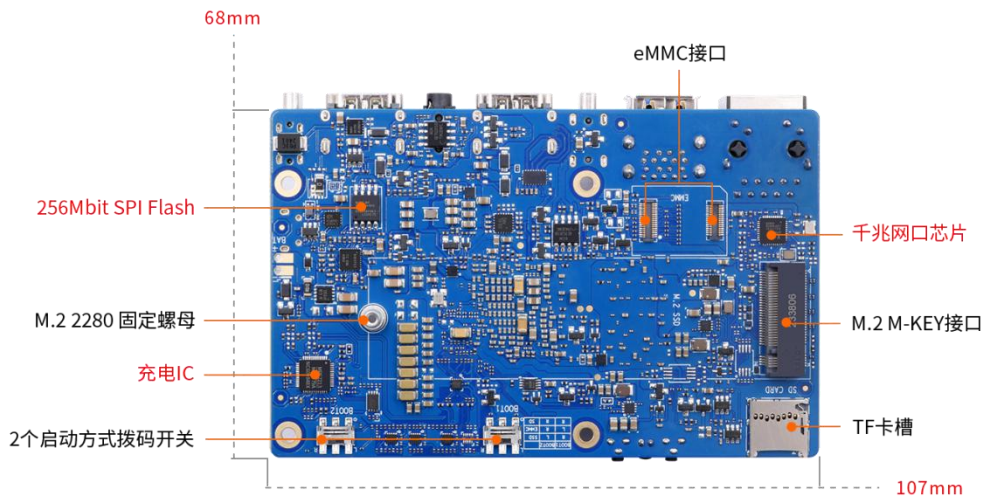


## 1.4. 开发板的接口详情图

顶层视图



底层视图



## 2. 开发板使用介绍

### 2.1. 准备需要的配件

1) TF 卡，最小 32GB 容量的 **class10** 级或以上的高速闪迪卡。强烈推荐使用 64GB 或以上容量的 TF 卡。



2) TF 卡读卡器，用于读写 TF 卡。



3) HDMI 转 HDMI 连接线，用于将开发板连接到 HDMI 显示器或者电视进行显示。



4) Type-C转USB3.0 转接线，用于Type-C接口连接USB3.0 的存储设备。



5) 电源，Type-C 接口的 20V PD-65W 适配器。



6) M.2 M-Key 2280 规格PCIe NVMe SSD。

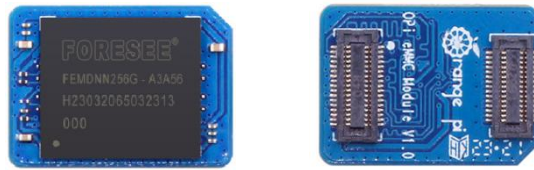
注意，NVMe SSD目前测试了樊想、金士顿和三星的，只有三星的NVMe SSD能稳定运行Linux系统。非三星品牌的NVMe SSD需要等后续软件更新才能正常使用。



7) M.2 M-Key 2280 规格的SATA SSD。



## 8) eMMC模块



## 9) USB接口的鼠标和键盘。



## 10) USB摄像头。



## 11) 金属配套外壳。

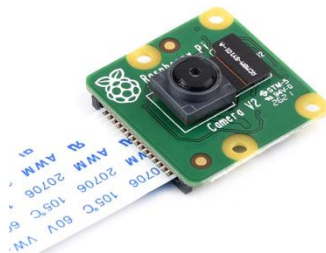


## 12) 百兆或者千兆网线，用于将开发板连接到因特网。



13) 12V 的散热风扇。

14) 树莓派 IMX219 型号的摄像头。



15) 树莓派 5 寸 MIPI LCD 显示屏。



16) Micro USB 接口的数据线，用于串口调试。



17) 安装有 Ubuntu 22.04 和 Windows 操作系统的 X64 电脑。

1	Ubuntu22.04 PC	可选，用于编译 Linux 源码
2	Windows PC	用于烧录 Ubuntu 和 openEuler 镜像

## 2.2. 下载开发板的镜像和相关的资料

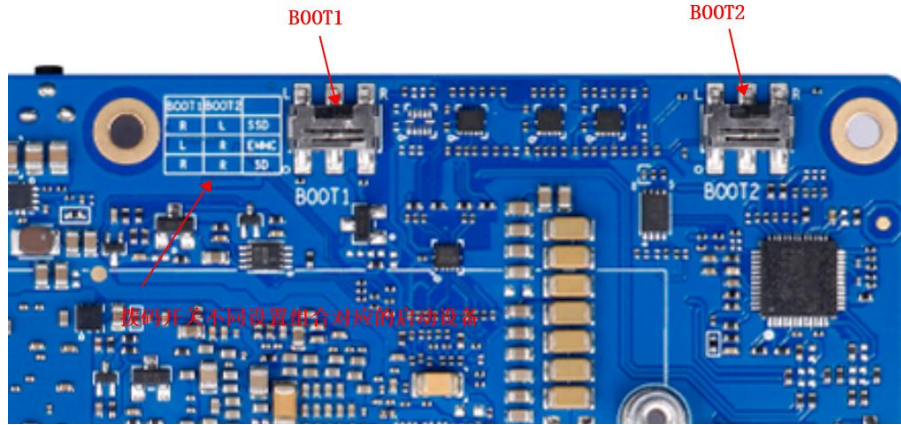
开发板资料下载页面的链接如下所示：

<http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-AIpro.html>



## 2.3. 控制启动设备的两个拨码开关的使用说明

开发板支持从 TF 卡、eMMC 和 SSD（支持 NVMe SSD 和 SATA SSD）启动。具体从哪个设备启动是由开发板背面的两个拨码（BOOT1 和 BOOT2）开关来控制的。



BOOT1 和 BOOT2 两个拨码开关都支持左右两种设置状态，所以总共有 4 种设置状态，开发板目前只使用了其中的三种。不同的设置状态对应的启动设备如下表所示：

拨码开关 BOOT1	拨码开关 BOOT2	对应的启动设备
左	左	未使用
右	左	SATA SSD 和 NVMe SSD
左	右	eMMC
右	右	TF 卡

注意，SATA SSD和NVMe SSD的启动方式对应的拨码开关的设置状态是一样的。这两种启动方式是通过M2\_TYPE引脚的电平来自动区分的。

另外请注意，切换拨码开关后必须重新拔插电源上下电才能让新的启动设备选项生效。通过开发板的复位按键来复位系统是不会让拨码开关新设置的配置生效的。

## 2.4. 烧写 Linux 镜像到 TF 卡中的方法

### 2.4.1. 基于 Windows PC 将 Linux 镜像烧写到 TF 卡的方法

注意，这里说的Linux镜像具体指的是从开发板的资料下载页面下载的Ubuntu或者openEuler镜像。

1) 首先准备一张 32GB 或更大容量的 TF 卡（推荐使用 64GB 或以上容量的 TF

卡)，TF 卡的传输速度必须为 **class10** 级或 **class10** 级以上，建议使用闪迪等品牌的 TF 卡。

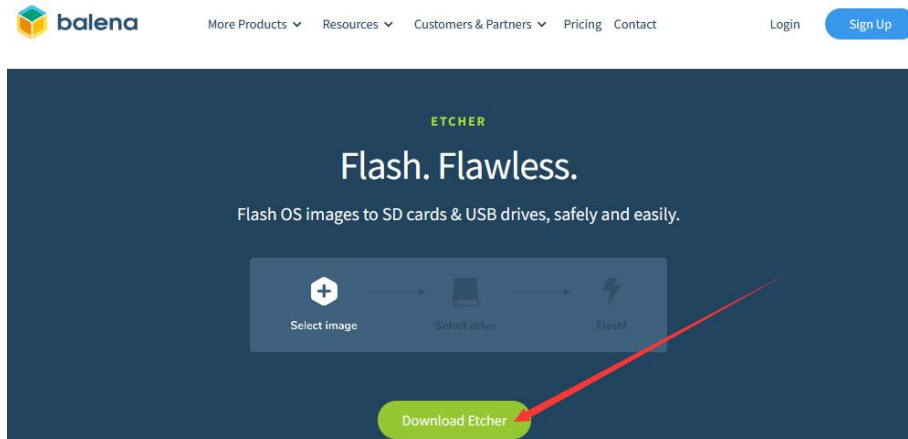
2) 然后把 TF 卡插入读卡器，再把读卡器插入电脑。

3) 从[开发板的资料下载页面](#)下载想要烧录的 Linux 镜像压缩包。

4) 然后下载用于烧录 Linux 镜像的软件——**balenaEtcher**，下载地址为：

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

5) 进入 balenaEtcher 下载页面后，点击绿色的下载按钮会跳到软件下载的地方。

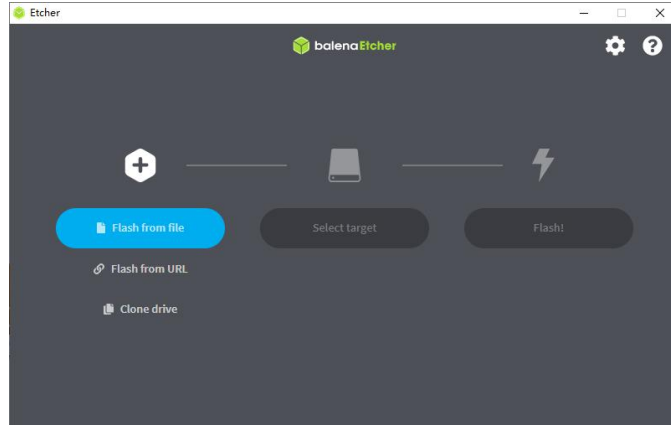


6) 然后可以选择下载 Portable 版本的 balenaEtcher，Portable 版本无需安装，双击打开就可以使用。

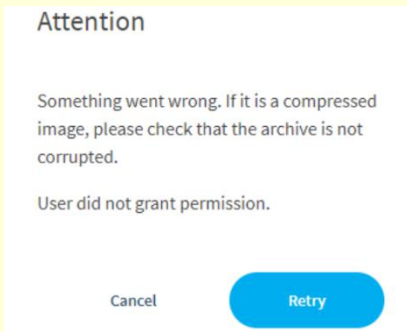


7) 打开后的 balenaEtcher 界面如下图所示：

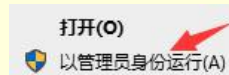




打开 balenaEtcher 时如果提示下面的错误:



请选择 balenaEtcher 后点击右键，然后选择以管理员身份运行。

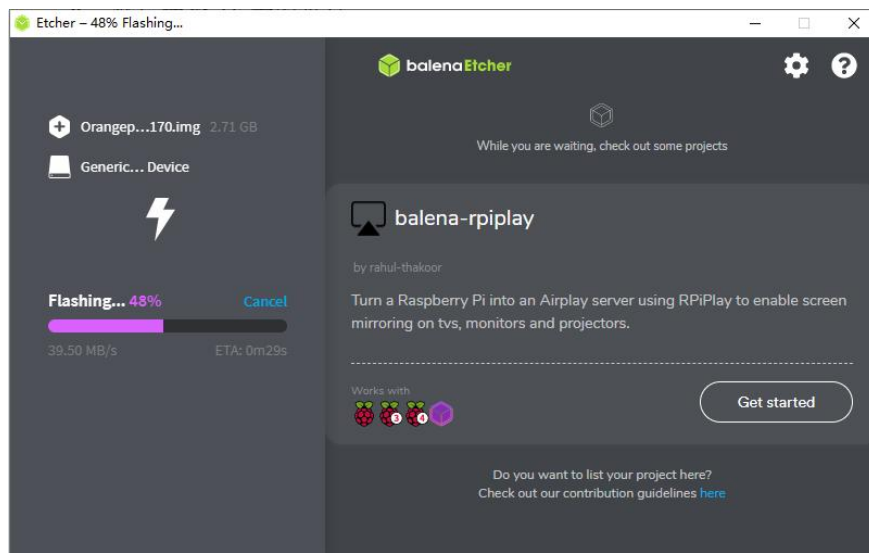


8) 使用 balenaEtcher 烧录 Linux 镜像的具体步骤如下所示:

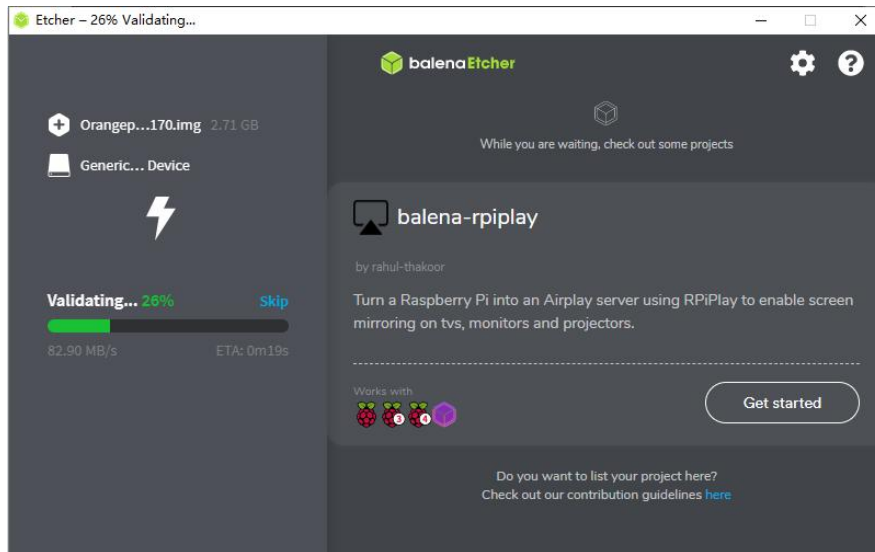
- a. 首先选择要烧录的 Linux 镜像文件的路径。
- b. 然后选择 TF 卡的盘符。
- c. 最后点击 Flash 就会开始烧录 Linux 镜像到 TF 卡中。



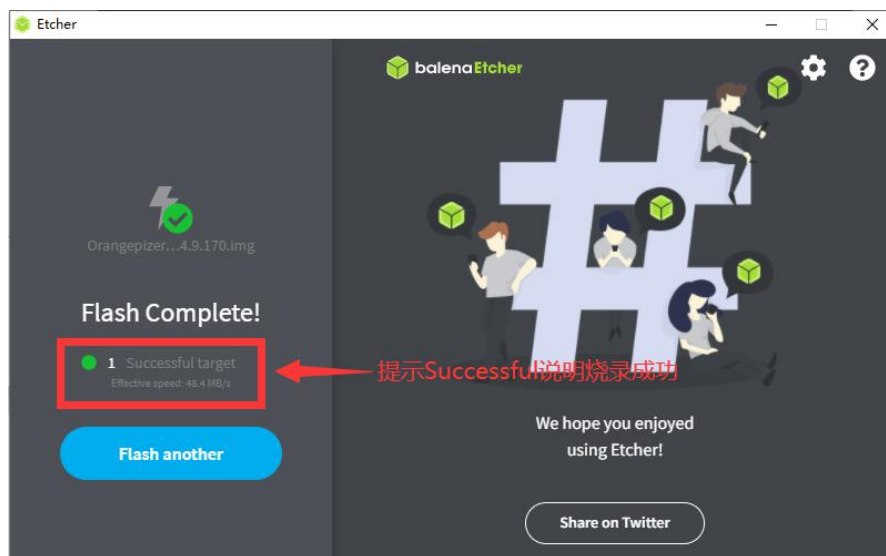
9) balenaEtcher 烧录 Linux 镜像的过程显示的界面如下图所示，另外进度条显示紫色表示正在烧录 Linux 镜像到 TF 卡中。



10) Linux 镜像烧录完后，balenaEtcher 默认还会对烧录到 TF 卡中的镜像进行校验，确保烧录过程没有出问题。如下图所示，显示绿色的进度条就表示镜像已经烧录完成，balenaEtcher 正在对烧录完成的镜像进行校验。



11) 成功烧录完成后 balenaEtcher 的显示界面如下图所示，如果显示绿色的指示图标说明镜像烧录成功，此时就可以退出 balenaEtcher，然后拔出 TF 卡插入到开发板的 TF 卡槽中使用了。



注意，启动系统前请确保拨码开关拨到了TF卡启动的位置了。拨码开关的使用说明请参考控制启动设备的两个拨码开关的使用说明一小节的说明。

### 2.4.2. 基于 Ubuntu PC 将 Linux 镜像烧写到 TF 卡的方法

注意，这里说的Linux镜像具体指的是从开发板的资料下载页面下载的Ubuntu 或者openEuler镜像。

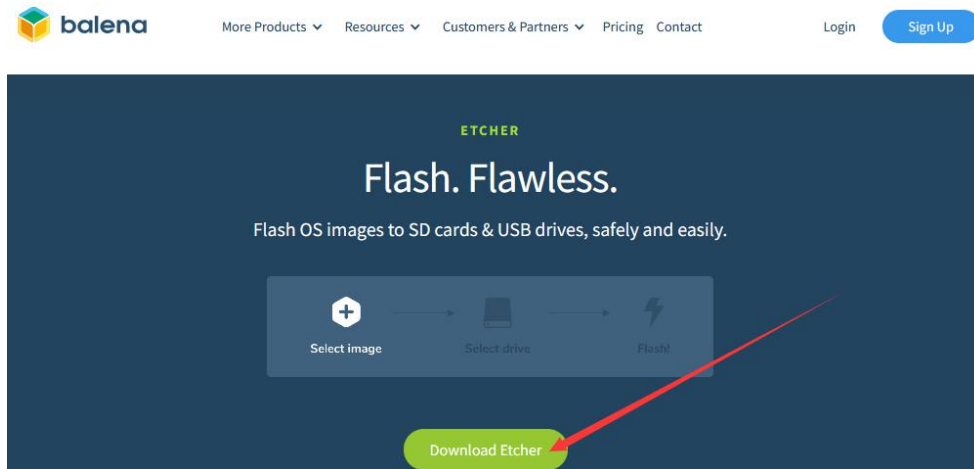
1) 首先准备一张 32GB 或更大容量的 TF 卡（推荐使用 64GB 或以上容量的 TF 卡），TF 卡的传输速度必须为 class10 级或 class10 级以上，建议使用闪迪等品牌的 TF 卡。

2) 然后把 TF 卡插入读卡器，再把读卡器插入电脑。

3) 下载 balenaEtcher 软件，下载地址为：

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

4) 进入 balenaEtcher 下载页面后，点击绿色的下载按钮会跳到软件下载的地方。



5) 然后选择下载 Linux 版本的软件即可。

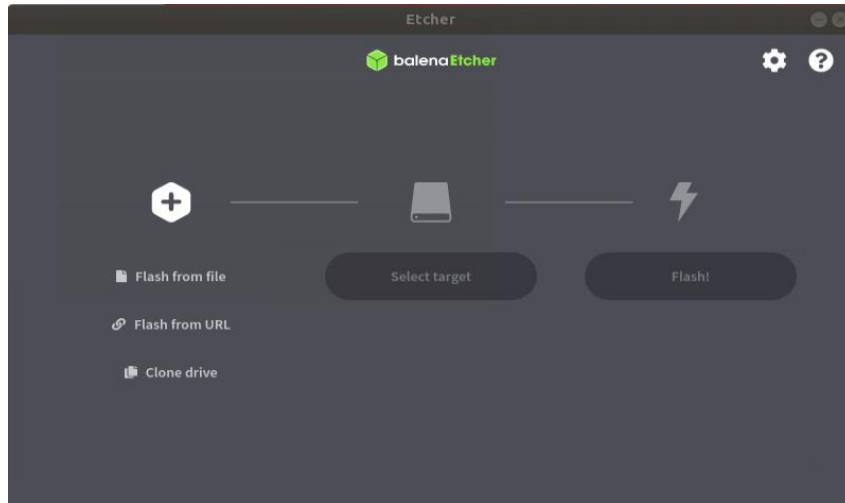
DOWNLOAD

## Download Etcher

ASSET	OS	ARCH	
ETCHER FOR WINDOWS (X86 X64) (INSTALLER)	WINDOWS	X86 X64	<a href="#">Download</a>
ETCHER FOR WINDOWS (X86 X64) (PORTABLE)	WINDOWS	X86 X64	<a href="#">Download</a>
ETCHER FOR WINDOWS (LEGACY 32 BIT) (X86 X64) (PORTABLE)	WINDOWS	X86 X64	<a href="#">Download</a>
ETCHER FOR MACOS	MACOS	X64	<a href="#">Download</a>
ETCHER FOR LINUX X64 (64-BIT) (APPIMAGE)	LINUX	X64	<a href="#">Download</a>
ETCHER FOR LINUX (LEGACY 32 BIT) (APPIMAGE)	LINUX	X86	<a href="#">Download</a>

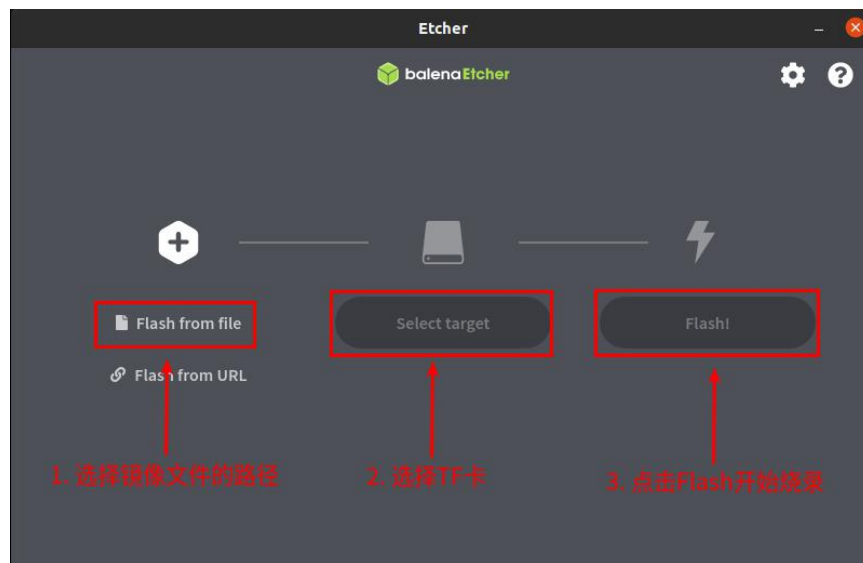
6) 从[开发板的资料下载页面](#)下载想要烧录的 Linux 镜像文件压缩包。

7) 然后在 Ubuntu PC 的图形界面双击 **balenaEtcher-x.x.x-x64.AppImage** 即可打开 balenaEtcher，balenaEtcher 打开后的界面显示如下图所示：

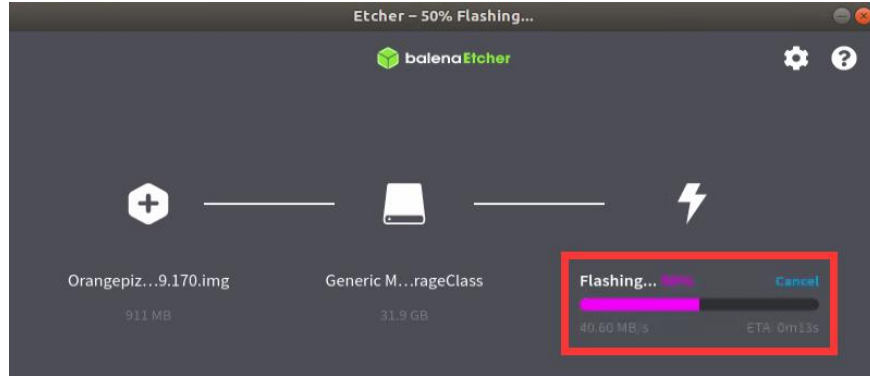


8) 使用 balenaEtcher 烧录 Linux 镜像的具体步骤如下所示：

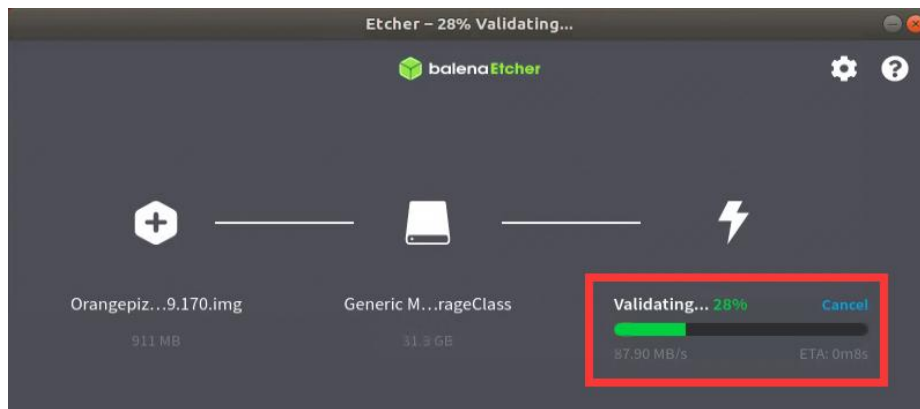
- a. 首先选择要烧录的 Linux 镜像文件的路径。
- b. 然后选择 TF 卡的盘符。
- c. 最后点击 Flash 就会开始烧录 Linux 镜像到 TF 卡中。



9) balenaEtcher 烧录 Linux 镜像的过程显示的界面如下图所示，另外进度条显示紫色表示正在烧录 Linux 镜像到 TF 卡中。



10) Linux 镜像烧录完后，balenaEtcher 默认还会对烧录到 TF 卡中的镜像进行校验，确保烧录过程没有出问题。如下图所示，显示绿色的进度条就表示镜像已经烧录完成，balenaEtcher 正在对烧录完成的镜像进行校验。



11) 成功烧录完成后 balenaEtcher 的显示界面如下图所示，如果显示绿色的指示图标说明镜像烧录成功，此时就可以退出 balenaEtcher，然后拔出 TF 卡插入到开发板的 TF 卡槽中使用了。

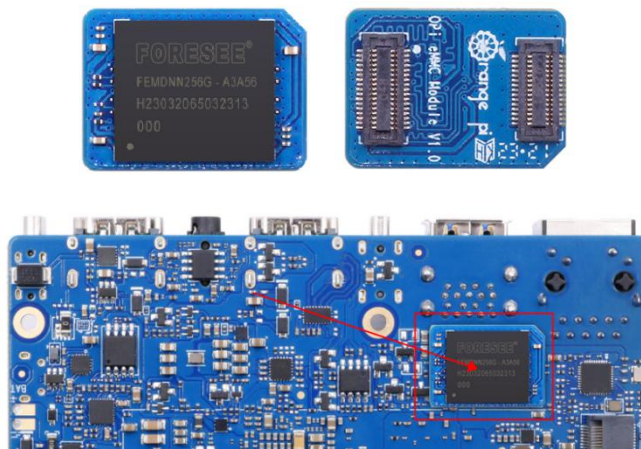


注意，启动系统前请确保拨码开关拨到了TF卡启动的位置了。拨码开关的使用说明请参考控制启动设备的两个拨码开关的使用说明一小节的说明。

## 2.5. 烧写 Linux 镜像到 eMMC 中的方法

注意，这里说的Linux镜像具体指的是从开发板的资料下载页面下载的Ubuntu或者openEuler镜像。

1) 开发板预留了 eMMC 模块的扩展接口，烧录系统到 eMMC 前，需要先购买一个与开发板 eMMC 接口相匹配的 eMMC 模块。然后将 eMMC 模块安装到开发板上。配套的 eMMC 模块和插入开发板的方法如下所示：



2) 烧录 Linux 镜像到 eMMC 中需要借助 TF 卡来完成，所以首先需要将 Linux 镜像烧录到 TF 卡上，然后使用 TF 卡启动开发板进入 Linux 系统。烧录 Linux 镜像到 TF 卡的方法请见[基于 Windows PC 将 Linux 镜像烧写到 TF 卡的方法](#)和[基于 Ubuntu PC 将 Linux 镜像烧写到 TF 卡的方法](#)两小节的说明。

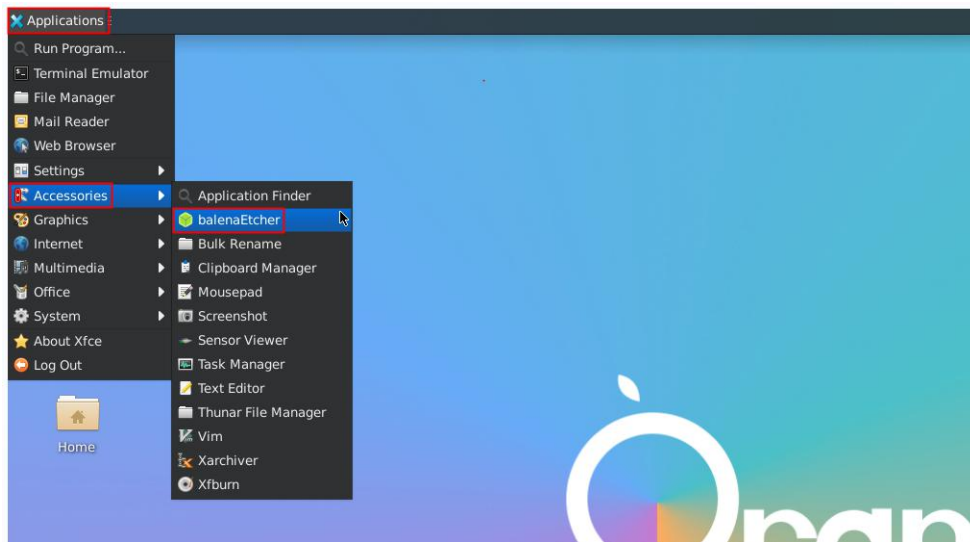
3) 启动进入 TF 卡的 Linux 系统后，请先确认下 eMMC 模块已经被开发板的 Linux 系统正常识别了。如果 eMMC 模块正常识别了的话，在 root 用户下使用 `fdisk -l` 命令就能看到 eMMC 模块的容量信息。

```
(base) root@orangepiaipro:~# fdisk -l
.....
Disk /dev/mmcbk0: 28.91 GiB, 31037849600 bytes, 60620800 sectors
.....
```

4) 然后将要烧录的 Linux 镜像文件压缩包上传到 TF 卡的 Linux 系统中。

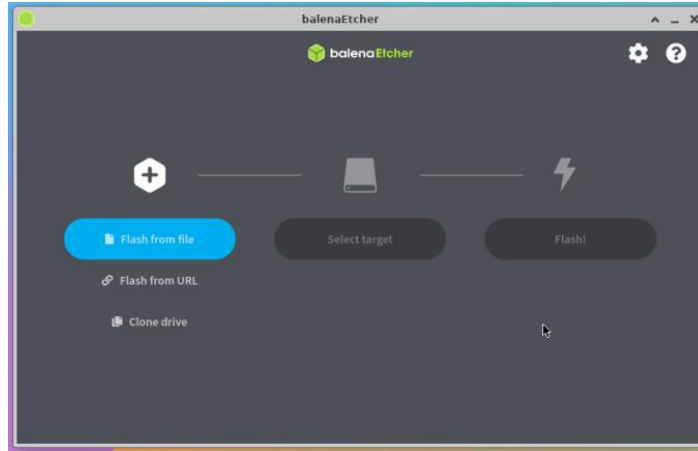
**注意，使用xz格式压缩的Linux镜像压缩包不需要解压，balenaEtcher烧录时会自动解压。**

5) 然后就可以开始使用 balenaEtcher 软件烧录镜像到 eMMC 中了。Linux 系统中已经预装了 balenaEtcher，打开方法如下所示：

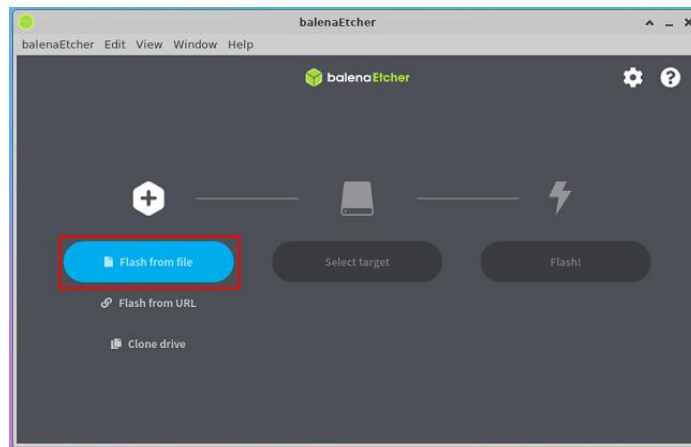


6) balenaEtcher 打开后的界面如下所示：



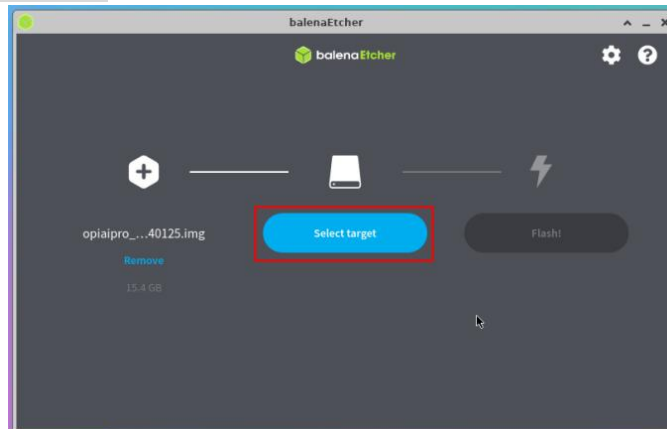


7) 然后点击 **Flash from file** 选择前面上传的想要烧录的镜像文件。

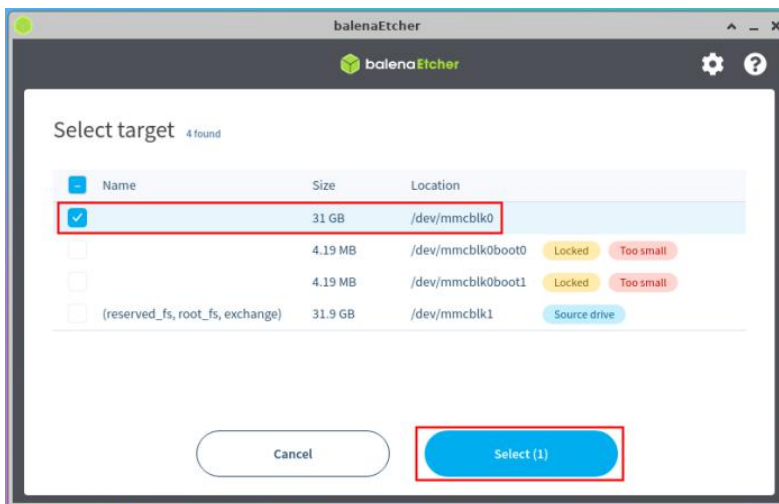


如果打开Linux镜像文件时提示没有权限，请使用 `sudo chmod 777 镜像文件名` 这条命令来给镜像文件添加权限。

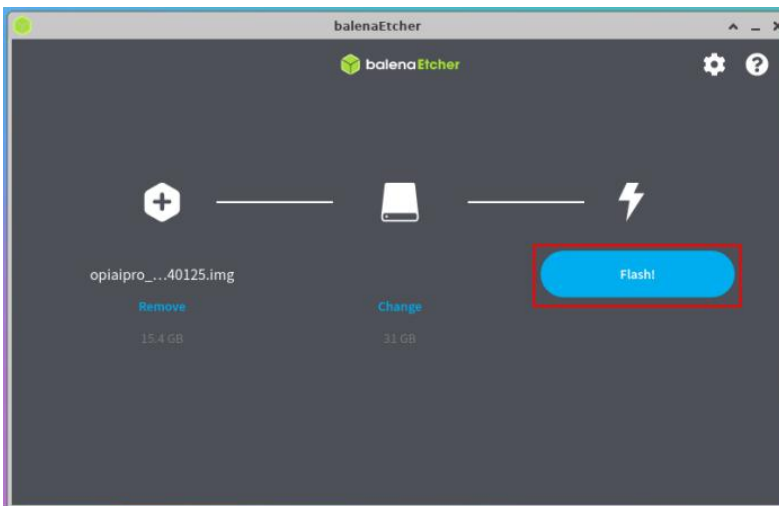
8) 然后点击 **Select target**。



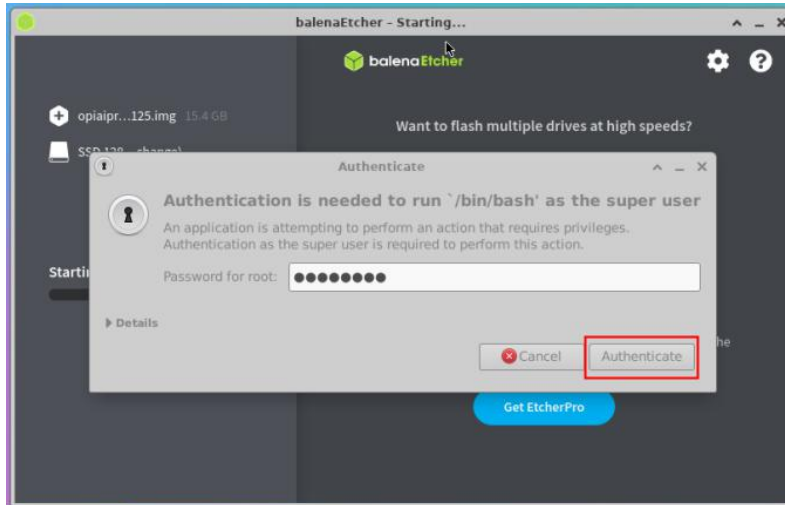
9) 然后选择 eMMC 对应的 `/dev/mmcblk0` 选项，再点击 **Select** 即可。



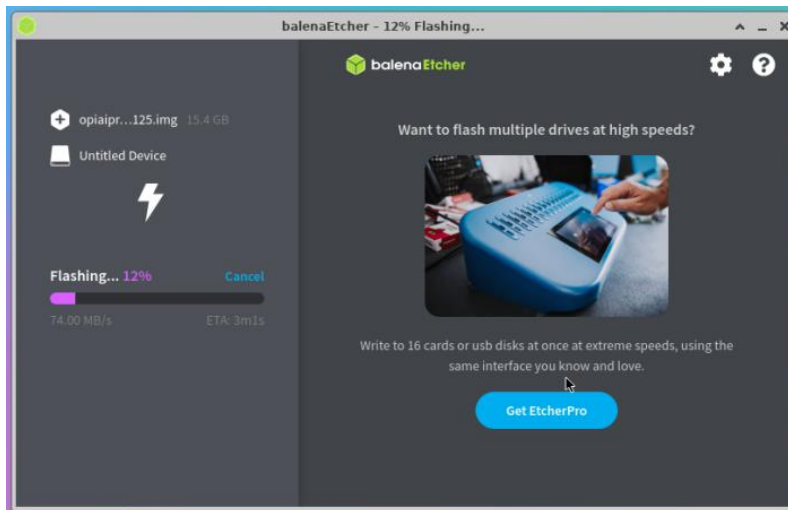
10) 然后点击 **Flash!** 开始烧录。



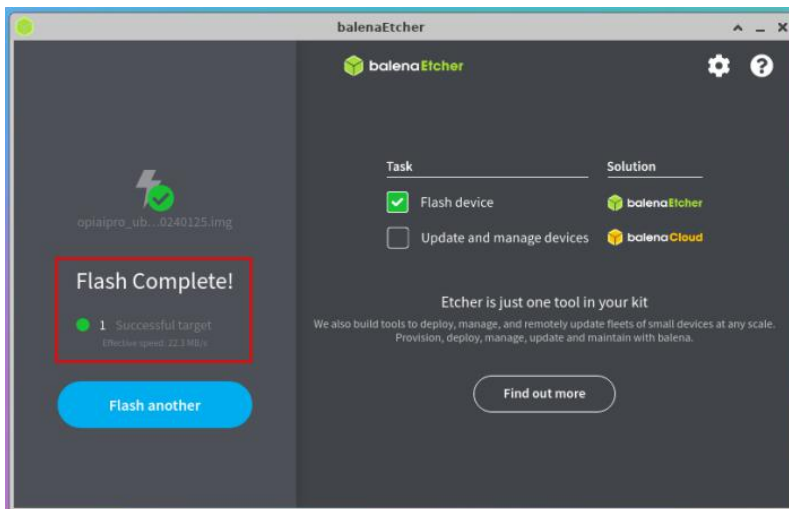
11) 然后输入 Linux 系统的密码: **Mind@123**。



12) 然后就会真正开始烧录 Linux 镜像到 eMMC 中了。



13) Linux 镜像烧录完后的显示如下所示：



14) 此时就可以关闭掉 Linux 系统，然后拔出 TF 卡，并断开 Type-C 电源。再将两个拨码开关拨到 eMMC 启动对应的位置，然后重新插入 Type-C 电源就可以启动 eMMC 中的 Linux 系统了。

注意，启动系统前请确保拨码开关拨到 eMMC 启动的位置了。拨码开关的使用说明请参考[控制启动设备的两个拨码开关的使用说明](#)一小节的说明。

## 2.6. 烧写 Linux 镜像到 NVMe SSD 中的方法

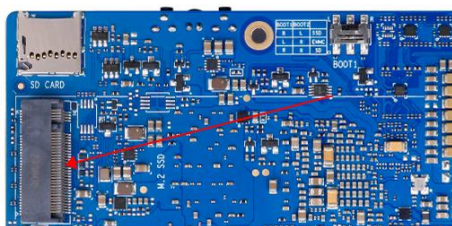
注意，这里说的 Linux 镜像具体指的是从开发板的资料下载页面下载的 Ubuntu 或者 openEuler 镜像。

注意，NVMe SSD 目前测试了 樊想、金士顿和三星的，只有三星的 NVMe SSD 能稳定运行 Linux 系统。非三星品牌的 NVMe SSD 需要等后续软件更新才能正常使用。

1) 首先需要准备一个 2280 规格 NVMe SSD，开发板 M.2 插槽支持的 PCIe 规格为 PCIe3.0x4。PCIe3.0 和 PCIe4.0 的 NVMe SSD 都是可以用的，只是 PCIe4.0 SSD 速度最高只有 PCIe3.0x4 的速度。2242 等其他规格的 SSD 也都是可以使用的，只是没法固定。



2) 然后把 NVMe SSD 插入开发板的 M.2 接口中，并固定好。



3) 烧录 Linux 镜像到 NVMe SSD 中需要借助 TF 卡来完成，所以首先需要将 Linux 镜像烧录到 TF 卡上，然后使用 TF 卡启动开发板进入 Linux 系统。烧录 Linux 镜像到 TF 卡的方法请见[基于 Windows PC 将 Linux 镜像烧写到 TF 卡的方法](#)和[基于 Ubuntu PC 将 Linux 镜像烧写到 TF 卡的方法](#)两小节的说明。

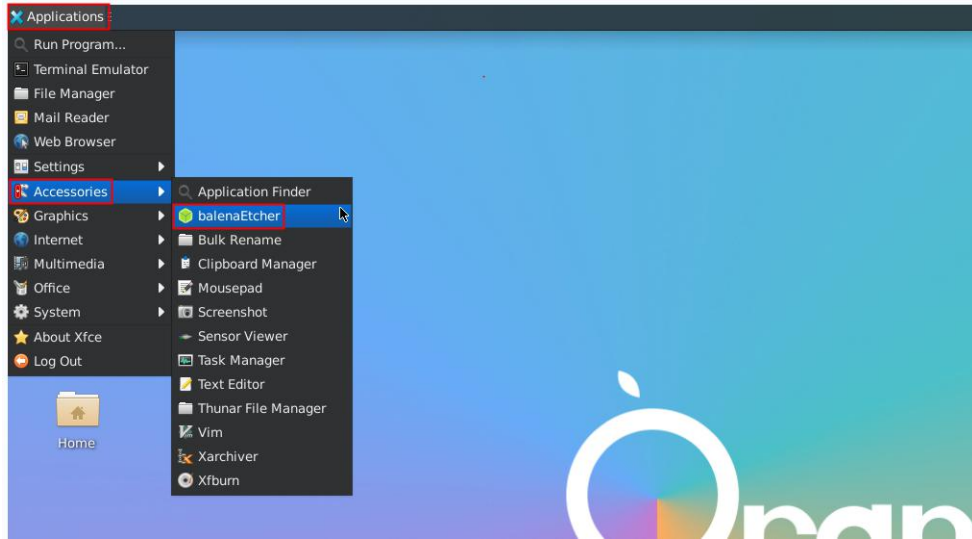
4) 启动进入 TF 卡的 Linux 系统后，请先确认下 NVMe SSD 已经被开发板的 Linux 系统正常识别了。如果 NVMe SSD 正常识别了的话，使用 `sudo fdisk -l` 命令就能看到 `nvme` 相关的信息。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo fdisk -l | grep "nvme0n1"
Disk /dev/nvme0n1: 238.47 GiB, 256060514304 bytes, 500118192 sectors
.....
```

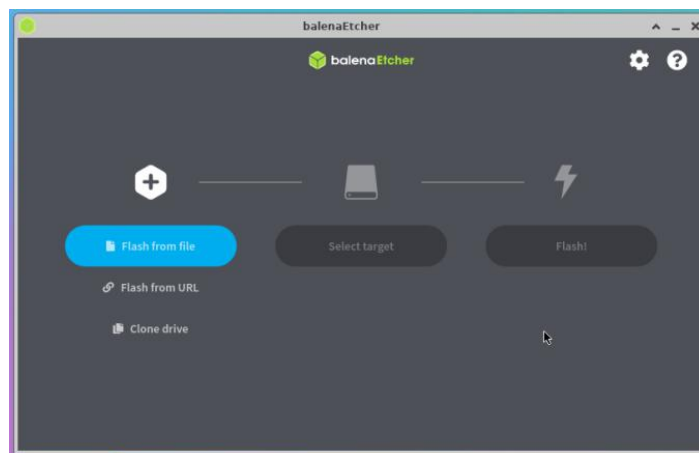
5) 然后将要烧录的 Linux 镜像文件压缩包上传到 TF 卡的 Linux 系统中。

**注意，使用xz格式压缩的Linux镜像压缩包不需要解压，balenaEtcher烧录时会自动解压。**

6) 然后就可以开始使用 balenaEtcher 软件烧录镜像到 SSD 中了。Linux 系统中已经预装了 balenaEtcher，打开方法如下所示：

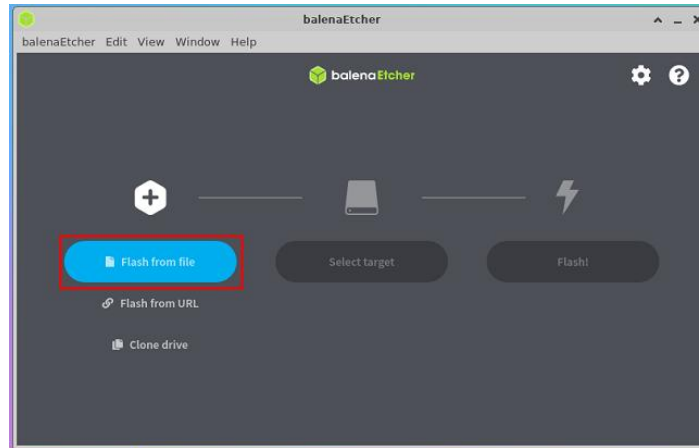


7) balenaEtcher 打开后的界面如下所示:



如果打开Linux镜像文件时提示没有权限，请使用 `sudo chmod 777 镜像文件名` 这条命令来给镜像文件添加权限。

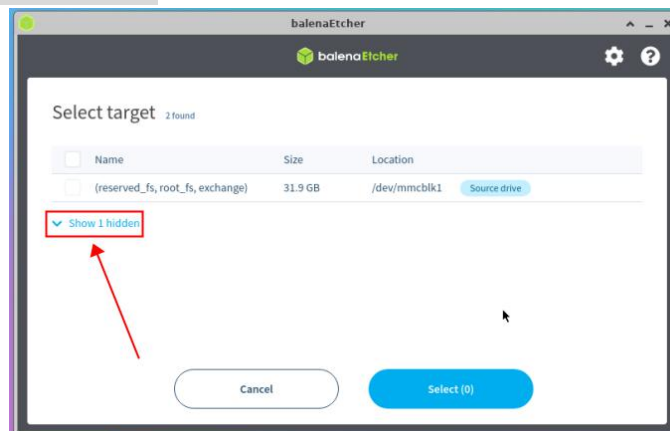
8) 然后点击 **Flash from file** 选择前面上传的想要烧录的镜像文件。



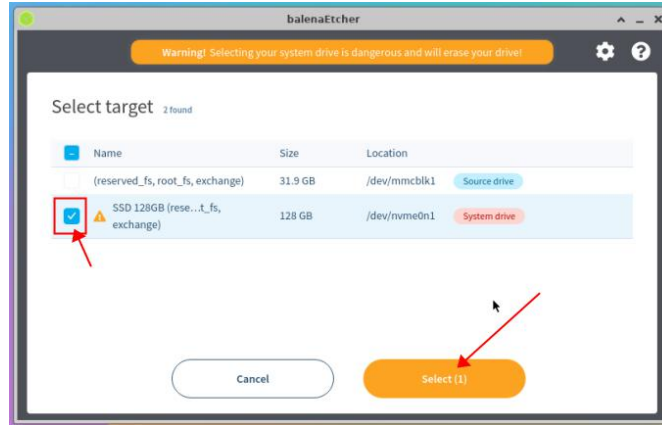
9) 然后点击 **Select target**。



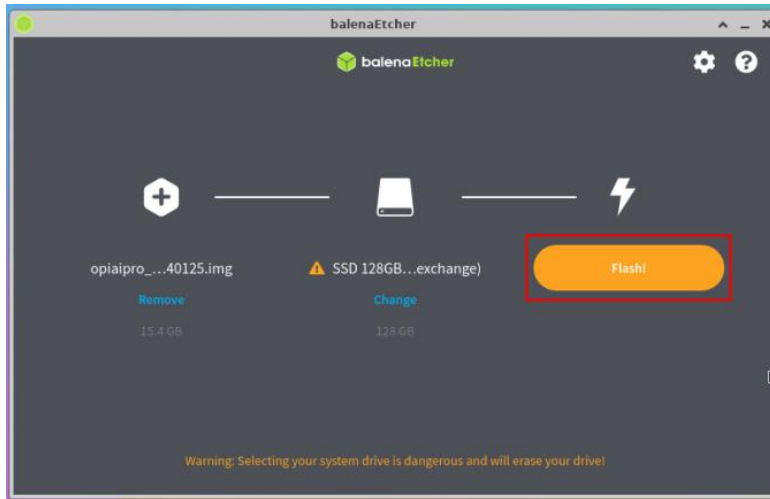
10) 然后点击 **Show 1 hidden**。



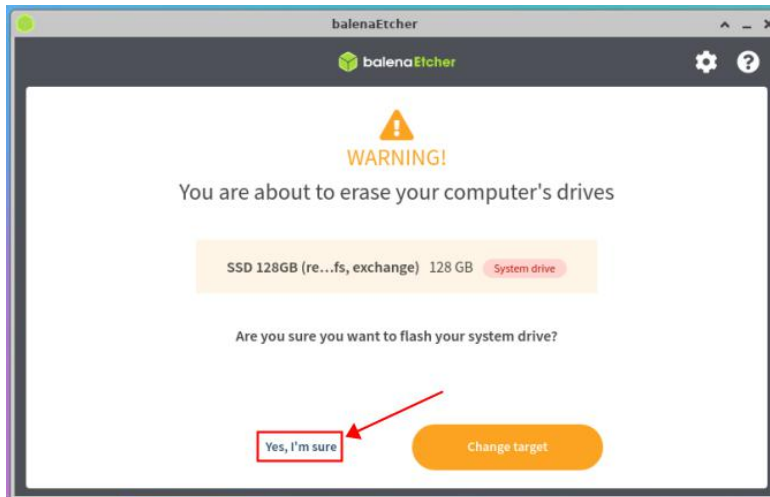
11) 然后选择 SSD 对应的选项，再点击 **Select** 即可。



12) 然后点击 **Flash!**开始烧录。

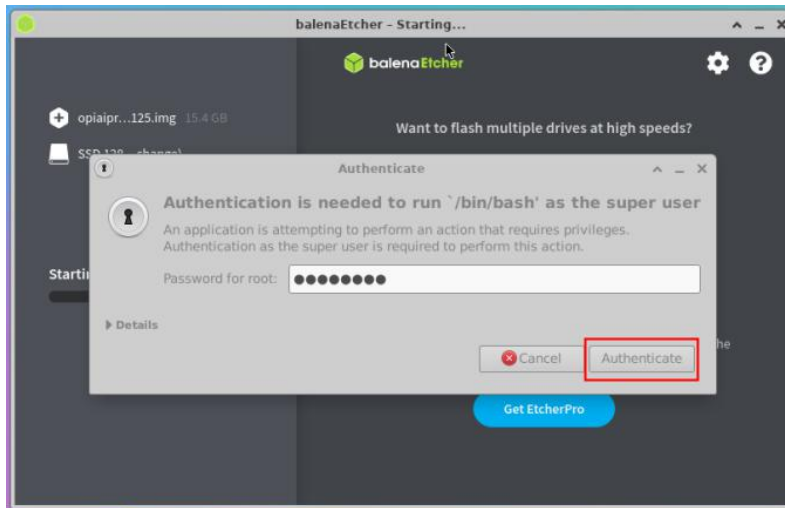


13) 然后选择 **Yes, I'm sure**。

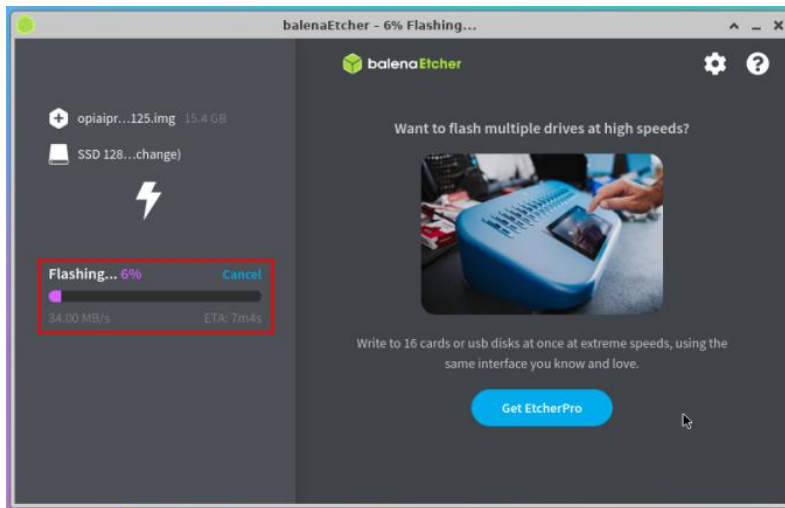




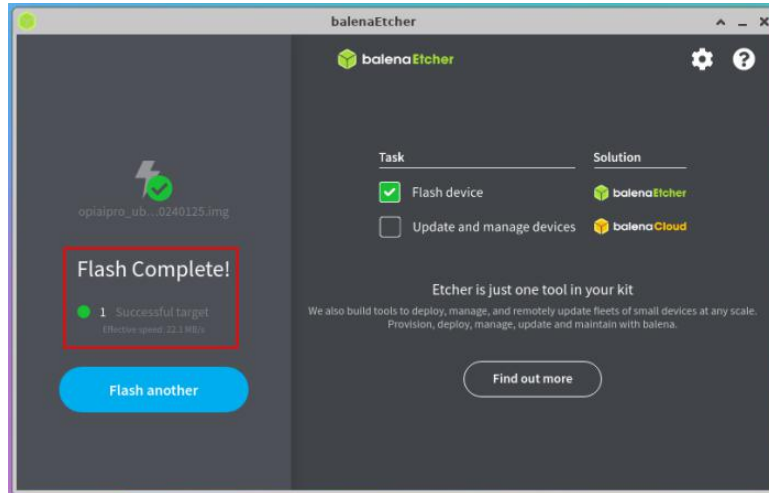
14) 然后输入 Linux 系统的密码: **Mind@123**。



15) 然后就会真正开始烧录 Linux 镜像到 SSD 中了。



16) Linux 镜像烧录完后的显示如下所示:



17) 此时就可以关闭掉 Linux 系统，然后拔出 TF 卡，并断开 Type-C 电源。再将两个拨码开关拨到 SSD 启动对应的位置，然后重新插入 Type-C 电源就可以启动 SSD 中的 Linux 系统了。

注意，启动系统前请确保拨码开关拨到了SSD启动的位置了。拨码开关的使用说明请参考[控制启动设备的两个拨码开关的使用说明](#)一小节的说明。

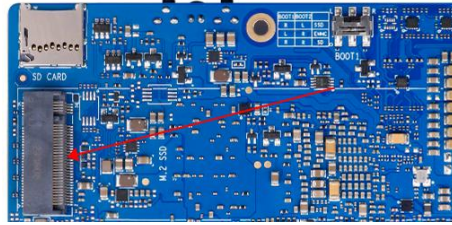
## 2.7. 烧写 Linux 镜像到 SATA SSD 中的方法

注意，这里说的Linux镜像具体指的是从开发板的资料下载页面下载的Ubuntu或者openEuler镜像。

1) 首先需要准备一个 M.2 2280 规格的 SATA SSD。2242 等其他规格的 SSD 也都是可以使用的，只是没法固定。



2) 然后把 SATA SSD 插入开发板的 M.2 接口中，并固定好。



3) 烧录 Linux 镜像到 SATA SSD 中需要借助 TF 卡来完成，所以首先需要将 Linux 镜像烧录到 TF 卡上，然后使用 TF 卡启动开发板进入 Linux 系统。烧录 Linux 镜像到 TF 卡的方法请见[基于 Windows PC 将 Linux 镜像烧写到 TF 卡的方法](#)和[基于 Ubuntu PC 将 Linux 镜像烧写到 TF 卡的方法](#)两小节的说明。

4) 启动进入 TF 卡的 Linux 系统后，首先需要更新下 SATA 对应的 dt.img 文件。步骤如下所示：

- a. 首先进入 `/opt/opi_test/sata` 文件夹。

```
(base) HwHiAiUser@orangepiaipro:~$ cd /opt/opi_test/sata
```

- b. 然后运行下 `update.sh` 脚本来更新 SATA 对应的 dt.img。

```
(base) HwHiAiUser@orangepiaipro:/opt/opi_test/sata$ sudo ./update.sh
```

- c. 运行完 `update.sh` 脚本后会自动重启 Linux 系统让配置生效。

- d. 一切顺利的话，重新进入 TF 卡的 Linux 系统后就能识别到 SATA SSD 了。

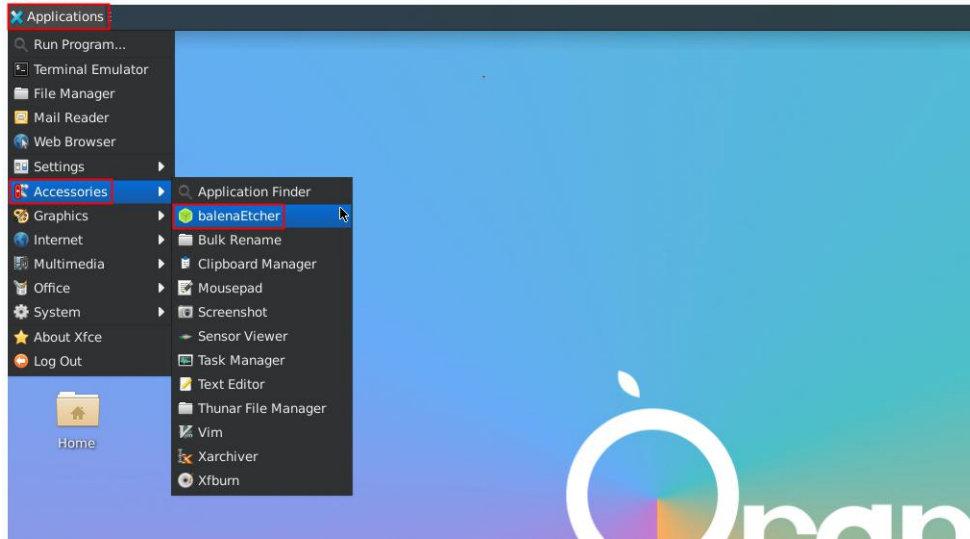
```
(base) HwHiAiUser@orangepiaipro:~$ sudo fdisk -l | grep "/dev/sd"
Disk /dev/sda: 238.47 GiB, 256060514304 bytes, 500118192 sectors
.....
```

开发板的 M.2 接口既支持 NVMe SSD 也支持 SATA SSD，只能二选一。Linux 系统中 dt.img 的配置默认打开的是 PCIe 的配置，如果 M.2 接口接的是 SATA SSD，需要手动切换下 SATA 对应的 dt.img 才能正常使用。

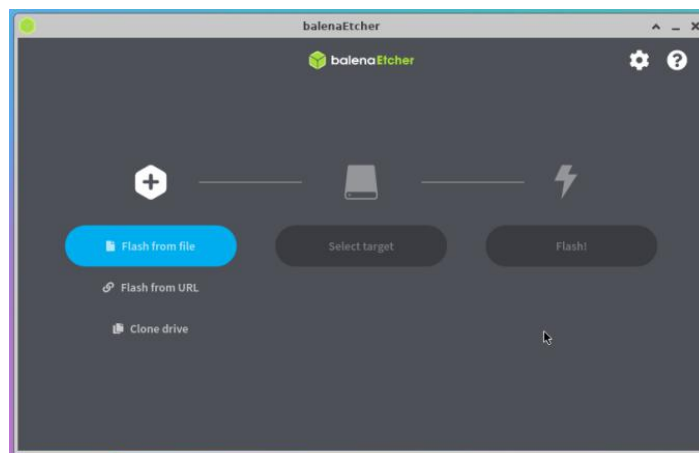
5) 然后将要烧录的 Linux 镜像文件压缩包上传到 TF 卡的 Linux 系统中。

注意，使用 xz 格式压缩的 Linux 镜像压缩包不需要解压，balenaEtcher 烧录时会自动解压。

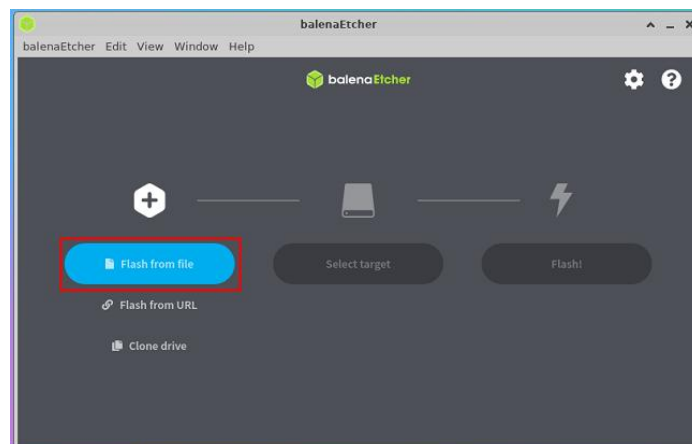
6) 然后就可以开始使用 balenaEtcher 软件烧录镜像到 SSD 中了。Linux 系统中已经预装了 balenaEtcher，打开方法如下所示：



7) balenaEtcher 打开后的界面如下所示:

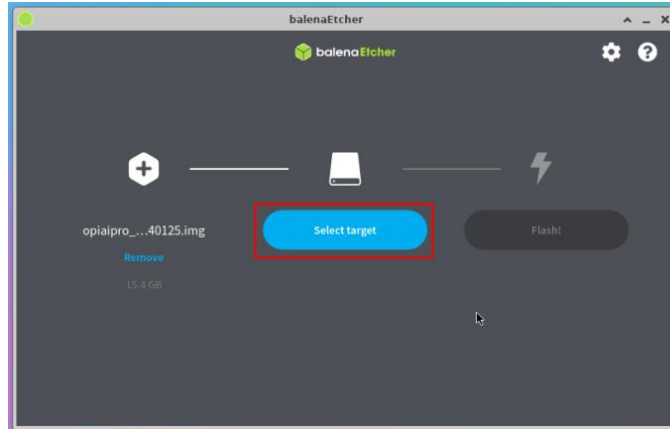


8) 然后点击 **Flash from file** 选择前面上传的想要烧录的镜像文件。

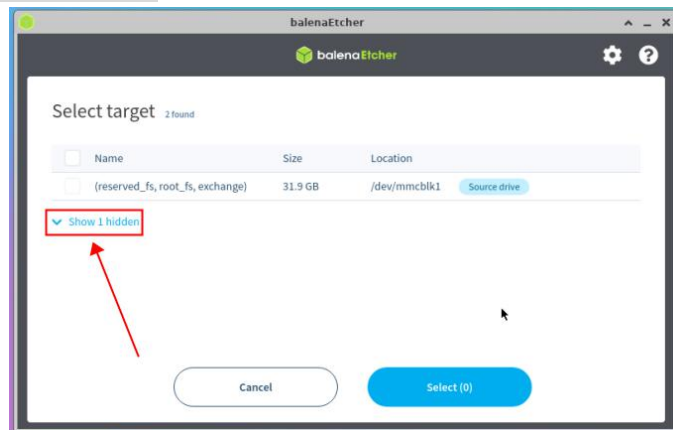


如果打开Linux镜像文件时提示没有权限，请使用 `sudo chmod 777` 镜像文件名这条命令来给镜像文件添加权限。

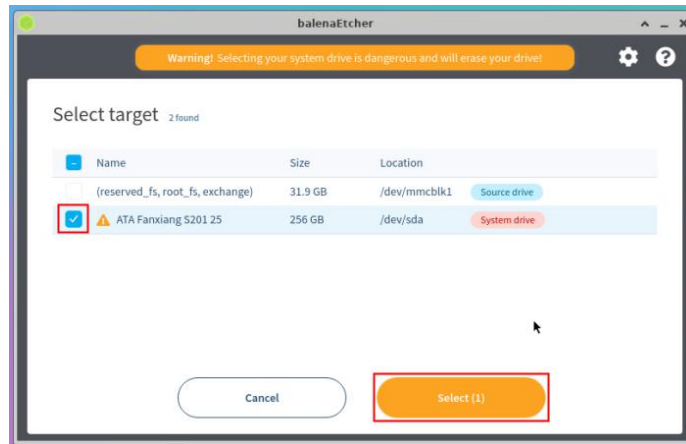
9) 然后点击 **Select target**。



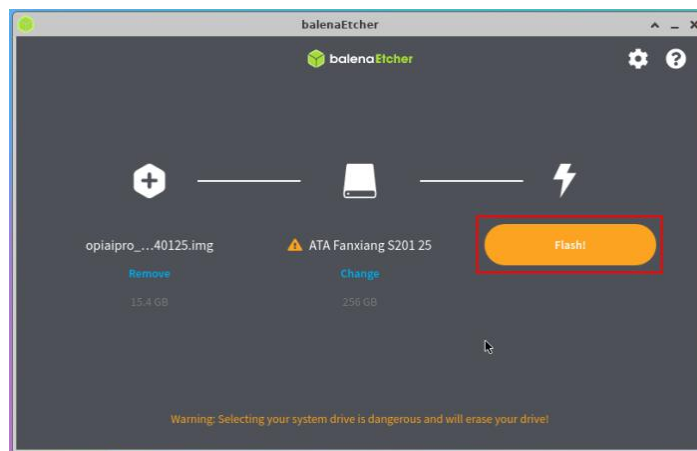
10) 然后点击 **Show 1 hidden**。



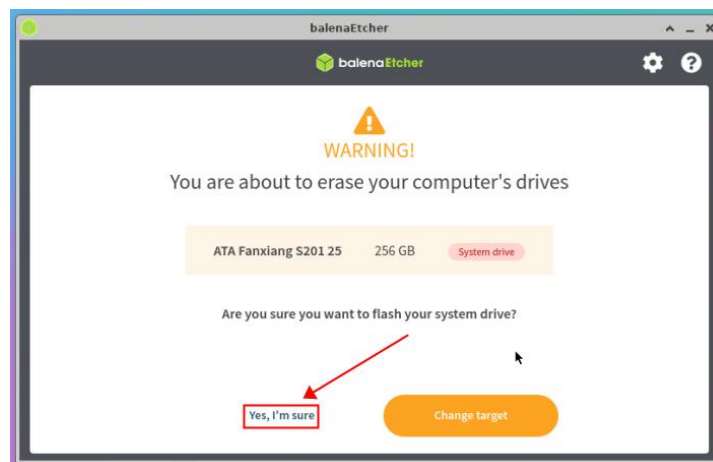
11) 然后选择 SSD 对应的选项，再点击 **Select** 即可。



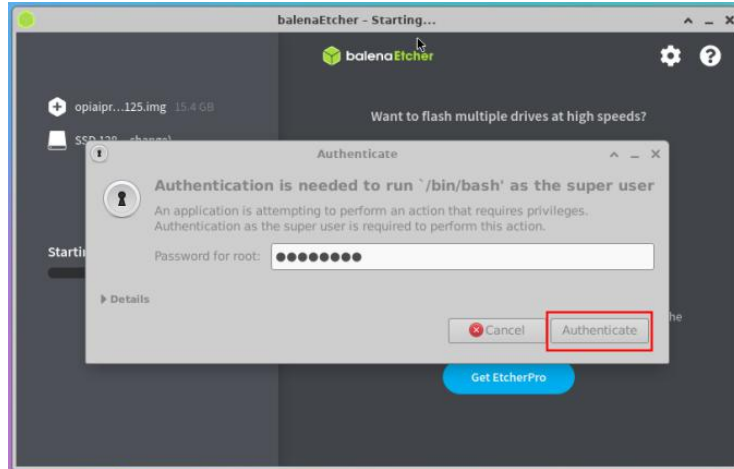
12) 然后点击 **Flash!**开始烧录。



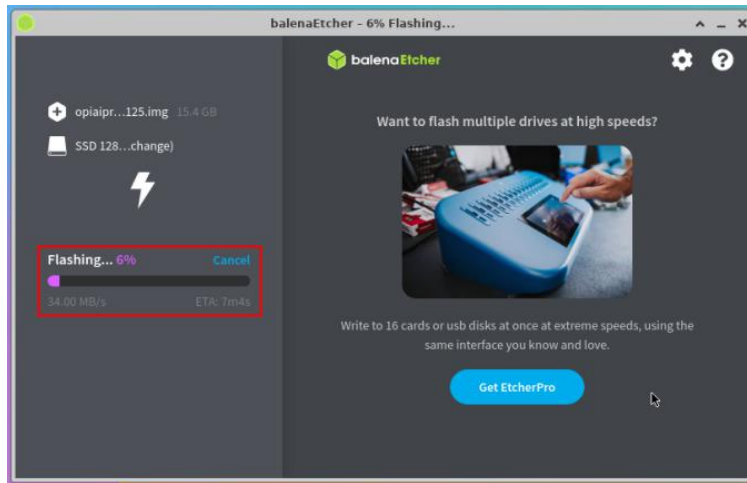
13) 然后选择 **Yes, I'm sure**。



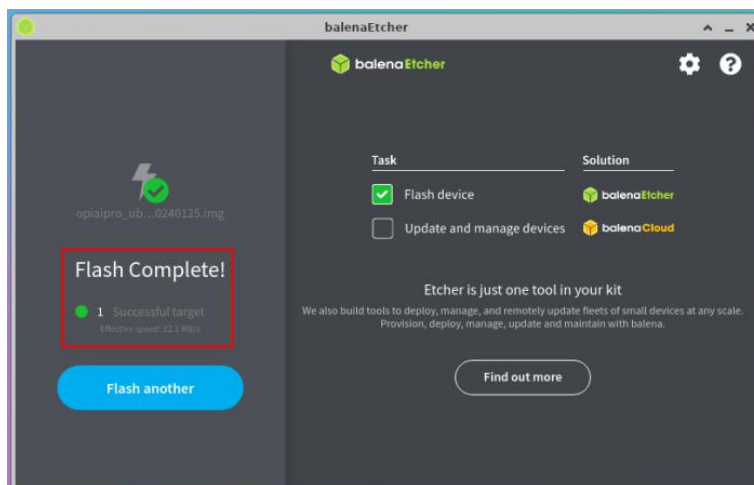
14) 然后输入 Linux 系统的密码: **Mind@123**。



15) 然后就会真正开始烧录 Linux 镜像到 SSD 中了。



16) Linux 镜像烧录完后的显示如下所示：



17) 烧录完成后，还需要将 SATA 版本的 dt.img 烧录到 SATA SSD 中，因为提供的镜像默认打开的都是 PCIe 的配置。具体命令如下所示：

```
sudo dd if=/opt/opi_test/dt_img/dt_drm_sata.img of=/dev/sda count=4096 seek=114688 bs=512
```

注意，上面的命令中，“of=”参数后面的/dev/sda为SSD对应的设备节点名。请根据实际情况进行修改。

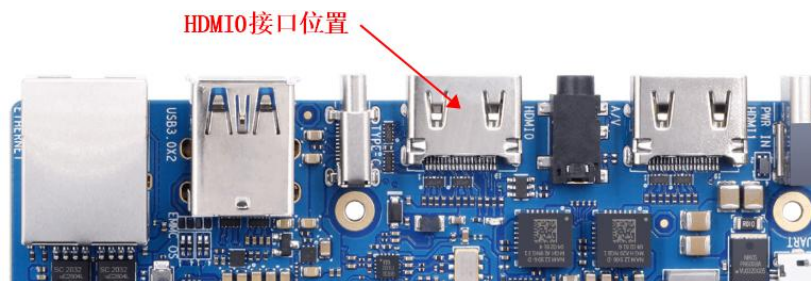
18) 此时就可以关闭掉 Linux 系统，然后拔出 TF 卡，并断开 Type-C 电源。再将两个拨码开关拨到 SSD 启动对应的位置，然后重新插入 Type-C 电源就可以启动 SSD 中的 Linux 系统了。

注意，启动系统前请确保拨码开关拨到了SSD启动的位置了。拨码开关的使用说明请参考[控制启动设备的两个拨码开关的使用说明](#)一小节的说明。

## 2.8. 启动开发板的步骤

1) 将烧录好镜像的 TF 卡或者 eMMC 模块或者 SSD 插入开发板对应的插槽中。

2) 开发板有两个 HDMI 接口（目前只有 HDMI0 支持显示 Linux 系统的桌面，HDMI1 显示 Linux 系统桌面的功能还需等软件更新），如果想显示 Linux 系统的桌面，可以将开发板的 HDMI0 接口连接到 HDMI 显示器。

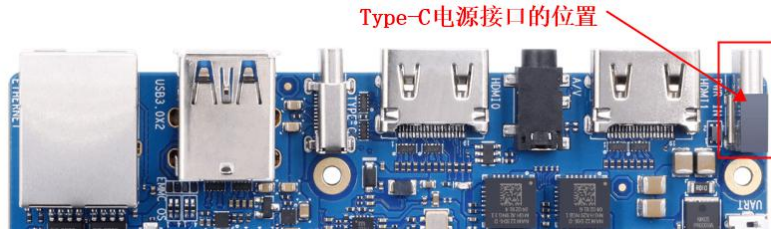


3) 开发板有 USB 接口，可以接上 USB 鼠标和键盘，来控制开发板。

4) 开发板有千兆以太网口，可以插入网线用来上网。

5) 然后需要连接一个 20V PD-65W 的 Type C 接口的电源，电源接口的位置如下图所示：





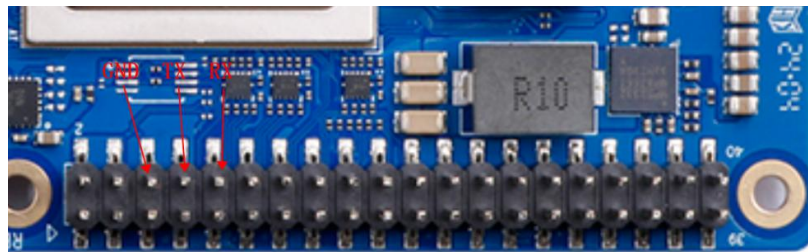
6) 然后打开电源适配器的开关，如果一切正常，等待一段时间后，HDMI 显示器就能看到 Linux 系统的登录界面了。

7) 如果想通过调试串口查看系统的输出信息，请使用串口线将开发板连接到电脑，串口的连接方法请参看[调试串口的使用方法](#)一节。

## 2.9. 调试串口的使用方法

开发板默认使用 `uart0` 做为调试串口。需要注意的是，`uart0` 的 `tx` 和 `rx` 引脚同时接到了两个地方，所以有两种使用调试串口的方法：

1) `uart0` 的 `tx` 和 `rx` 引脚接到了 40 pin 扩展接口中的 8 号和 10 号引脚，此种方式需要准备一个 **3.3v** 的 USB 转 TTL 模块和相应的杜邦线，然后才能正常使用开发板的调试串口功能。



2) `uart0` 的 `tx` 和 `rx` 引脚又接到了开发板的 CH343P 芯片上，再通过 CH343P 芯片引出到 Micro USB 接口上。此种方式只需要一根 Micro USB 接口的数据线将开发板连接到电脑的 USB 接口就可以开始使用开发板的调试串口功能了，无需购买 USB 转 TTL 模块。这种方法是推荐的方法。



另外请注意，上面的两种方法只能二选一，请不要同时使用。

### 2.9.1. 通过 Micro USB 接口来使用调试串口的连接说明

1) 首先需要准备一根 Micro USB 接口的数据线



2) 然后将 Micro USB 接口一端插入开发板的 Micro USB 接口中。



3) 再将数据线的另一端插入电脑的 USB 接口中即可。

### 2.9.2. 通过 40 pin 接口中的 uart0 来使用调试串口的连接说明

1) 首先需要准备一个 3.3v 的 USB 转 TTL 模块，然后将 USB 转 TTL 模块的 USB 接口一端插入到电脑的 USB 接口中。

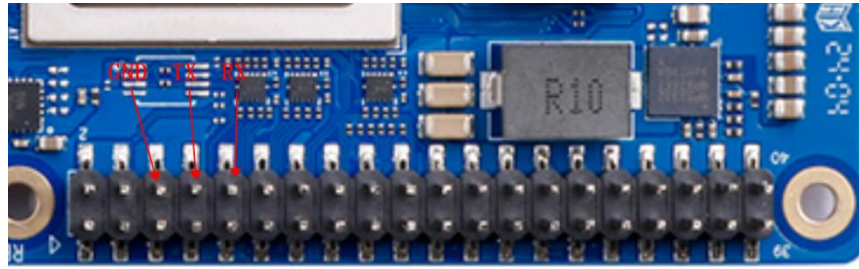
香橙派

USB转TTL模块



- USB转TTL模块的3.3V不需要接
- USB转TTL模块的TXD接开发板调试串口的RXD
- USB转TTL模块的RXD接开发板调试串口的TXD
- USB转TTL模块的GND接开发板调试串口的GND
- USB转TTL模块的5V不需要接

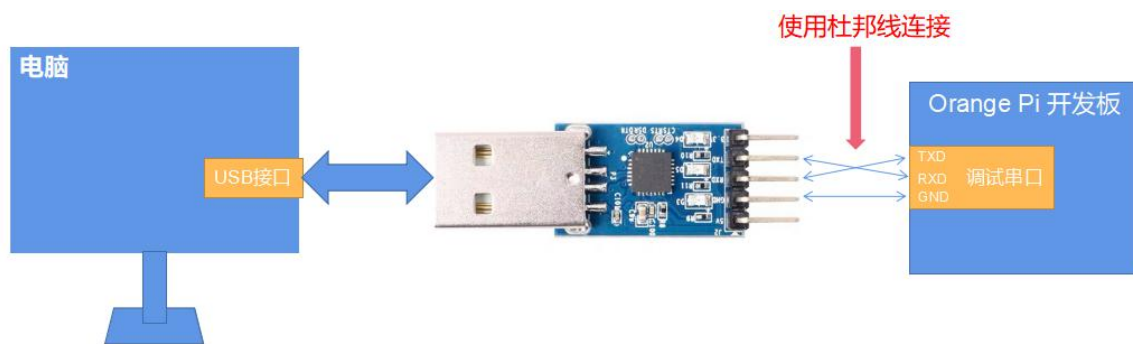
2) 开发板的调试串口 GND、TX 和 RX 引脚的对应关系如下图所示：



3) USB 转 TTL 模块 GND、TX 和 RX 引脚需要通过杜邦线连接到开发板的调试串口上。

- a. USB 转 TTL 模块的 GND 接到开发板的 GND 上。
- b. USB 转 TTL 模块的 **RX** 接到开发板的 **TX** 上。
- c. USB 转 TTL 模块的 **TX** 接到开发板的 **RX** 上。

4) USB 转 TTL 模块连接电脑和开发板的示意图如下所示：



USB转TTL模块连接电脑和 Orange Pi 开发板的示意图

串口的 TX 和 RX 是需要交叉连接的，如果不想仔细区分 TX 和 RX 的顺序，可以把串口的 TX 和 RX 先随便接上，如果测试串口没有输出再交换下 TX 和 RX 的顺序，这样就总有一种顺序是对的。

### 2.9.3. Ubuntu 平台调试串口的使用方法

Linux 下可以使用的串口调试软件有很多，如 **putty**、**minicom** 等，下面演示下 **putty** 的使用方法。

1) 首先请按照通过 40 pin 接口中的 **uart0** 来使用调试串口的连接说明或通过 **Micro USB 接口来使用调试串口的连接说明** 一小节的说明（两种方法请根据自己的情况二选一）将开发板和电脑连接起来，如果串口模块识别正常的话，在 Ubuntu PC 的

`/dev` 下就可以看到对应的设备名，请记住这个设备名，后面设置串口软件时会用到。

a. 如果使用 40 pin 接口中的 `uart0` 显示的设备名一般为 `/dev/ttyUSB0`。

```
test@test:~$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

b. 如果使用 Micro USB 接口显示的设备名一般为 `/dev/ttyACM0`。

```
test@test:~$ ls /dev/ttyACM*
/dev/ttyACM0
```

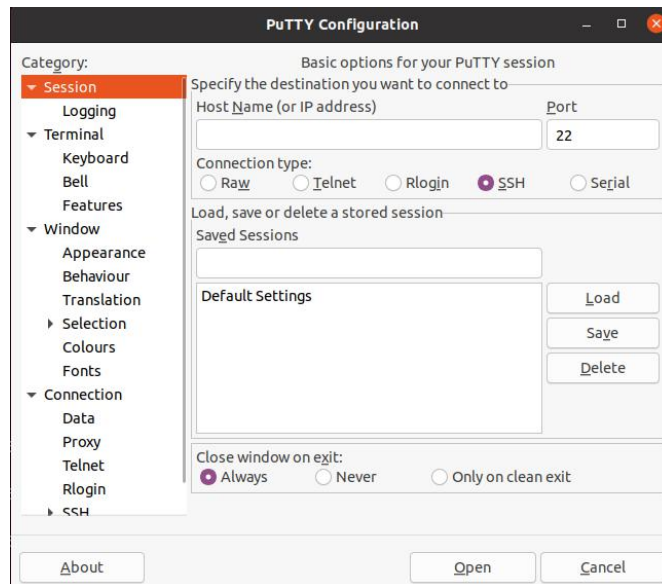
2) 然后使用下面的命令在 Ubuntu PC 上安装下 `putty`。

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install -y putty
```

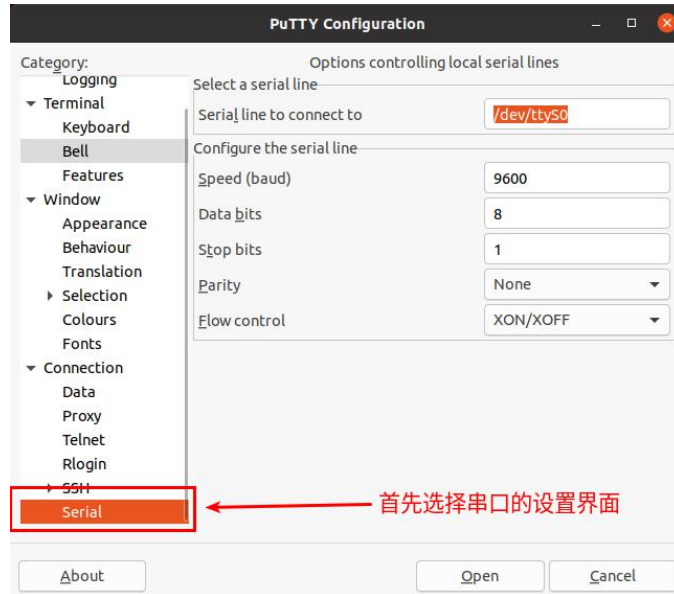
3) 然后运行 `putty`，记得加 `sudo` 权限。

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

4) 执行 `putty` 命令后会弹出下面的界面。

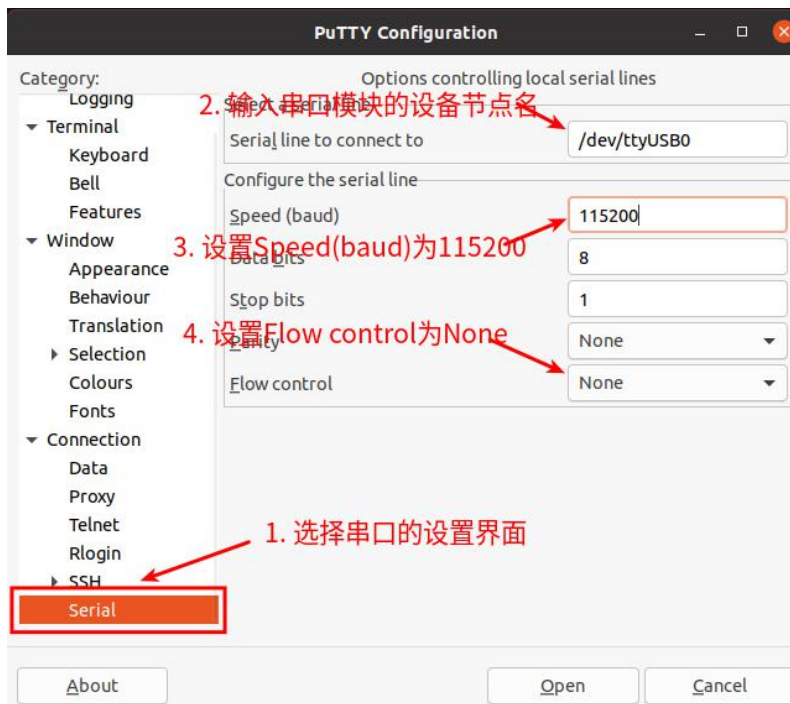


5) 首先选择串口的设置界面。



6) 然后设置串口的参数。

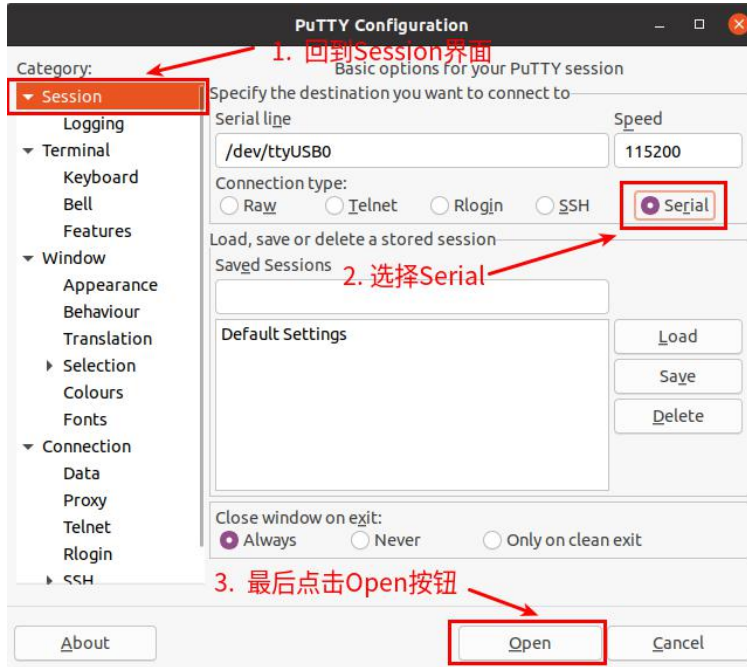
- a. 设置 **Serial line to connect to** 为 `/dev/ttyUSB0` 或者 `/dev/ttyACM0`（请根据实际情况进行修改）。
- b. 设置 **Speed(baud)** 为 **115200**（串口的波特率）。
- c. 设置 **Flow control** 为 **None**。



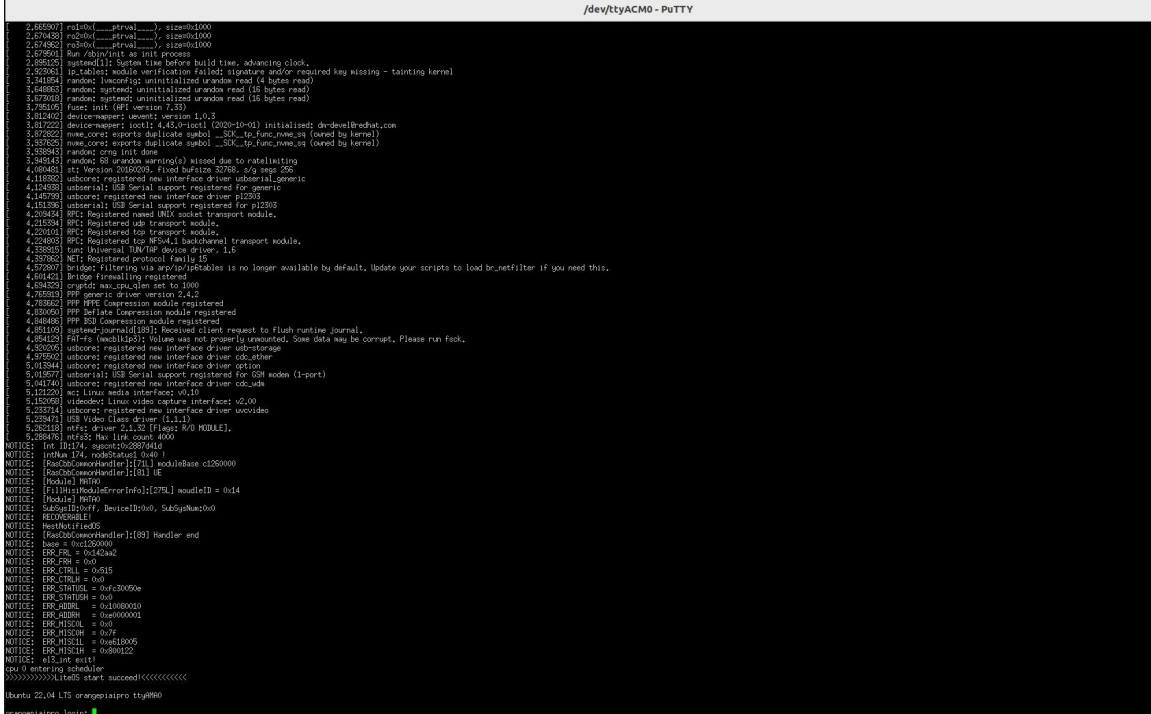
7) 在串口的设置界面设置完后，再回到 Session 界面。



- a. 首先选择 **Connection type** 为 **Serial**。
- b. 然后点击 **Open** 按钮连接串口。



8) 然后启动开发板，就能从打开的串口终端中看到系统输出的 Log 信息了。



9) 当看到登录界面时，就可以使用下面的账号和密码来登录 Linux 系统了。

账号	密码
root	Mind@123
HwHiAiUser	Mind@123

### 2.9.4. Windows 平台调试串口的使用方法

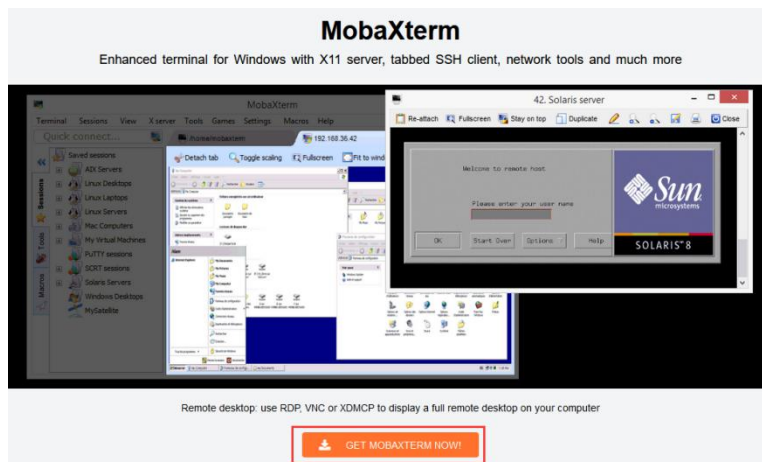
Windows 下可以使用的串口调试软件有很多，如 SecureCRT、MobaXterm 等，下面演示 MobaXterm 的使用方法，这款软件有免费版本，无需购买序列号即可使用。

1) 首先下载 MobaXterm。

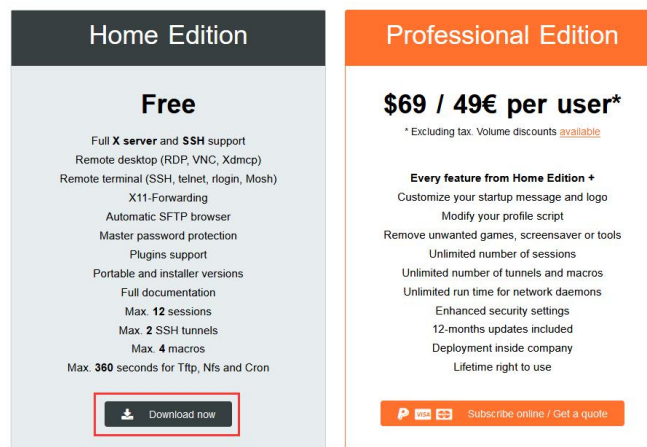
a. 下载 MobaXterm 网址如下：

<https://mobaxterm.mobatek.net/>

b. 进入 MobaXterm 下载网页后点击 **GET XOBATERM NOW!**。



c. 然后选择下载 Home 版本。



d. 然后选择 Portable 便携式版本，下载完后无需安装，直接打开就可以使用。



2) 下载完后使用解压缩软件解压下载的压缩包，即可得到 MobaXterm 的可执软件，然后双击打开。

MobaXterm_Personal_23.6.exe	2023/12/21 6:15	应用程序	16,556 KB
CygUtils.plugin	2023/12/21 5:08	PLUGIN 文件	17,748 KB
CygUtils64.plugin	2023/12/21 5:08	PLUGIN 文件	11,723 KB

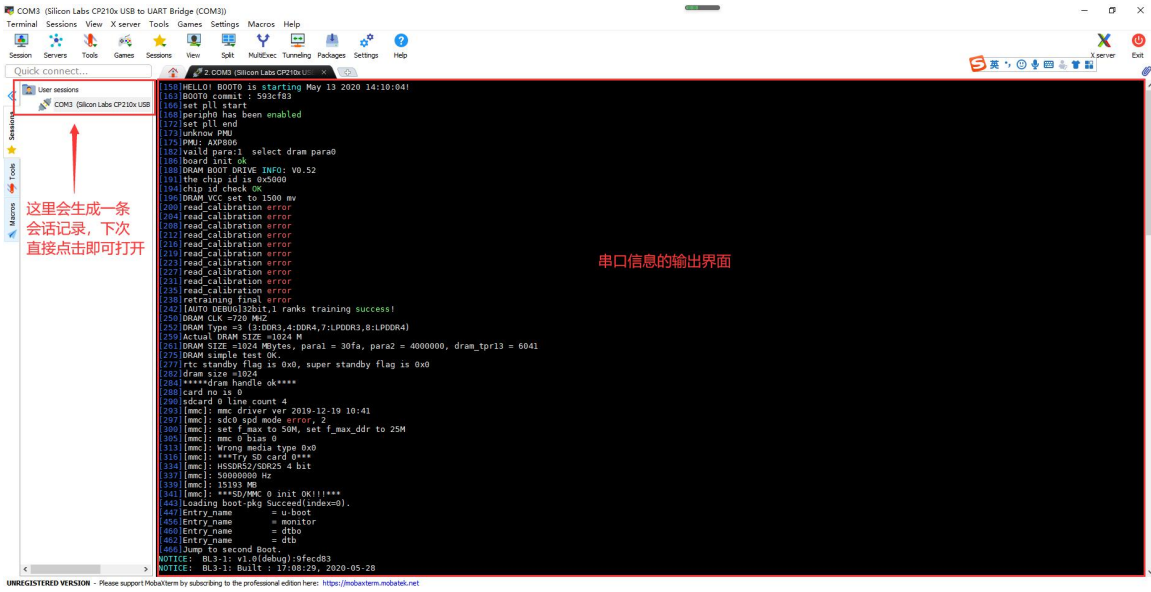
3) 打开软件后，设置串口连接的步骤如下：

- a. 打开会话的设置界面。
- b. 选择串口类型。
- c. 选择串口的端口号（根据实际情况选择对应的端口号），如果看不到端口号，请使用 [360 驱动大师](#) 扫描安装 USB 转 TTL 串口芯片的驱动。
- d. 选择串口的波特率为 **115200**。
- e. 最后点击 “OK” 按钮完成设置。



4) 点击 “OK” 按钮后会进入下面的界面，此时启动开发板就能看到串口的输出信息了。





5) 当看到登录界面时, 就可以使用下面的账号和密码来登录 Linux 系统了。

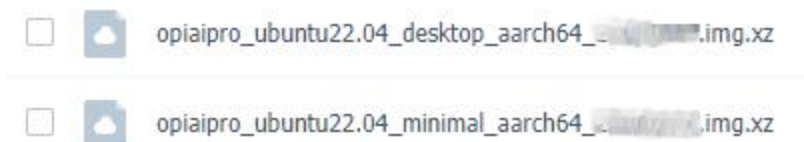
账号	密码
root	Mind@123
HwHiAiUser	Mind@123

### 3. Ubuntu Xfce 桌面系统使用说明

进入 Ubuntu 镜像的下载链接后可以看到下图所示的两个 ubuntu 镜像，他们的区别是：

1) **minimal** 镜像是一个只有最基础功能的镜像，像 Linux 桌面、CANN 和 AI 示例代码等都没有预装。此镜像只建议想自己从头定制安装 Linux 桌面和 AI 相关软件的开发者使用。

2) **desktop** 镜像预装了 Linux 桌面、CANN、AI 示例代码和一系列测试程序。如果想正常使用开发板的功能，请使用这个镜像。本章的内容都是基于 desktop 镜像编写的。



#### 3.1. 已支持的 Ubuntu 镜像类型和内核版本

Linux 镜像类型	内核版本	桌面版
Ubuntu 22.04 - Jammy	Linux5.10	支持

#### 3.2. Linux 系统功能适配情况

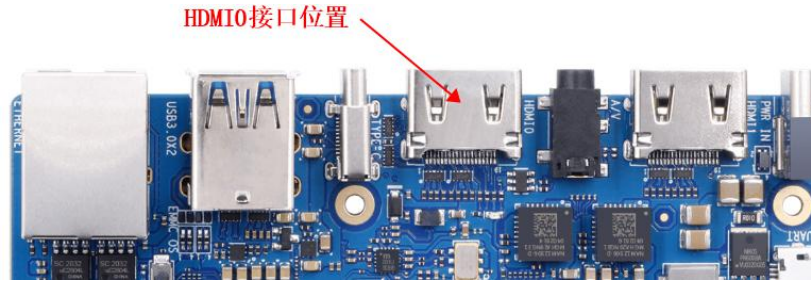
功能	是否能测试	Linux 内核驱动
HDMI0 显示	OK	OK
HDMI0 音频	OK	NO
HDMI1 显示	OK	NO
HDMI1 音频	OK	NO
耳机播放	OK	NO
耳机 MIC 录音	OK	NO
Type-C USB3.0 (无 USB2.0)	OK	OK
USB3.0 Host x 2	OK	OK
千兆网口	OK	OK
千兆网口灯	OK	OK

WIFI	OK	OK
蓝牙	OK	OK
Micro USB 调试串口	OK	OK
复位按键	OK	OK
关机按键（无开机功能）	OK	OK
烧录按键	OK	OK
MIPI 摄像头 0	OK	NO
MIPI 摄像头 1	OK	NO
MIPI LCD 显示	OK	NO
电源指示灯	OK	OK
软件可控的 LED 灯	OK	OK
风扇接口	OK	OK
电池接口	OK	OK
TF 卡启动	OK	OK
TF 卡启动识别 eMMC	OK	OK
TF 卡启动识别 NVMe SSD	OK	OK
TF 卡启动识别 SATA SSD	OK	OK
eMMC 启动	OK	OK
SATA SSD 启动	OK	OK
NVMe SSD 启动	OK	OK
2 个拨码开关	OK	OK
40 pin-GPIO	OK	OK
40 pin-UART	OK	OK
40 pin-SPI	OK	OK
40 pin-I2C	OK	OK
40 pin-PWM	OK	NO

### 3.3. Linux 系统登录说明

#### 3.3.1. 登录 Linux 系统桌面的方法

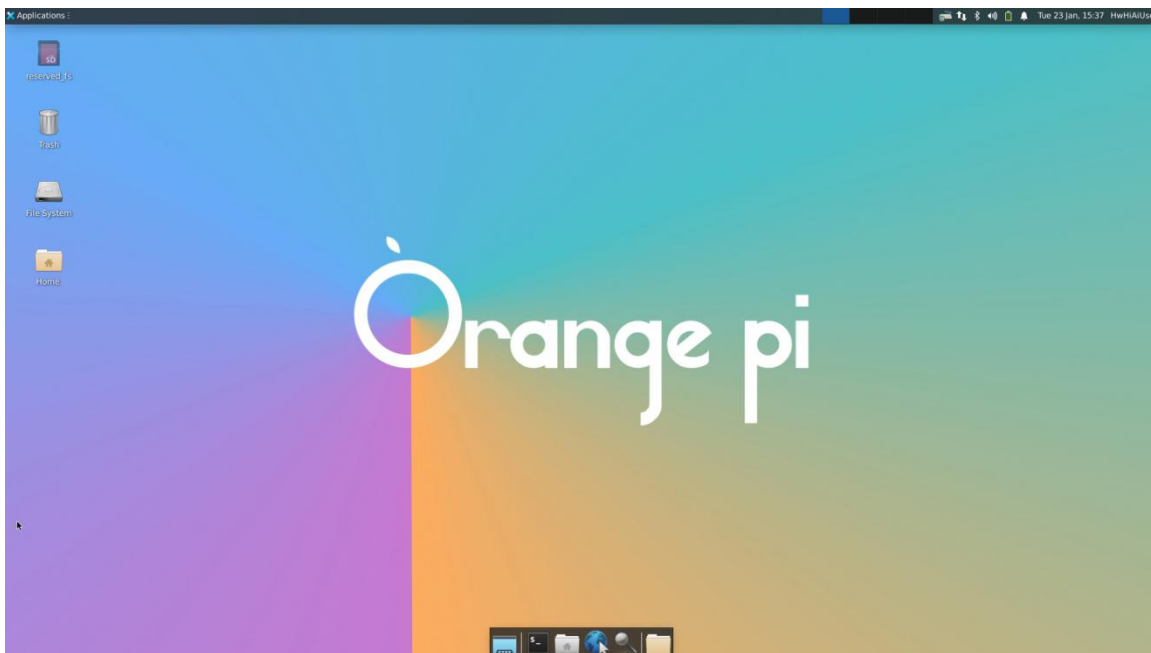
开发板有两个 HDMI 接口，目前只有 HDMI0 支持显示 Linux 系统的桌面，HDMI1 还需等软件更新。如果想显示 Linux 系统的桌面，请将开发板的 HDMI0 接口连接到 HDMI 显示器。



开发板上电开机后，需要等待一段时间，HDMI 显示器才会显示 Linux 系统的登录界面，登录界面如下图所示：



Linux 桌面系统的默认登录用户为 **HwHiAiUser**，登录密码为 **Mind@123**。目前没有打开 root 用户登录的通道。成功登录后显示的 Linux 系统桌面如下图所示：



### 3.3.2. Linux 系统默认登录账号和密码

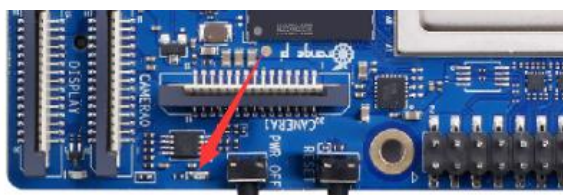
账号	密码
root	Mind@123
HwHiAiUser	Mind@123

当输入密码提示错误，或者 ssh 连接有问题，请注意，只要使用的是 Orange Pi 提供的 Linux 镜像，**就请不要怀疑上面的密码不对**，而是要找其他的原因。

### 3.4. 板载 LED 灯测试说明

开发板上有两个绿色的 LED 灯，作用如下所示：

1) 靠近关机按键的绿灯：此绿灯为电源指示灯，由硬件控制其亮灭，软件无法控制。只要开发板接入了 Type-C 电源并上电了，此绿灯就会点亮。



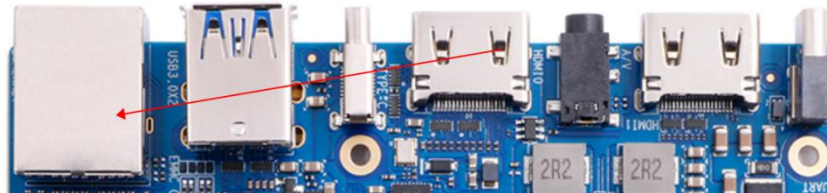
2) MIPI LCD 和 CAMERA0 之间的绿灯：此绿灯由 **GPIO4\_19** 控制其亮灭，可以作为 SATA 硬盘的指示灯或者其他需要的用途。目前发布的 Linux 系统默认在 DTS 中将其点亮。当看到此灯点亮后，至少可以说明 Linux 内核已经启动了。



### 3.5. 网络连接测试

#### 3.5.1. 以太网口测试

1) 首先将网线的一端插入开发板的以太网接口，网线的另一端接入交换机或者路由器，并确保网络是畅通的。



2) 系统启动后会通过 **DHCP** 自动给以太网口分配 IP 地址。

3) 在开发板的 Linux 系统中查看 IP 地址的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ ip a s eth0
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether 2c:52:af:89:11:11 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.100/24 brd 192.168.2.255 scope global dynamic noprefixroute eth0
        valid_lft 42077sec preferred_lft 42077sec
    inet6 fe80::913d:474c:4834:a1a4/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

4) 测试网络连通性的命令如下所示，如果能 ping 通百度说明开发板的网络连接正常，**ping** 命令可以通过 **Ctrl+C** 快捷键来中断运行。

```
(base) HwHiAiUser@orangepiaipro:~$ ping www.baidu.com -I eth0
```

```
PING www.a.shifen.com (183.2.172.185) from 192.168.2.100 eth0: 56(84) bytes of data.
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=1 ttl=52 time=10.0 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=2 ttl=52 time=9.77 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=3 ttl=52 time=9.94 ms
64 bytes from 183.2.172.185 (183.2.172.185): icmp_seq=4 ttl=52 time=9.94 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 9.770/9.931/10.065/0.105 ms
```

### 3.5.2. WIFI 连接测试

请不要通过修改/etc/network/interfaces 配置文件的方式来连接 WIFI，通过这种方式连接 WIFI 网络使用会有问题。

#### 3.5.2.1. 通过 nmcli 命令连接 WIFI 的方法

1) 先登录 Linux 系统，有下面三种方式：

- a. 如果开发板连接了网线，可以通过 [ssh 远程登录 Linux 系统](#)。
- a. 如果开发板连接好了调试串口，可以使用串口终端登录 Linux 系统。
- b. 如果连接了开发板到 HDMI 显示器，可以通过 HDMI 显示的终端登录到 Linux 系统。

2) 然后使用 `nmcli dev wifi` 命令扫描周围的 WIFI 热点。

```
(base) HwHiAiUser@orangepiaipro:~$ nmcli dev wifi
```

3) 然后使用 `nmcli` 命令连接扫描到的 WIFI 热点，其中：

- a. `wifi_name` 需要换成想连接的 WIFI 热点的名字。
- b. `wifi_passwd` 需要换成想连接的 WIFI 热点的密码。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo nmcli dev wifi connect wifi_name password wifi_passwd
Device 'wlan0' successfully activated with 'cf937f88-ca1e-4411-bb50-61f402eef293'.
```

4) 通过 `ip addr show wlan0` 命令可以查看 wifi 的 IP 地址。

```
(base) HwHiAiUser@orangepiaipro:~$ ip a s wlan0
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
```



```
link/ether 54:f2:9f:7b:ba:36 brd ff:ff:ff:ff:ff:ff
inet 10.31.2.93/16 brd 10.31.255.255 scope global dynamic noprefixroute wlan0
    valid_lft 43191sec preferred_lft 43191sec
inet6 fe80::5297:7036:a33c:bb93/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
```

5) 使用 **ping** 命令可以测试 wifi 网络的连通性，**ping** 命令可以通过 **Ctrl+C** 快捷键来中断运行。

```
(base) HwHiAiUser@orangepiaipro:~$ ping www.orangepi.org -I wlan0
PING www.orangepi.org (123.57.147.237) from 10.31.2.93 wlan0: 56(84) bytes of data.
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=1 ttl=53 time=47.1 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=2 ttl=53 time=44.3 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=3 ttl=53 time=45.0 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=4 ttl=53 time=71.0 ms
^C
--- www.orangepi.org ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 44.377/51.902/71.082/11.119 ms
```

### 3.5.2.2. 通过 nmtui 图形化方式连接 WIFI 的方法

1) 先登录 Linux 系统，有下面三种方式：

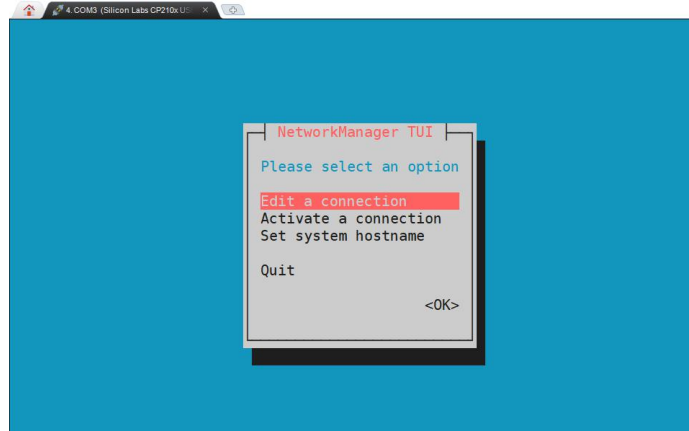
- a. 如果开发板连接了网线，可以通过 [ssh 远程登录 Linux 系统](#)。
- b. 如果开发板连接好了调试串口，可以使用串口终端登录 Linux 系统。
- c. 如果连接了开发板到 HDMI 显示器，可以通过 HDMI 显示的终端登录到 Linux 系统。

2) 然后在命令行中输入 nmtui 命令打开 wifi 连接的界面。

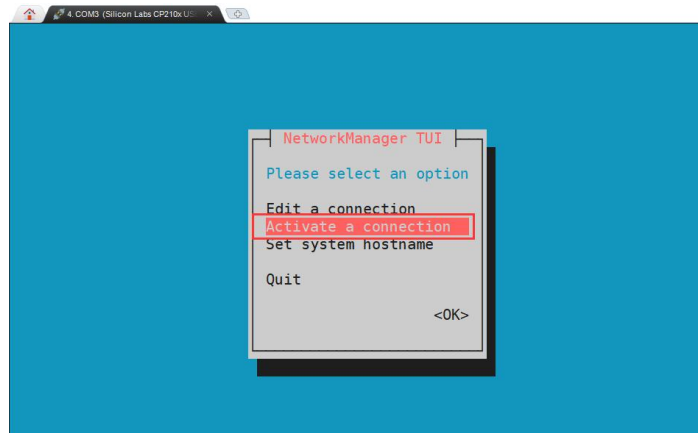
```
(base) HwHiAiUser@orangepiaipro:~$ sudo nmtui
```

3) 输入 nmtui 命令打开的界面如下所示：

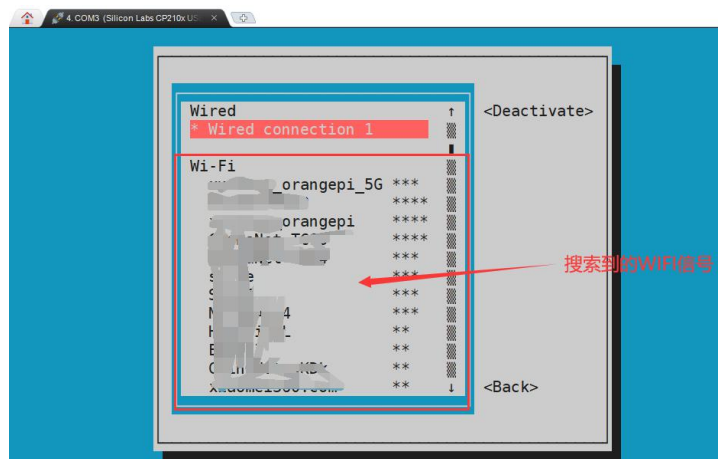




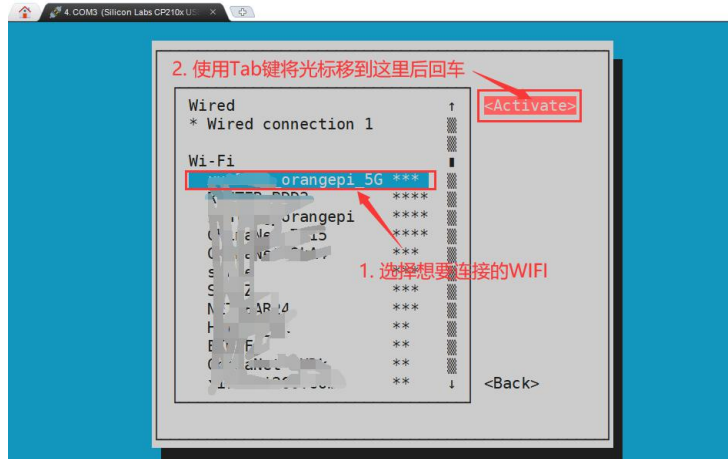
4) 选择 **Activate a connect** 后回车。



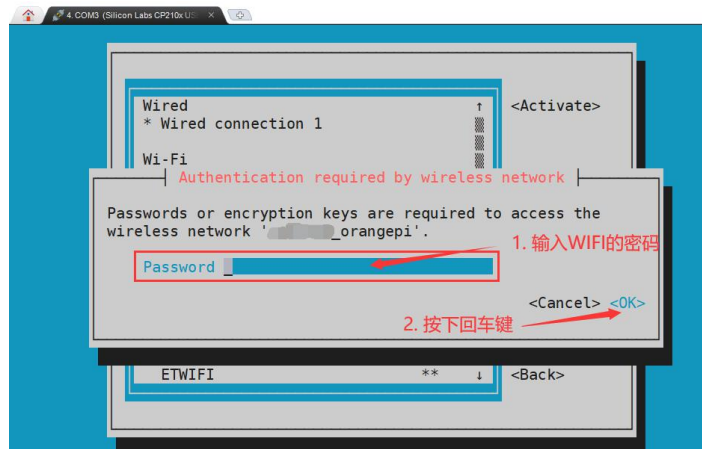
5) 然后就能看到所有搜索到的 WIFI 热点。



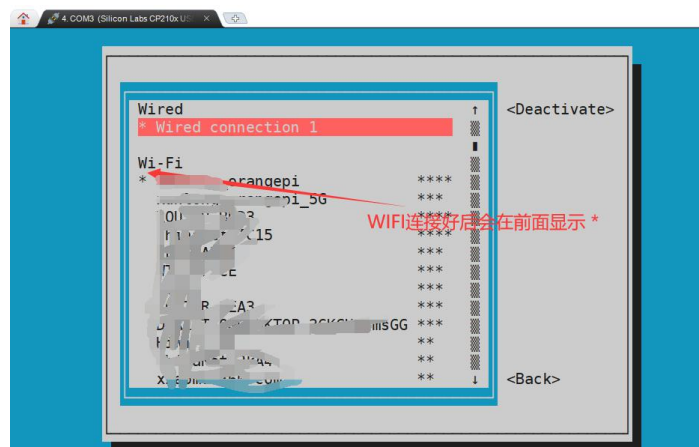
6) 选择想要连接的 WIFI 热点后再使用 Tab 键将光标定位到 **Activate** 后回车。



7) 然后会弹出输入密码的对话框，在 **Password** 内输入对应的密码然后回车就会开始连接 WIFI。



8) WIFI 连接成功后会在已连接的 WIFI 名称前显示一个 “\*”。



9) 通过 **ip a s wlan0** 命令可以查看 wifi 的 IP 地址。

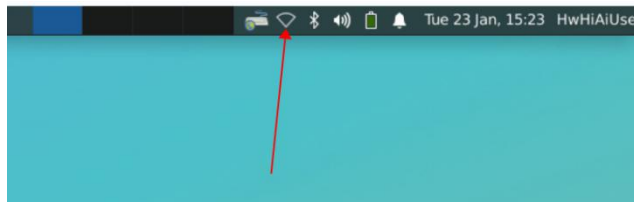
```
(base) HwHiAiUser@orangepiaipro:~$ ip a s wlan0
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether 54:f2:9f:7b:ba:36 brd ff:ff:ff:ff:ff:ff
    inet 10.31.2.93/16 brd 10.31.255.255 scope global dynamic noprefixroute wlan0
        valid_lft 43003sec preferred_lft 43003sec
    inet6 fe80::5297:7036:a33c:bb93/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

10) 使用 **ping** 命令可以测试 wifi 网络的连通性，**ping** 命令可以通过 **Ctrl+C** 快捷键来中断运行。

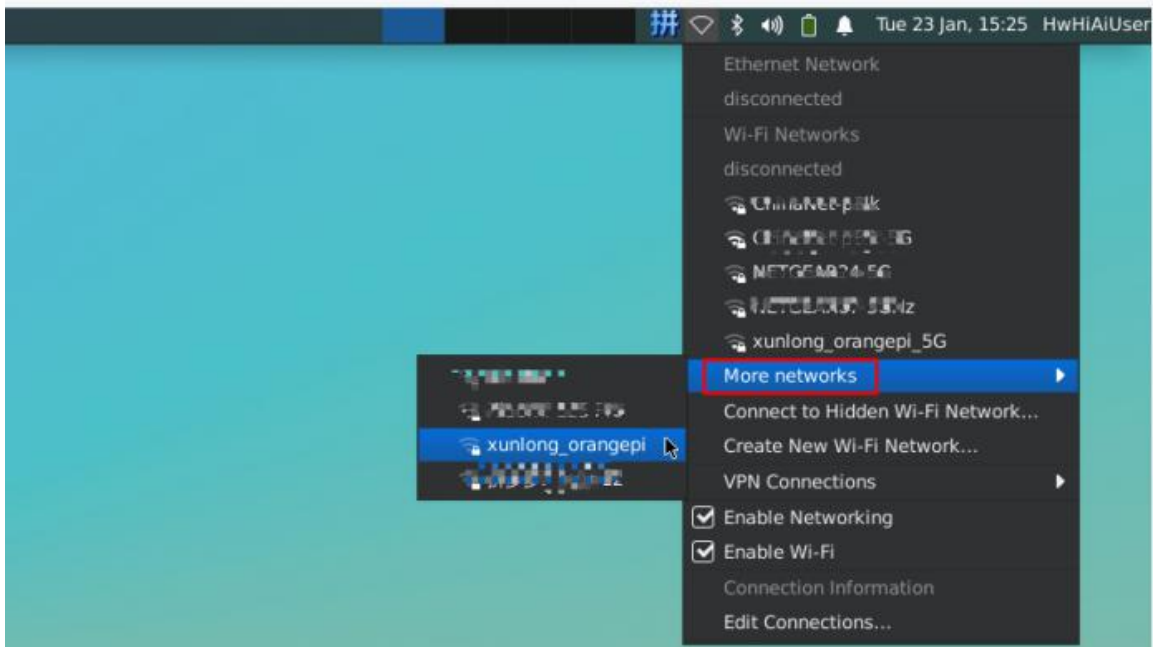
```
(base) HwHiAiUser@orangepiaipro:~$ ping www.orangepi.org -I wlan0
PING www.orangepi.org (123.57.147.237) from 10.31.2.93 wlan0: 56(84) bytes of data.
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=1 ttl=53 time=47.1 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=2 ttl=53 time=44.3 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=3 ttl=53 time=45.0 ms
64 bytes from 123.57.147.237 (123.57.147.237): icmp_seq=4 ttl=53 time=71.0 ms
^C
--- www.orangepi.org ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 44.377/51.902/71.082/11.119 ms
```

### 3.5.2.3. 桌面版镜像的测试方法

1) 首先点击桌面右上角的网络配置图标。



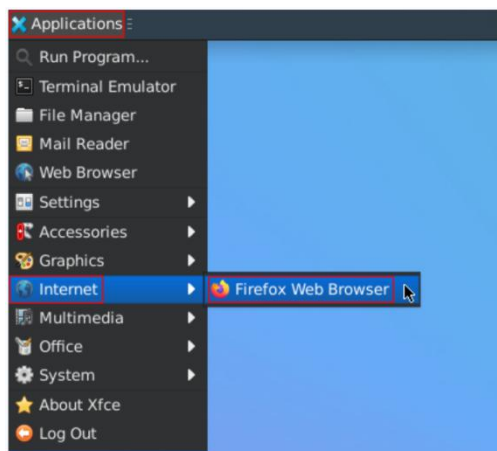
2) 在弹出的下拉框中点击 **More networks** 可以看到所有扫描到的 WIFI 热点，然后选择想要连接的 WIFI 热点。



3) 然后输入 WIFI 热点的密码，再点击 **Connect** 就会开始连接 WIFI。



4) 连接好 WIFI 后，可以打开浏览器查看是否能上网，浏览器的入口如下图所示：



5) 打开浏览器后如果能打开其他网页说明 WIFI 连接正常。



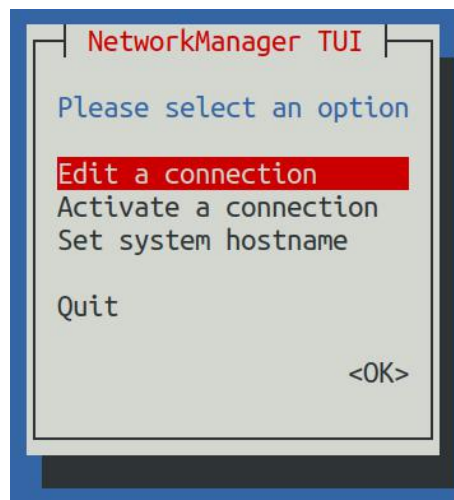
### 3.5.3. 设置静态 IP 地址的方法

#### 3.5.3.1. 使用 nmtui 命令来设置静态 IP 地址

1) 首先运行 **nmtui** 命令。

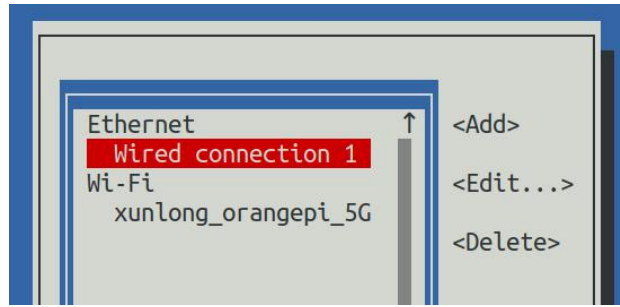
```
(base) HwHiAiUser@orangepiaipro:~$ sudo nmtui
```

2) 然后选择 **Edit a connection** 并按下回车键。

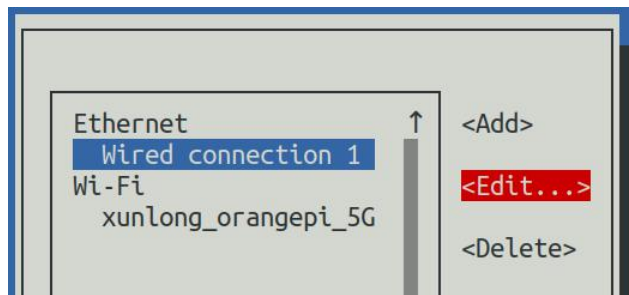


3) 然后选择需要设置静态 IP 地址的网络接口，比如设置 **Ethernet** 接口的静态 IP 地

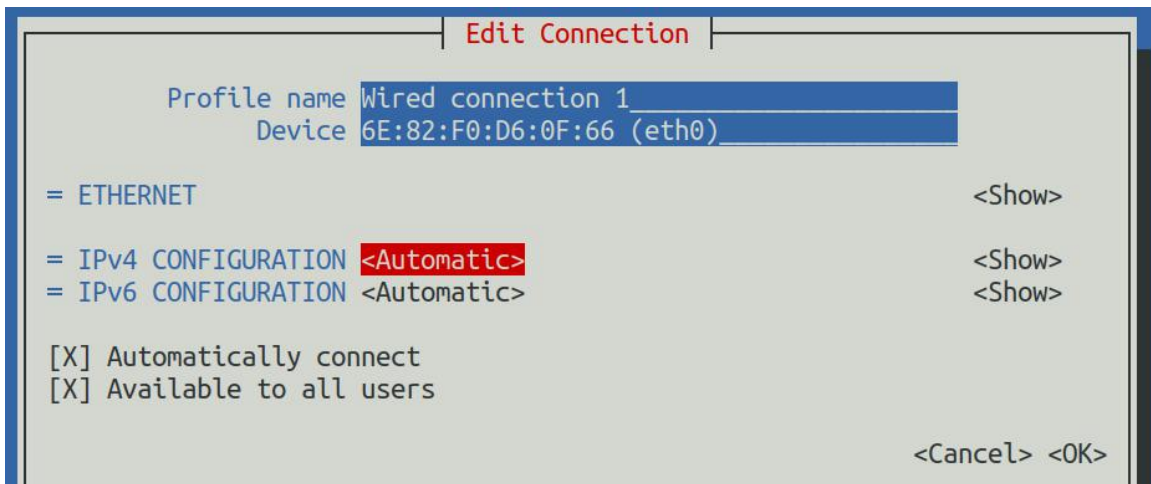
址选择 **Wired connection 1** 就可以了。



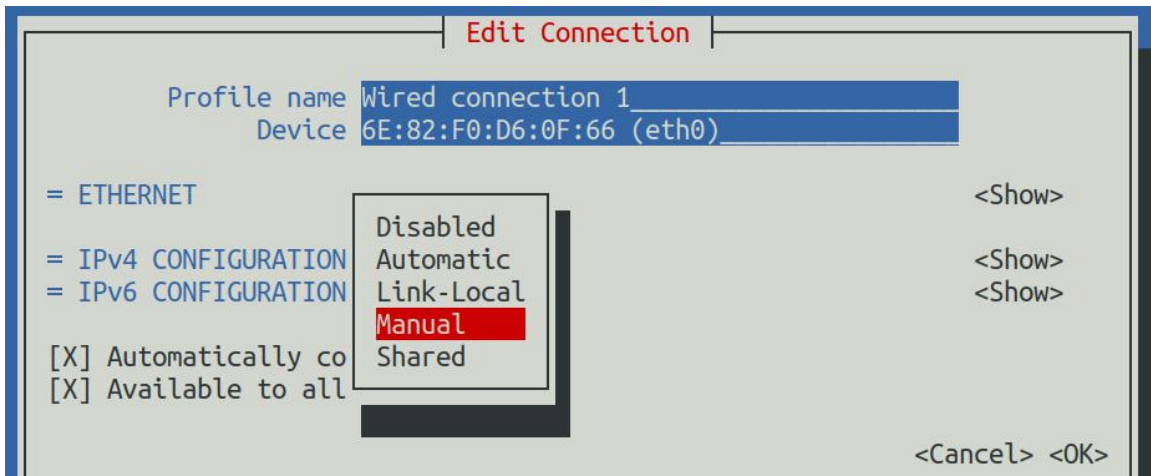
4) 然后通过 **Tab** 键选择 **Edit** 并按下回车键。



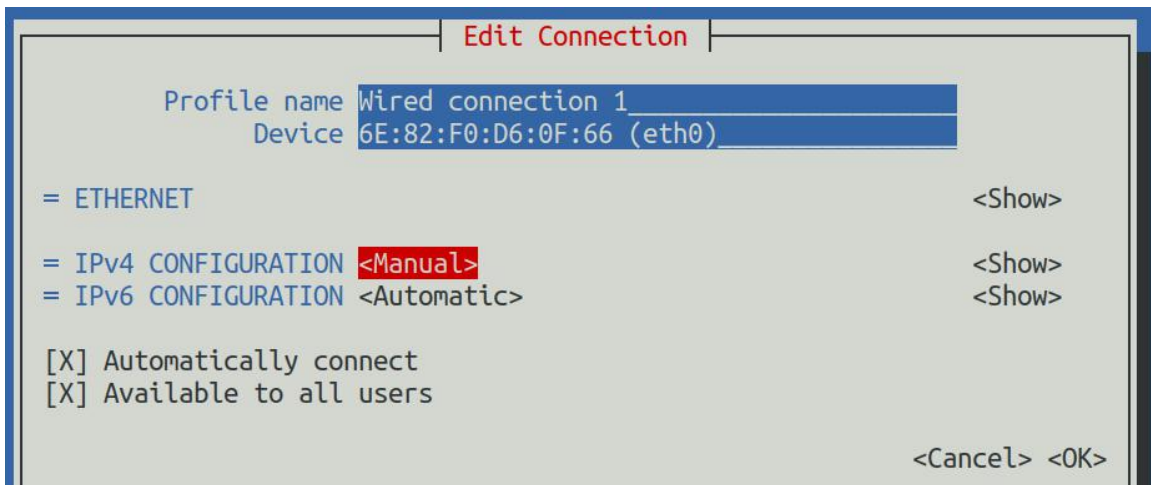
5) 然后通过 **Tab** 键将光标移动到下图所示的 **<Automatic>** 位置进行 IPv4 的配置。



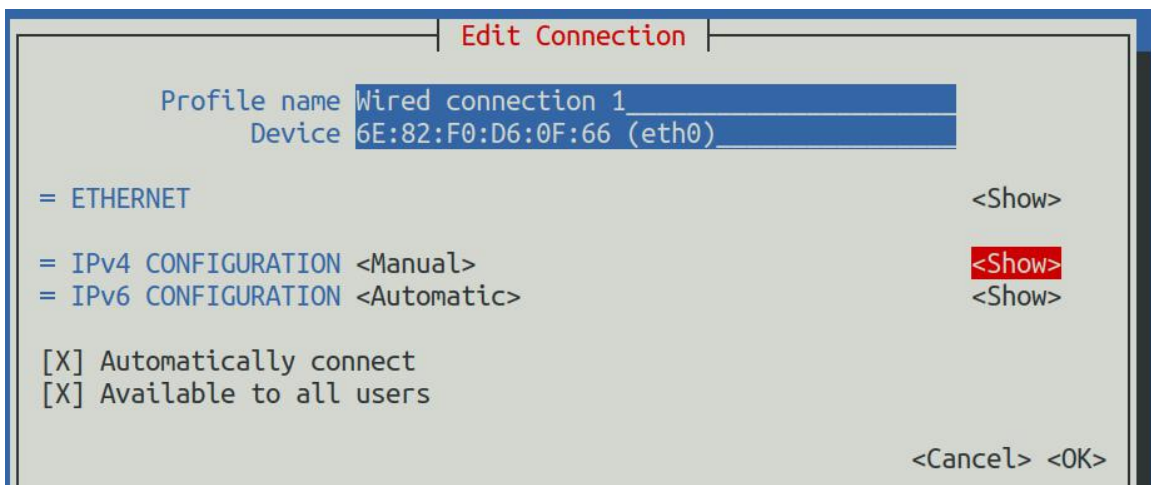
6) 然后回车，通过上下方向键选择 **Manual**，然后回车确定。



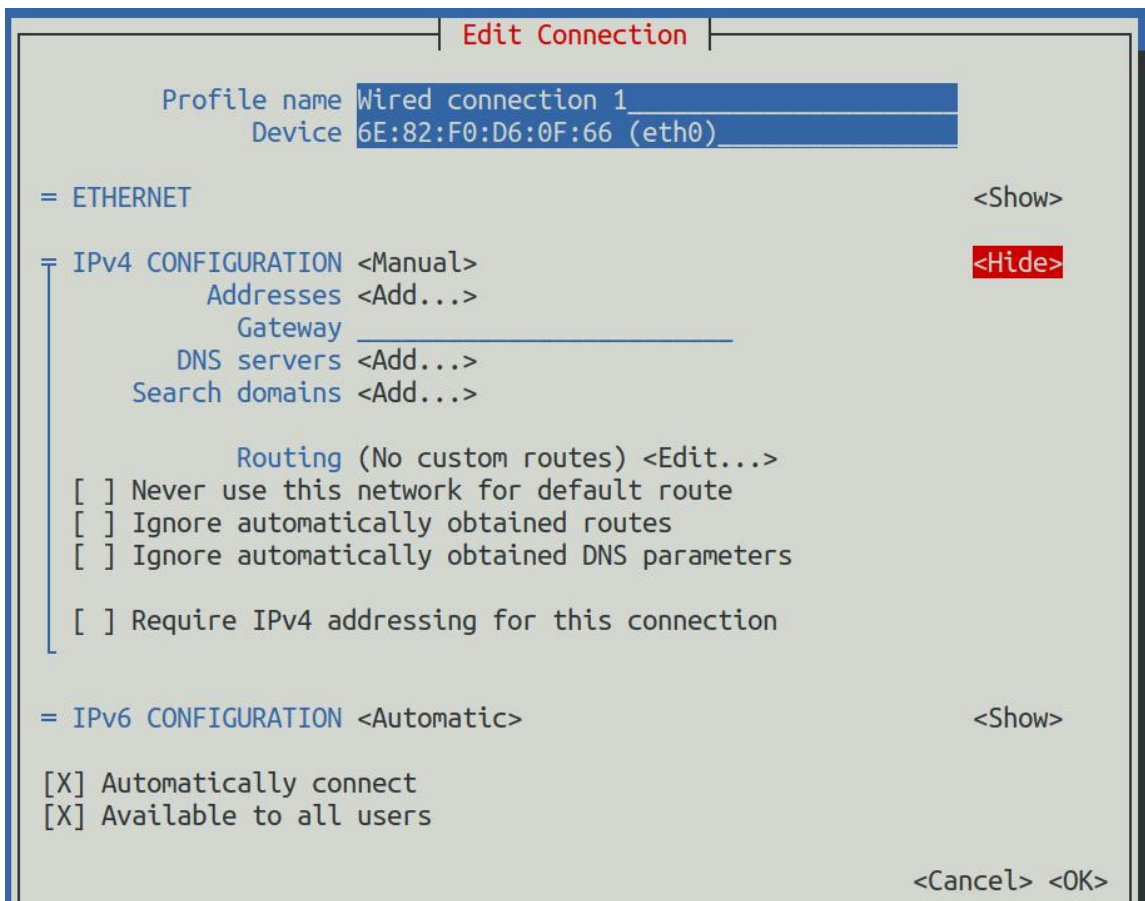
7) 选择完后的显示如下图所示:



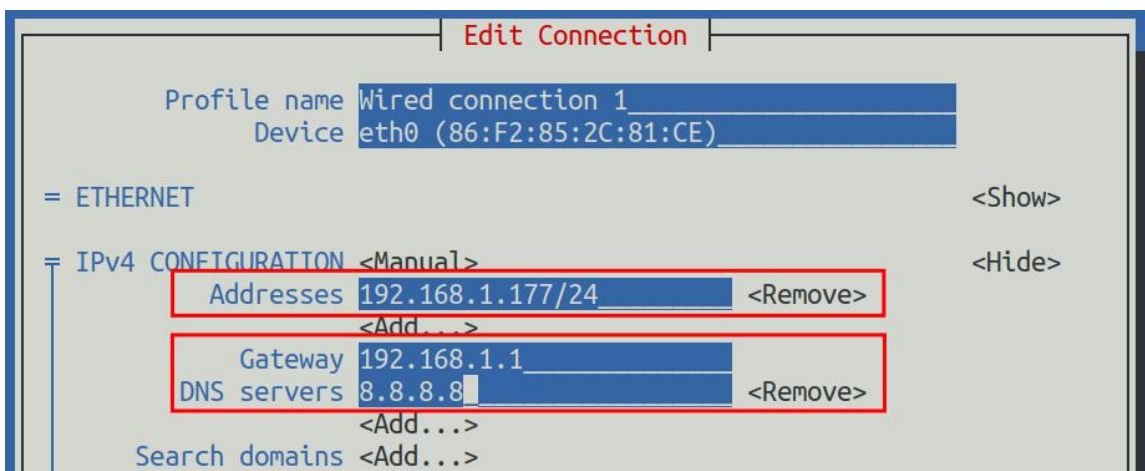
8) 然后通过 Tab 键将光标移动到<Show>。



9) 然后回车，回车后会弹出下面的设置界面。

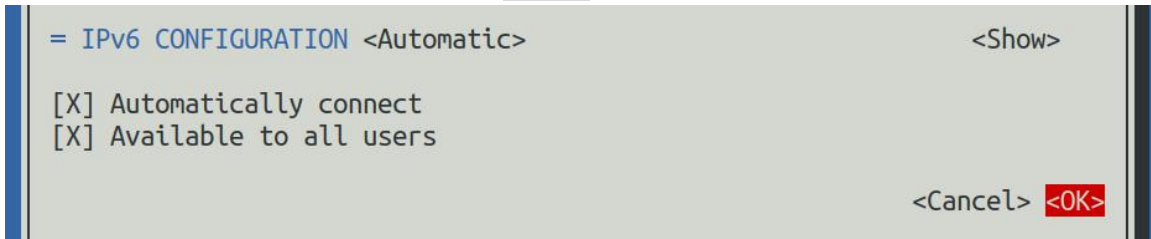


10) 然后就可以在下图所示的位置设置 IP 地址(Addresses)、网关(Gateway)和 DNS 服务器的地址（里面还有很多其他设置选项，请自行探索），**请根据自己的具体需求来设置，下图中设置的值只是一个示例。**

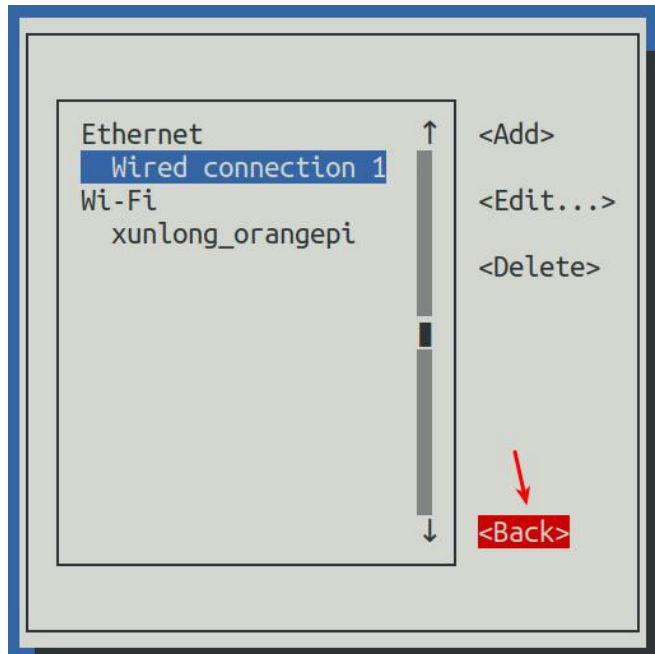




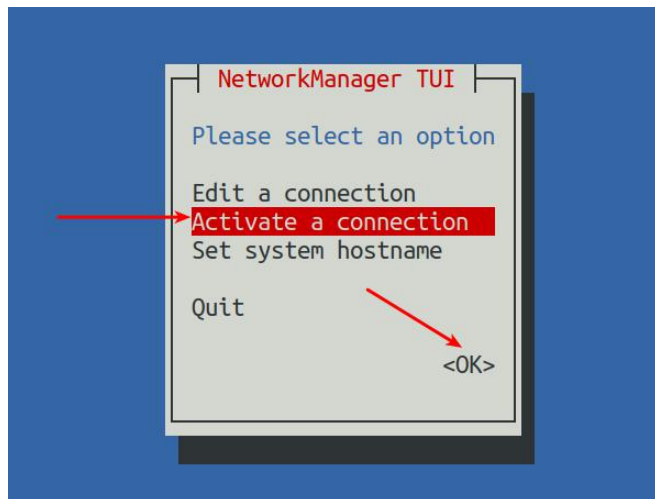
11) 设置完后将光标移动到右下角的 **<OK>**，然后回车确认。



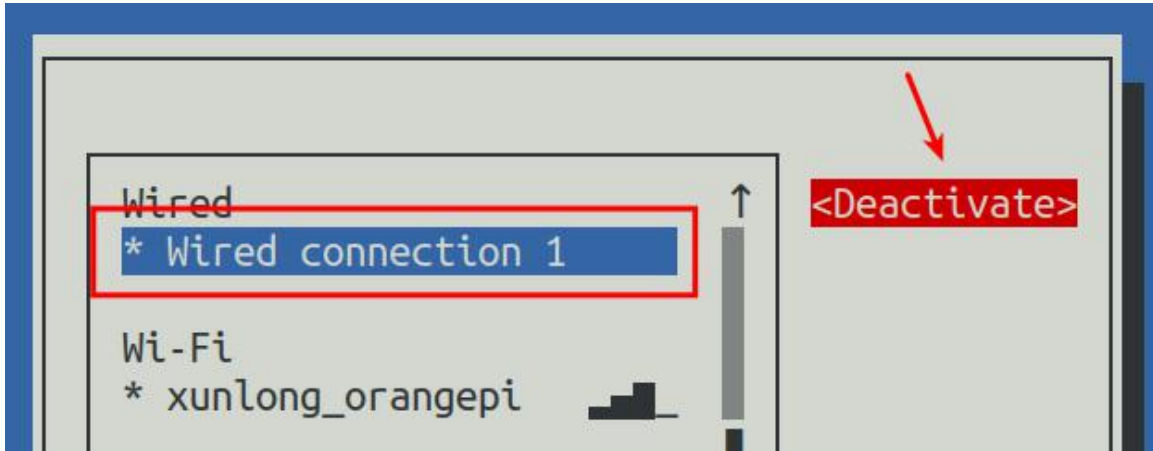
12) 然后点击 **<Back>** 回退到上一级选择界面。



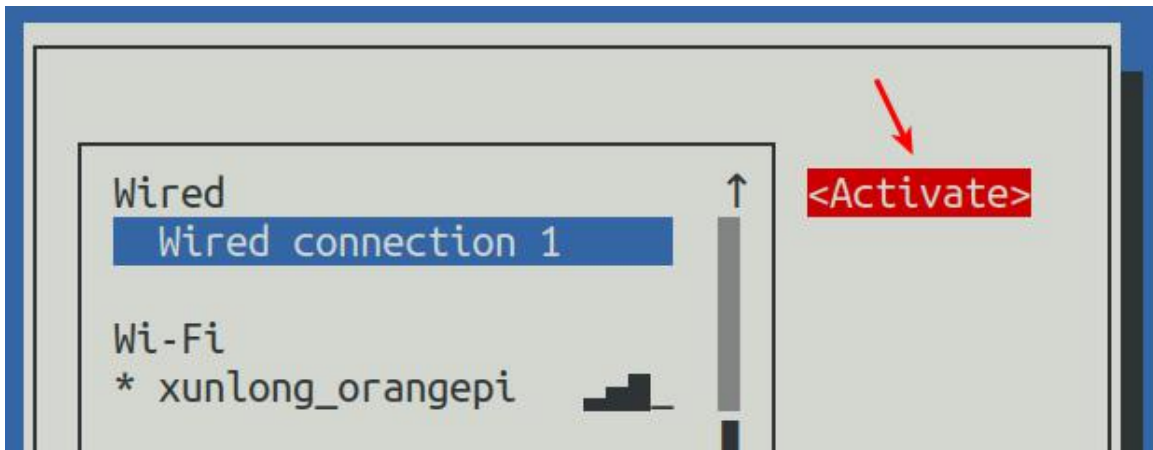
13) 然后选择 **Activate a connection**，再将光标移动到 **<OK>**，最后点击回车。



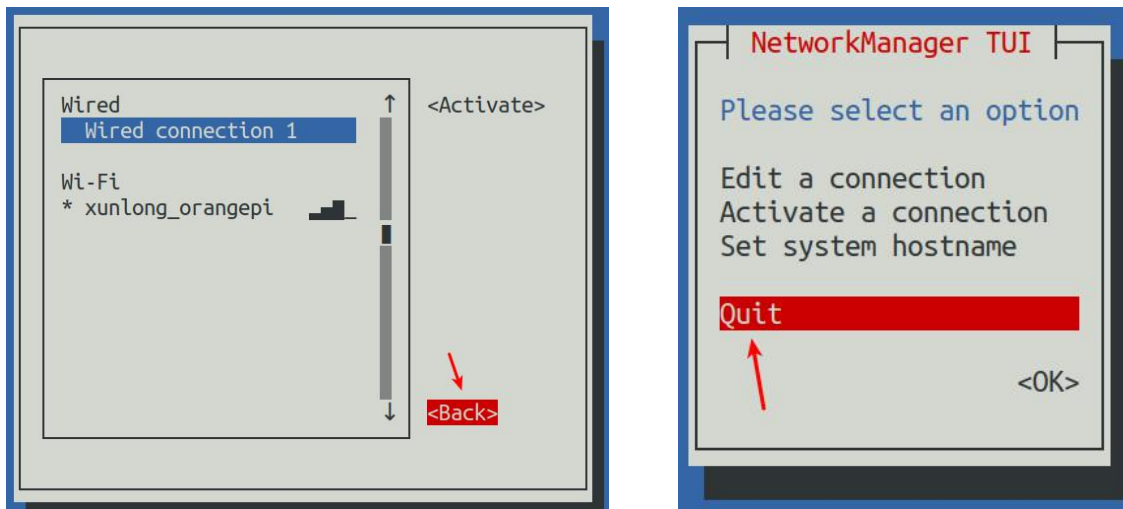
14) 然后选择需要设置的网络接口，比如 **Wired connection 1**，然后将光标移动到 **<Deactivate>**，再按下回车键禁用 **Wired connection 1**。



15) 然后请不要移动光标，再按下回车键重新使能 **Wired connection 1**，这样前面设置的静态 IP 地址就会生效了。



16) 然后通过 **<Back>** 和 **Quit** 按钮就可以退出 nmtui。



17) 然后通过 `ip a s eth0` 就能看到网口的 IP 地址已经变成前面设置的静态 IP 地址了。

```
(base) HwHiAiUser@orangepiaipro:~$ ip a s eth0
```

18) 然后就可以测试网络的连通性来检查 IP 地址是否配置 OK 了，`ping` 命令可以通过 `Ctrl+C` 快捷键来中断运行。

```
(base) HwHiAiUser@orangepiaipro:~$ ping 192.168.x.xxx -I eth0
```

### 3.5.3.2. 使用 nmcli 命令来设置静态 IP 地址

1) 如果要设置网口的静态 IP 地址，请先将网线插入开发板，如果需要设置 WIFI 的静态 IP 地址，请先连接好 WIFI，然后再开始设置静态 IP 地址。

2) 然后通过 `nmcli con show` 命令可以查看网络设备的名字，如下所示：

- a. `orangepi` 为 WIFI 网络接口的名字（名字不一定相同）。
- b. `Wired connection 1` 为以太网接口的名字。

```
(base) HwHiAiUser@orangepiaipro:~$ nmcli con show
NAME          UUID                                TYPE  DEVICE
orangepi      cfc4f922-ae48-46f1-84e1-2f19e9ec5e2a  wifi  wlan0
Wired connection 1 9db058b7-7701-37b8-9411-efc2ae8bfa30  ethernet  eth0
```

3) 然后输入下面的命令，其中：

- a. `"Wired connection 1"` 表示设置以太网口的静态 IP 地址，如果需要设置



WIFI 的静态 IP 地址，请修改为 WIFI 网络接口对应的名字（通过 `nmcli con show` 命令可以获取到）。

- b. `ipv4.addresses` 后面是要设置的静态 IP 地址，可以修改为自己想要设置的值。
- c. `ipv4.gateway` 表示网关的地址。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo nmcli con mod "Wired connection 1" \
ipv4.addresses "192.168.1.110" \
ipv4.gateway "192.168.1.1" \
ipv4.dns "8.8.8.8" \
ipv4.method "manual"
```

4) 然后重启 Linux 系统。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo reboot
```

5) 然后重新进入 Linux 系统使用 `ip addr show eth0` 命令就可以看到 IP 地址已经设置为想要的值了。

```
(base) HwHiAiUser@orangepiaipro:~$ ip addr show eth0
```

### 3.6. SSH 远程登录开发板

Linux 系统默认都开启了 ssh 远程登录，并且允许 root 用户登录系统。ssh 登录前首先需要确保以太网或者 wifi 网络已连接，然后使用 `ip addr` 命令或者通过查看路由器的方式获取开发板的 IP 地址。

#### 3.6.1. Ubuntu 下 SSH 远程登录开发板

1) 获取开发板的 IP 地址。

2) 然后就可以通过 ssh 命令远程登录 Linux 系统。

```
test@test:~$ ssh root@192.168.2.xxx          (需要替换为开发板的 IP 地址，不要照抄)
root@192.168.2.xx's password:              (在这里输入密码，默认密码为 Mind@123)
```

注意，输入密码的时候，**屏幕上是不会显示输入的密码的具体内容的**，请不要以为是有有什么故障，输入完后直接回车即可。

如果提示拒绝连接，只要使用的是 Orange Pi 提供的镜像，**就请不要怀疑**

**Mind@123 这个密码是不是不对，而是要找其他原因。**

3) 成功登录系统后的显示如下图所示：

```
orangepi@orangepi-M600:~$ ssh root@192.168.2.100
root@192.168.2.100's password:
Welcome to Orange Pi Ai Pro
This system is based on Ubuntu 22.04 LTS (GNU/Linux 5.10.0+ aarch64)

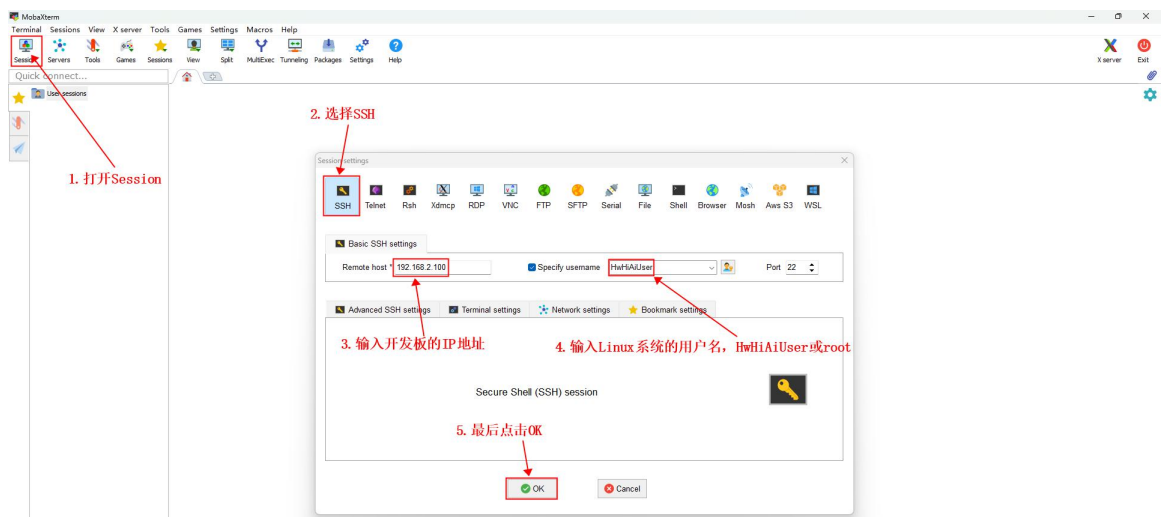
This system is only applicable to individual developers and cannot be used for commercial purposes.

By using this system, you have agreed to the Huawei Software License Agreement.
Please refer to the agreement for details on https://www.hiascend.com/software/protocol

Last login: Fri Jan 26 16:55:15 2024 from 192.168.2.220
-bash: warning: setlocale: LC_ALL: cannot change locale (en_US.UTF-8)
(base) root@orangepiaipro:~#
```

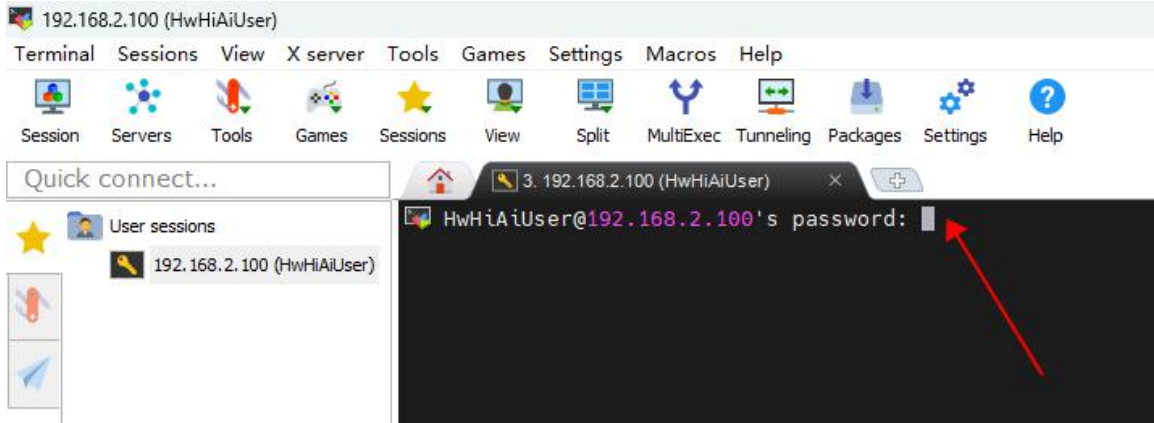
### 3.6.2. Windows 下 SSH 远程登录开发板

- 1) 首先获取开发板的 IP 地址。
- 2) 在 Windows 下可以使用 MobaXterm 远程登录开发板，首先新建一个 ssh 会话。
  - a. 打开 **Session**。
  - b. 然后在 **Session Setting** 中选择 **SSH**。
  - c. 然后在 **Remote host** 中输入开发板的 IP 地址。
  - d. 然后在 **Specify username** 中输入 Linux 系统的用户名 **root** 或 **HwHiAiUser**。
  - e. 最后点击 **OK** 即可。

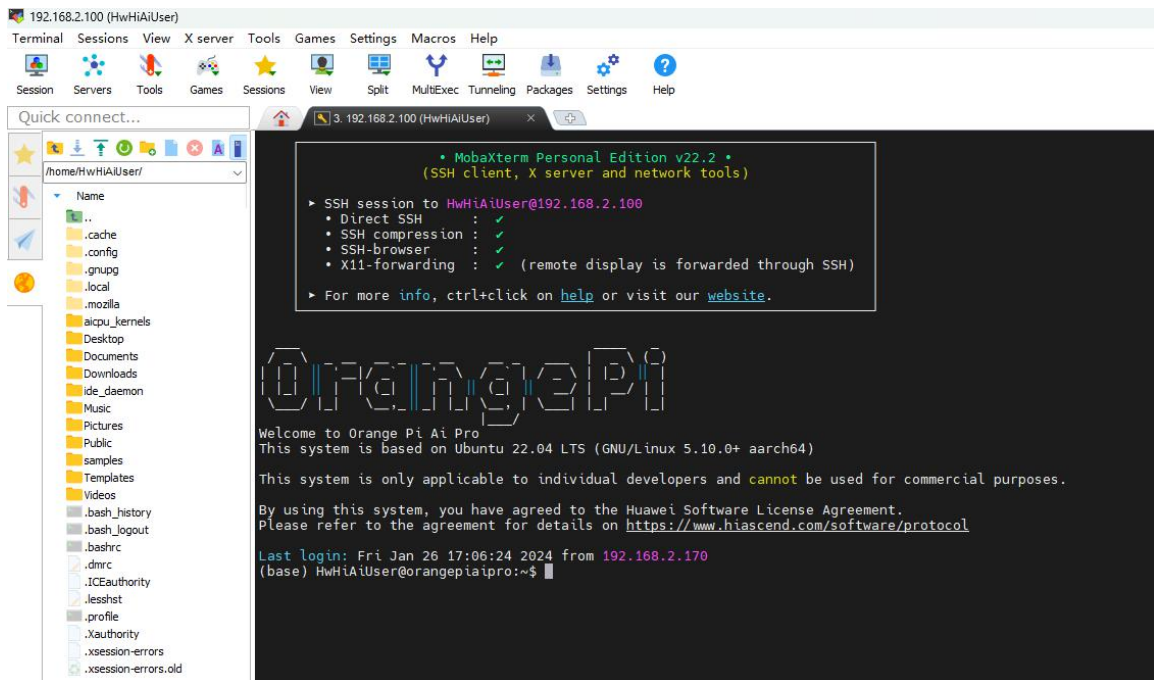


3) 然后会提示输入密码，默认 **root** 和 **HwHiAiUser** 用户的密码都为 **Mind@123**。

注意，输入密码的时候，**屏幕上是不会显示输入的密码的具体内容的**，请不要以为是有什么故障，输入完后直接回车即可。



4) 成功登录系统后的显示如下图所示



## 3.7. HDMI 接口的使用说明

### 3.7.1. HDMI 显示 Linux 桌面的说明

开发板有两个 HDMI2.0 接口，目前只有 HDMI0 支持显示 Linux 系统的桌面，

HDMI0 显示 Linux 系统桌面的详细说明请查看[登录 Linux 系统桌面的方法](#)一小节的说明。

### 3.7.2. 使用 HDMI 接口显示图片和播放音频的方法

当 Linux 系统的桌面系统关闭时，HDMI0 和 HDMI1 还可以用于 NVR 二次开发场景输出图片。

测试 HDMI0 输出一张图片的方法如下所示：

- 1) 首先连接 HDMI0 接口到 HDMI 显示器。
- 2) 然后切换到 root 用户，并进入 HDMI0 测试程序所在路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/hdmi0_pic
(base) root@orangepiaipro:/opt/opi_test/hdmi0_pic# ls
sample_hdmi test.sh update_dt.sh ut_1920x1080_nv12.yuv
```

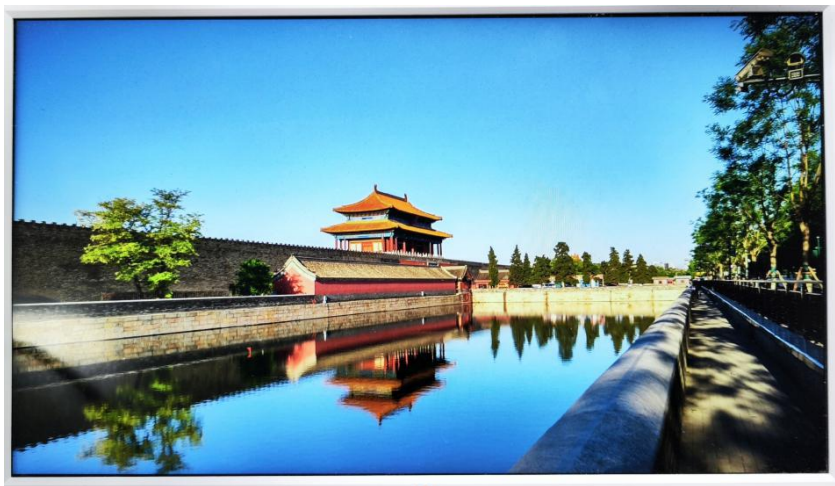
- 3) 然后运行 **update\_dt.sh** 脚本更新下 dt.img（关闭 drm 的配置，打开 vdp 的配置）。**update\_dt.sh** 脚本运行完后会自动重启 Linux 系统。

```
(base) root@orangepiaipro:/opt/opi_test/hdmi0_pic# ./update_dt.sh
```

- 4) 重启后再次进入 HDMI0 测试程序所在的路径，然后运行 **test.sh** 脚本就会播放一张图片到 HDMI 显示器（默认显示 10 秒），并且同时会播放一段音频到 HDMI 显示器，如果 HDMI 显示器支持播放音频的话，还能听到声音。

```
(base) root@orangepiaipro:/opt/opi_test/hdmi0_pic# ./test.sh
```

- 5) HDMI 显示的图片如下所示：



测试 HDMI1 输出一张图片的方法和 HDMI0 是一样的，只是测试程序的路径

为:

```
/opt/opi_test/hdmi_pic
```

测试完 HDMI 播放图片后，再切换回 HDMI0 显示 Linux 系统桌面的方法如下所示:

1) 首先切换到 root 用户，并进入 `/opt/opi_test/hdmi_desktop` 目录。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/hdmi_desktop
(base) root@orangepiaipro:/opt/opi_test/hdmi_desktop# ls
update_dt.sh
```

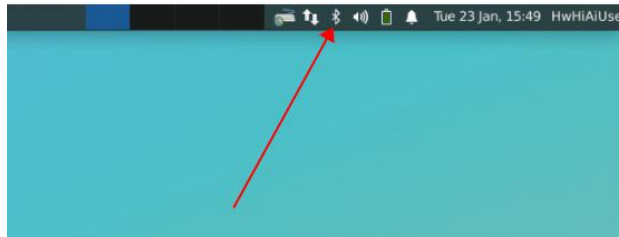
2) 然后运行 `update_dt.sh` 脚本更新下 dt.img（关闭 vdp 的配置，打开 drm 的配置）。`update_dt.sh` 脚本运行完后会自动重启 Linux 系统。

```
(base) root@orangepiaipro:/opt/opi_test/hdmi_desktop# ./update_dt.sh
```

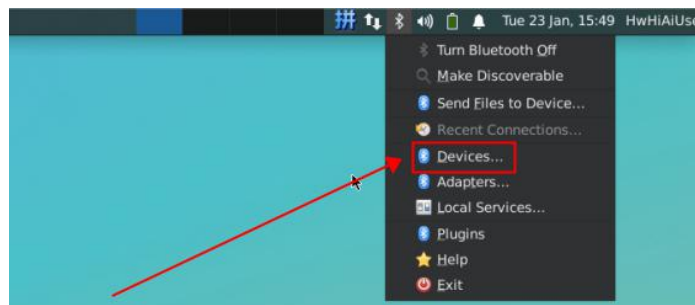
3) 重启后就能看到 HDMI 显示器会显示 Linux 系统的桌面了。

### 3.8. 蓝牙使用方法

1) 点击桌面右上角的蓝牙图标

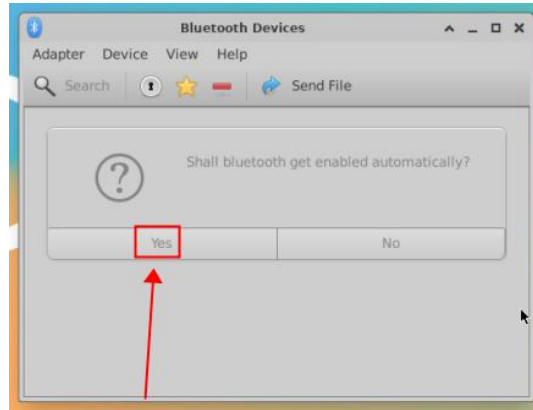


2) 然后打开蓝牙设备的配置界面

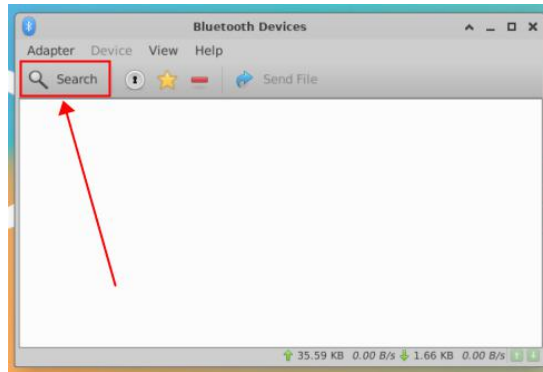


3) 然后在下面的界面中选择 **Yes**

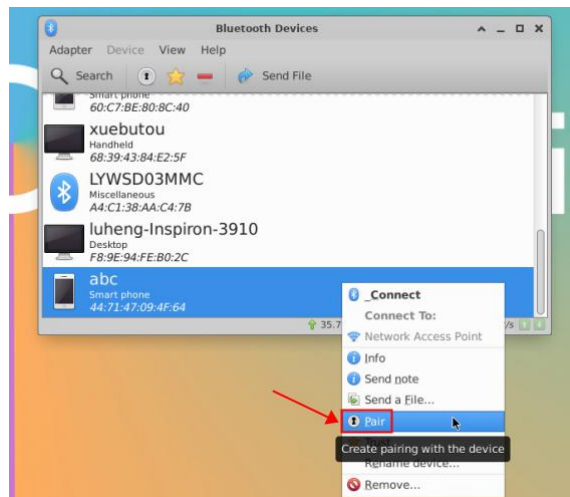




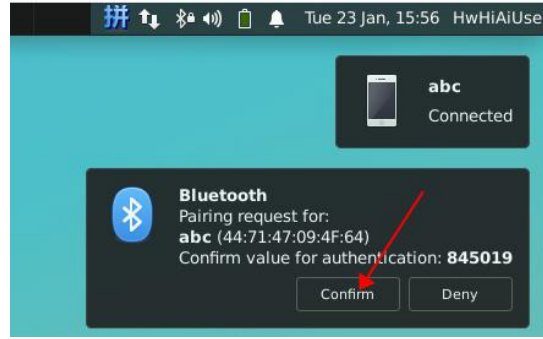
4) 点击 **Search** 即可开始扫描周围的蓝牙设备



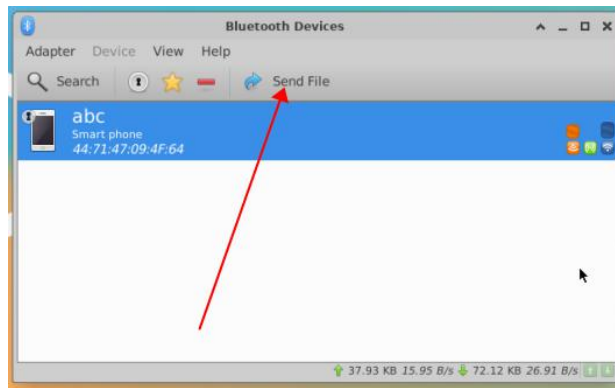
5) 然后选择想要连接的蓝牙设备，再点击鼠标右键就会弹出对此蓝牙设备的操作界面，选择 **Pair** 即可开始配对，这里演示的是和 Android 手机配对。



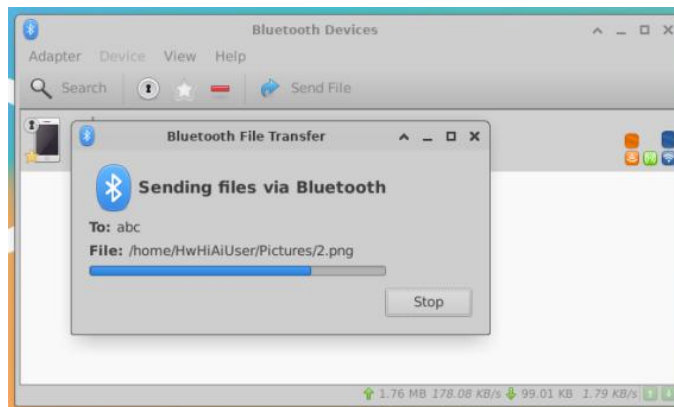
6) 配对时，桌面的右上角会弹出配对确认框，选择 **Confirm** 确认即可，此时手机上也同样需要进行确认。



7) 和手机配对完后，可以选择已配对的蓝牙设备，然后选择 **Send a File** 即可开始给手机发送一张图片。



8) 发送图片的界面如下所示：



## 3.9. USB 接口测试

### 3.9.1. Type-C USB 接口使用注意事项

开发板有一个 Type-C USB 接口，其所在位置如下图所示：



需要注意的是，Type-C USB 接口只有 USB3.0 的功能，没有 USB2.0 和 USB1.1 的功能（Soc 没有多余的 D+ 和 D- 信号线可以用），所以无法接 USB 鼠标和键盘以及 USB2.0 的存储设备。测试此接口时，可以使用下图所示的 Type-C 转 USB3.0 转接线，来连接一个 USB3.0 的存储设备。



### 3.9.2. 连接 USB 鼠标或键盘测试

开发板有两个 USB3.0 HOST 接口可以接鼠标和键盘，其所在位置如下所示。如果 USB 接口不够用的话，还可以通过 USB Hub 来扩展 USB 接口的数量。



将 USB 接口的鼠标和键盘插入开发板的 USB 3.0 接口中，然后连接开发板到的 HDMI0 接口到 HDMI 显示器，等 HDMI 显示器显示 Linux 系统的桌面后就可以使用鼠标键盘来操作 Linux 系统了。

### 3.9.3. USB 摄像头测试

1) 首先将 USB 摄像头插入到开发板的 USB3.0 HOST 接口中。



2) 然后通过 `v4l2-ctl` 命令就可以看到 USB 摄像头的设备节点信息为 `/dev/video0`

```
(base) HwHiAiUser@orangepiaipro:~$ sudo apt-get update
(base) HwHiAiUser@orangepiaipro:~$ sudo apt-get install -y v4l-utils
(base) HwHiAiUser@orangepiaipro:~$ sudo v4l2-ctl --list-devices
Q8 HD Webcam: Q8 HD Webcam (usb-xhci-hcd.3.auto-1):
    /dev/video0
    /dev/video1 #这个是用来采集 metadata 的，先忽略。
    /dev/media0
```

注意 `v4l2` 中的 `l` 是小写字母 `l`，不是数字 `1`。  
另外 `video` 的序号不一定是 `video0`，请以实际看到的为准。

3) 使用 `fswebcam` 测试 USB 摄像头

a. 安装 `fswebcam`

```
(base) HwHiAiUser@orangepiaipro:~$ sudo apt-get update
(base) HwHiAiUser@orangepiaipro:~$ sudo apt-get install -y fswebcam
```

b. 安装完 `fswebcam` 后可以使用下面的命令来拍照

- a) `-d` 选项用于指定 USB 摄像头的设备节点
- b) `--no-banner` 用于去除照片的水印
- c) `-r` 选项用于指定照片的分辨率
- d) `-S` 选项用于设置于跳过前面的帧数
- e) `./image.jpg` 用于设置生成的照片的名字和路径

```
(base) HwHiAiUser@orangepiaipro:~$ sudo fswebcam -d /dev/video0 --no-banner -r 1280x720 -S 5 ./image.jpg
```

c. 然后就可以通过 HDMI 显示器在 Linux 桌面直接打开查看拍摄的图片。

4) 使用内置的 USBCamera 样例代码测试 USB 摄像头。

a. 首先进入 USBCamera 样例代码的路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/USBCamera
(base) root@orangepiaipro:/opt/opi_test/USBCamera# ls
main main.cpp readme.md
```

b. 然后运行下面的命令就可以使用 USB 摄像头拍一张照片：

```
(base) root@orangepiaipro:/opt/opi_test/USBCamera# ./main /dev/video0
```

c. 运行成功后，在 USBCamera 样例目录下会生成一个 yuyv422 格式、1280\*720 分辨率的 **out.yuv** 文件。

```
(base) root@orangepiaipro:/opt/opi_test/USBCamera# ls
main main.cpp out.yuv readme.md
```

d. 然后在 Linux 桌面中使用下面的命令可以查看 **out.yuv** 文件的内容。

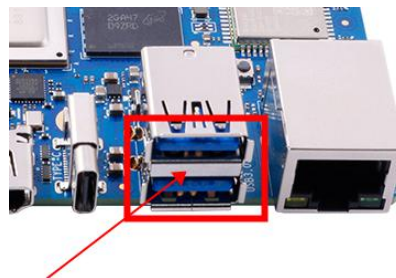
```
(base) root@orangepiaipro:/opt/opi_test/USBCamera# ffplay -pix_fmt yuyv422 -video_size 1280*720 out.yuv
```

### 3.9.4. USB 音频测试

1) 首先需要准备一个带录音功能的 USB 接口的耳机。



2) 然后将 USB 接口的耳机插入开发的 USB 接口中。





3) 然后使用 **arecord -l** 命令查看下录音设备的编号，如下面的输出所示，其中 **card 0** 中的 **0** 表示录音设备编号为 **0**。

```
(base) HwHiAiUser@orangepiaipro:~$ arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: Audio [AB13X USB Audio], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

4) 然后进入 USB 音频测试代码的路径中。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/USBAudio
(base) root@orangepiaipro:/opt/opi_test/USBAudio# ls
Readme.md main main.c
```

5) 然后使用下面的命令可以使用 USB 音频设备录制一段音频。其中 **0** 表示录音设备编号，需根据实际情况填写。

```
(base) root@orangepiaipro:/opt/opi_test/USBAudio# ./main plughw:0
```

6) 录制结束后，在终端界面输入 **over** 即可退出录制。

```
(base) root@orangepiaipro:/opt/opi_test/USBAudio# ./main plughw:0
Start record!
over      #输入 over 结束录制音频。
(base) root@orangepiaipro:/opt/opi_test/USBAudio#
```

7) 录音成功后，在 USBAudio 样例目录下会生成音频文件 **audio.pcm**。

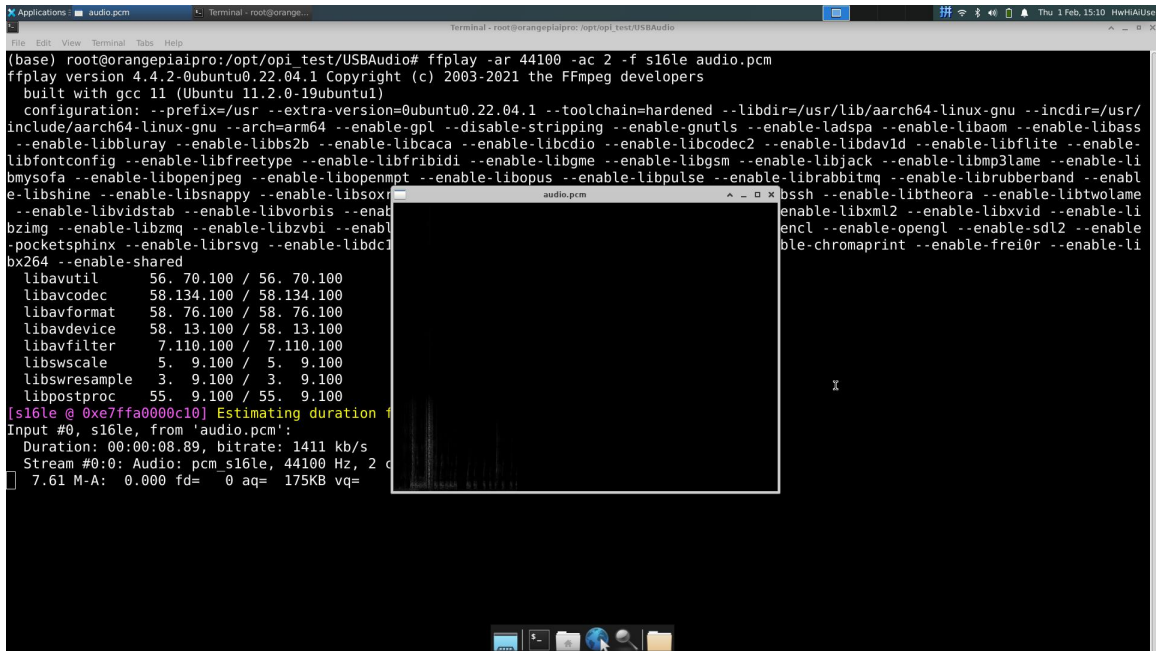
```
(base) root@orangepiaipro:/opt/opi_test/USBAudio# ls *.pcm
audio.pcm
```

8) 然后确保使用 **aplay -l** 命令能看到 USB 的播音设备。

```
(base) root@orangepiaipro:/opt/opi_test/USBAudio# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: Audio [AB13X USB Audio], device 0: USB Audio [USB Audio]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

9) 然后在 Linux 系统桌面中，使用下面的命令可以将录制的音频播放到 USB 耳机。

```
(base) root@orangeipipro:/opt/opi_test/USBAudio# ffplay -ar 44100 -ac 2 -f s16le audio.pcm
```

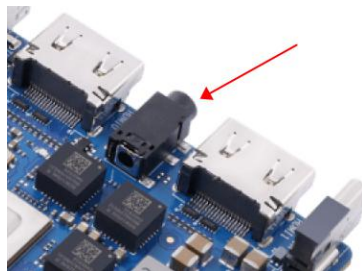


### 3. 10. 音频测试

**Linux 内核没有适配耳机和 HDMI 等的 ALSA 音频驱动，此部分驱动还在开发中，目前只能通过音频样例代码来测试耳机、HDMI 的音频播放和板载 MIC 的录音功能。**

#### 3. 10. 1. 耳机接口播放音频测试

1) 首先将耳机插入开发板的 3.5mm 耳机接口中。



2) 然后进入音频测试程序所在的目录中。

```
(base) HwHiAiUser@orangeipipro:~$ sudo -i
(base) root@orangeipipro:~# cd /opt/opi_test/audio
```

```
(base) root@orangepiaipro:/opt/opi_test/audio# ls
capture.sh play.sh qzgy_48k_16_mono_30s.pcm sample_audio
```

3) 然后使用下的命令就可以播放测试音频到耳机了。

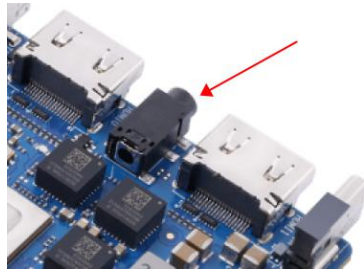
```
(base) root@orangepiaipro:/opt/opi_test/audio# ./sample_audio play 2 qzgy_48k_16_mono_30s.pcm
```

### 3. 10. 2. HDMI 音频播放测试

请参考[使用 HDMI 接口显示图片和播放音频的方法](#)一小节的说明。

### 3. 10. 3. 耳机 MIC 录音测试

1) 首先将带 MIC 功能的耳机插入开发板的 3.5mm 耳机接口中。



2) 然后进入音频测试程序所在的目录中。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/audio
(base) root@orangepiaipro:/opt/opi_test/audio# ls
capture.sh play.sh qzgy_48k_16_mono_30s.pcm sample_audio
```

3) 然后可以使用下面的命令录制一段 5 秒钟的音频。

```
(base) root@orangepiaipro:/opt/opi_test/audio# ./sample_audio capture test.pcm
```

4) 录音完成后会在当前目录下生成一个 **test.pcm** 的录音文件。

```
(base) root@orangepiaipro:/opt/opi_test/audio# ls test.pcm
test.pcm
```

5) 然后使用下面的命令可以将录制的音频文件播放到耳机。

```
(base) root@orangepiaipro:/opt/opi_test/audio# ./sample_audio play 2 test.pcm
```



### 3.11. MIPI LCD 屏幕的测试说明

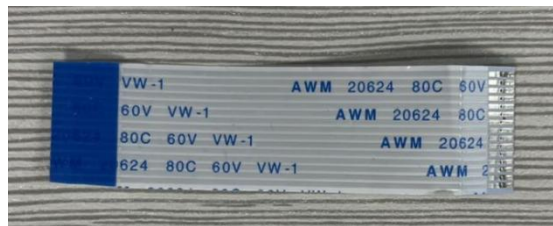
#### 3.11.1. 树莓派 5 寸屏幕的组装方法

1) 首先准备需要的配件。

a. 树莓派 5 寸 MIPI LCD 显示屏。



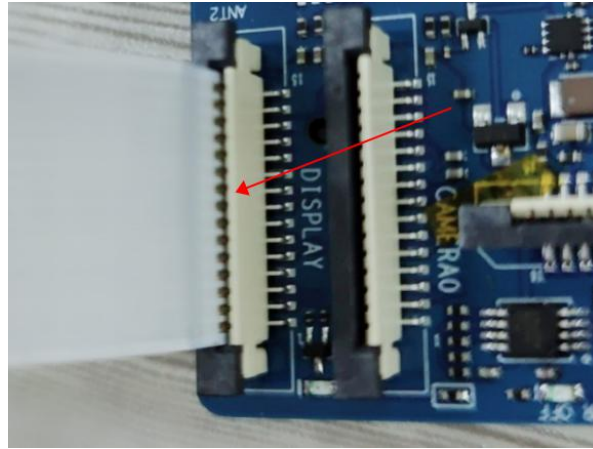
b. 15pin MIPI 排线。



2) 然后将 15pin MIPI 排线按照下图所示的方式连接到树莓派 5 寸屏幕上（注意绝缘面的朝向）。



3) 最后将 LCD 屏幕连接到开发板的 MIPI LCD 接口上。



### 3. 11. 2. 输出图片到 LCD 屏幕的方法

使用 MIPI LCD 屏幕显示 Linux 系统桌面的功能还在开发中。目前 MIPI LCD 屏幕只能用样例程序显示一张图片，步骤如下所示：

1) 首先连接好树莓派 5 寸 LCD 屏幕，具体方法情况参考[树莓派 5 寸屏幕的组装方法](#)一小节的说明。

2) 然后进入测试程序所在的路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/lcd
(base) root@orangepiaipro:/opt/opi_test/lcd# ls
raspberry_demo test.sh update_dt.sh ut_800x480_nv12.yuv
```

3) 然后运行 **update\_dt.sh** 脚本更新下 dt.img（关闭 drm 的配置，打开 vdp 的配置）。**update\_dt.sh** 脚本运行完后会自动重启 Linux 系统。

```
(base) root@orangepiaipro:/opt/opi_test/lcd# ./update_dt.sh
```

4) 重启后再次进入 LCD 测试程序所在的路径，然后运行 **test.sh** 脚本就会播放一张图片到 LCD 屏幕。

```
(base) root@orangepiaipro:/opt/opi_test/lcd# ./test.sh
```



5) 使用 **Ctrl+C** 快捷键可以停止显示图片。

6) 测试完 LCD 播放图片后，再切换回 HDMI0 显示 Linux 系统桌面的方法如下所示：

a. 首先切换到 root 用户，并进入 `/opt/opi_test/hdmi_desktop` 目录。

```
(base) root@orangepiaipro:~# cd /opt/opi_test/hdmi_desktop
(base) root@orangepiaipro:/opt/opi_test/hdmi_desktop# ls
update_dt.sh
```

b. 然后运行 `update_dt.sh` 脚本更新下 `dt.img`（关闭 `vdp` 的配置，打开 `drm` 的配置）。`update_dt.sh` 脚本运行完后会自动重启 Linux 系统。

```
(base) root@orangepiaipro:/opt/opi_test/hdmi_desktop# ./update_dt.sh
```

c. 重启后就能看到 HDMI 显示器会显示 Linux 系统的桌面了。

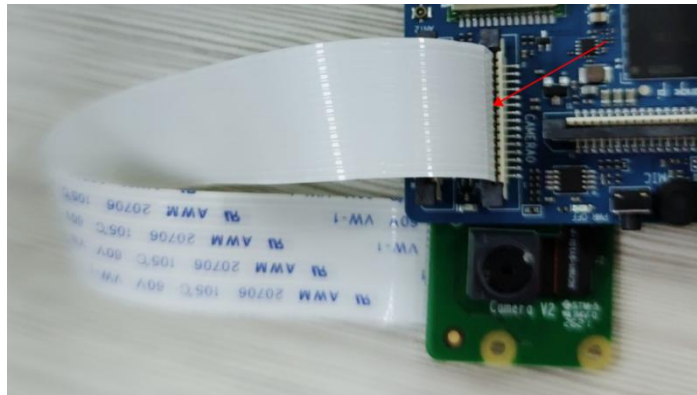
### 3.12. MIPI 摄像头的测试说明

目前 MIPI 摄像头测试只能用样例程序拍一张图片，步骤如下所示：

1) 首先需要准备一个树莓派 IMX219 摄像头。



2) 然后将摄像头连接到开发板的 MIPI 摄像头接口中。测试一个摄像头拍照时，请使用 CAMERA0 接口。



3) 然后登录 Linux 系统的桌面，打开一个终端，再进入测试程序所在的路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/camera
(base) root@orangepiaipro:/opt/opi_test/camera# ls
sample_hdmi test_one.sh test_two.sh update_dt.sh vi_l1_sample
```

4) 然后运行下面的命令就会使用 CAMERA0 拍摄一张照片。

```
(base) root@orangepiaipro:/opt/opi_test/camera# ./vi_l1_sample 1 1 1
```

5) 拍摄好的图片如下所示：

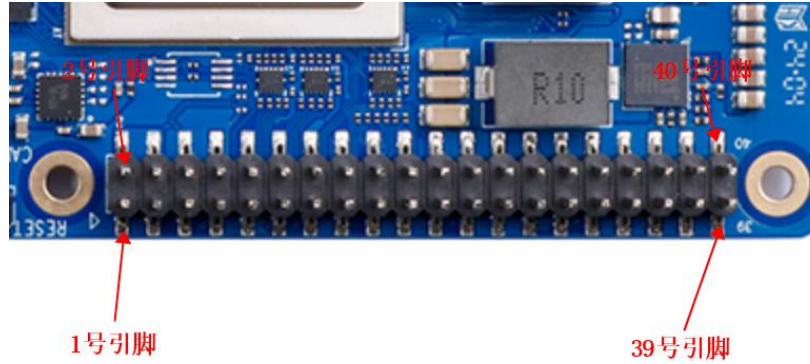
```
(base) root@orangepiaipro:/opt/opi_test/camera# ls *.yuv
vi_pipe0_chn0_w1920_h1080.yuv
```

6) 然后在 Linux 桌面中使用 ffplay 命令可以查看下拍摄的图片。

```
(base) root@orangepiaipro:/opt/opi_test/camera# ffplay -pix_fmt yuv420p -video_size 1920*1080 ./vi_pipe0_chn0_w1920_h1080.yuv
```

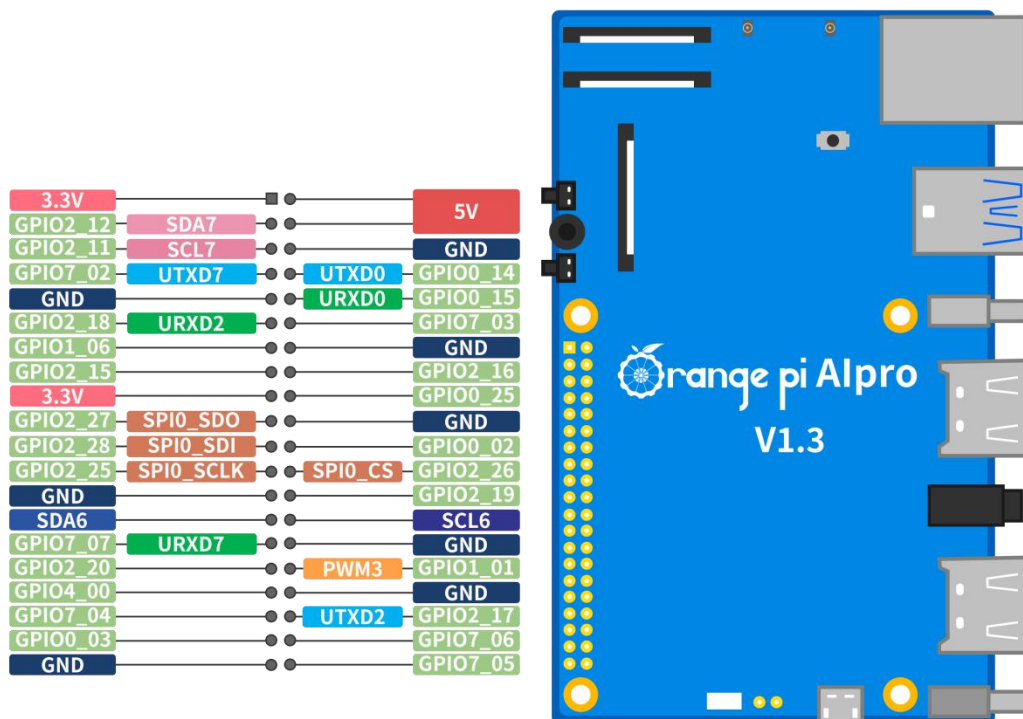
### 3.13. 40 Pin 接口引脚功能说明

开发板的 40 pin 接口引脚的顺序如下图所示：



开发板 40 pin 接口引脚的功能如下表所示：

GPIO序号	GPIO	功能	引脚	引脚	功能	GPIO	GPIO序号
		3.3V	1	2	5V		
76	GPIO2_12	SDA7	3	4	5V		
75	GPIO2_11	SCL7	5	6	GND		
226	GPIO7_02	UTXD7	7	8	UTXD0	GPIO0_14	14
		GND	9	10	URXD0	GPIO0_15	15
82	GPIO2_18	URXD2	11	12		GPIO7_03	227
38	GPIO1_06		13	14	GND		
79	GPIO2_15		15	16		GPIO2_16	80
		3.3V	17	18		GPIO0_25	25
91	GPIO2_27	SPI0_SDO	19	20	GND		
92	GPIO2_28	SPI0_SDI	21	22		GPIO0_02	2
89	GPIO2_25	SPI0_SCLK	23	24	SPI0_CS	GPIO2_26	90
		GND	25	26		GPIO2_19	83
		SDA6	27	28	SCL6		
231	GPIO7_07	URXD7	29	30	GND		
84	GPIO2_20		31	32	PWM3	GPIO1_01	33
128	GPIO4_00		33	34	GND		
228	GPIO7_04		35	36	UTXD2	GPIO2_17	81
3	GPIO0_03		37	38		GPIO7_06	230
		GND	39	40		GPIO7_05	229



40 pin 接口使用注意事项如下所示：

- 1) 40 pin 接口中总共有 26 个 GPIO 口，但 8 号和 10 号引脚默认是用于调试串口功能的，并且这两个引脚和 Micro USB 调试串口是连接在一起的，所以这两个引脚请不要设置为 GPIO 等功能。
- 2) 所有的 GPIO 口的电压都是 3.3v。
- 3) 40 pin 接口中 27 号和 28 号引脚只有 I2C 的功能，没有 GPIO 等其他复用功能，另外这两个引脚的电压默认都为 1.8v。

### 3.14. 40 pin 接口 GPIO、I2C、UART、SPI 和 PWM 测试

#### 3.14.1. 40 pin GPIO 口的测试方法

开发板 40 pin 接口引脚的功能如下表所示，其中标红部分的引脚默认配置为 GPIO 功能，可以直接使用，其他具有 GPIO 复用功能的引脚需要修改 DTS 配置才能正常使用 GPIO 的功能。

GPIO序号	GPIO	功能	引脚
		3.3V	1
76	GPIO2_12	SDA7	3

引脚	功能	GPIO	GPIO序号
2	5V		
4	5V		



75	GPI02_11	SCL7	5
226	GPI07_02	UTXD7	7
		GND	9
82	GPI02_18	URXD2	11
38	GPI01_06		13
79	GPI02_15		15
		3.3V	17
91	GPI02_27	SPI0_SDO	19
92	GPI02_28	SPI0_SDI	21
89	GPI02_25	SPI0_SCLK	23
		GND	25
		SDA6	27
231	GPI07_07	URXD7	29
84	GPI02_20		31
128	GPI04_00		33
228	GPI07_04		35
3	GPI00_03		37
		GND	39

6	GND		
8	UTXD0	GPI00_14	14
10	URXD0	GPI00_15	15
12		GPI07_03	227
14	GND		
16		GPI02_16	80
18		GPI00_25	25
20	GND		
22		GPI00_02	2
24	SPI0_CS	GPI02_26	90
26		GPI02_19	83
28	SCL6		
30	GND		
32	PWM3	GPI01_01	33
34	GND		
36	UTXD2	GPI02_17	81
38		GPI07_06	230
40		GPI07_05	229

Linux 镜像中预装了 **gpio\_operate** 工具用于设置 GPIO 管脚的输入与输出方向，也可将每个 GPIO 管脚独立的设为 0 或 1。**gpio\_operate** 工具的详细使用方法如下所示：

1) **gpio\_operate** 工具必须使用 **root** 帐号执行。

2) **gpio\_operate -h** 命令可以获取 **gpio\_operate** 工具的帮助信息：

```
(base) root@orangepiaipro:~# gpio_operate -h
Usage: gpio_operate <Command|-h> [Options...]
gpio_operate Command:
  -h                : This command's help information.
  set_value         : Set gpio pin value.
  get_value         : Get gpio pin value.
  set_direction     : Set gpio pin direction value.
  get_direction     : Get gpio pin direction value.
```

3) **gpio\_operate get\_direction gpio\_group gpio\_pin** 用于查询 GPIO 管脚方向。



a. `gpio_group` 和 `gpio_pin` 参数说明如下所示:

类型	描述
<code>gpio_group</code>	GPIO 组号, 取值为[0, 8]
<code>gpio_pin</code>	GPIO 管脚号, 取值为[0, 31]

b. 比如 40 pin 中的第 31 号引脚对应的 GPIO 为 GPIO2\_20, 那么其 GPIO 组号为 2, GPIO 管脚号为 20, 获取其方向的命令为:

```
(base) root@orangepiaipro:~# gpio_operate get_direction 2 20
Get gpio pin direction value succeeded, value is 0.
```

c. 输出的打印信息说明

字段	说明
<code>direction</code>	GPIO 管脚方向, 取值为[0, 1] <ul style="list-style-type: none"> <li>• 0: 输入方向</li> <li>• 1: 输出方向</li> </ul>

4) `gpio_operate set_direction gpio_group gpio_pin direction` 用于设置 GPIO 管脚方向。

a. `gpio_group`、`gpio_pin` 和 `direction` 参数说明如下所示:

类型	描述
<code>gpio_group</code>	GPIO 组号, 取值为[0, 8]
<code>gpio_pin</code>	GPIO 管脚号, 取值为[0, 31]
<code>direction</code>	GPIO 管脚方向, 取值为[0, 1] <ul style="list-style-type: none"> <li>• 0: 输入方向</li> <li>• 1: 输出方向</li> </ul>

b. 比如 40 pin 中的第 31 号引脚对应的 GPIO 为 GPIO2\_20, 那么其 GPIO 组号为 2, GPIO 管脚号为 20, 设置其方向为输出的命令为:

```
(base) root@orangepiaipro:~# gpio_operate set_direction 2 20 1
Set gpio pin direction value succeeded.
```

5) `gpio_operate get_value gpio_group gpio_pin` 命令用于查询 GPIO 管脚值。

a. `gpio_group` 和 `gpio_pin` 参数说明如下所示:



类型	描述
<i>gpio_group</i>	GPIO 组号, 取值为[0, 8]
<i>gpio_pin</i>	GPIO 管脚号, 取值为[0, 31]

b. 比如 40 pin 中的第 31 号引脚对应的 GPIO 为 GPIO2\_20, 那么其 GPIO 组号为 2, GPIO 管脚号为 20, 查询其管脚值的命令如下所示:

```
(base) root@orangepiaipro:~# gpio_operate get_value 2 20
Get gpio pin value succeeded, value is 0. #这里查询到的值为 0, 也就是低电平
```

6) **gpio\_operate set\_value *gpio\_group gpio\_pin value*** 命令用于设置 GPIO 管脚值为高电平或者低电平, **注意设置管脚值前, 请确保已将 GPIO 管脚的方向设置为输出了。**

a. *gpio\_group*、*gpio\_pin* 和 *value* 参数说明如下所示:

类型	描述
<i>gpio_group</i>	GPIO 组号, 取值为[0, 8]
<i>gpio_pin</i>	GPIO 管脚号, 取值为[0, 31]
<i>value</i>	GPIO 管脚值, 取值为[0, 1] 当 GPIO 管脚方向为输入方向时, 不允许设置 GPIO 管脚值。

b. 比如 40 pin 中的第 31 号引脚对应的 GPIO 为 GPIO2\_20, 那么其 GPIO 组号为 2, GPIO 管脚号为 20, 设置其输出为高电平的命令为:

```
(base) root@orangepiaipro:~# gpio_operate set_value 2 20 1
```

### 3. 14. 2. 40 pin SPI 回环测试

开发板 40 pin 接口引脚的功能如下表所示, 其中标红部分的引脚具有 SPI 功能, 并且 Linux 系统默认配置为了 SPI 功能, 可以直接使用。

GPIO序号	GPIO	功能	引脚	引脚	功能	GPIO	GPIO序号
		3.3V	1	2	5V		
76	GPIO2_12	SDA7	3	4	5V		
75	GPIO2_11	SCL7	5	6	GND		
226	GPIO7_02	UTXD7	7	8	UTXD0	GPIO0_14	14
		GND	9	10	URXD0	GPIO0_15	15

82	GPI02_18	URXD2	11
38	GPI01_06		13
79	GPI02_15		15
		3.3V	17
91	GPI02_27	SPI0_SDO	19
92	GPI02_28	SPI0_SDI	21
89	GPI02_25	SPI0_SCLK	23
		GND	25
		SDA6	27
231	GPI07_07	URXD7	29
84	GPI02_20		31
128	GPI04_00		33
228	GPI07_04		35
3	GPI00_03		37
		GND	39

12		GPI07_03	227
14	GND		
16		GPI02_16	80
18		GPI00_25	25
20	GND		
22		GPI00_02	2
24	SPI0_CS	GPI02_26	90
26		GPI02_19	83
28	SCL6		
30	GND		
32	PWM3	GPI01_01	33
34	GND		
36	UTXD2	GPI02_17	81
38		GPI07_06	230
40		GPI07_05	229

40 pin 接口中的 SPI 总线为 SPI0，测试前请先确保 `/dev` 下存在 `spidev0.0` 设备节点。

```
(base) HwHiAiUser@orangepiaipro:~$ ls /dev/spidev0.0
/dev/spidev0.0
```

然后先不短接 SPI0 的 mosi 和 miso 两个引脚，运行 `spidev_test` 的输出结果如下所示，可以看到 TX 和 RX 的数据不一致。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo spidev_test -v -D /dev/spidev0.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D |.....@.....|
RX | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 |.....|
```

然后用杜邦线短接 SPI1 的 mosi（40 pin 接口中的第 19 号引脚）和 miso（40 pin 接口中的第 21 号引脚）两个引脚再运行 `spidev_test` 的输出如下，可以看到发送和接收的数据一样，说明回环测试成功。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo spidev_test -v -D /dev/spidev0.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D | .....@.....
RX | FF FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D | .....@.....
```

### 3. 14. 3. 40 pin I2C 测试

开发板 40 pin 接口引脚的功能如下表所示，其中标红部分的引脚具有 I2C 功能，并且 Linux 系统默认配置为了 I2C 功能，可以直接使用。

GPIO序号	GPIO	功能	引脚	引脚	功能	GPIO	GPIO序号
		3.3V	1	2	5V		
76	GPIO2_12	SDA7	3	4	5V		
75	GPIO2_11	SCL7	5	6	GND		
226	GPIO7_02	UTXD7	7	8	UTXD0	GPIO0_14	14
		GND	9	10	URXD0	GPIO0_15	15
82	GPIO2_18	URXD2	11	12		GPIO7_03	227
38	GPIO1_06		13	14	GND		
79	GPIO2_15		15	16		GPIO2_16	80
		3.3V	17	18		GPIO0_25	25
91	GPIO2_27	SPI0_SDO	19	20	GND		
92	GPIO2_28	SPI0_SDI	21	22		GPIO0_02	2
89	GPIO2_25	SPI0_SCLK	23	24	SPI0_CS	GPIO2_26	90
		GND	25	26		GPIO2_19	83
		SDA6	27	28	SCL6		
231	GPIO7_07	URXD7	29	30	GND		
84	GPIO2_20		31	32	PWM3	GPIO1_01	33
128	GPIO4_00		33	34	GND		
228	GPIO7_04		35	36	UTXD2	GPIO2_17	81
3	GPIO0_03		37	38		GPIO7_06	230
		GND	39	40		GPIO7_05	229

启动 Linux 系统后，先确认下 `/dev` 下存在 `i2c6` 和 `i2c7` 的设备节点。

```
(base) HwHiAiUser@orangepiaipro:~$ ls /dev/i2c-6
/dev/i2c-6
(base) HwHiAiUser@orangepiaipro:~$ ls /dev/i2c-7
/dev/i2c-7
```

然后在 40 pin 接口的 i2c6 或者 i2c7 引脚上接一个 i2c 设备。

	i2c6	i2c7
sda 引脚	对应 40 pin 中 27 号引脚	对应 40 pin 中 3 号引脚
scl 引脚	对应 40 pin 中 28 号引脚	对应 40 pin 中 5 号引脚
3.3v 引脚	对应 40 pin 中 1 号引脚	对应 40 pin 中 1 号引脚
gnd 引脚	对应 40 pin 中 6 号引脚	对应 40 pin 中 6 号引脚

然后使用 `i2cdetect` 命令如果能检测到连接的 i2c 设备的地址，就说明 i2c 能正常使用。

1) i2c6 使用的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ sudo i2cdetect -y -r 6
```

2) i2c7 使用的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ sudo i2cdetect -y -r 7
```

不同的 i2c 设备地址是不同的，下图 0x38 地址只是一个示例。请以实际看到的为准。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo i2cdetect -y -r 7
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  38  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
(base) HwHiAiUser@orangepiaipro:~$
```

### 3.14.4. 40 pin UART 测试

开发板 40 pin 接口引脚的功能如下表所示，其中标红部分的引脚具有 uart 功能，并且 Linux 系统默认配置为了 uart 功能，可以直接使用。另外请注意 uart0 默认设

置为调试串口功能，请不要将其当成普通串口使用。

GPIO序号	GPIO	功能	引脚	引脚	功能	GPIO	GPIO序号
		3.3V	1	2	5V		
76	GPIO2_12	SDA7	3	4	5V		
75	GPIO2_11	SCL7	5	6	GND		
226	GPIO7_02	UTXD7	7	8	UTXD0	GPIO00_14	14
		GND	9	10	URXD0	GPIO00_15	15
82	GPIO2_18	URXD2	11	12		GPIO7_03	227
38	GPIO1_06		13	14	GND		
79	GPIO2_15		15	16		GPIO2_16	80
		3.3V	17	18		GPIO00_25	25
91	GPIO2_27	SPI0_SDO	19	20	GND		
92	GPIO2_28	SPI0_SDI	21	22		GPIO00_02	2
89	GPIO2_25	SPI0_SCLK	23	24	SPI0_CS	GPIO2_26	90
		GND	25	26		GPIO2_19	83
		SDA6	27	28	SCL6		
231	GPIO7_07	URXD7	29	30	GND		
84	GPIO2_20		31	32	PWM3	GPIO1_01	33
128	GPIO4_00		33	34	GND		
228	GPIO7_04		35	36	UTXD2	GPIO2_17	81
3	GPIO00_03		37	38		GPIO7_06	230
		GND	39	40		GPIO7_05	229

启动 Linux 系统后，先确认下 `/dev` 下存在 `uart` 的设备节点。

```
(base) HwHiAiUser@orangepiaipro:~$ ls /dev/ttyAMA*
/dev/ttyAMA0 /dev/ttyAMA1 /dev/ttyAMA2
```

uart 设备节点和 `uart` 对应关系如下所示：

uart 设备节点	uart 接口
<code>/dev/ttyAMA1</code>	<code>uart2</code>
<code>/dev/ttyAMA2</code>	<code>uart7</code>

然后开始测试 `uart` 接口，先使用杜邦线短接要测试的 `uart` 接口的 `rx` 和 `tx` 引脚。不同的 `uart` 的 `rx` 和 `tx` 引脚对应的 40 pin 接口中的引脚如下所示：

uart 接口	rx 引脚	tx 引脚
---------	-------	-------

<b>uart2</b>	<b>40pin 的 11 号引脚</b>	<b>40pin 的 36 号引脚</b>
<b>uart7</b>	<b>40pin 的 29 号引脚</b>	<b>40pin 的 7 号引脚</b>

然后进入串口测试程序的路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
(base) root@orangepiaipro:~# cd /opt/opi_test/uart
(base) root@orangepiaipro:/opt/opi_test/uart# ls
serial serial.c
```

串口测试程序 serial 的使用方法如下所示：

```
(base) root@orangepiaipro:/opt/opi_test/uart# ./serial
Usage: ./serial <serialport>
```

使用 serial 测试程序可以测试下串口的自收自发。serial 程序会打开对应的串口发送一个字符串——**Hello, Serial Port!**，然后打印接收到的字符串。如果自发自收的字符串相同，说明测试成功。

1) uart2 测试命令如下所示：

```
(base) root@orangepiaipro:/opt/opi_test/uart# ./serial /dev/ttyAMA1
W: Hello, Serial Port!
R: Hello, Serial Port!
```

2) uart7 测试命令如下所示：

```
(base) root@orangepiaipro:/opt/opi_test/uart# ./serial /dev/ttyAMA2
W: Hello, Serial Port!
R: Hello, Serial Port!
```

### 3. 14. 5. 40 pin PWM 测试

开发板 40 pin 接口引脚的功能如下表所示，其中标红部分的引脚具有 pwm 功能。

GPIO序号	GPIO	功能	引脚
		3.3V	1
76	GPIO2_12	SDA7	3
75	GPIO2_11	SCL7	5
226	GPIO7_02	UTXD7	7
		GND	9
82	GPIO2_18	URXD2	11

引脚	功能	GPIO	GPIO序号
2	5V		
4	5V		
6	GND		
8	UTXD0	GPIO0_14	14
10	URXD0	GPIO0_15	15
12		GPIO7_03	227



38	GPI01_06		13
79	GPI02_15		15
		3.3V	17
91	GPI02_27	SPI0_SDO	19
92	GPI02_28	SPI0_SDI	21
89	GPI02_25	SPI0_SCLK	23
		GND	25
		SDA6	27
231	GPI07_07	URXD7	29
84	GPI02_20		31
128	GPI04_00		33
228	GPI07_04		35
3	GPI00_03		37
		GND	39

14	GND		
16		GPI02_16	80
18		GPI00_25	25
20	GND		
22		GPI00_02	2
24	SPI0_CS	GPI02_26	90
26		GPI02_19	83
28	SCL6		
30	GND		
32	PWM3	GPI01_01	33
34	GND		
36	UTXD2	GPI02_17	81
38		GPI07_06	230
40		GPI07_05	229

目前只能通过操作寄存器的方式来测试 PWM3 引脚输出一个波形。测试步骤如下所示：

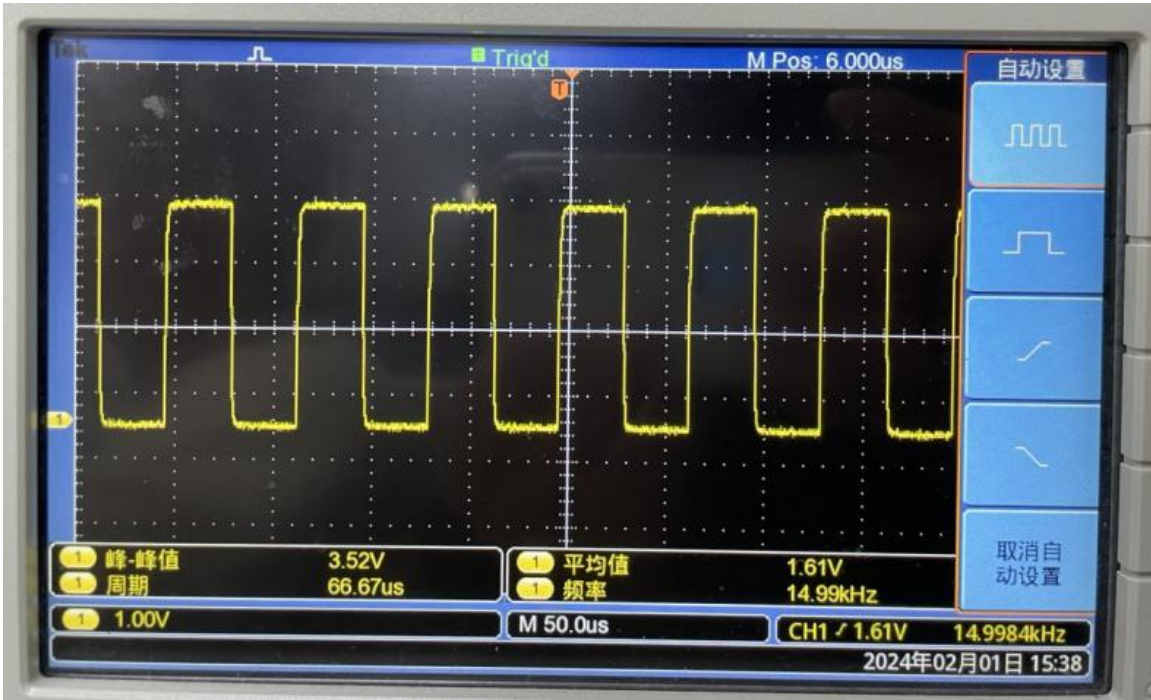
1) 首先进入 pwm 的测试代码的路径。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo -i
[sudo] password for HwHiAiUser:
(base) root@orangepiaipro:~# cd /opt/opi_test/pwm
(base) root@orangepiaipro:/opt/opi_test/pwm# ls
test.sh
```

2) 然后运行 test.sh 脚本即可输出一个 50% 占空比的方波。

```
(base) root@orangepiaipro:/opt/opi_test/pwm# ./test.sh
```

3) 然后用示波器测量 40 pin 中的第 32 号引脚就可以查看 PWM 的输出波形，如下所示：



### 3. 15. 上传文件到开发板 Linux 系统中的方法

#### 3. 15. 1. 在 Ubuntu PC 中上传文件到开发板 Linux 系统中的方法

##### 3. 15. 1. 1. 使用 scp 命令上传文件的方法

1) 使用 scp 命令可以在 Ubuntu PC 中上传文件到开发板的 Linux 系统中，具体命令如下所示：

- a. **file\_path**: 需要替换为要上传文件的路径。
- b. **root**: 为开发板 Linux 系统的用户名，也可以替换成其他的，比如 **HwHiAiUser**。
- c. **192.168.xx.xx**: 为开发板的 IP 地址，请根据实际情况进行修改。
- d. **/root**: 开发板 Linux 系统中的路径，也可以修改为其他的路径。

```
test@test:~$ scp file_path root@192.168.xx.xx:/root/
```

2) 如果要上传文件夹，需要加上 **-r** 参数。

```
test@test:~$ scp -r dir_path root@192.168.xx.xx:/root/
```



3) scp 还有更多的用法，请使用下面的命令查看 man 手册。

```
test@test:~$ man scp
```

### 3.15.1.2. 使用 filezilla 上传文件的方法

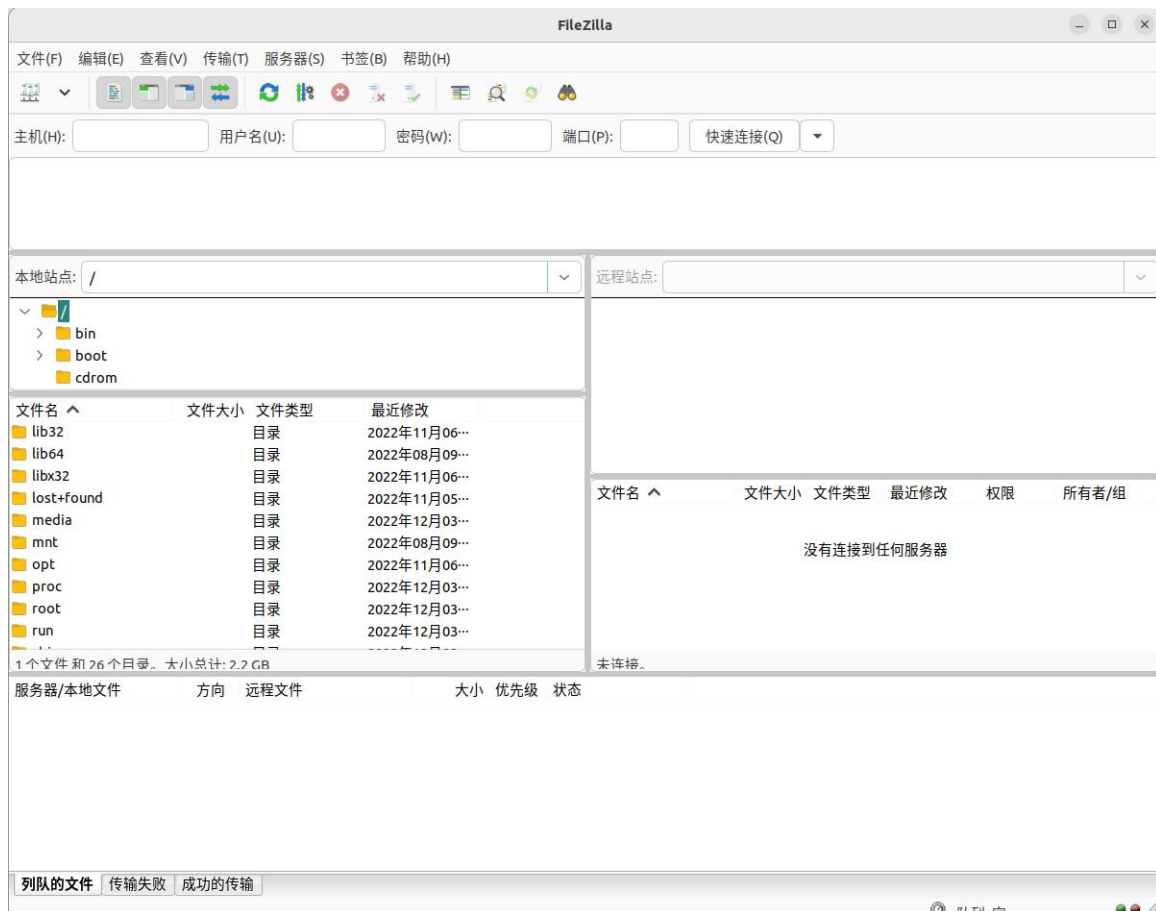
1) 首先在 Ubuntu PC 中安装 filezilla。

```
test@test:~$ sudo apt-get install -y filezilla
```

2) 然后使用下面的命令打开 filezilla。

```
test@test:~$ filezilla
```

3) filezilla 打开后的界面如下所示，此时右边远程站点下面显示的是空的。



4) 连接开发板的方法如下图所示：



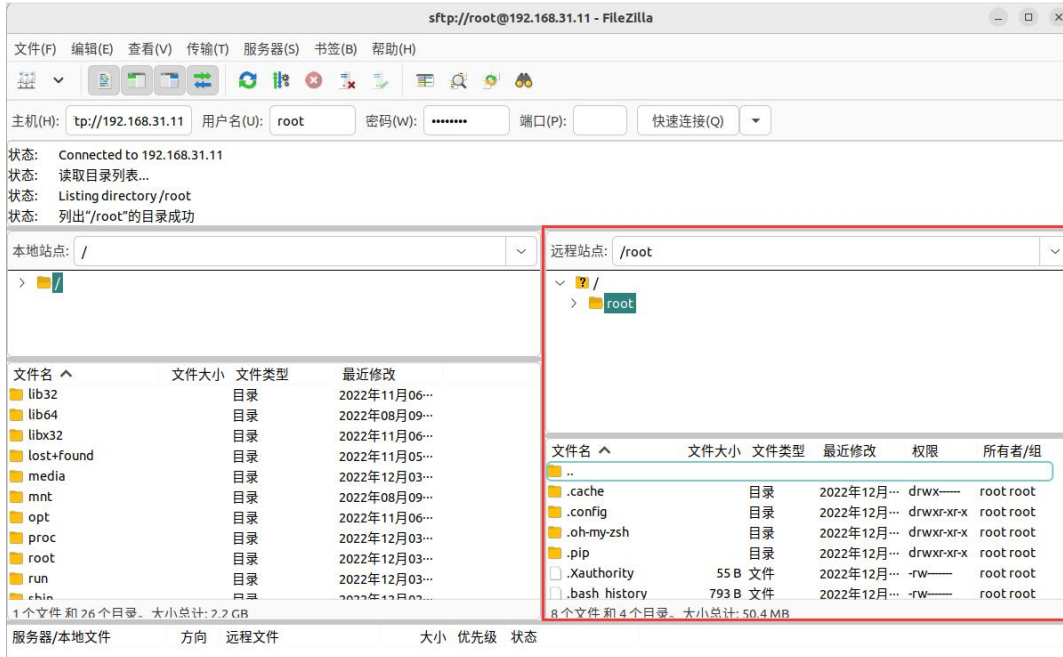
5) 然后选择**保存密码**，再点击**确定**。



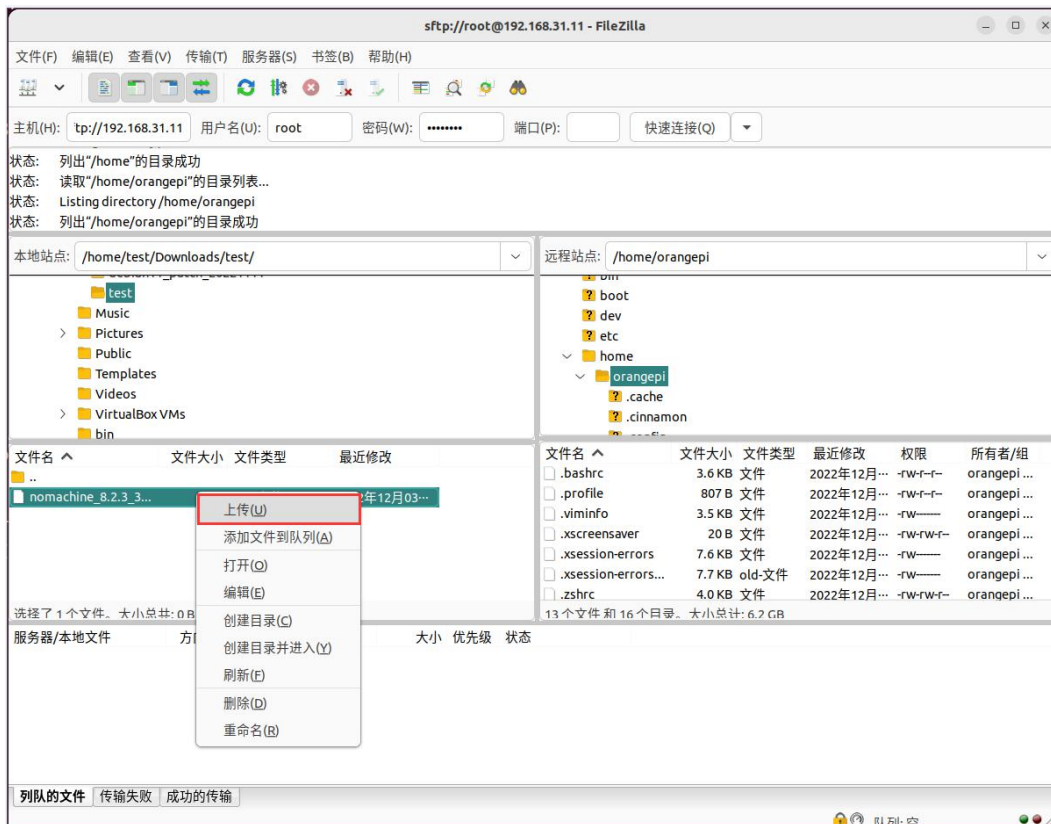
6) 然后选择**总是信任该主机**，再点击**确定**。



7) 连接成功后在 filezilla 软件的右边就可以看到开发板 linux 文件系统的目录结构了。



8) 然后在 filezilla 软件的右边选择要上传到开发板中的路径，再在 filezilla 软件的左边选中 Ubuntu PC 中要上传的文件，再点击鼠标右键，再点击上传选项就会开始上传文件到开发板中了。



9) 上传完成后就可以去开发板 Linux 系统中的对应路径中查看上传的文件了。

10) 上传文件夹的方法和上传文件的方法是一样的，这里就不再赘述了。

### 3. 15. 2. 在 Windows PC 中上传文件到开发板 Linux 系统中的方法

#### 3. 15. 2. 1. 使用 filezilla 上传文件的方法

1) 首先下载 filezilla 软件 Windows 版本的安装文件，下载链接如下所示：

<https://filezilla-project.org/download.php?type=client>



**Please select your edition of FileZilla Client**

	FileZilla	FileZilla with manual	FileZilla Pro	FileZilla Pro + CLI
Standard FTP	Yes	Yes	Yes	Yes
FTP over TLS	Yes	Yes	Yes	Yes
SFTP	Yes	Yes	Yes	Yes
Comprehensive PDF manual	-	Yes	Yes	Yes
Amazon S3	-	-	Yes	Yes
Backblaze B2	-	-	Yes	Yes
Dropbox	-	-	Yes	Yes
Microsoft OneDrive	-	-	Yes	Yes
Google Drive	-	-	Yes	Yes
Google Cloud Storage	-	-	Yes	Yes
Microsoft Azure Blob + File Storage	-	-	Yes	Yes
WebDAV	-	-	Yes	Yes
OpenStack Swift	-	-	Yes	Yes
Box	-	-	Yes	Yes
Site Manager synchronization	-	-	Yes	Yes
Command-line interface	-	-	-	Yes
Batch transfers	-	-	-	Yes

然后选择这里下载 → **Download** **Select** **Select** **Select**

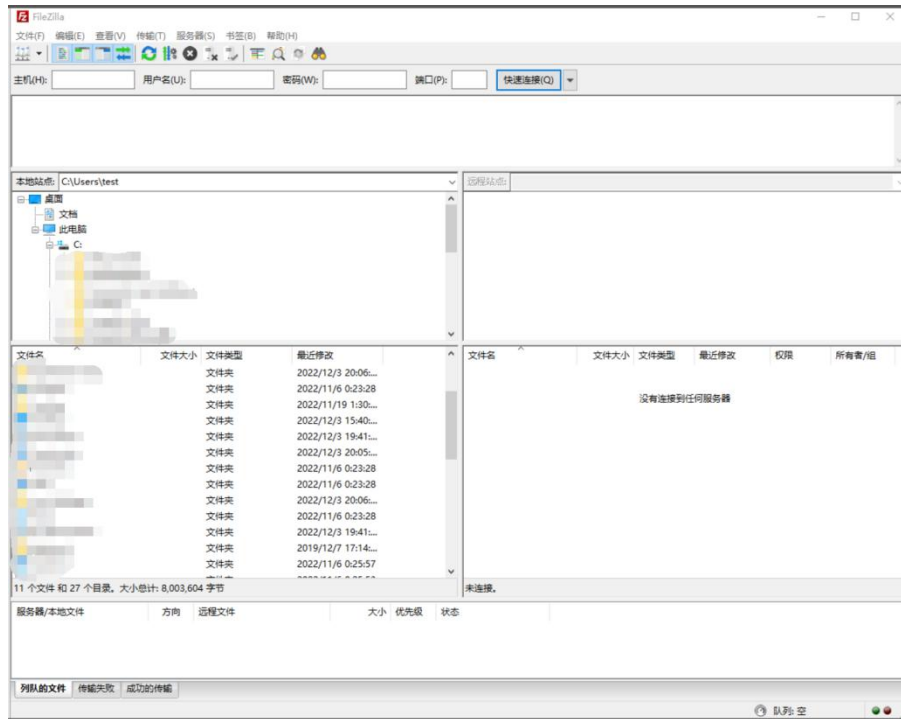
2) 下载的安装包如下所示，然后双击直接安装即可。

**FileZilla\_Server\_x.x.x\_win64-setup.exe**

安装过程中，下面的安装界面请选择 **Decline**，然后再选择 **Next>**。



3) filezilla 打开后的界面如下所示，此时右边远程站点下面显示的是空的。



4) 连接开发板的方法如下图所示：



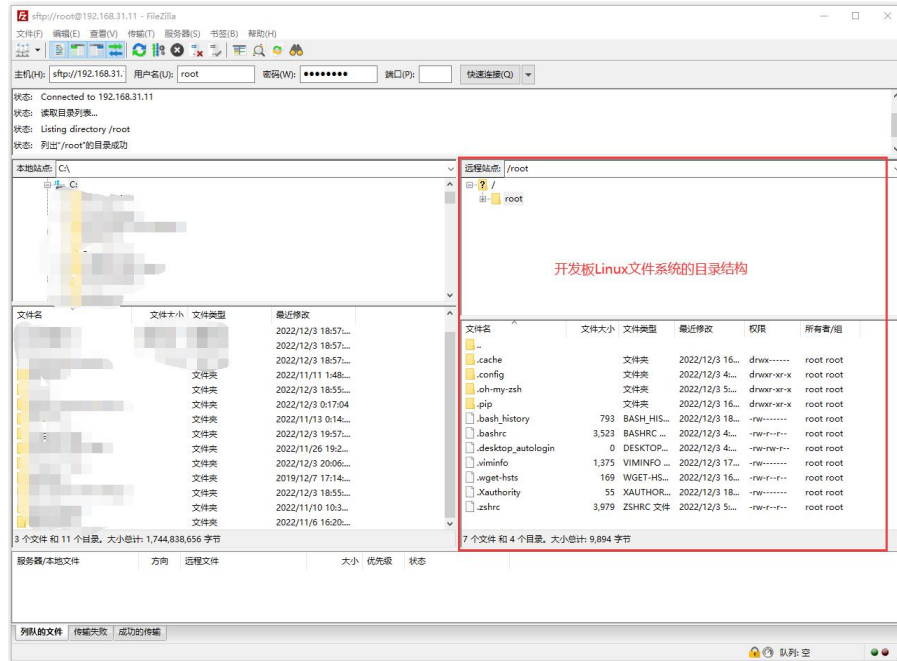
5) 然后选择**保存密码**，再点击**确定**。



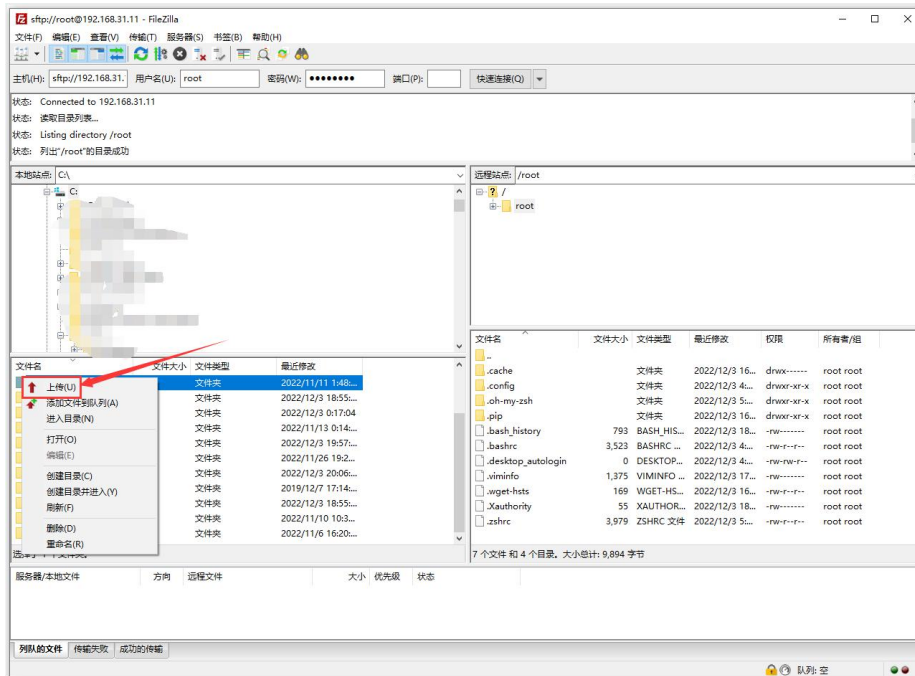
6) 然后选择**总是信任该主机**，再点击**确定**。



7) 连接成功后在 filezilla 软件的右边就可以看到开发板 linux 文件系统的目录结构了。



8) 然后在 filezilla 软件的右边选择要上传到开发板中的路径，再在 filezilla 软件的左边选中 Windows PC 中要上传的文件，再点击鼠标右键，再点击上传选项就会开始上传文件到开发板中了。

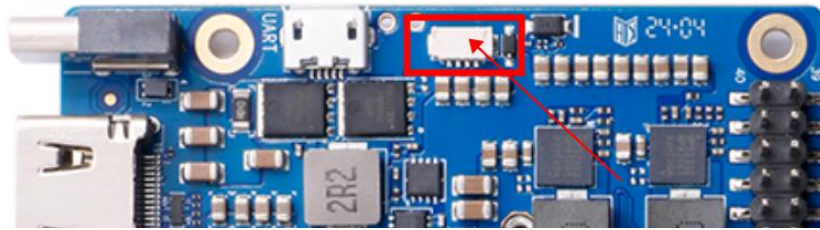


9) 上传完成后就可以去开发板 Linux 系统中的对应路径中查看上传的文件了。

10) 上传文件夹的方法和上传文件的方法是一样的，这里就不再赘述了。

### 3.16. 散热风扇的使用方法

开发板散热风扇的接口所在的位置如下所示：



开发板使用的散热风扇为 12V 的，接口为 4pin，0.8mm 间距规格。可以通过 PWM 来控制风扇的转速。

使用 `npu-smi` 命令可以查询和控制 PWM 风扇，详细用法如下所示：

1) 查询风扇模式的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ sudo npu-smi info -t pwm-mode
pwm-mode          : auto
```

字段	说明
pwm-mode	风扇模式。 有如下两种模式： <ul style="list-style-type: none"> <li>• manual: 手动模式</li> <li>• auto: 自动模式</li> </ul> 默认为自动模式。

2) 查询风扇调速比的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ sudo npu-smi info -t pwm-duty-ratio
pwm-duty-ratio(%) : 15
```

3) 设置风扇模式为手动模式的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ sudo npu-smi set -t pwm-mode -d 0
```



描述
风扇使能模式。分为手动模式、自动模式。默认为自动模式。 <ul style="list-style-type: none"> <li>• 0: 手动模式</li> <li>• 1: 自动模式</li> </ul>

4) 将风扇设置为手动模式后就可以通过下面的命令来设置风扇的调速比了。比如下面的命令会将风扇调速比设置为 100，设置完后风扇会用最大的转速运行。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo npu-smi set -t pwm-duty-ratio -d 100
```

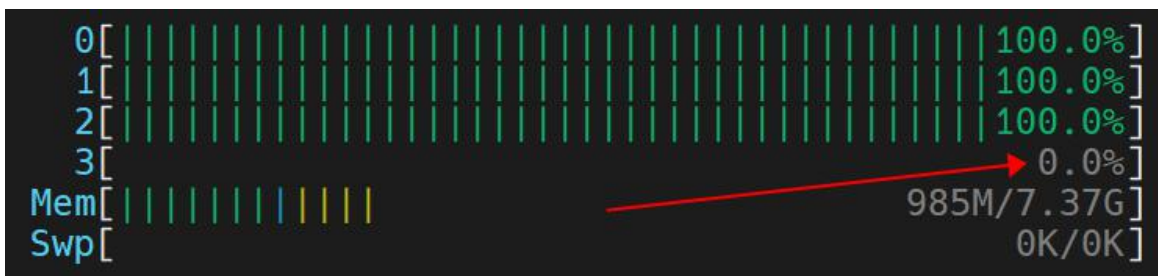
描述
风扇调速比 取值范围: [0-100]

### 3. 17. AI CPU 和 control CPU 的设置方法

开发板使用的昇腾 SOC 总共有 4 个 CPU，这 4 个 CPU 既可以设置为 control CPU，也可以设置为 AI CPU。默认情况下，control CPU 和 AI CPU 的分配数量为 3:1。使用 **npu-smi info** 命令可以查看下 control CPU 和 AI CPU 的分配数量。

```
(base) HwHiAiUser@orangepiaipro:~$ npu-smi info -t cpu-num-cfg -i 0 -c 0
Current AI CPU number      : 1
Current control CPU number : 3
Current data CPU number    : 0
(base) HwHiAiUser@orangepiaipro:~$
```

当 Linux 系统跑满后，使用 **htop** 命令会看到有一个 CPU 的占用率始终接近 0，请注意，这是正常的。因为这个 CPU 默认用于 AI CPU。





如果当前环境模型中无 AI CPU 算子，且运行业务时查询 AI CPU 占用率持续为 0，则可以将 AI CPU 的数量配置为 0。查询 AI CPU 占用率的命令如下所示：

```
(base) HwHiAiUser@orangepiaipro:~$ npu-smi info -t usages -i 0 -c 0
Memory Capacity(MB)      : 7545
Memory Usage Rate(%)     : 20
Hugepages Total(page)    : 15
Hugepages Usage Rate(%)  : 100
Aicore Usage Rate(%)     : 0
Aicpu Usage Rate(%)     : 0
Ctrlcpu Usage Rate(%)    : 1
Memory Bandwidth Usage Rate(%) : 1
```

如果不需要使用 AI CPU，使用下面的命令可以将 4 个 CPU 都设置为 control CPU。设置完后需要重启系统让配置生效。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo npu-smi set -t cpu-num-cfg -i 0 -c 0 -v 0:4:0
Status      : OK
Message : The cpu-num-cfg of the chip is set successfully. Reset system for the configuration to take effect.
```

当 4 个 CPU 都设置为 control CPU 后，再运行任务让所有 CPU 跑满，使用 htop 命令就能看到 4 个 CPU 的占用率都能达到 100%了。

```
0[|||||||||||||||||||||||||||||||||100.0%]
1[|||||||||||||||||||||||||||||||||100.0%]
2[|||||||||||||||||||||||||||||||||100.0%]
3[|||||||||||||||||||||||||||||||||100.0%]
Mem[|||||||||||||||||913M/7.37G]
Swp[|||||0K/0K]
```

### 3.18. 设置 Swap 内存的方法

虽然开发板有 8GB 或 16GB 的大内存，但有些应用需要的内存大于 8GB 或 16GB，此时我们可以通过 Swap 内存来扩展系统能使用的最大内存容量。方法如下所示：

1) 首先创建一个 swap 文件，下面的命令会创建一个 16GB 大小的 swap 文件，容量大小请根据自己的需求进行修改。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo fallocate -l 16G /swapfile
```

2) 然后修改文件权限，确保只有 root 用户可以读写。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo chmod 600 /swapfile
```

3) 然后把这个文件设置成 swap 空间。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo mkswap /swapfile
```

4) 然后启用 swap。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo swapon /swapfile
```

5) 完成以上步骤后，可以通过下面的命令可以检查 swap 内存是否已经添加成功。

```
(base) HwHiAiUser@orangepiaipro:~$ free -h
```

	total	used	free	shared	buff/cache	available
Mem:	7.4Gi	1.1Gi	5.5Gi	27Mi	835Mi	6.1Gi
Swap:	<b>15Gi</b>	0B	15Gi			

6) 如果需要 swap 设置在重启之后依然有效，请运行下面命令将对应的配置添加到 `/etc/fstab` 文件中。

```
(base) HwHiAiUser@orangepiaipro:~$ echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

### 3.19. 测试 MindSpore 的方法

开发板的 Ubuntu 系统中已经预装了 MindSpore，使用下面的方法可以测试下 MindSpore 是否能正常使用。

1) 首先请根据 [设置 Swap 内存的方法](#) 一小节的说明给系统额外添加 16GB 的 Swap 内存。

2) 然后在 **HwHiAiUser** 用户中执行下面的命令：

```
(base) HwHiAiUser@orangepiaipro:~$ python -c \
"import mindspore;mindspore.set_context(device_target='Ascend');mindspore.run_check()"
```

3) 等待一段时间后，如果能看到下面的输出就说明 MindSpore 能正常使用。

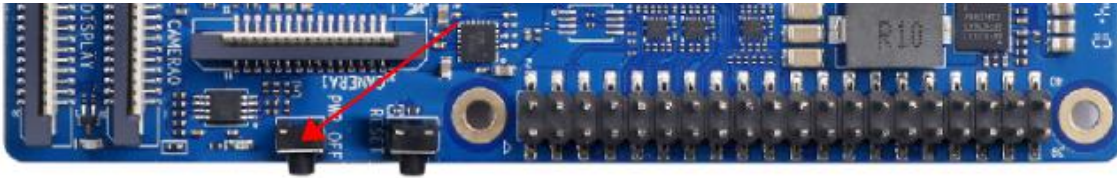
```
MindSpore version: 2.2.12.20240312
The result of multiplication calculation is correct, MindSpore has been installed on platform [Ascend] successfully!
```

### 3.20. 关机和重启开发板的方法

1) 在 Linux 系统运行的过程中，如果直接拔掉电源断电，可能会导致文件系统丢失某些数据，建议断电前先使用 **poweroff** 命令关闭开发板的 Linux 系统，然后再拔掉电源。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo poweroff
```

2) 除了 **poweroff** 命令可以关闭 Linux 系统外，还可以使用开发板上的关机按键来关闭开发板的 Linux 系统，然后再拔掉电源。请注意，关机按键是没有开机功能的。



3) 使用 **reboot** 命令即可重启开发板中的 Linux 系统。

```
(base) HwHiAiUser@orangepiaipro:~$ sudo reboot
```

## 4. 体验 AI 应用样例

我们在镜像中预装了 Jupyter Lab 软件。Jupyter Lab 软件是一个基于 web 的交互式开发环境，集成了代码编辑器、终端、文件管理器等功能，使得开发者可以在一个界面中完成各种任务。并且我们在镜像中也预置了一些可以在 Jupyter Lab 软件中运行的 AI 应用样例。这些样例都是使用 Python 编写的，并调用了 Python 版本的 AscendCL 编程接口。本章节介绍如何登录 jupyter lab 并在 jupyter lab 中运行这些预置的 AI 应用样例。

### 4.1. 登录 jupyter lab

1) 首先登录 Linux 系统桌面，然后打开终端，再切换到保存 AI 应用样例的目录下。

```
(base) HwHiAiUser@orangepiaipro:~$ cd samples/notebooks/
```

2) 在当前目录下有 9 个文件夹和 1 个 shell 文件，分别对应 9 个 AI 应用样例和 Jupyter Lab 启动脚本 **start\_notebook.sh**。

```
(base) HwHiAiUser@orangepiaipro:~/samples/notebooks$ ls
01-yolov5          06-human_protein_map_classification
02-ocr             07-Unet++
03-resnet          08-portrait_pictures
04-image-HDR-enhance 09-speech-recognition
05-cartoonGAN_picture start_notebook.sh
```

3) 然后执行 start\_notebook.sh 脚本启动 Jupyter Lab。

```
(base) HwHiAiUser@orangepiaipro:~/samples/notebooks$ ./start_notebook.sh
```

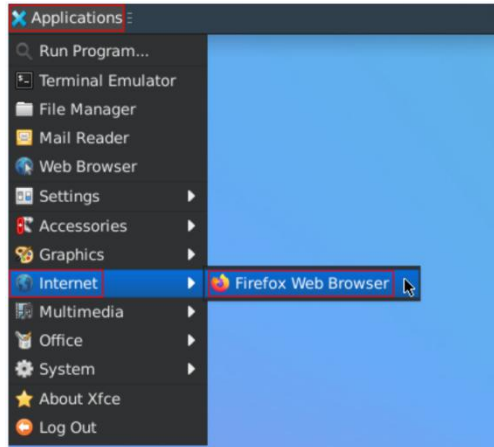
4) 在执行该脚本后，终端会出现如下打印信息，在打印信息中会有登录 Jupyter Lab 的网址链接。



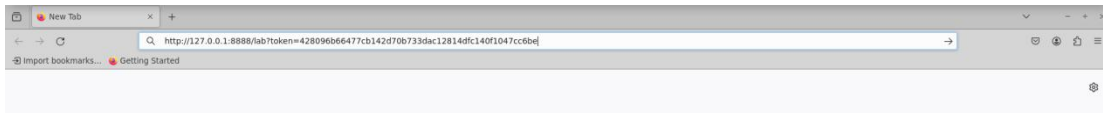
```
(base) HwHiAiUser@orangeipipro:~/samples/notebooks$ ./start notebook.sh
[W 2024-01-26 20:58:32.899 ServerApp] A `jupyter_server_extension_points` function was not found in notebook shim. Instead, a `jupyter_server_extension_paths` function was found and will be used for now. This function name will be deprecated in future releases of Jupyter Server.
[I 2024-01-26 20:58:32.902 ServerApp] jupyter lsp | extension was successfully linked.
[I 2024-01-26 20:58:32.922 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2024-01-26 20:58:32.946 ServerApp] jupyterlab | extension was successfully linked.
[I 2024-01-26 20:58:33.976 ServerApp] notebook_shim | extension was successfully linked.
[I 2024-01-26 20:58:34.068 ServerApp] notebook_shim | extension was successfully loaded.
[I 2024-01-26 20:58:34.197 ServerApp] jupyter lsp | extension was successfully loaded.
[I 2024-01-26 20:58:34.201 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2024-01-26 20:58:34.209 LabApp] JupyterLab application directory is /usr/local/miniconda3/share/jupyter/lab
[I 2024-01-26 20:58:34.212 LabApp] Extension Manager is 'pypi'.
[I 2024-01-26 20:58:34.224 ServerApp] jupyterlab | extension was successfully loaded.
[I 2024-01-26 20:58:34.226 ServerApp] Serving notebooks from local directory: /home/HwHiAiUser/samples/notebooks
[I 2024-01-26 20:58:34.226 ServerApp] Jupyter Server 2.12.5 is running at:
[I 2024-01-26 20:58:34.227 ServerApp] http://127.0.0.1:8888/lab?token=428096b66477cb142d70b733dac12814dfc140f1047cc6be
[I 2024-01-26 20:58:34.227 ServerApp] http://127.0.0.1:8888/lab?token=428096b66477cb142d70b733dac12814dfc140f1047cc6be
[I 2024-01-26 20:58:34.240 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

To access the server, open this file in a browser:
file:///home/HwHiAiUser/.local/share/jupyter/runtime/jpserver-5744-open.html
Or copy and paste one of these URLs:
http://127.0.0.1:8888/lab?token=428096b66477cb142d70b733dac12814dfc140f1047cc6be
http://127.0.0.1:8888/lab?token=428096b66477cb142d70b733dac12814dfc140f1047cc6be
[I 2024-01-26 20:58:34.339 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server-nodejs, javascript-typscript-langserver, jedi-language-server, julia-language-server, pyright, python-language-server, python-lsp-server, r-languageserver, sql-language-server, texlab, typescript-language-server, unified-language-server, vscode-css-languageserver-bin, vscode-html-languageserver-bin, vscode-json-languageserver-bin, yaml-language-server
```

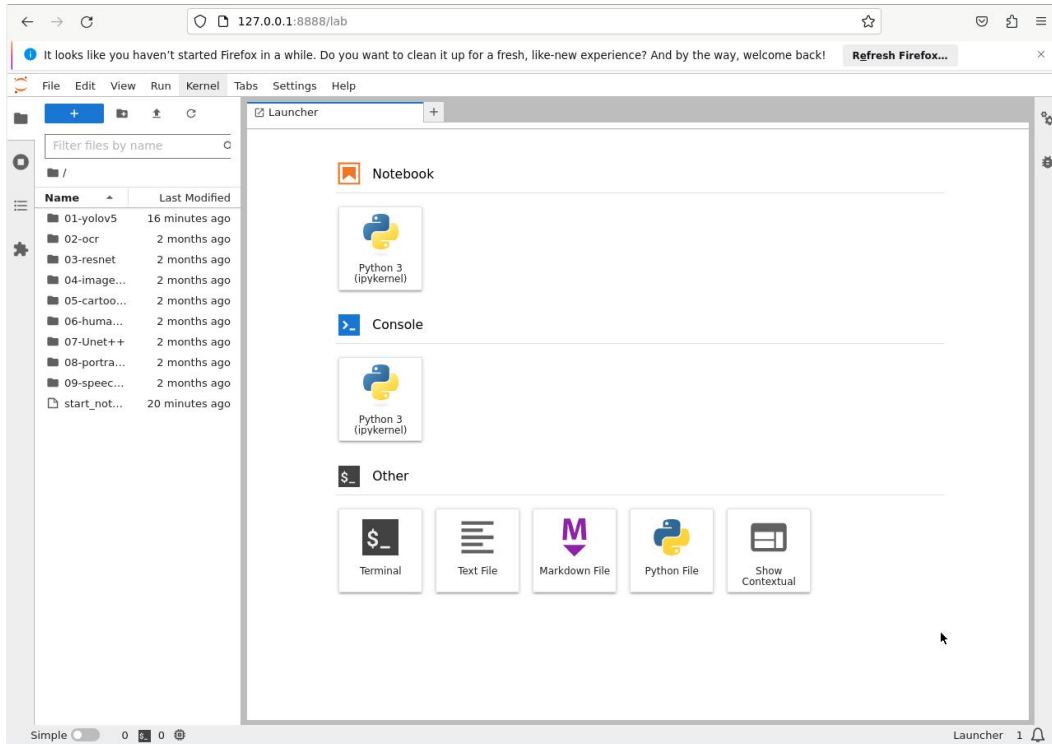
5) 然后打开火狐浏览器。



6) 再在浏览器中输入上面看到的网址链接，就可以登录 Jupyter Lab 软件了。



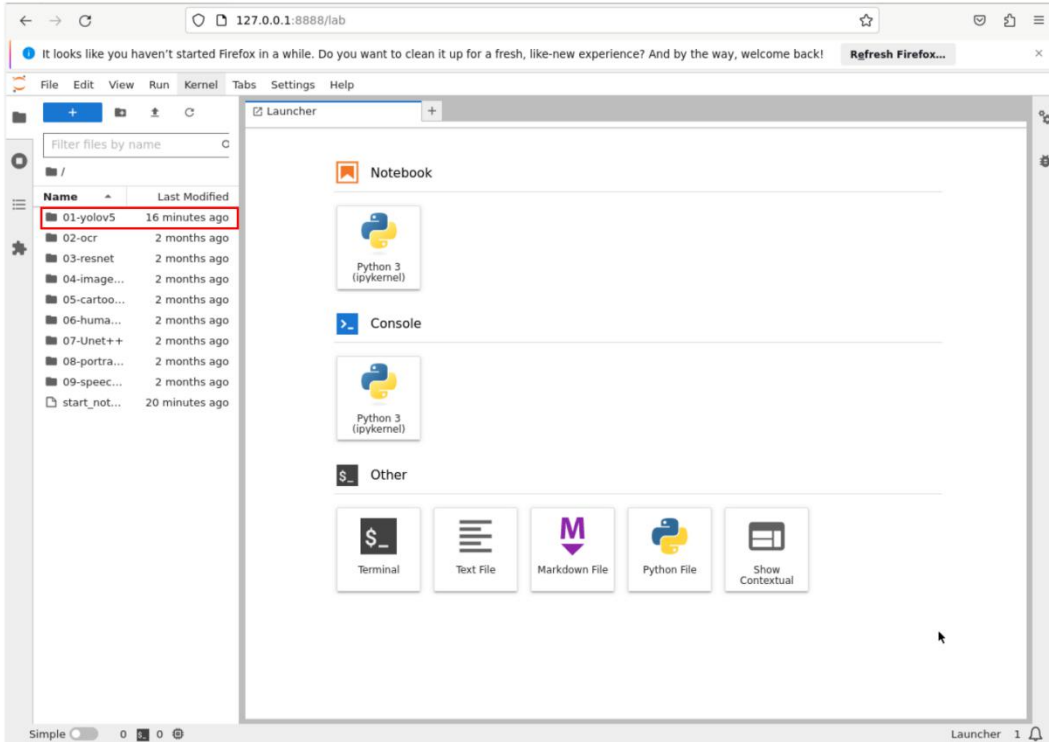
7) 登录 Jupyter Lab 后的界面如下所示，左侧文件管理器中是 9 个 AI 应用样例和 Jupyter Lab 启动脚本。



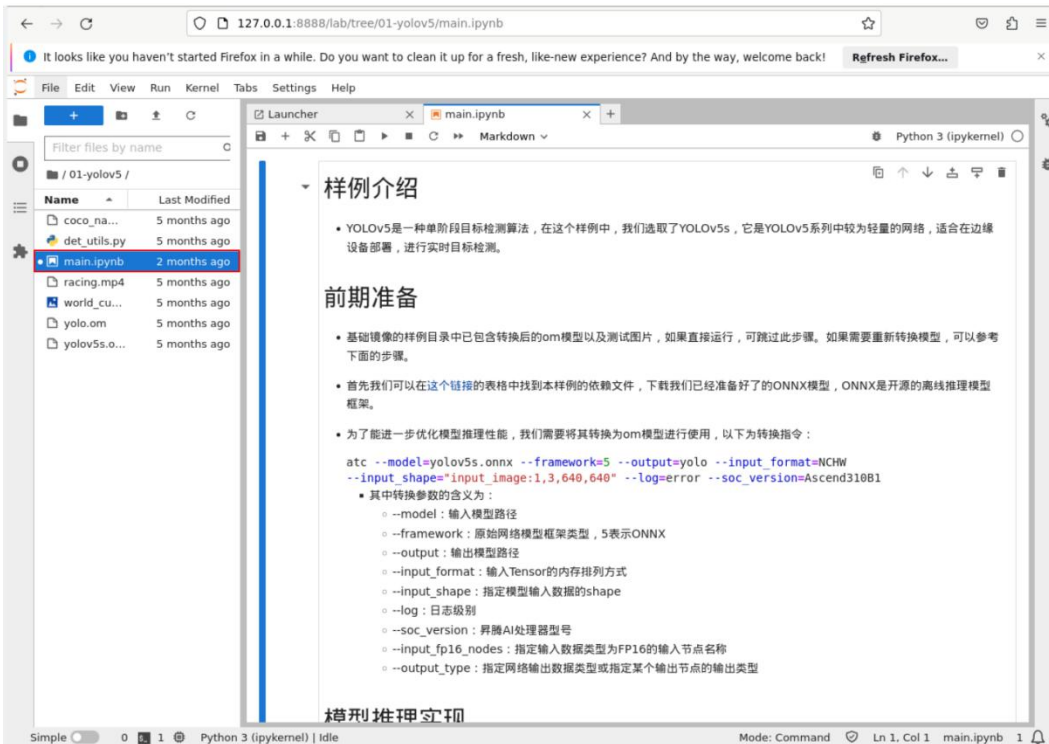
## 4.2. 运行目标检测样例

YOLOv5 是一种单阶段目标检测器算法，在这个样例中，我们选取了 YOLOv5s，它是 YOLOv5 系列中较为轻量的网络模型，适合在边缘设备部署，进行实时目标检测。在样例中已经包含转换后的 om 模型和测试视频、图片，可以按照以下流程在 Jupyter Lab 中运行该样例。

1) 首先在 jupyter lab 界面双击“01-yolov5”，进入到该目录下。

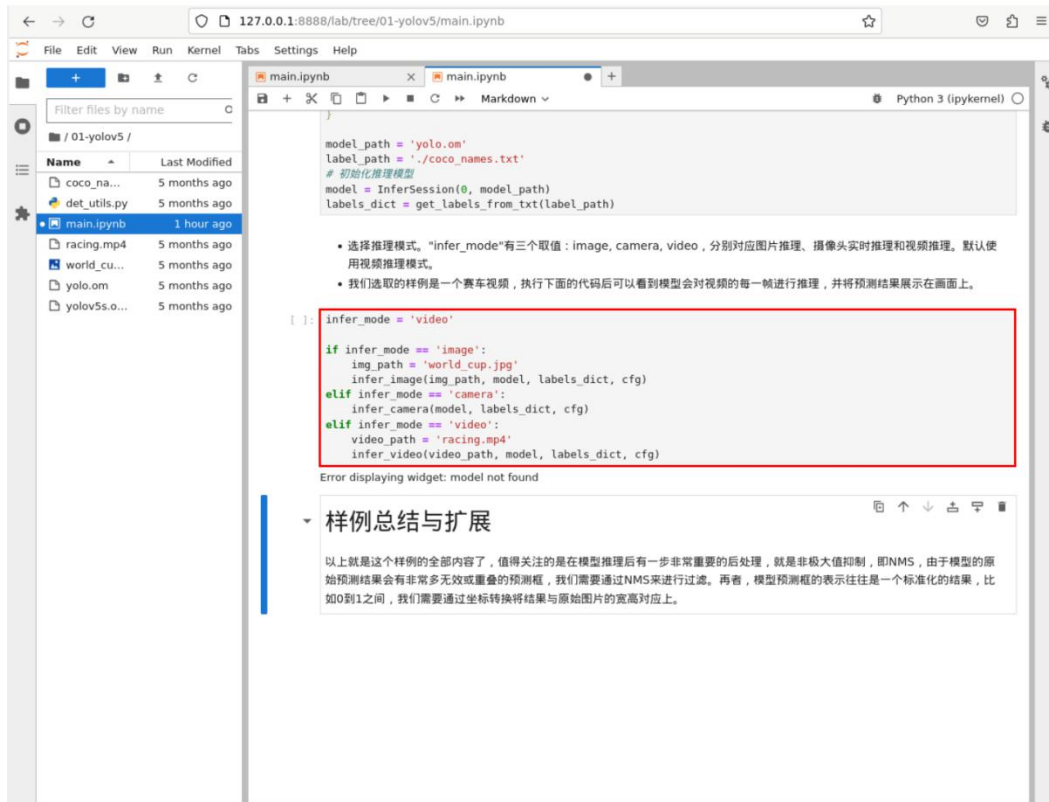


2) 在该目录下有运行该示例的所有资源，其中 main.ipynb 是在 Jupyter Lab 中运行该示例的文件，双击打开 main.ipynb，在右侧窗口中会显示 main.ipynb 文件中的内容。

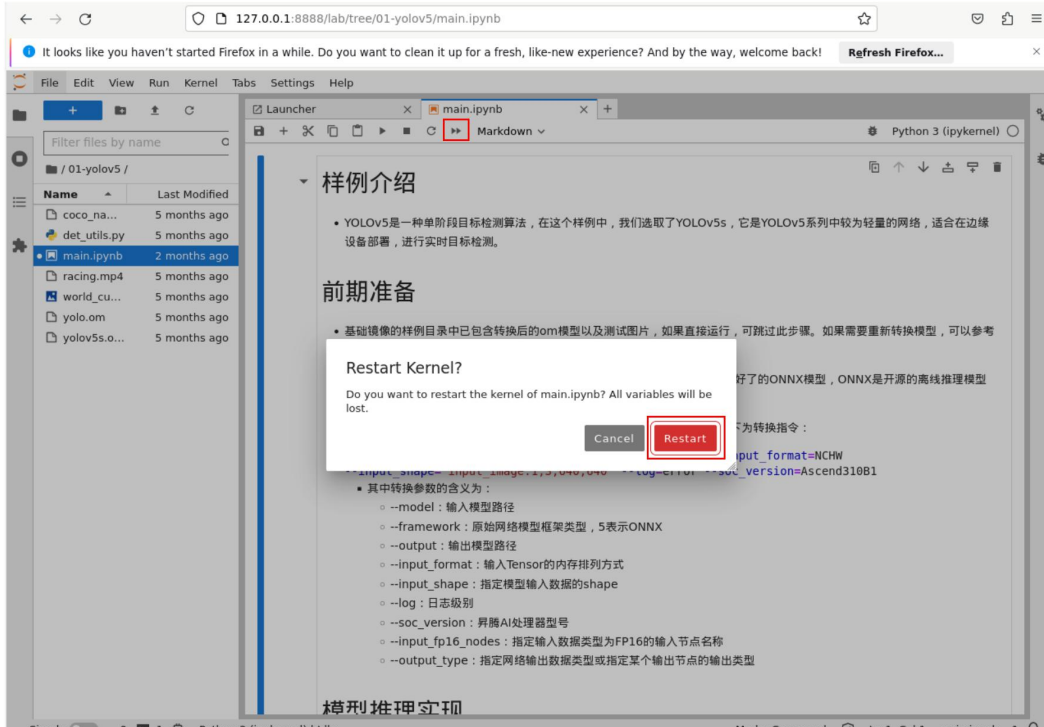




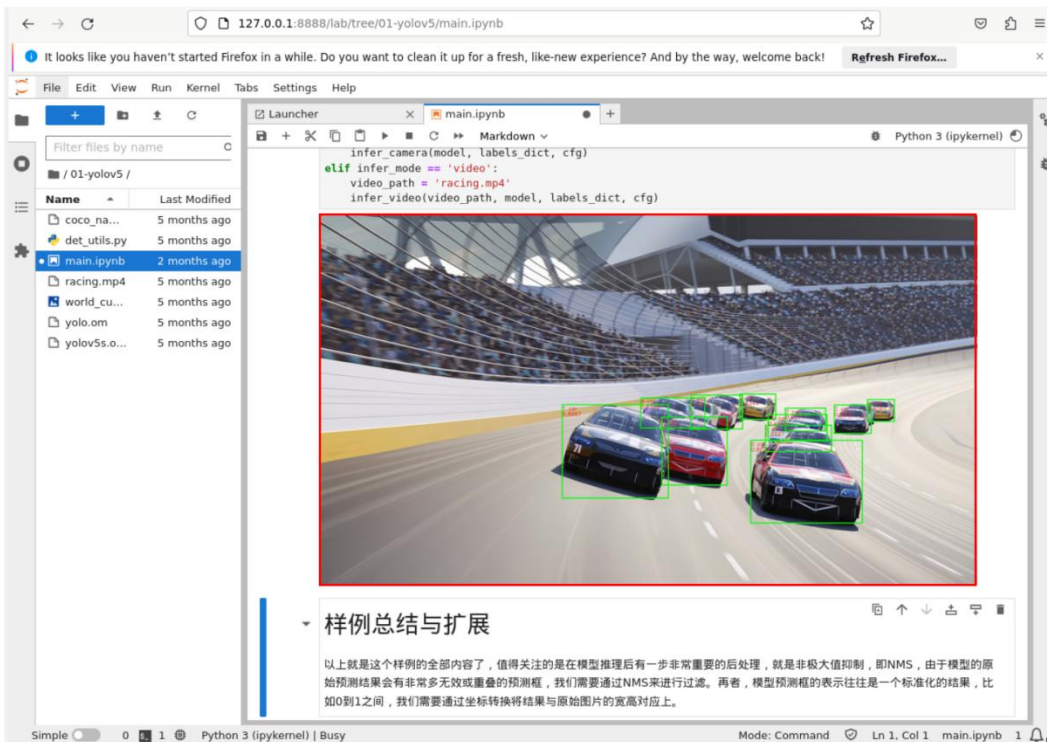
3) 在 main.ipynb 文件中 infer\_mode 的值可赋值为 image、video 和 camera，分别对应图片、视频、USB 摄像头中的内容进行目标检测，默认值为 video。



4) 单击 ▶▶ 按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



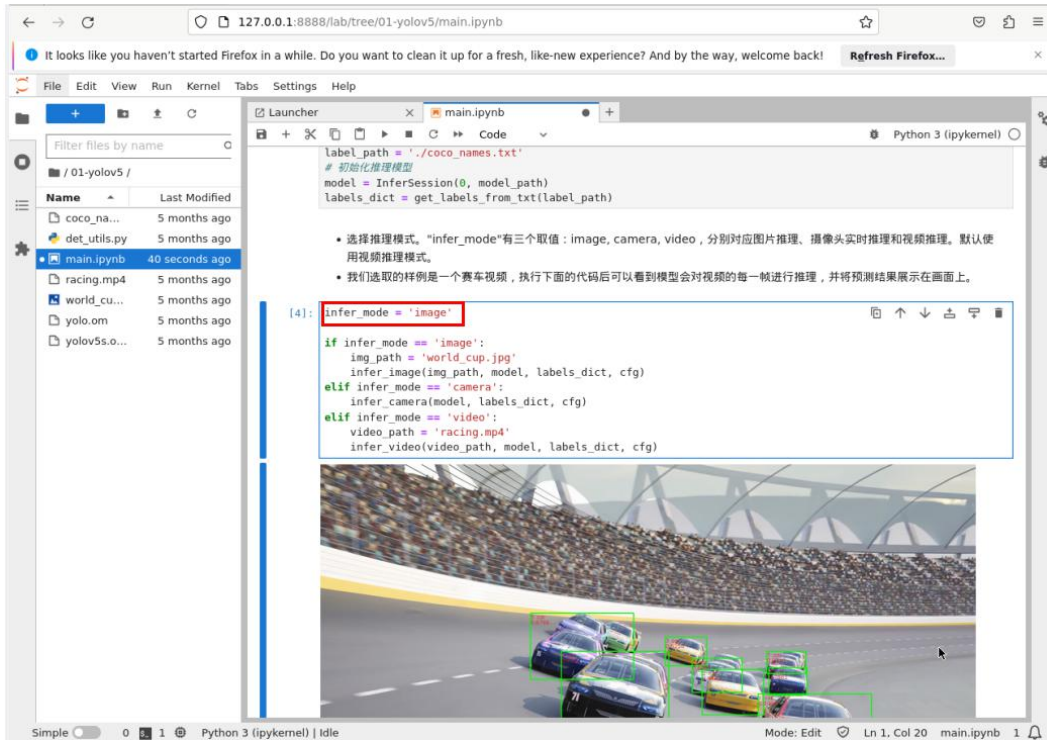
5) 若干秒后，在窗口中出现了一段赛车的视频，我们可以看到模型对视频的每一帧进行推理，并将检测到的赛车标注了出来。



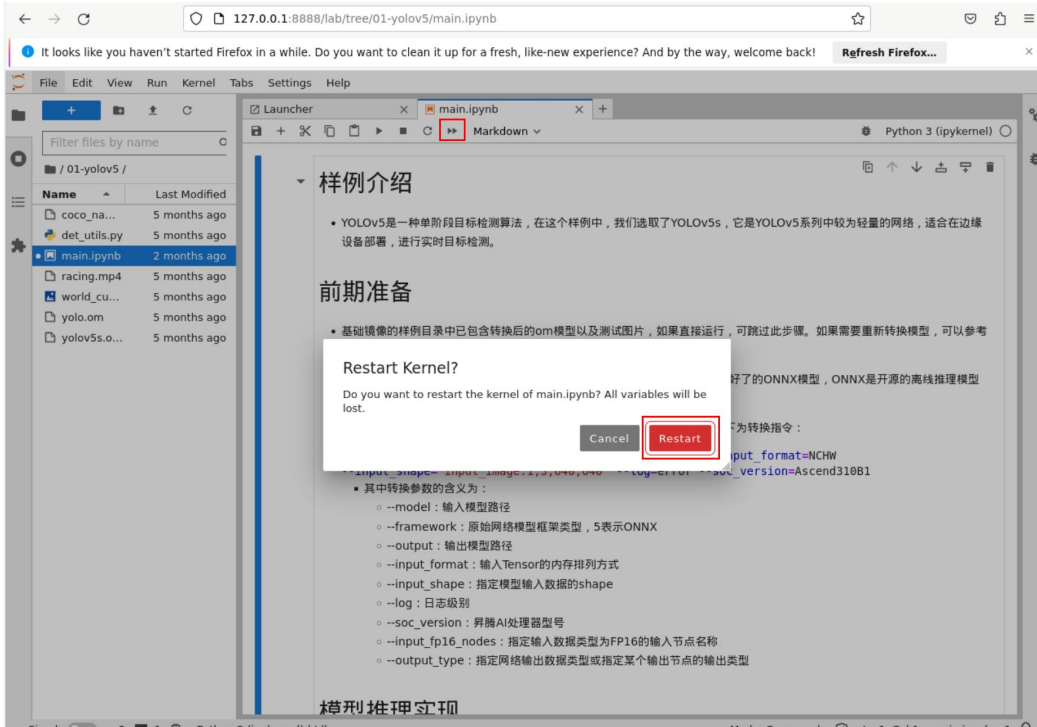
6) 测试视频的保存路径如下所示:

**/home/HwHiAiUser/samples/notebooks/01-yolov5/racing.mp4**

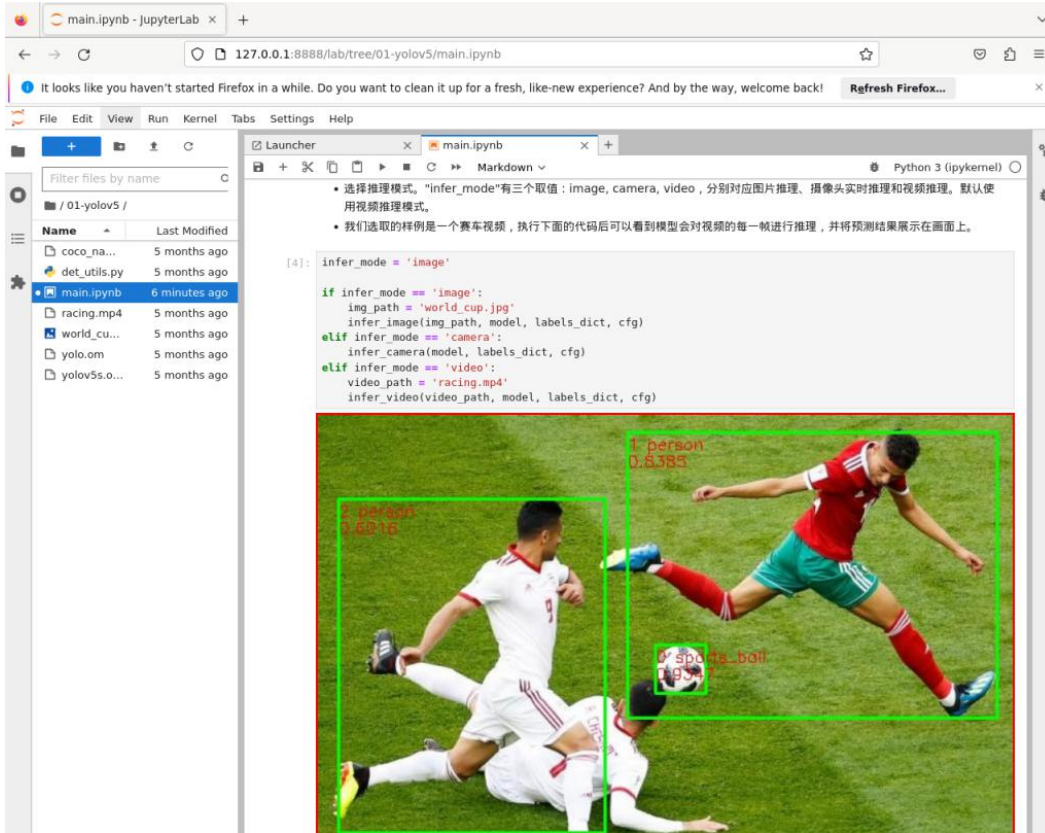
7) 然后我们在窗口中将 infer\_mode 的值修改 image，切换为图片推理模式。



8) 单击 ▶▶ 按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



9) 若干秒后，在窗口中出现了一张足球比赛的图片，我们可以看到模型对图片进行推理，识别到了图片中的三个人和一个足球。



10) 测试图片的保存路径如下所示:

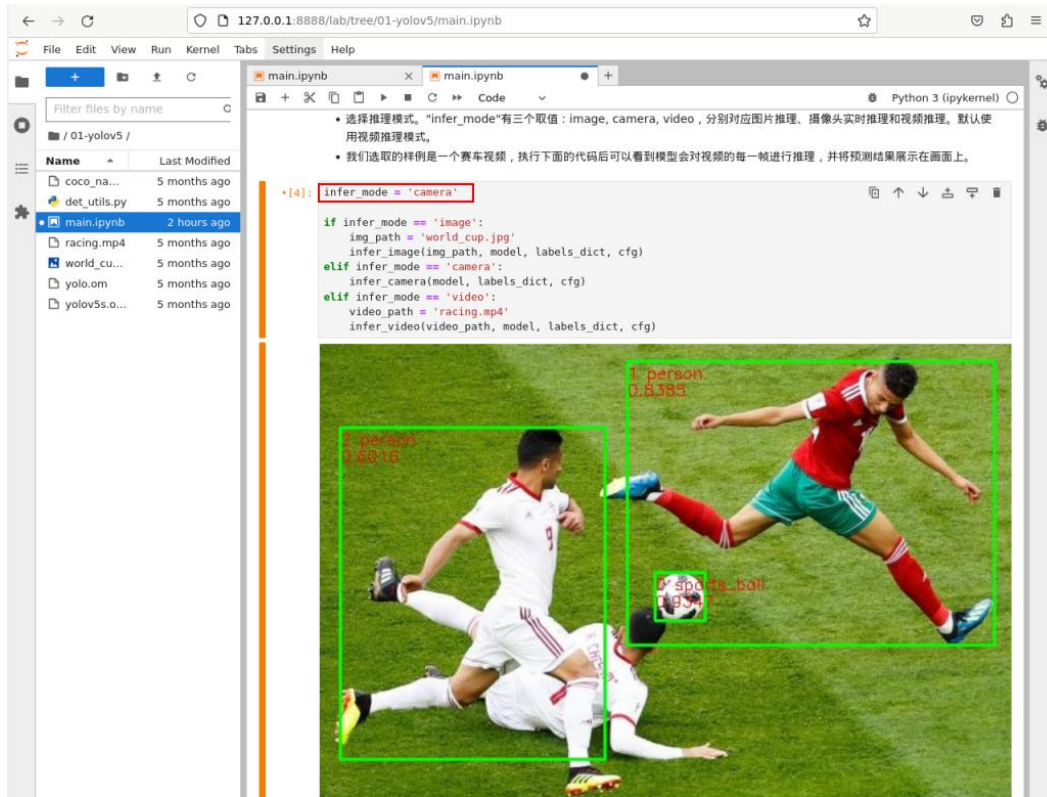
`/home/HwHiAiUser/samples/notebooks/01-yolov5/world_cup.jpg`

11) 测试图片的内容如下所示:

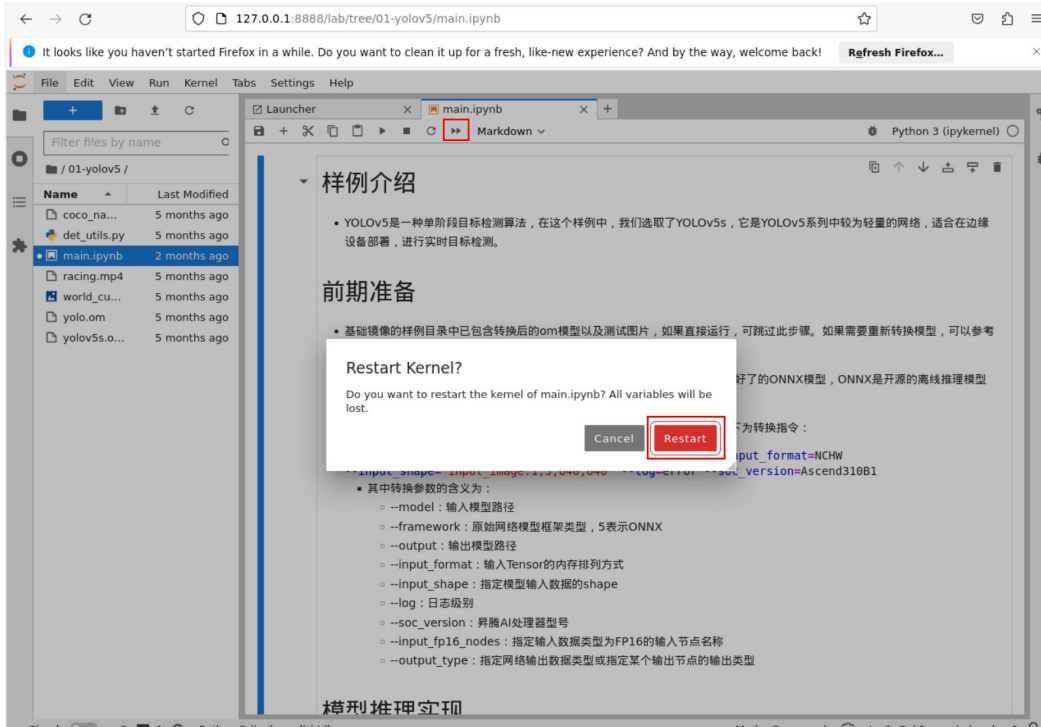


12) 然后我们在窗口中修改 infer\_mode 的值为 camera，切换为 USB 摄像头推理模

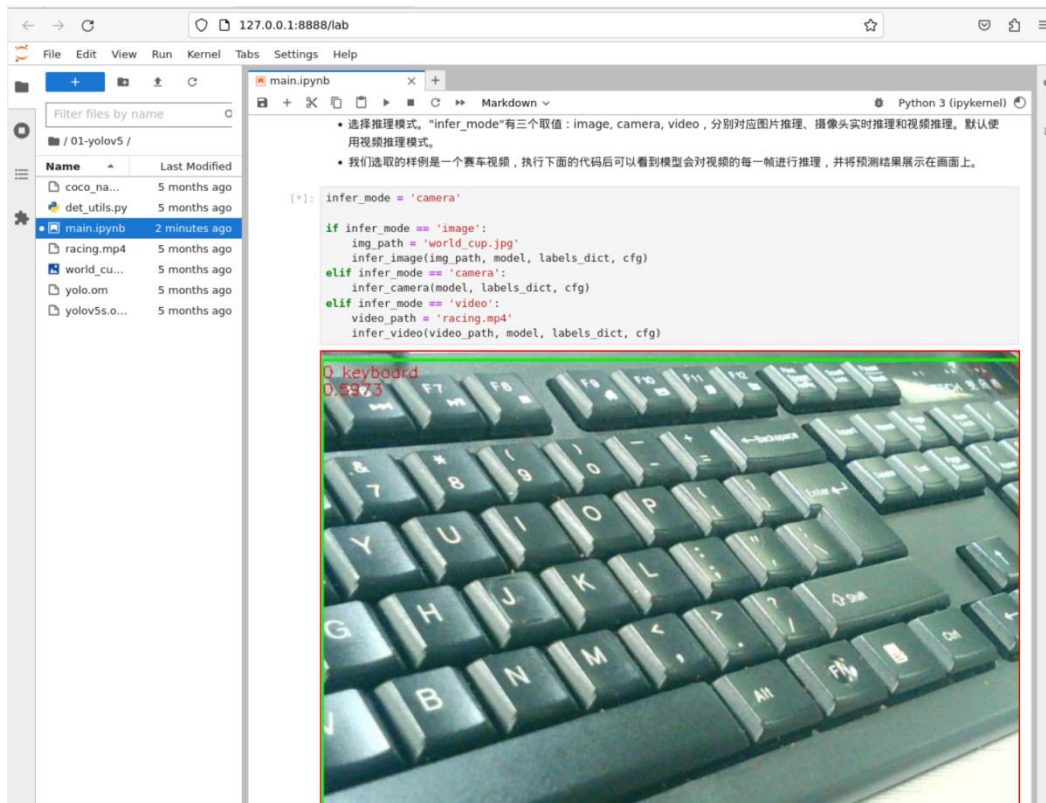
式（此模式需要在开发板上接 USB 摄像头）。



13) 单击 ▶▶ 按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时样例开始运行。



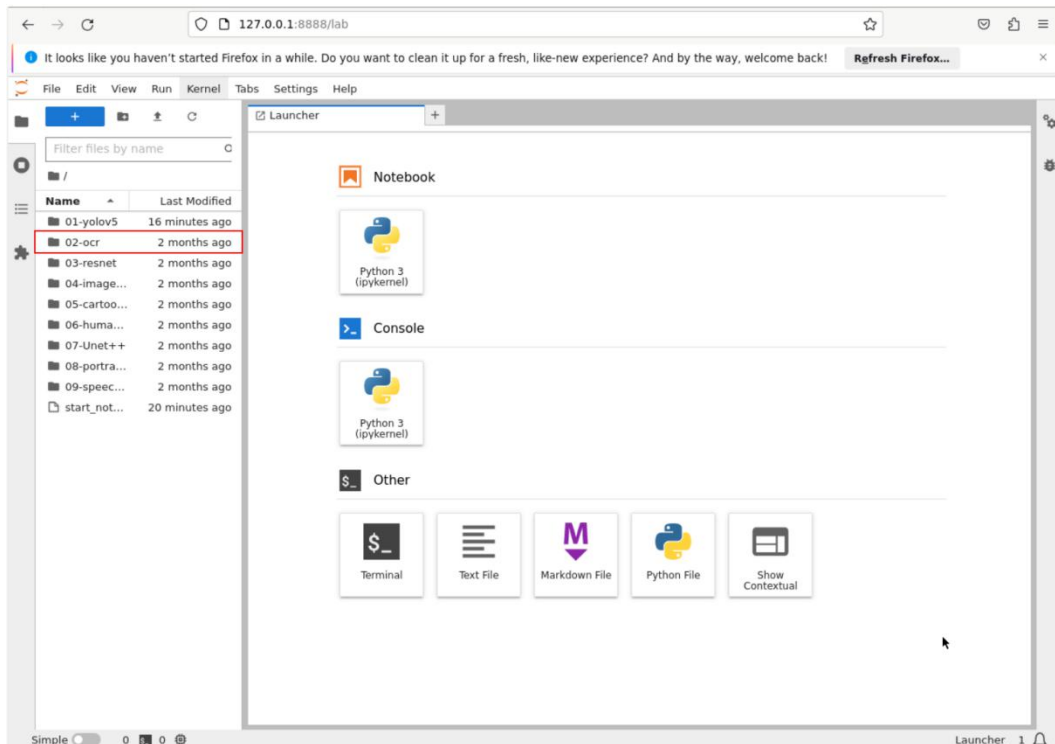
14) 我们可以看到模型对摄像头拍到的画面进行推理, 比如在拍摄到键盘时识别到了画面中的键盘, 并将推理结果展示在画面上。



### 4.3. 运行文字识别样例

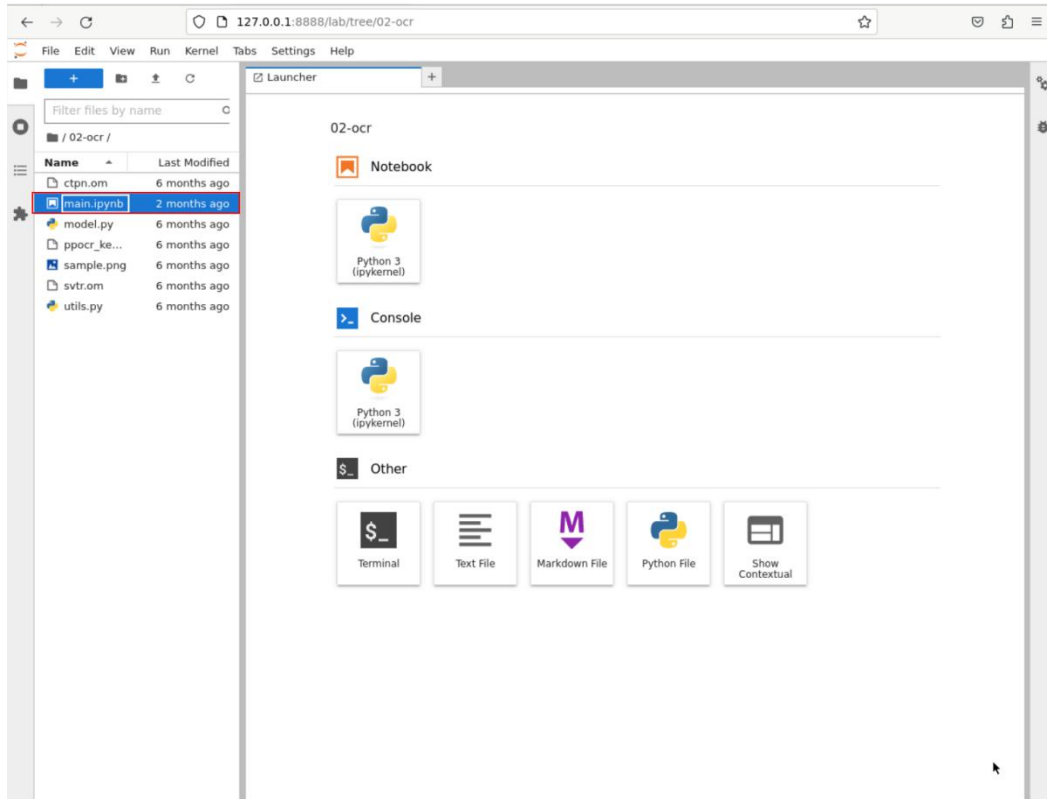
OCR 一般指 Scene Text Recognition（场景文字识别），主要面向自然场景。OCR 两阶段方法一般包含两个模型，检测模型负责找出图像或视频中的文字位置，识别模型负责将图像信息转换为文本信息。在样例中已经包含转换后的 om 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。


1) 首先在 Jupyter Lab 界面双击“02-ocr”，进入到该目录下。

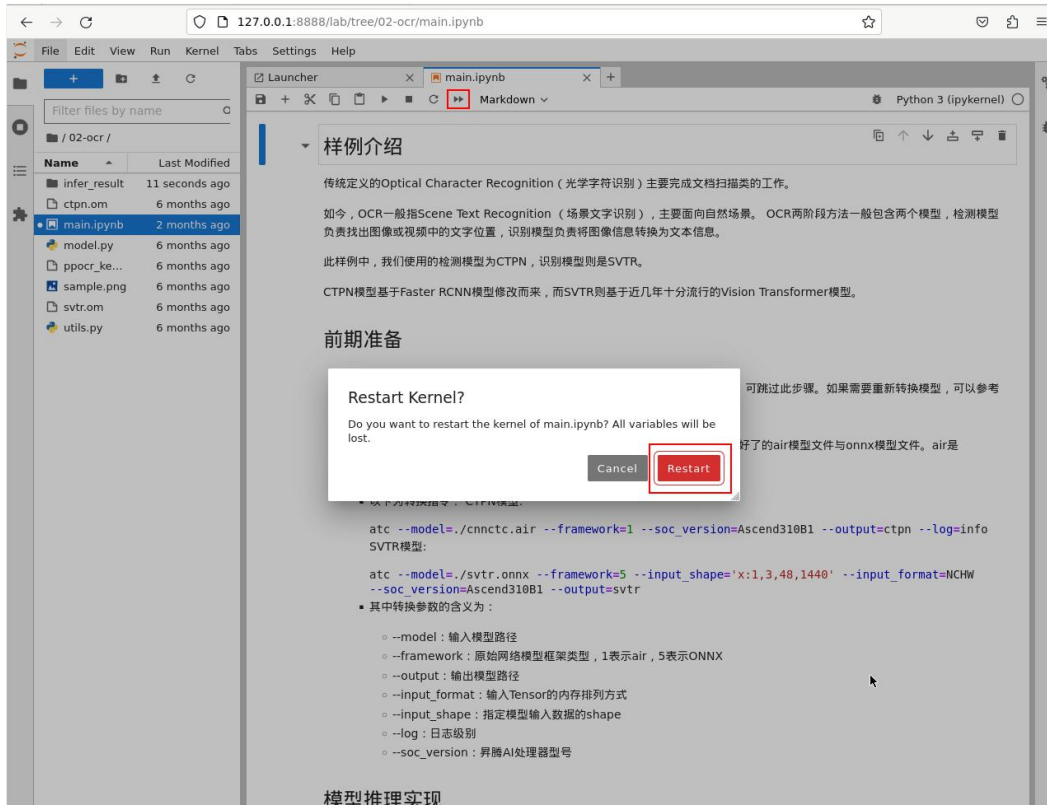


2) 在该目录下有运行该示例的所有资源，其中 main.ipynb 是在 Jupyter Lab 中运行该样例的文件，双击打开 main.ipynb，在右侧窗口中会显示 main.ipynb 文件中的内容。

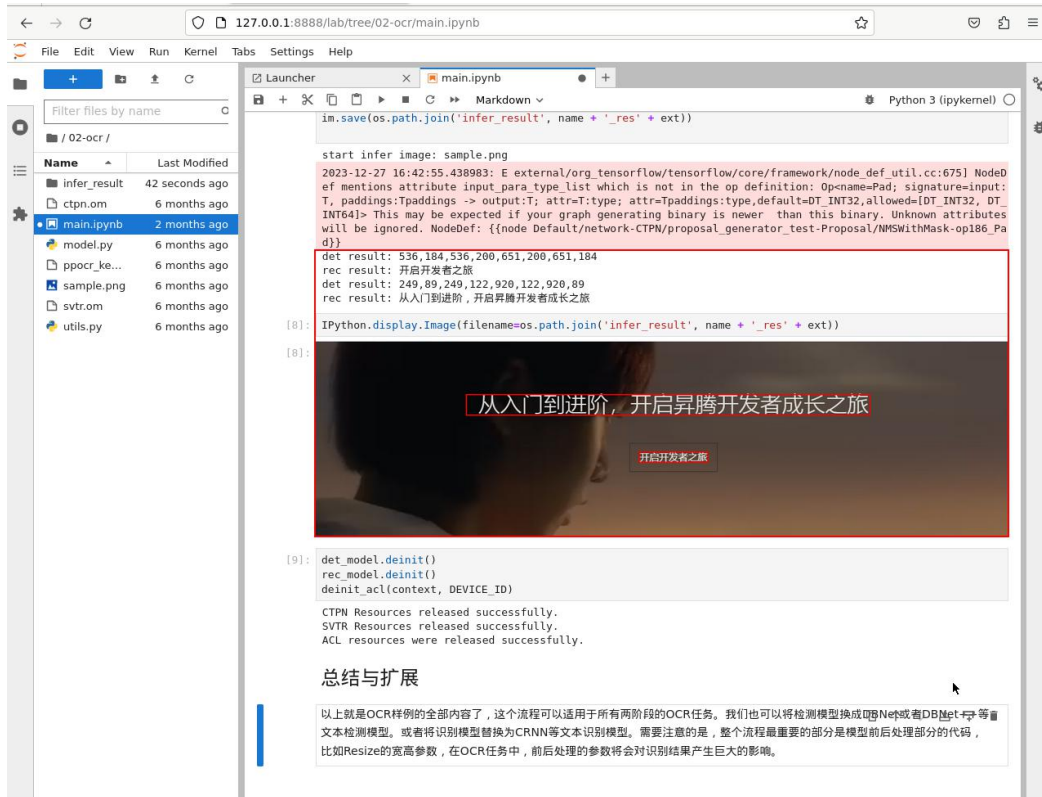




3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后, 在窗口中出现了一张带文字的图片。我们可以看到模型对图片进行推理, 成功获取了图片中的文字信息“开启开发者之旅”、“从入门到进阶, 开启昇腾开发者成长之旅”。



5) 测试图片的保存路径如下所示：

`/home/HwHiAiUser/samples/notebooks/02-ocr/sample.png`

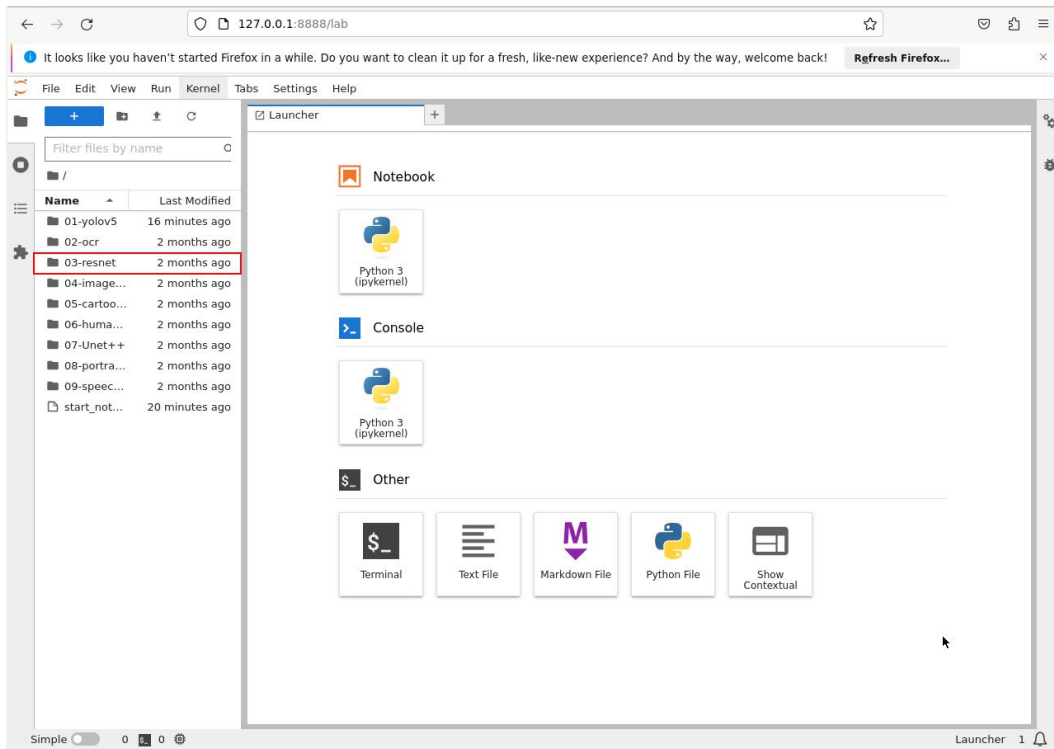
6) 测试图片的内容如下所示：



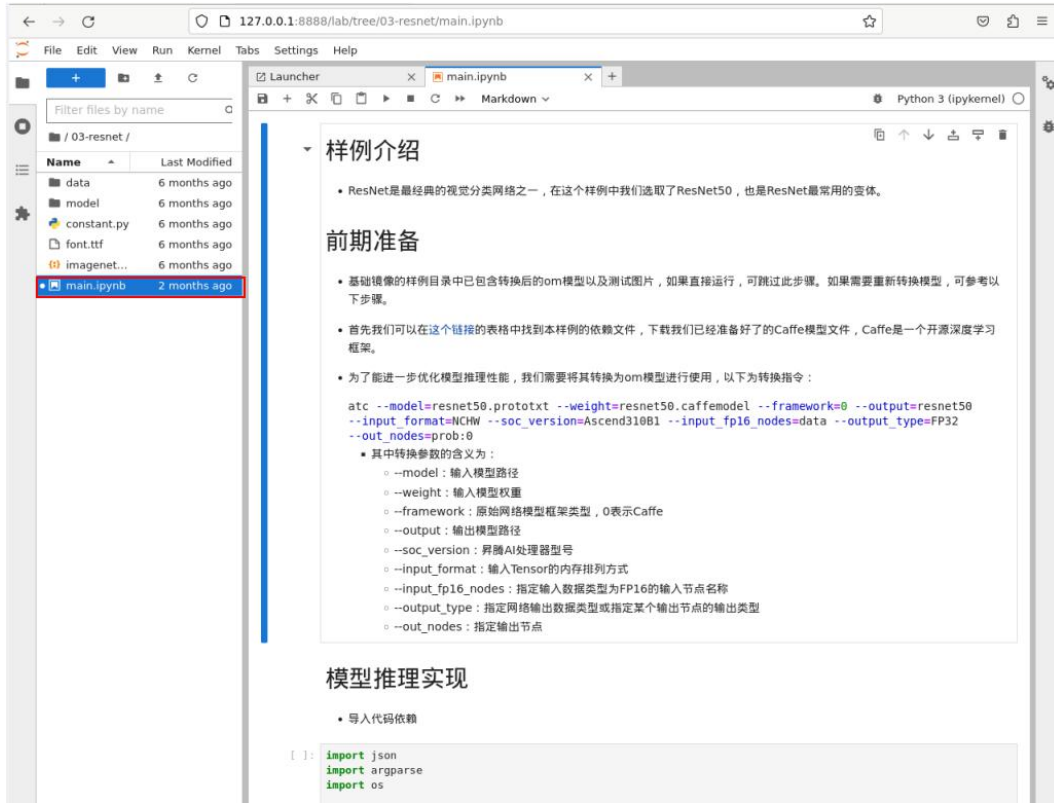
#### 4.4. 运行目标分类样例


ResNet 是最经典的视觉分类网络之一，在这个样例中我们选取了 ResNet50，也是 ResNet 最常用的变体。在样例中已经包含转换后的 om 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。

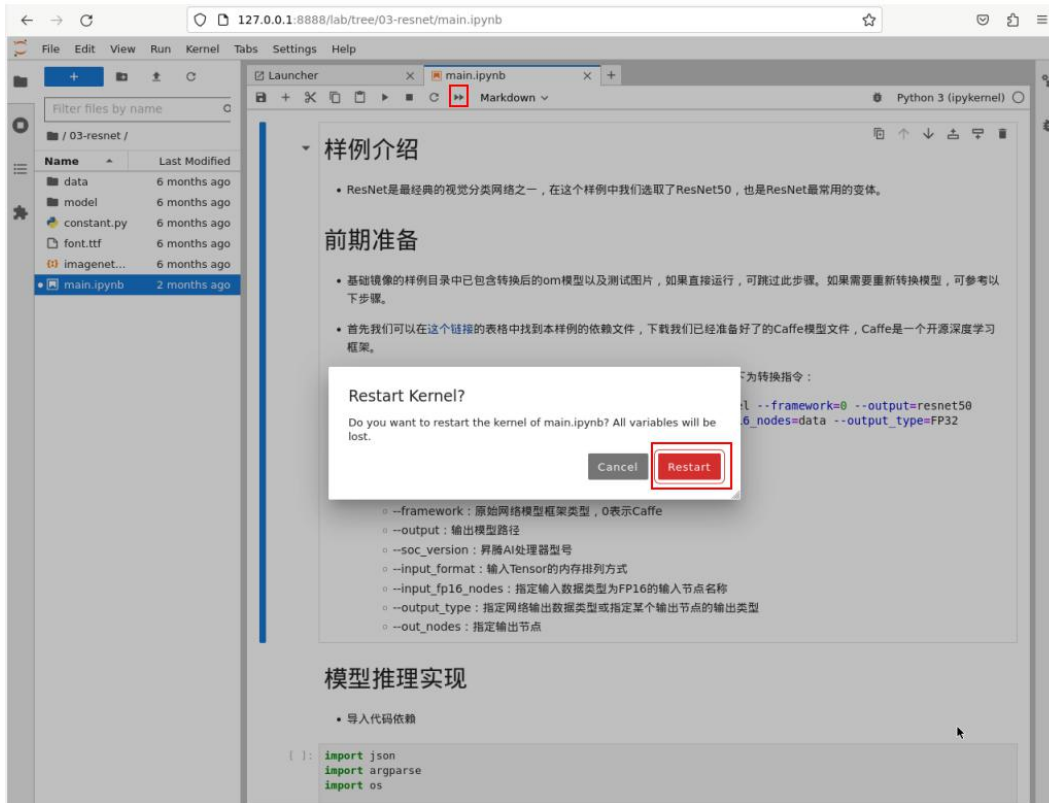
1) 首先在 Jupyter Lab 界面双击“03-resnet”，进入到该目录下。



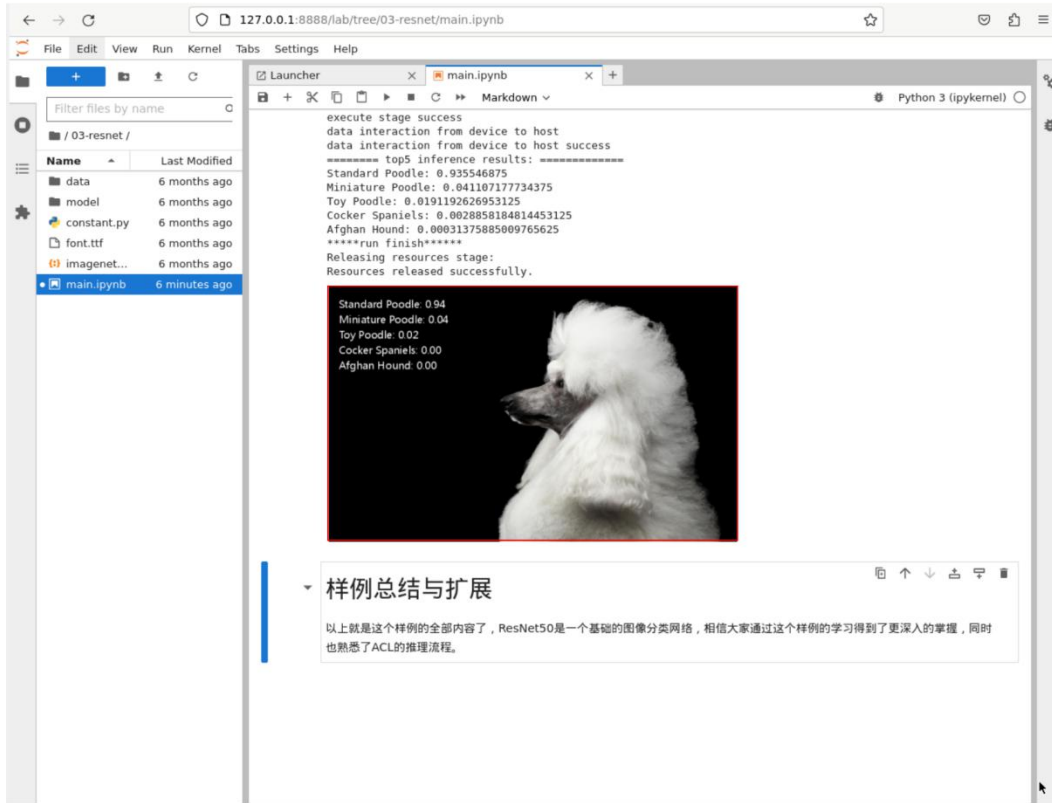
2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该样例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。



3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了一张小狗的图片，我们可以看到模型对图片进行推理，展示了图片中物体最有可能的五种类别以及相应的置信度，其中置信度最高的类别是 Standard Poodle，该类别正是对应图片中的小狗。



5) 测试图片的保存路径如下所示:

**/home/HwHiAiUser/samples/notebooks/03-resnet/data/dog2\_1024\_683.jpg**

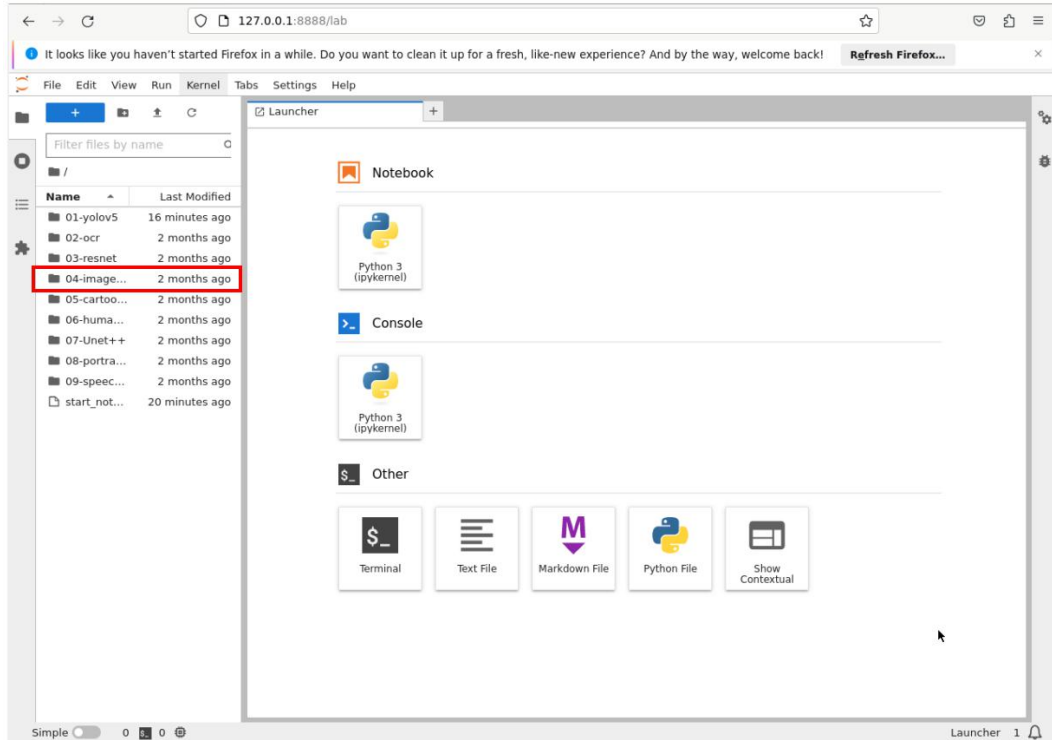
6) 测试图片的内容如下所示:



## 4.5. 运行图像曝光增强样例

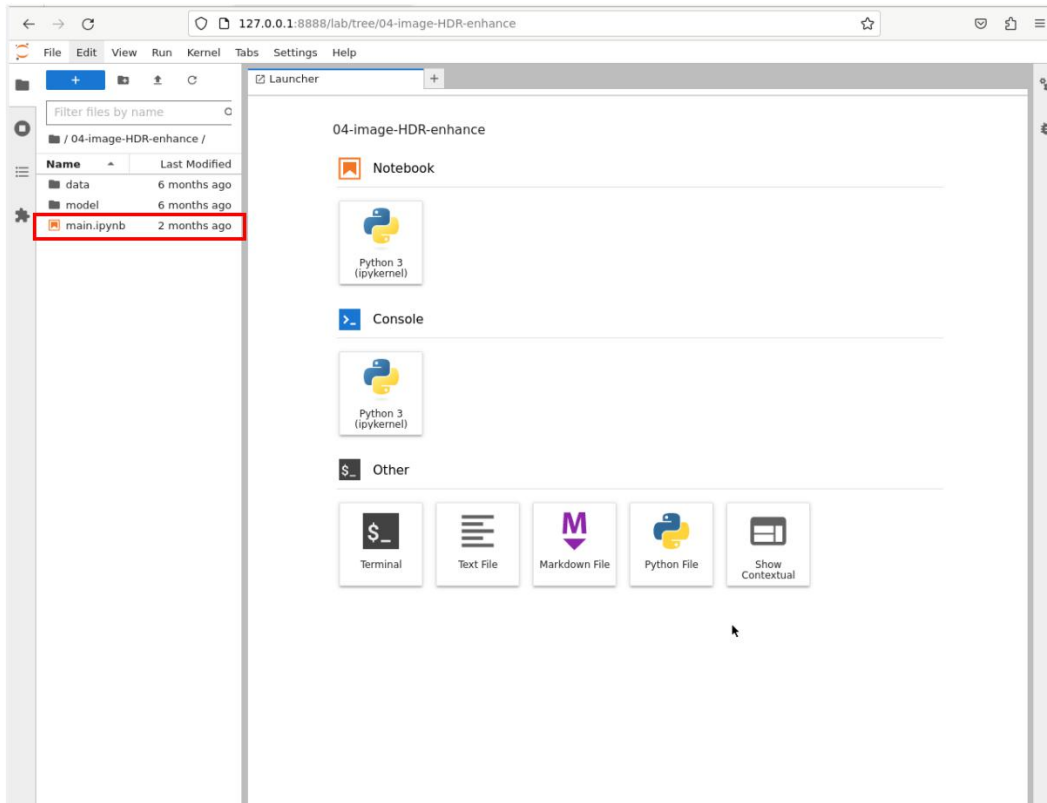
该样例可以对曝光不足的输入图片进行 HDR 效果增强。在样例中已经包含转换后的 om 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。


1) 首先在 Jupyter Lab 界面双击“04-image-HDR-enhance”，进入到该目录下。

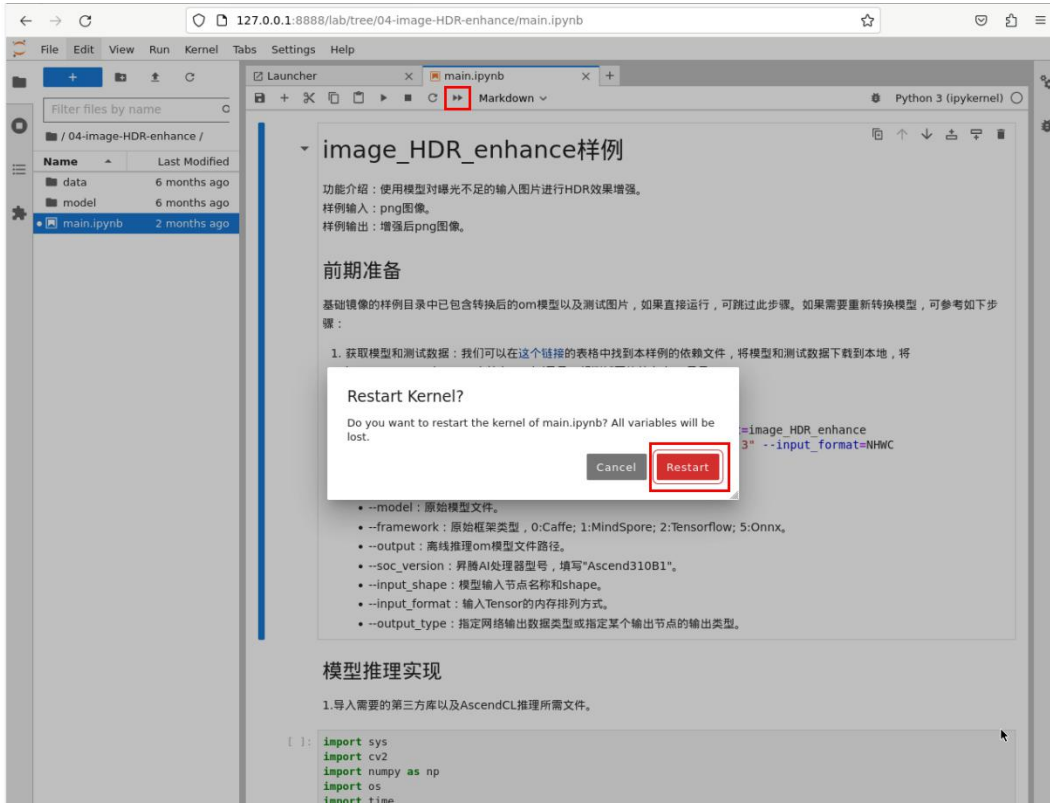


2) 在该目录下有运行该示例的所有资源，其中 main.ipynb 是在 Jupyter Lab 中运行该样例的文件，双击打开 main.ipynb，在右侧窗口中会显示 main.ipynb 文件中的内容。

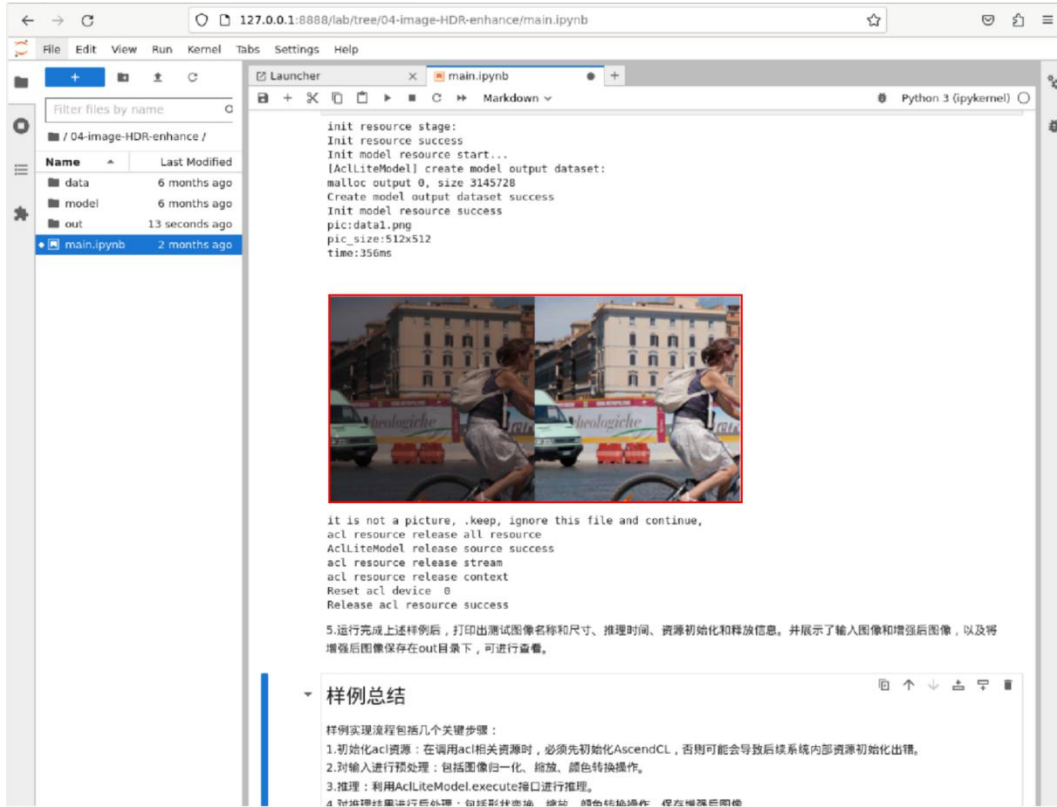




3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了两张图片，左侧是图片原本的样子，右侧是经过模型处理后的样子，我们可以看到模型对图片进行了曝光增强，提升了图片的画面感。



5) 测试图片的保存路径如下所示：

**/home/HwHiAiUser/samples/notebooks/04-image-HDR-enhance/data/data1.png**

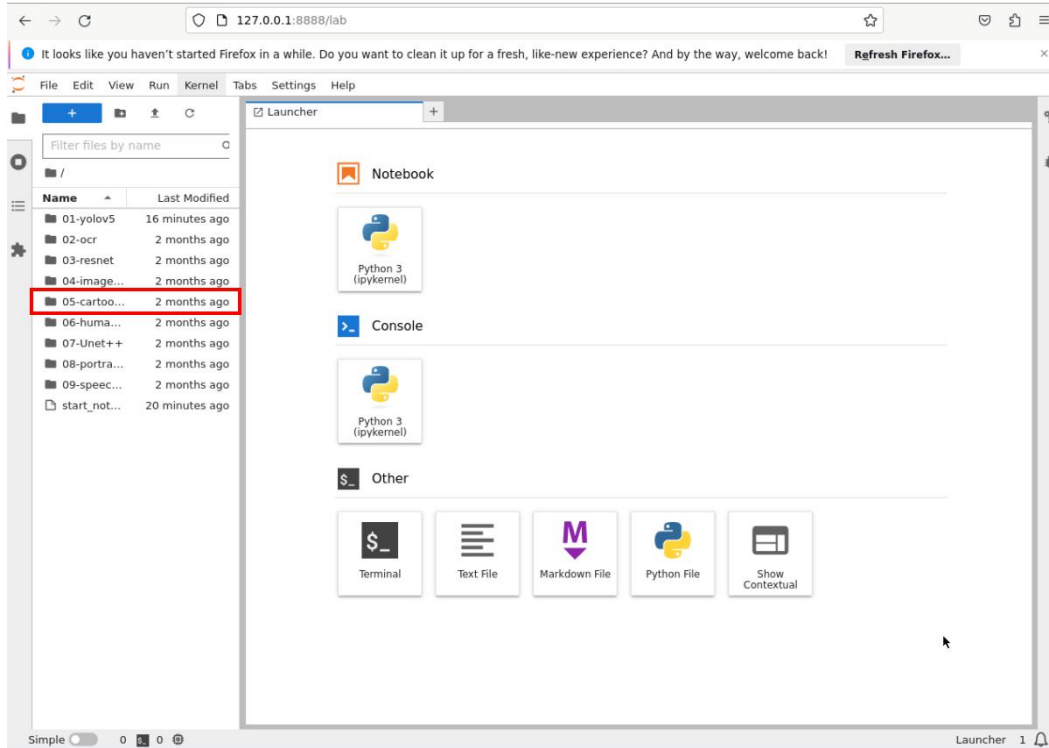
6) 测试图片的内容如下所示：



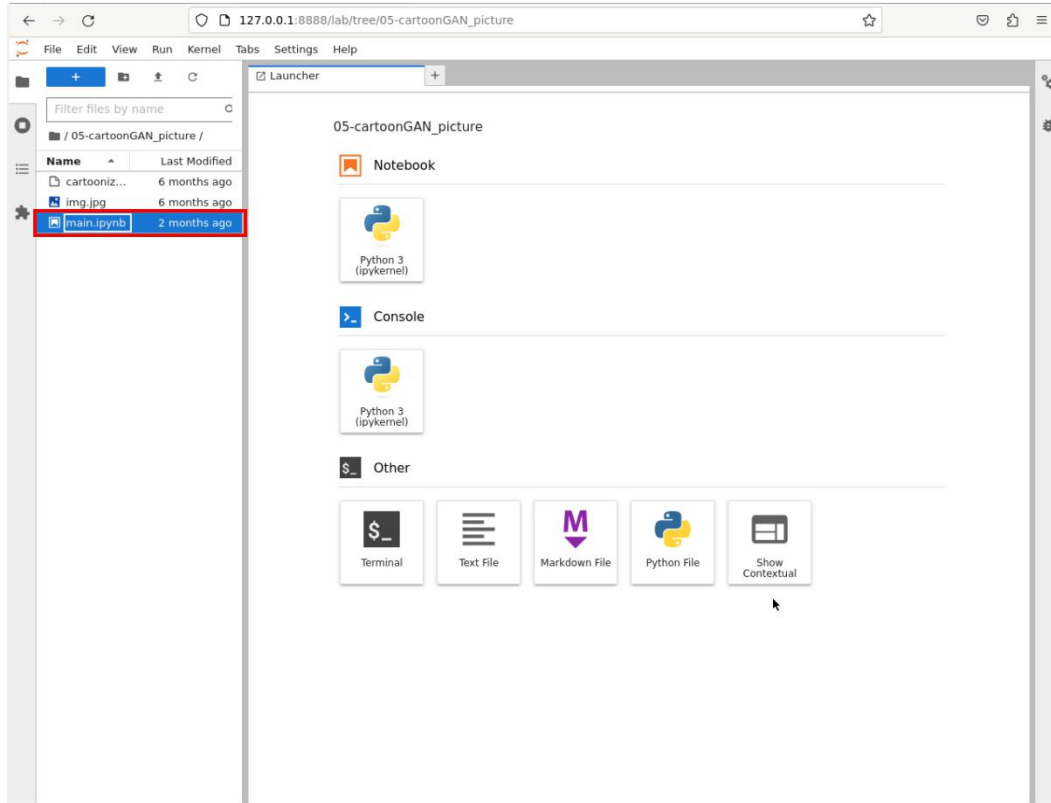
## 4.6. 运行卡通图像生成样例


该样例使用 `cartoonGAN` 模型对输入图片进行卡通化处理。在样例中已经包含转换后的 `om` 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。

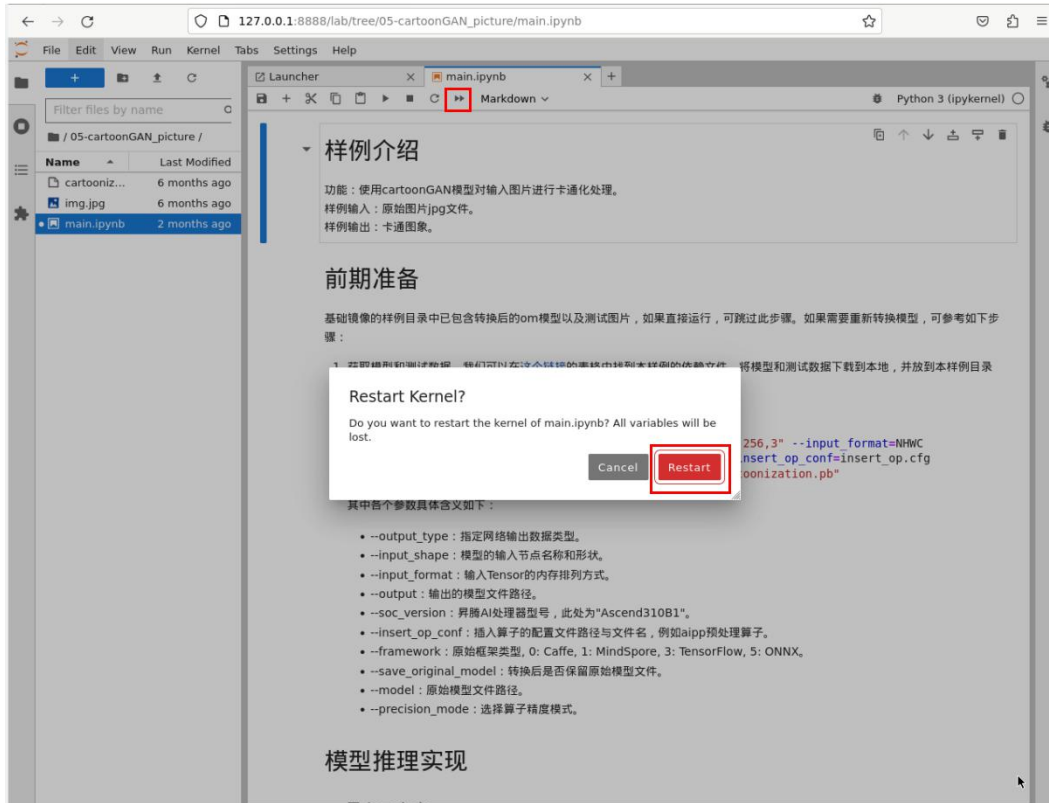
- 1) 首先在 Jupyter Lab 界面双击“05-cartoonGAN\_picture”，进入到该目录下。



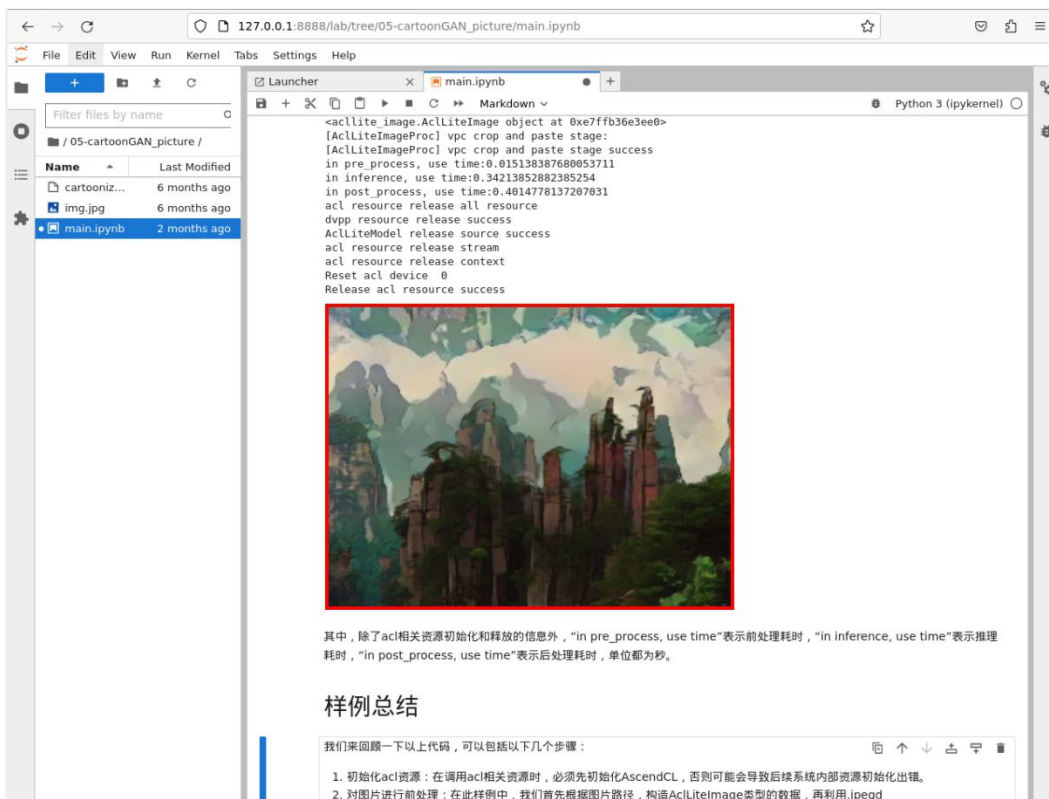
2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该示例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。



3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了一张卡通画风格的风景图。



5) 测试图片的保存路径如下所示:

```
/home/HwHiAiUser/samples/notebooks/05-cartoonGAN_picture/img.jpg
```

6) 测试图片的内容如下所示:



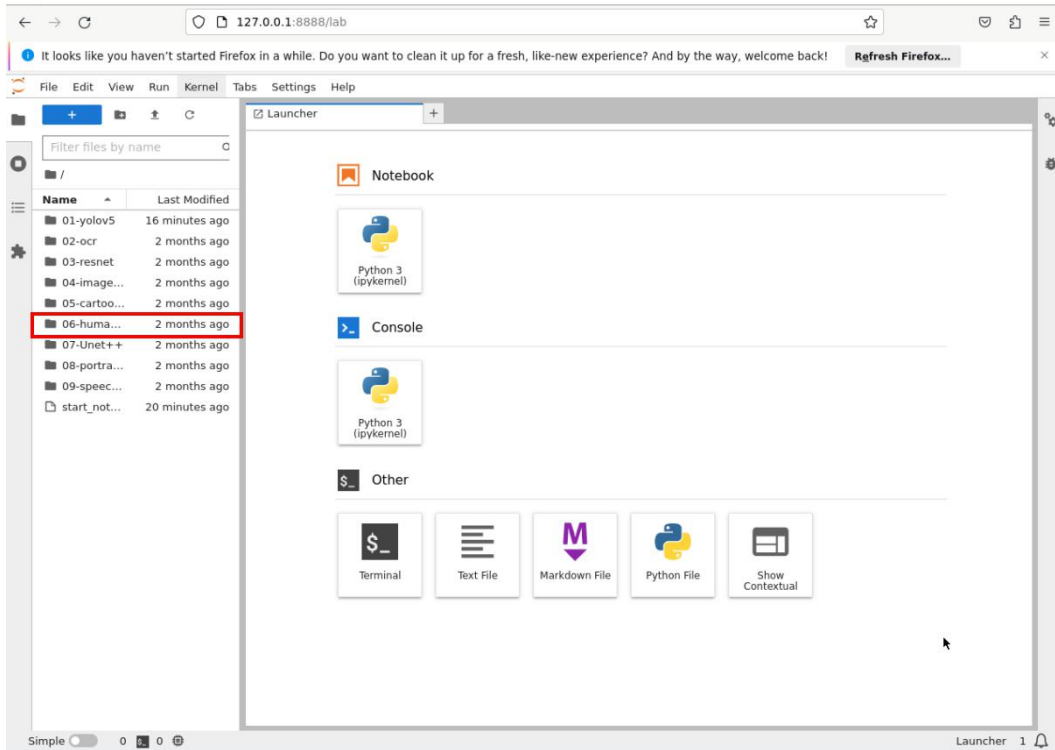
7) 我们可以看到模型对测试图片进行处理，将测试图片转换为卡通画风格的图片。

## 4.7. 运行蛋白质分类评估样例

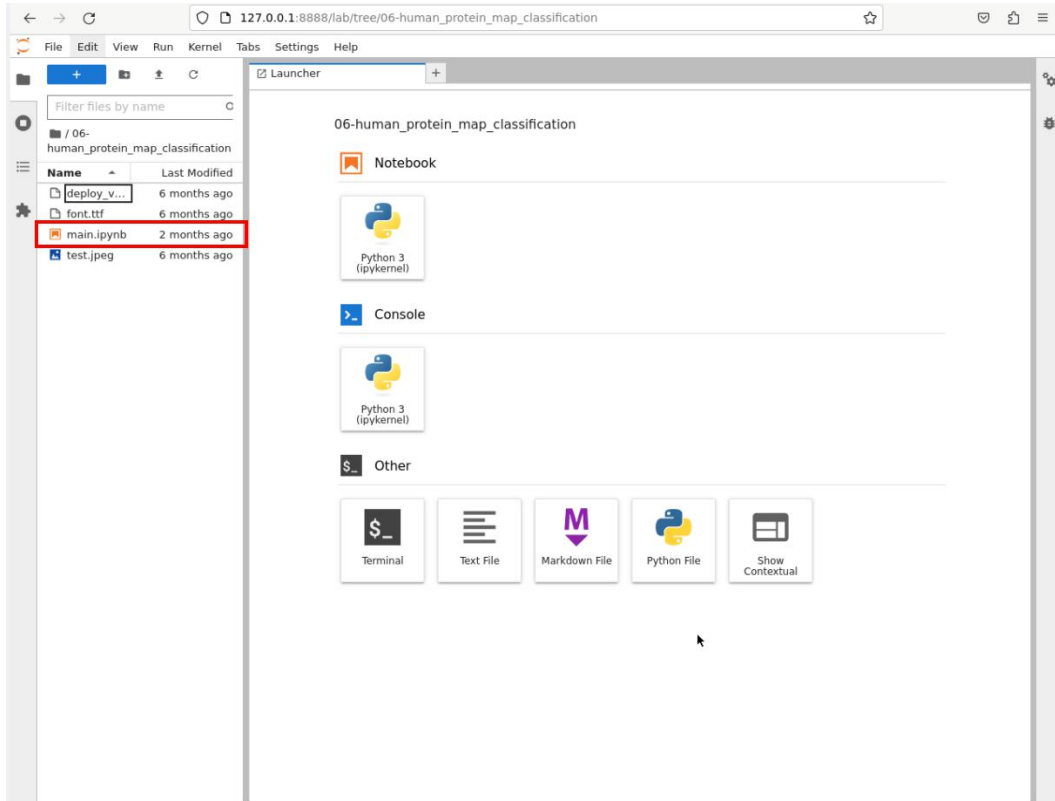
该样例可以对蛋白质图像进行自动化分类评估。在样例中已经包含转换后的om模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。


1) 首先在 Jupyter Lab 界面双击“06-human\_protein\_map\_classification”，进入到该目录下。

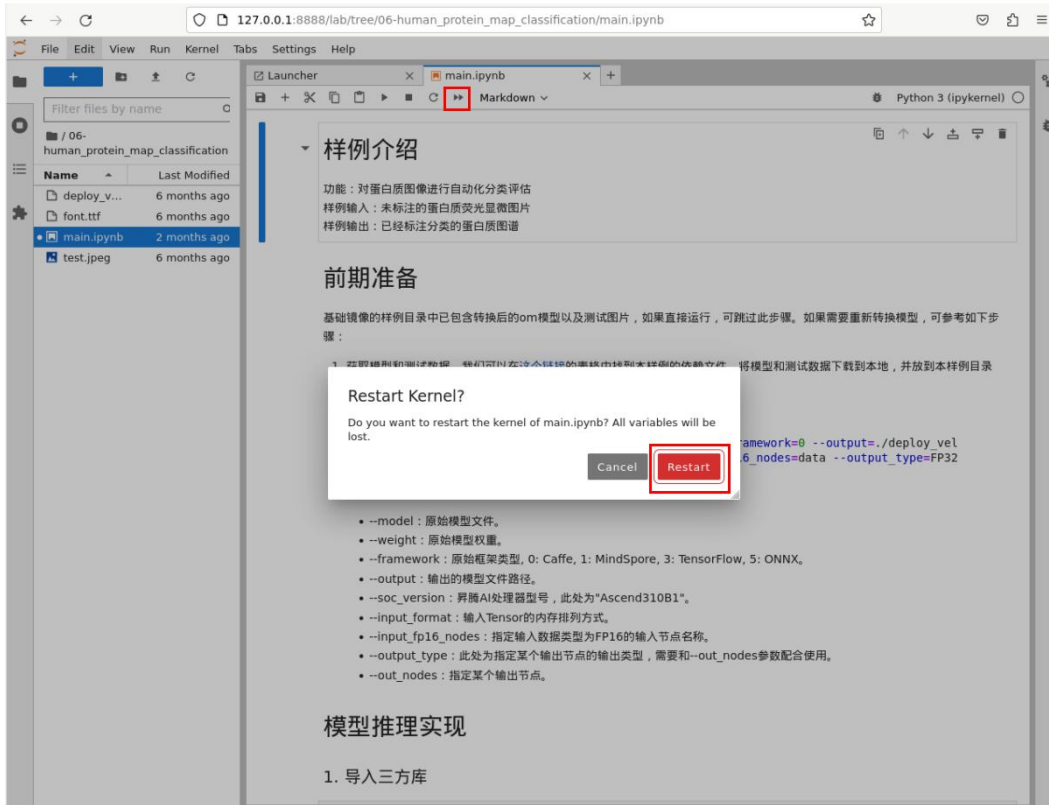




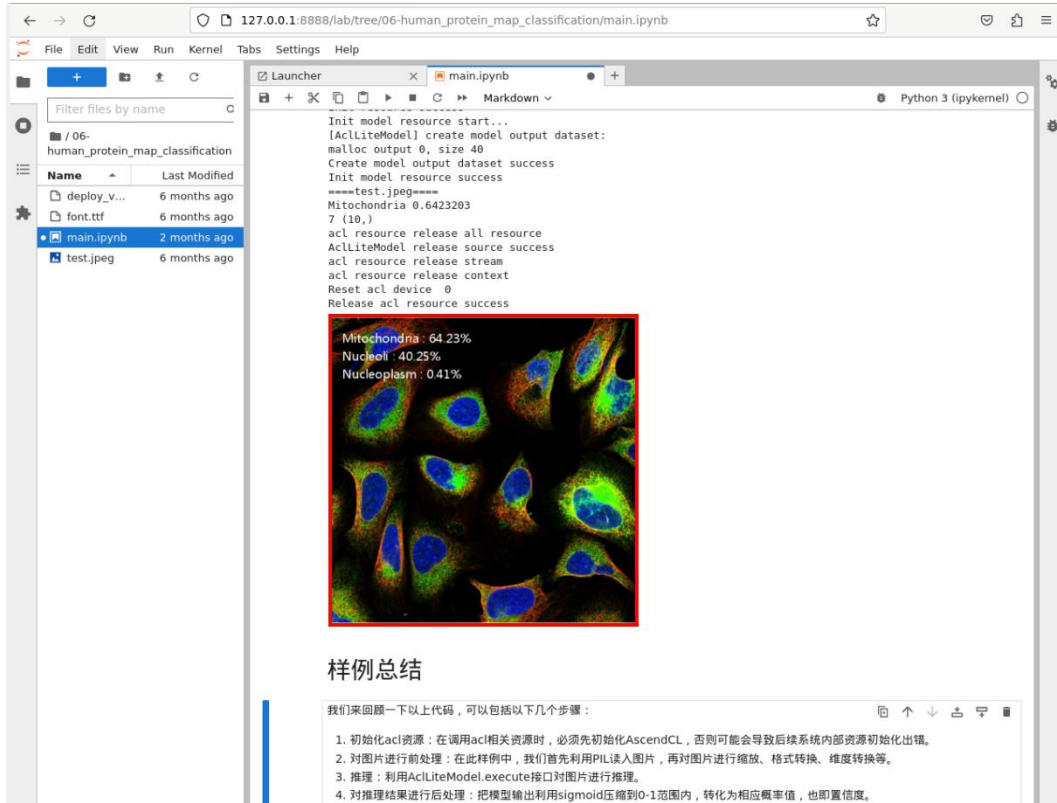
2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该示例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。



3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



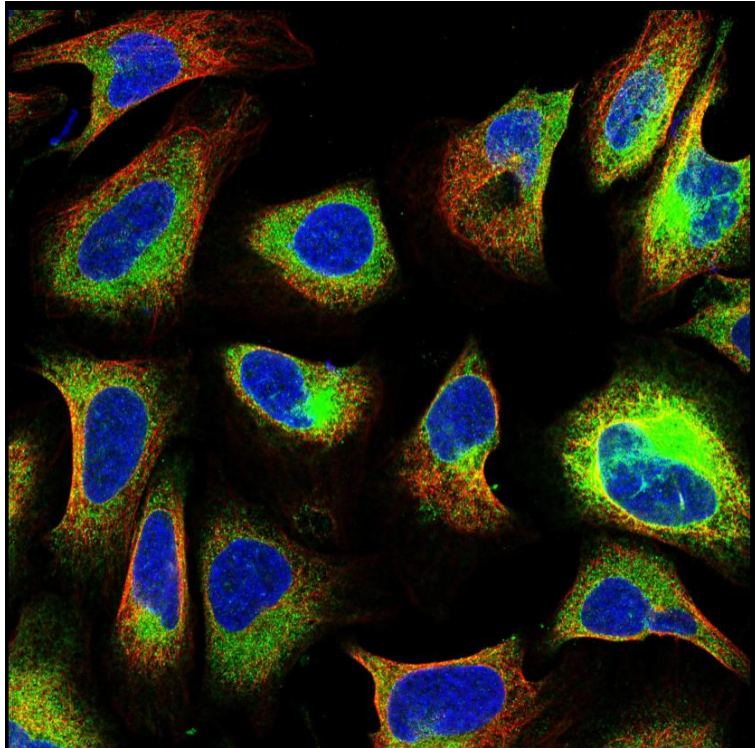
4) 若干秒后，在窗口中出现了一张蛋白质荧光显微图片。我们可以看到模型对图片进行推理，展示了图片中蛋白质最有可能的三种类别以及相应的置信度，其中置信度最高的类别是 Mitochondna，为 64.23%。



5) 测试图片的保存路径如下所示：

**/home/HwHiAiUser/samples/notebooks/06-human\_protein\_map\_classification/test.jpeg**

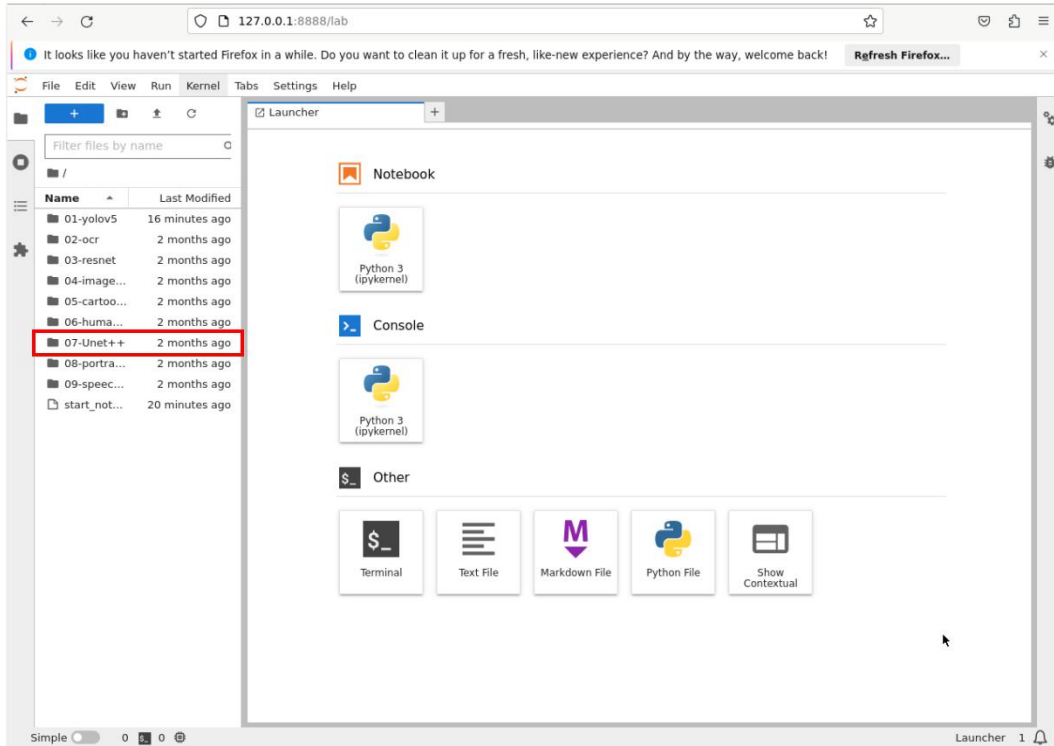
6) 测试图片的内容如下所示：



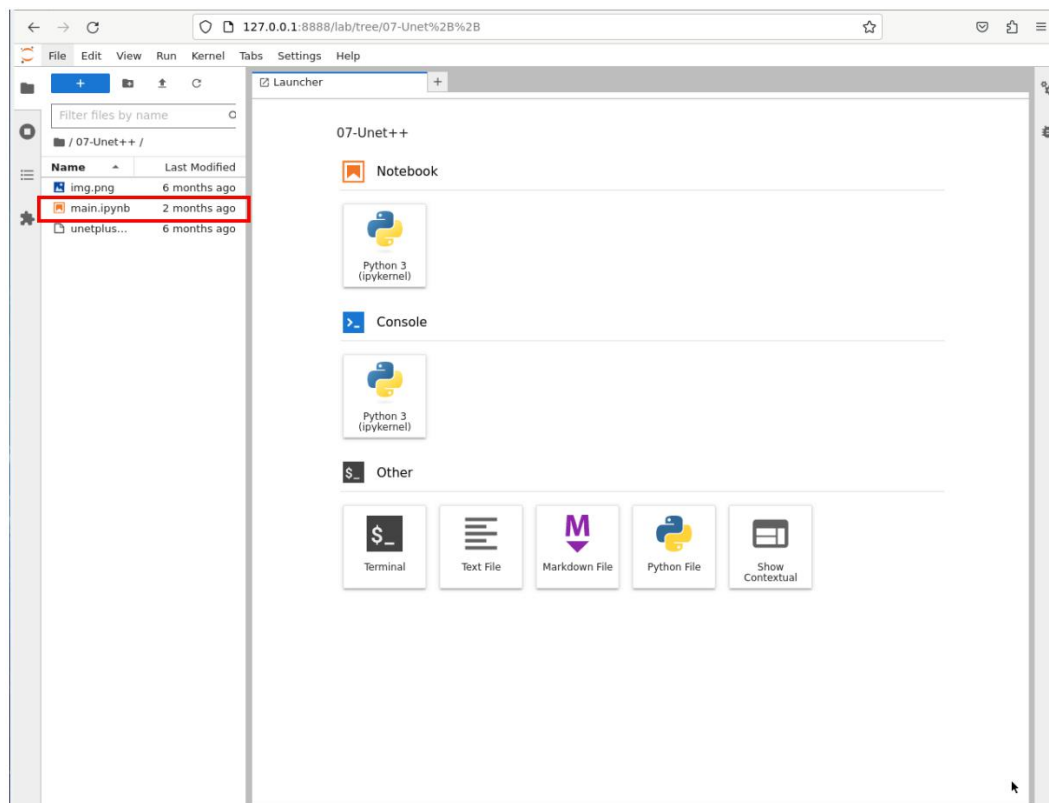
#### 4.8. 运行细胞图像分割样例

该样例可以对图像中的细胞核进行分割。在样例中已经包含转换后的 om 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。

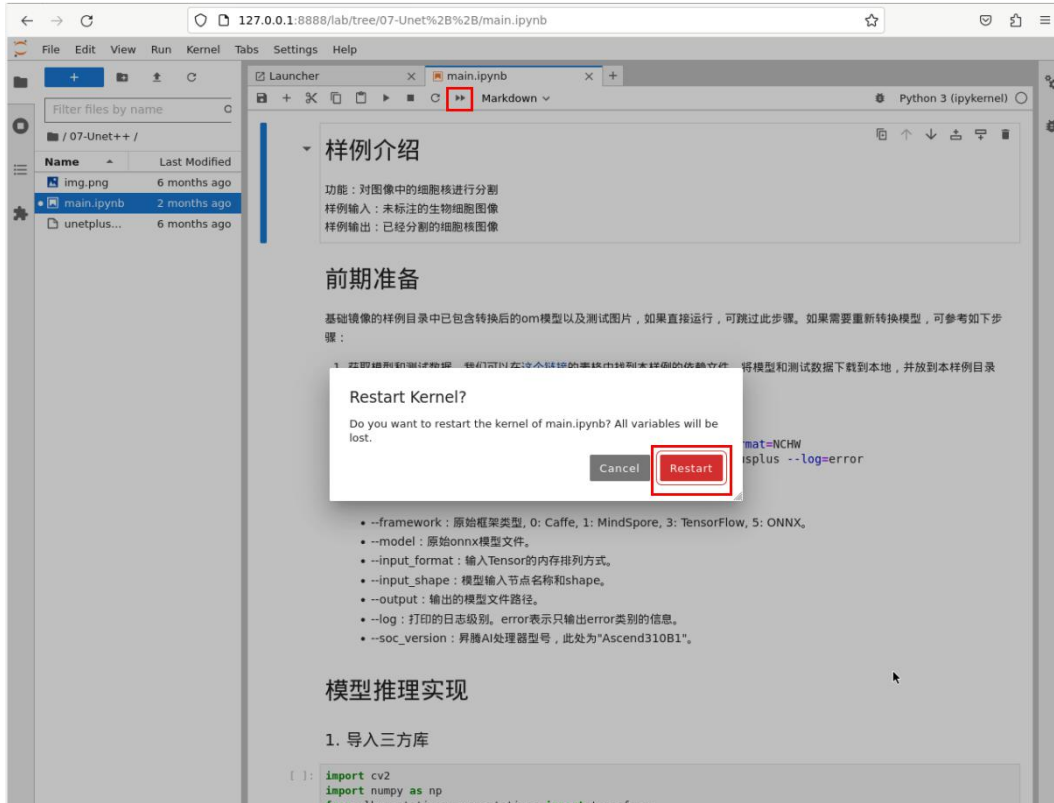
1) 首先在 Jupyter Lab 界面双击“07-Unet++”，进入到该目录下。



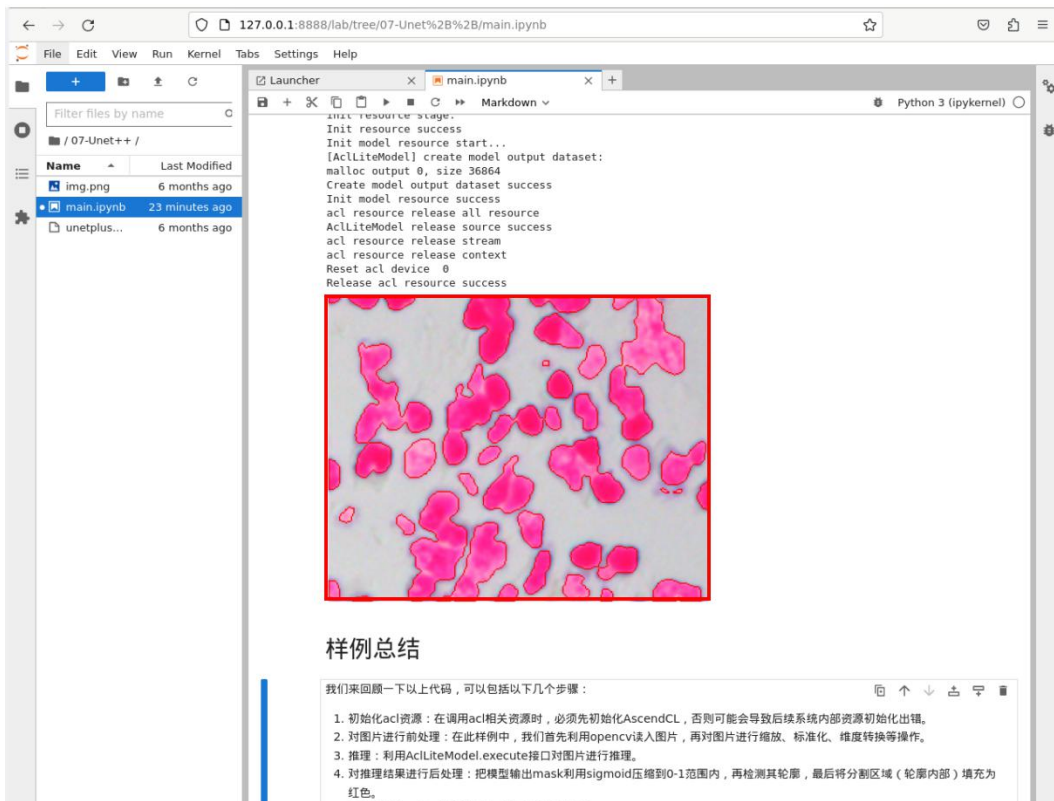
2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该示例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。



3) 单击▶▶按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了一张细胞核的图片。

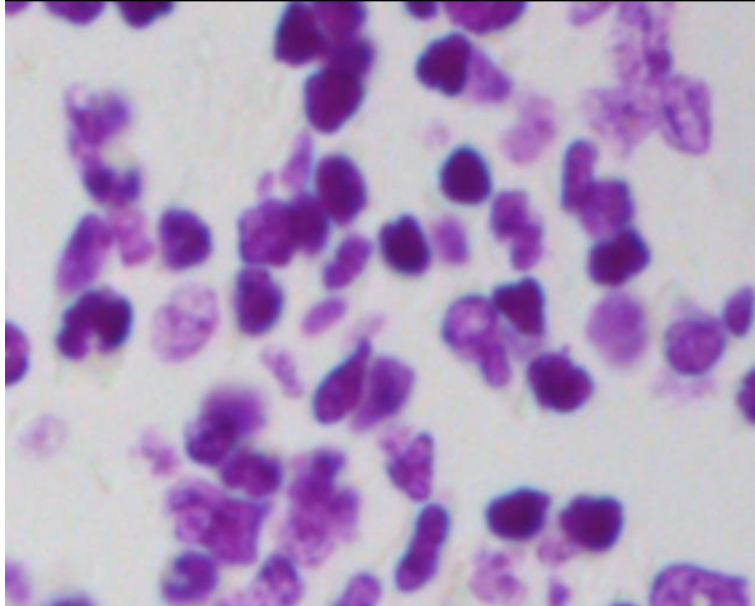




5) 测试图片的保存路径如下所示:

```
/home/HwHiAiUser/samples/notebooks/07-Unet++/img_png
```

6) 测试图片的内容如下所示:

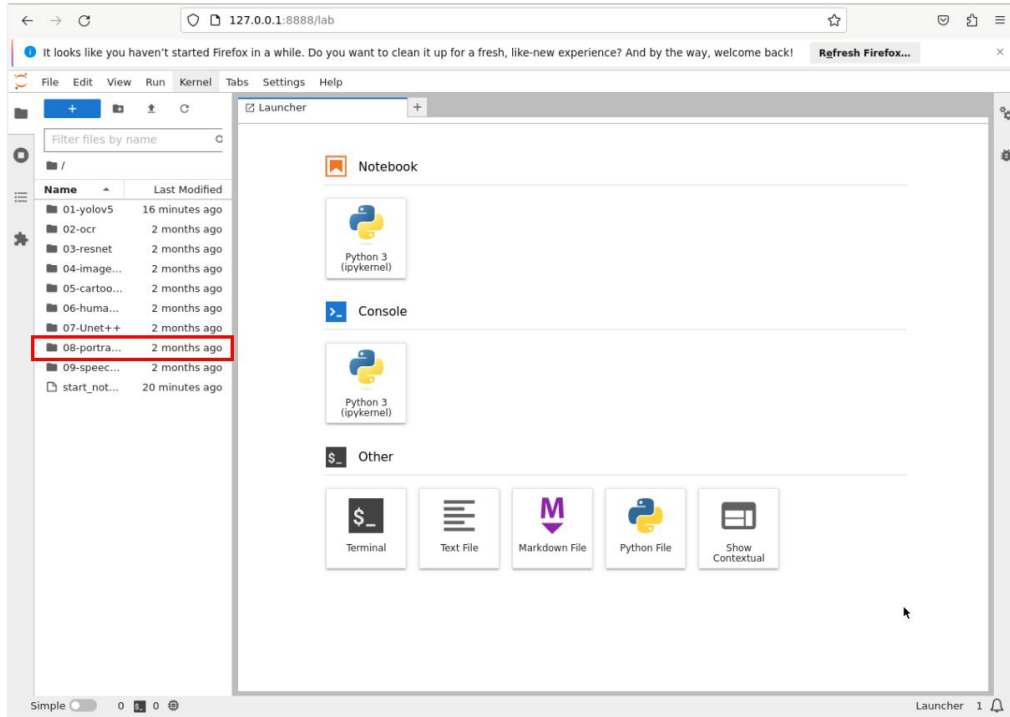


7) 我们可以看到模型对测试图片进行处理，将图片中的细胞核分割出来，并将分割区域（轮廓内部）填充为红色。

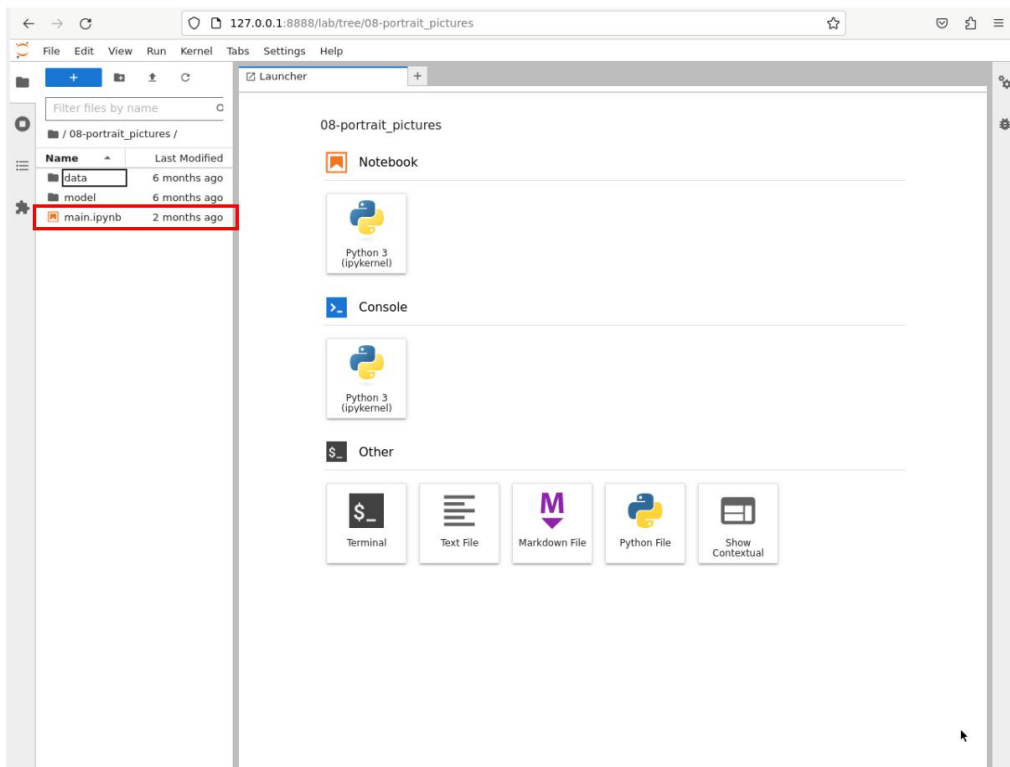
## 4.9. 运行人像分割与背景替换样例


在这个样例中，我们使用了深度学习神经网络 PortraitNet，该模型能够进行人像分割和背景替换。在样例中已经包含转换后的 om 模型和测试图片，可以按照以下流程在 Jupyter Lab 中运行该样例。

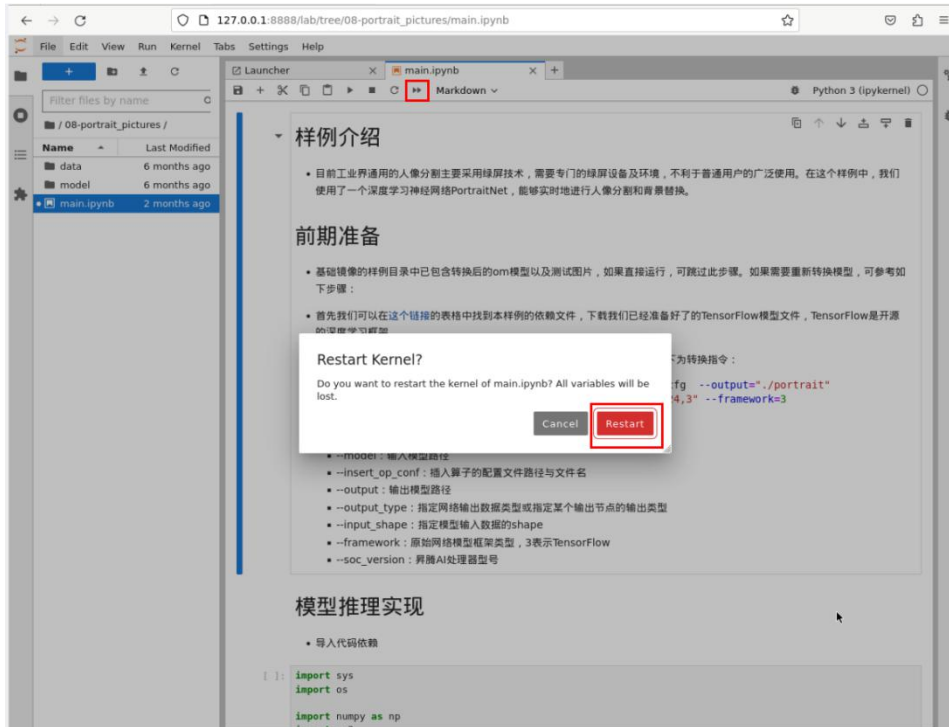
1) 首先在 Jupyter Lab 界面双击“08-portrait\_pictures”进入到该目录下。



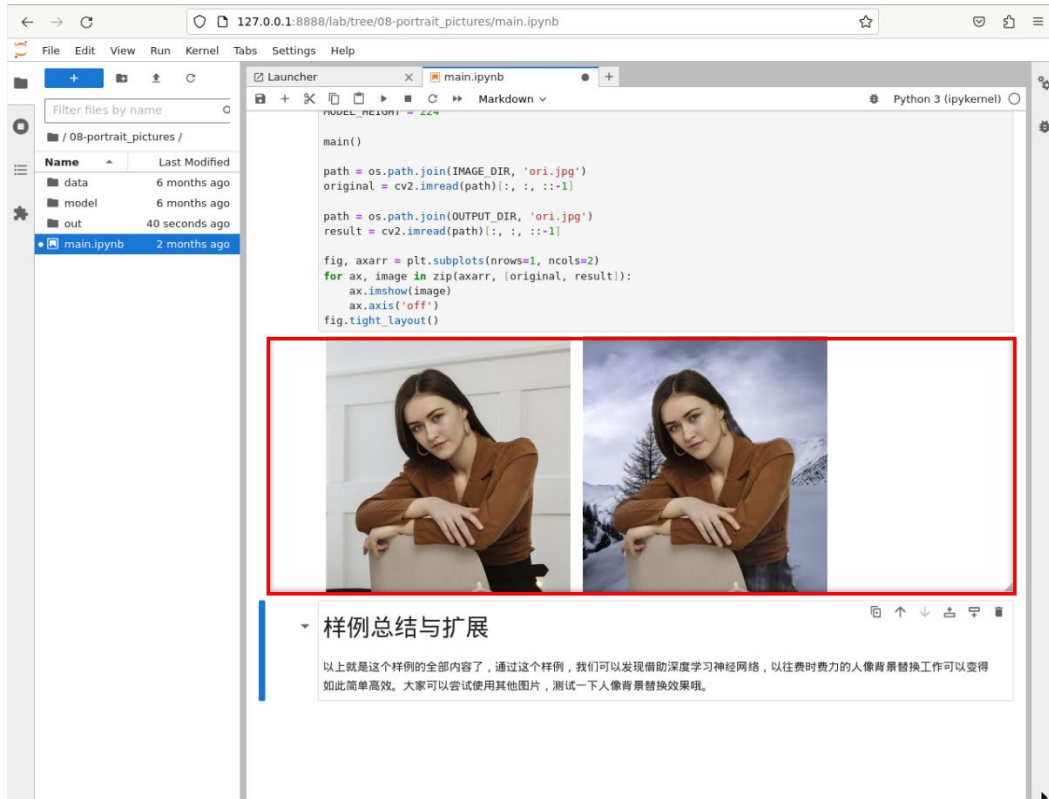
2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该样例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。



3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了两张图片，我们可以看到模型对左侧的图片进行处理，将图片中的人像分割出来，然后把背景替换成了右侧图片中的背景。



5) 该样例用到的测试图片有两张，保存路径如下所示：

**/home/HwHiAiUser/samples/notebooks/08-portrait\_pictures/data/ori.jpg**  
**/home/HwHiAiUser/samples/notebooks/08-portrait\_pictures/data/background.jpg**

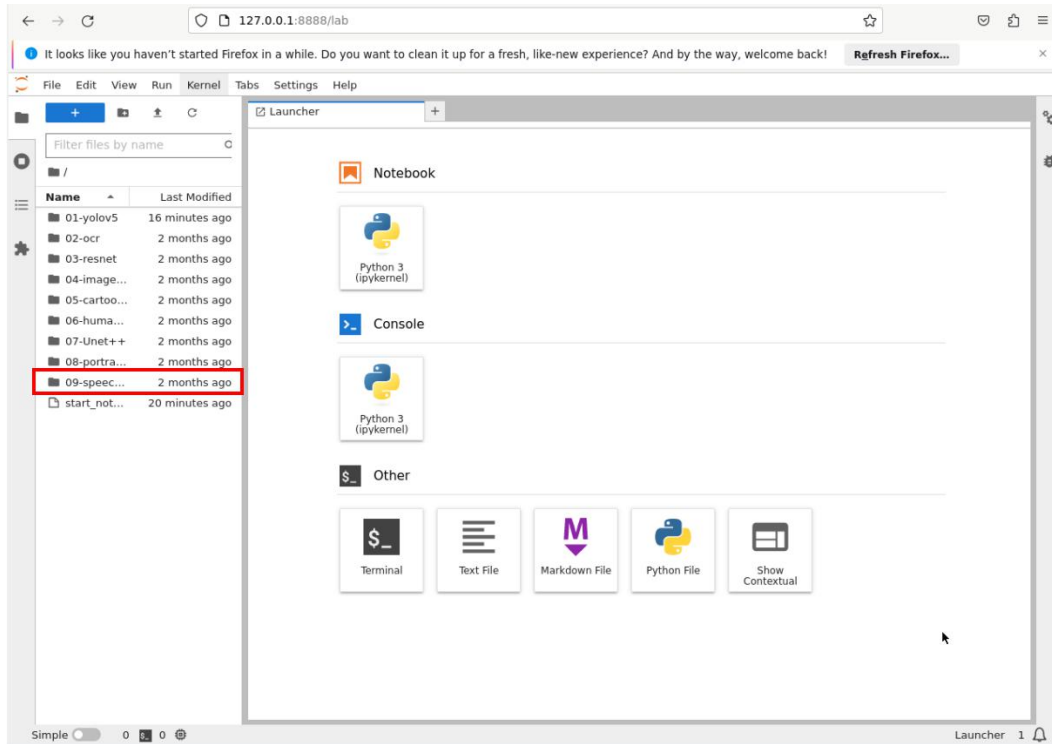
6) 测试图片的内容如下所示：



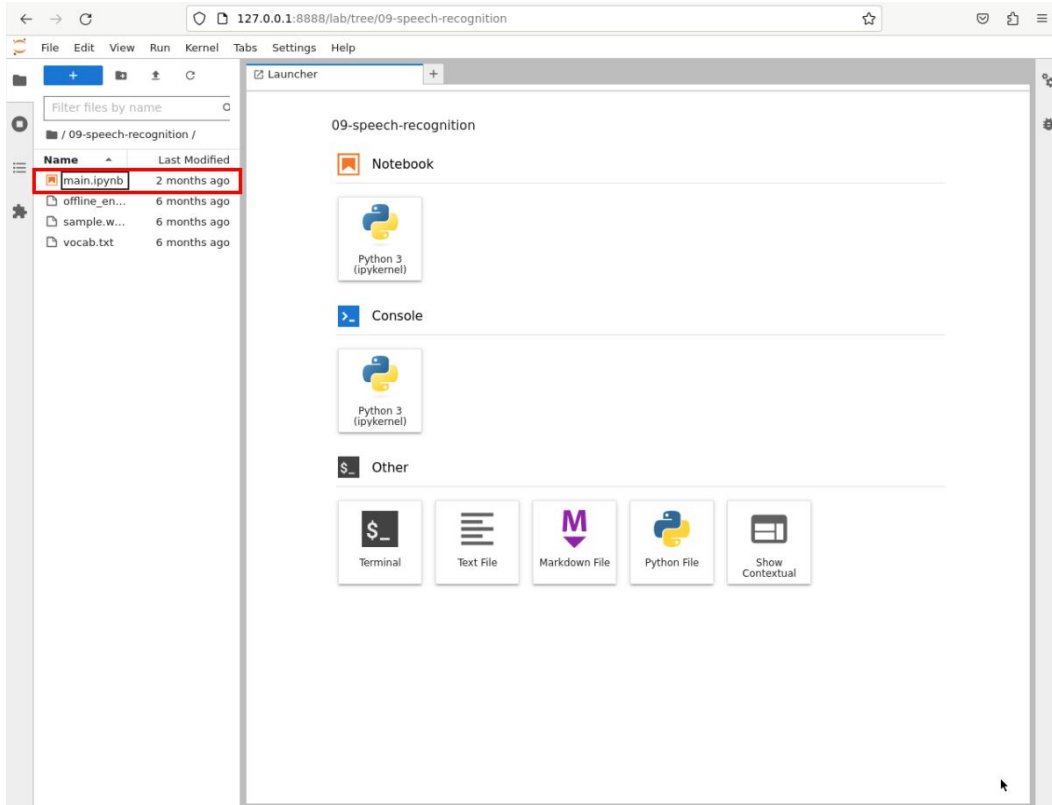
#### 4. 10. 运行语音识别样例


在这个样例中，我们使用了语音识别模型 WeNet，该模型可以指将语音转换为文本。在样例中已经包含转换后的 om 模型和测试语音，可以按照以下流程在 Jupyter Lab 中运行该样例。

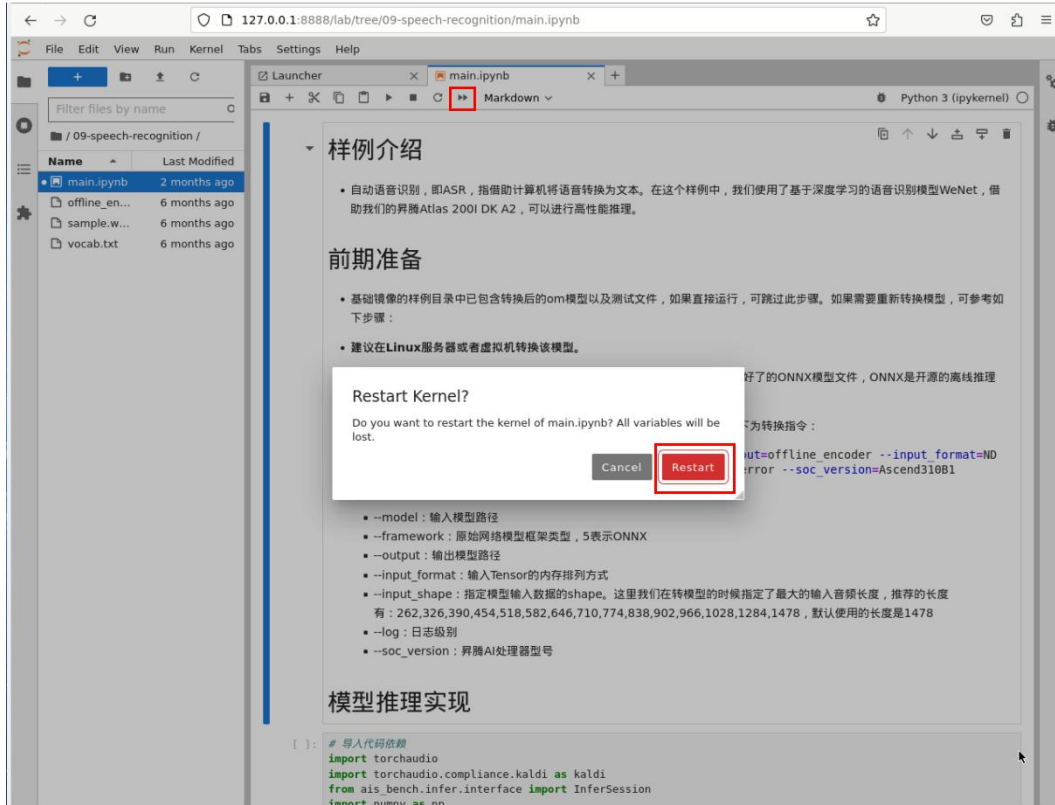
1) 首先在 Jupyter Lab 界面双击“09-speech-recognition”，进入到该目录下。



2) 在该目录下有运行该示例的所有资源，其中 `mian.ipynb` 是在 Jupyter Lab 中运行该样例的文件，双击打开 `main.ipynb`，在右侧窗口中会显示 `main.ipynb` 文件中的内容。

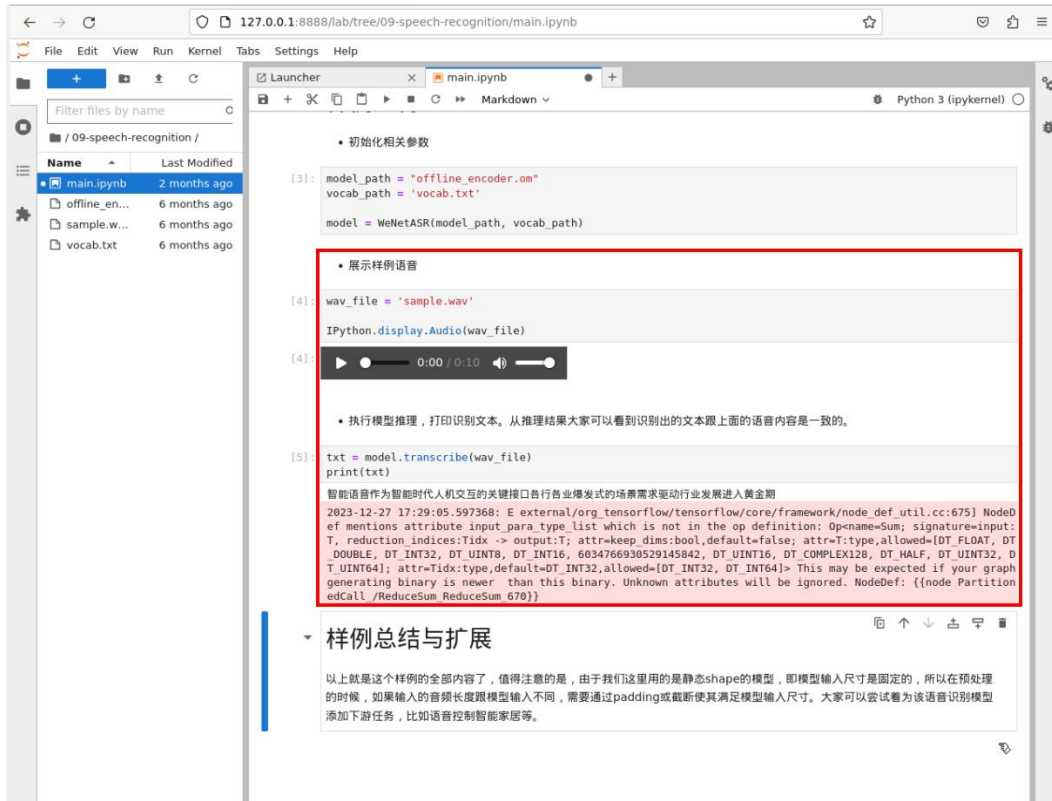


3) 单击  按钮运行样例，在弹出的对话框中单击“Restart”按钮，此时该样例开始运行。



4) 若干秒后，在窗口中出现了如下内容。我们可以看到模型对测试语音进行推理，识别出了语音中的文本信息为“智能语音作为智能时代人机交互的关键接口各行业爆发式的场景需求驱动行业发展进入黄金期”。





5) 测试语音的保存路径如下所示:

**/home/HwHiAiUser/samples/notebooks/09-speech-recognition/sample.wav**

## 5. Linux 内核源码包的使用说明

### 5.1. 编译主机系统的需求

目前的 Linux 内核源码包只在 **Ubuntu 22.04** 的 X64 电脑上测试过，所以首先请确保自己电脑安装的 Ubuntu 版本是 Ubuntu 22.04。查看电脑已安装的 Ubuntu 版本的命令如下所示，如果 Release 字段显示的不是 **22.04**，说明当前使用的 Ubuntu 版本不符合要求，请更换系统后再进行下面的操作。

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:  Ubuntu 22.04 LTS
Release:         22.04
Codename:       jammy
```

如果电脑安装的是 Windows 系统，没有安装有 Ubuntu 22.04 的电脑，可以考虑使用 **VirtualBox** 或者 **VMware** 来在 Windows 系统中安装一个 Ubuntu 22.04 虚拟机。但是请注意，Linux 内核源码包没有在 WSL 虚拟机中测试过，所以无法确保能在 WSL 中正常运行。Ubuntu 22.04 **amd64** 版本的安装镜像下载地址为：

```
https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/22.04/ubuntu-22.04-desktop-amd64.iso
```

在电脑中或者虚拟机中安装完 Ubuntu 22.04 后，请先设置 Ubuntu 22.04 的软件源为清华源（或者其他速度快的国内源），不然后面安装软件的时候很容易由于网络原因而出错。替换清华源的步骤如下所示：

- 1) 替换清华源的方法参考这个网页的说明即可。

```
https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/
```

- 2) 注意 Ubuntu 版本需要切换到 22.04。



## Ubuntu 镜像使用帮助

Ubuntu 的软件源配置文件是 `/etc/apt/sources.list`。将系统自带的该文件做个备份，将该文件替换为下面内容，即可使用 TUNA 的软件源镜像。

选择你的ubuntu版本:

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

3) 需要替换的 `/etc/apt/sources.list` 文件的内容为:

```
test@test:~$ sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak
test@test:~$ sudo vim /etc/apt/sources.list
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

4) 替换完后需要更新下软件包列表，并确保没有报错。

```
test@test:~$ sudo apt-get update
```

## 5.2. 安装交叉编译工具链和依赖包

1) 交叉编译工具链的压缩包可以从开发板的资料下载页面下载到。步骤为:

a. 打开下面的链接:

<http://www.orange-pi.cn/html/hardWare/computerAndMicrocontrollers/service-and->

[support/Orange-Pi-AIpro.html](#)

b. 然后选择官方工具。



c. 然后下载交叉编译工具链文件夹中的 **toolchain.tar.gz** 压缩包。



2) 然后在 Ubuntu 22.04 电脑中执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

3) 然后安装下面的依赖包

```
root@test:~# apt install -y python3 make gcc unzip pigz bison flex libncurses-dev cmake
root@test:~# apt install -y squashfs-tools bc device-tree-compiler libssl-dev rpm2cpio g++
```

4) 然后执行如下命令，创建 **/opt/compiler** 目录，并进入到 **/opt/compiler** 目录。

```
root@test:~# mkdir /opt/compiler
root@test:~# cd /opt/compiler
```

5) 然后将下载的 **toolchain.tar.gz** 复制到 **/opt/compiler** 中，再使用下面的命令将 **toolchain.tar.gz** 解压到 **/opt/compiler** 中。

```
root@test:/opt/compiler# tar -xvf toolchain.tar.gz
```

6) 解压后的交叉编译工具链如下所示：

```
root@test:/opt/compiler# ls toolchain
aarch64-target-linux-gnu bin include lib lib64 libexec sysroot
```

7) 然后在配置文件中增加交叉编译工具链路径。

```
root@test:~# echo "export PATH=/opt/compiler/toolchain/bin:\$PATH: " >> /etc/profile
```

8) 然后执行如下命令，使环境变量生效。

```
root@test:~# source /etc/profile
```

9) 然后可以执行如下命令，查看交叉编译工具链版本。如果显示有版本信息，则表明安装工具链成功。

```
root@test:~# aarch64-target-linux-gnu-gcc -v
Using built-in specs.
COLLECT_GCC=aarch64-target-linux-gnu-gcc
COLLECT_LTO_WRAPPER=/opt/compiler/toolchain/bin/../libexec/gcc/aarch64-target-linux-gnu/7.3.0/lto-wrapper
Target: aarch64-target-linux-gnu
.....
Thread model: posix
gcc version 7.3.0 (Do-Compiler V100R001C13B001)
```

### 5.3. 下载解压 Linux 内核源码包

1) Linux 内核源码压缩包可以从开发板的资料下载页面下载到。步骤为：

a. 打开下面的链接：

```
http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-AIpro.html
```

b. 然后选择 Linux 源码。

### 官方资料



c. 然后下载 Linux 内核源码压缩包 **Ascend310B-source-opi.tar.gz**。



2) 然后在 Ubuntu 22.04 电脑中，然后执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

3) 然后将下载好的 Linux 内核源码压缩包 **Ascend310B-source-opi.tar.gz** 拷贝到 Ubuntu 22.04 电脑的 **/opt** 目录下，然后使用下面的命令解压 Linux 内核源码压缩包。

```
root@test:/opt # tar xzf Ascend310B-source-opi.tar.gz
```

4) 解压后的 Linux 内核源码包的内容如下所示：

```
root@test:/opt # cd Ascend310B-source-opi
root@test:/opt/Ascend310B-source-opi# ls
abl build.sh config driver dtb kernel scripts tools
```

## 5.4. 编译并生效内核 Image 文件的方法

1) 首先进入 Ubuntu 22.04 电脑中，然后执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```



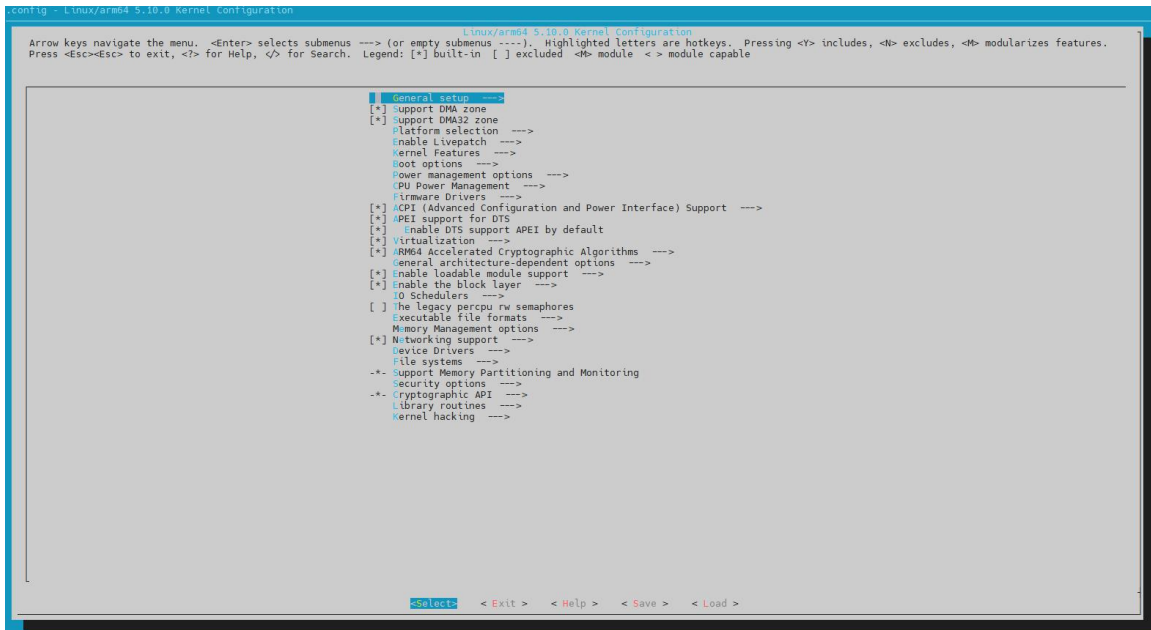
2) 然后执行如下命令，进入“Ascend310B-source-opi”目录。

```
root@test:~# cd /opt/Ascend310B-source-opi
```

3) 然后执行如下命令，即可开始编译内核。

```
root@test:/opt/Ascend310B-source-opi# bash build.sh kernel
```

4) 执行过程会弹出内核配置选项的图形界面，如果不需要修改，直接选择 **Exit** 退出即可。



5) 编译完成后会打印下面的信息：

```
generate /opt/Ascend310B-source-opi/output/kernel_modules success!
```

```
generate /opt/Ascend310B-source-opi/output/Image success!
```

```
sign /opt/Ascend310B-source-opi/output/Image success!
```

6) 编译后的 Image 文件会存放于 **Ascend310B-source/output** 目录下。

```
root@test:/opt/Ascend310B-source-opi# ls output/Image
output/Image
```

7) 编译后更新内核 Image 文件的方法如下所示：

- a. 首先登录开发板的 Linux 系统。
- b. 然后将编译好的 **Image** 文件上传至开发板 Linux 系统的任意目录下，例如



**/root** 目录下。

- c. 然后进入**/root** 目录。
- d. 然后执行如下命令，更新 Image 文件。
  - a) NVMe SSD 启动:

```
dd if=Image of=/dev/nvme0n1 count=61440 seek=32768 bs=512
```

b) SATA SSD 启动:

```
dd if=Image of=/dev/sda count=61440 seek=32768 bs=512
```

c) eMMC 启动:

```
dd if=Image of=/dev/mmcblk0 count=61440 seek=32768 bs=512
```

d) TF 卡启动:

```
dd if=Image of=/dev/mmcblk1 count=61440 seek=32768 bs=512
```

## 5.5. 编译并生效内核 DTB 文件的方法

1) 首先进入 Ubuntu 22.04 电脑中，然后执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

2) 然后执行如下命令，进入 **Ascend310B-source-opi** 目录。

```
root@test:~# cd /opt/Ascend310B-source-opi
```

3) 开发板使用的 DTS 文件如下所示，

```
Ascend310B-source-opi/dtb/dts/hi1910b/hi1910BL/hi1910B-default.dts
```

4) 然后执行如下命令，即可开始编译 DTB。

```
root@test:/opt/Ascend310B-source-opi# bash build.sh dtb
```

5) 如果最后打印如下信息表示编译内核 DTB 文件成功。

```
generate /opt/Ascend310B-source-opi/output/dt.img success!  
sign /opt/Ascend310B-source-opi/output/dt.img success!
```

6) 生成的 DTB 文件为 **dt.img**，存放在 **output** 目录下:

```
root@orange-pi-M600:/opt/Ascend310B-source-opi# ls output/dt.img  
output/dt.img
```

7) 编译后更新内核 DTB 文件的方法如下所示:





- a. 首先登录开发板的 Linux 系统。
- b. 然后将编译好的 **dt.img** 文件上传至开发板 Linux 系统的任意目录下，例如 **/root** 目录下。
- c. 然后进入 **/root** 目录。
- d. 然后执行如下命令，更新 **dt.img** 文件。DTB 文件有主备两份，下面的两条命令中第一条是更新主区的 DTB 文件，第二条是更新备区的 DTB 文件。测试时，一般只需更新主区的 DTB 文件，等测试没问题后再更新备区的 DTB 文件。

a) NVMe SSD 启动:

```
dd if=dt.img of=/dev/nvme0n1 count=4096 seek=114688 bs=512
dd if=dt.img of=/dev/nvme0n1 count=4096 seek=376832 bs=512
```

b) SATA SSD 启动:

```
dd if=dt.img of=/dev/sda count=4096 seek=114688 bs=512
dd if=dt.img of=/dev/sda count=4096 seek=376832 bs=512
```

c) eMMC 启动:

```
dd if=dt.img of=/dev/mmcblk0 count=4096 seek=114688 bs=512
dd if=dt.img of=/dev/mmcblk0 count=4096 seek=376832 bs=512
```

d) TF 卡启动:

```
dd if=dt.img of=/dev/mmcblk1 count=4096 seek=114688 bs=512
dd if=dt.img of=/dev/mmcblk1 count=4096 seek=376832 bs=512
```

## 6. Linux 镜像编译脚本的使用说明

### 6.1. 编译主机系统的需求

目前的 Linux 镜像编译脚本只在 **Ubuntu 22.04** 的 X64 电脑上测试过，所以首先请确保自己电脑安装的 Ubuntu 版本是 Ubuntu 22.04。查看电脑已安装的 Ubuntu 版本的命令如下所示，如果 Release 字段显示的不是 **22.04**，说明当前使用的 Ubuntu 版本不符合要求，请更换系统后再进行下面的操作。

```
test@test:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:   Ubuntu 22.04 LTS
Release:       22.04
Codename:      jammy
```

如果电脑安装的是 Windows 系统，没有安装有 Ubuntu 22.04 的电脑，可以考虑使用 **VirtualBox** 或者 **VMware** 来在 Windows 系统中安装一个 Ubuntu 22.04 虚拟机。但是请注意，Linux 镜像编译脚本没有在 WSL 虚拟机中测试过，所以无法确保能在 WSL 中正常运行。Ubuntu 22.04 **amd64** 版本的安装镜像下载地址为：

<https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/22.04/ubuntu-22.04-desktop-amd64.iso>

在电脑中或者虚拟机中安装完 Ubuntu 22.04 后，请先设置 Ubuntu 22.04 的软件源为清华源（或者其他速度快的国内源），不然后面安装软件的时候很容易由于网络原因而出错。替换清华源的步骤如下所示：

- 1) 替换清华源的方法参考这个网页的说明即可。

<https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/>

- 2) 注意 Ubuntu 版本需要切换到 22.04。



## Ubuntu 镜像使用帮助

Ubuntu 的软件源配置文件是 `/etc/apt/sources.list`。将系统自带的该文件做个备份，将该文件替换为下面内容，即可使用 TUNA 的软件源镜像。

选择你的ubuntu版本:

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

3) 需要替换的 `/etc/apt/sources.list` 文件的内容为:

```
test@test:~$ sudo mv /etc/apt/sources.list cat /etc/apt/sources.list.bak
test@test:~$ sudo vim /etc/apt/sources.list
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-updates main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-backports main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-security main restricted universe multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ jammy-proposed main restricted universe multiverse
```

4) 替换完后需要更新下软件包列表，并确保没有报错。

```
test@test:~$ sudo apt-get update
```

## 6.2. 制作 Linux 镜像需要准备的东西

制作 Linux 镜像所需软硬件条件如下:

1) 一台带有 USB 接口、系统为 Ubuntu 22.04 的 X64 电脑。

- 2) 一个 TF 卡读卡器。
- 3) 一张容量至少为 32GB 的 TF 卡。
- 4) 请确保电脑的网络畅通。

### 6.3. 下载 Linux 镜像编译脚本的源码压缩包

1) 编译 Linux 镜像需要用到的脚本、驱动包、CANN 包、基础 rootfs 等软件全部都打包成了一个压缩包放在了百度网盘上。下载步骤如下所示：

- a. 首先打开开发板的资料下载页面：

<http://www.orange-pi.cn/html/hardware/computerAndMicrocontrollers/service-and-support/Orange-Pi-AIpro.html>

- b. 然后选择 Linux 源码。



- c. 然后下载 **image-builder.tar.gz** 压缩包。



2) 然后在 Ubuntu 22.04 电脑中执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

3) 然后将下载的 **image-builder.tar.gz** 拷贝到 **/opt** 目录下，再使用下面的命令将其解压。

```
root@test:/opt# tar xzf image-builder.tar.gz
root@test:/opt# cd image-builder/src
root@test:/opt/image-builder/src# ls
complete compress minimal
```

4) 解压后的 **image-builder/src** 目录中包含最小镜像(minimal)、完整镜像(complete)、压缩扩容镜像(compress) 三个模块，他们对应制作镜像的三个步骤。

模块名称	模块名称	功能简介
最小镜像	src/minimal	可以在开发板上启动但缺少部分依赖的镜像
最小镜像	src/minimal	完整依赖镜像
压缩扩容镜像	src/compress	带有压缩扩容功能的完整依赖镜像

## 6.4. 制作最小镜像的方法

1) 首先请确保 Ubuntu22.04 系统没有设置为中文环境，如果有的话，请改回英文环境，不然制作最小镜像时会失败。

2) 然后将 TF 卡插入读卡器，然后将读卡器插入电脑的 USB 接口中。

3) 然后在 Ubuntu 22.04 电脑中执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

4) 然后安装下面的依赖包

```
root@test:~# apt-get install -y qemu qemu-user qemu-user-static binfmt-support
```

5) 然后将 **emmc-head** 命令依赖的两个库文件拷贝到 **/usr/lib64** 下面，步骤如下所示：

a. 首先打开开发板的资料下载页面：

```
http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-AIpro.html
```

b. 然后选择 Linux 源码。

## 官方资料



c. 然后下载依赖的库文件。



d. 再将下载好的库文件拷贝到 Ubuntu 22.04 系统的 `/usr/lib64` 目录下。

```
root@test:~# cp library/* /usr/lib64/
```

e. 然后运行下 `emmc-head` 命令，如果输出和下面一样，说明库文件安装正确。

```
root@test:~# cd /opt/image-builder/src/minimal
root@test:/opt/image-builder/src/minimal# ./ubuntu/22.04/download/emmc-head --help
Usages: emmc-head firmware_path boot_a_devname boot_b_devname [force_recover]
The following files must be contained in firmware_path:
Image, itrustee.img, dt.img, initrd.
boot_a_devname: A Partition boot device name, for example, eMMC:mmcblk0p2, SD:mmcblk1p2
boot_b_devname: B Partition boot device name, for example, eMMC:mmcblk0p3, SD:mmcblk1p3
force_recover: force recover flag.
Example: /var/davinci/driver/emmc-head ./firmware /dev/mmcblk0p2 /dev/mmcblk0p3
```

6) 然后进入 `image-builder` 源码所在的目录，然后再进入 `src/minimal` 目录。

```
root@test:~# cd /opt/image-builder
root@test:/opt/image-builder# cd src/minimal
root@test:/opt/image-builder/src/minimal# ls
base.sh openEuler ubuntu
```



7) 镜像制作过程中需要用到的，已预先下载好的软件存放的路径为：

a. openEuler 对应的 download 文件夹的路径：

```
openEuler/22.03/download
```

b. ubuntu 对应的 download 文件夹的路径：

```
ubuntu/22.04/download
```

8) 然后使用 `fdisk -l` 命令查看下 TF 卡的磁盘编号，如 `/dev/sdb`。

```
root@test:~# fdisk -l
.....
Disk /dev/sdb: 29.72 GiB, 31914983424 bytes, 62333952 sectors
Disk model: MassStorageClass
.....
```

9) 然后开始制作最小镜像到 TF 卡中，命令如下所示：

a. Ubuntu 22.04 镜像的命令如下所示：

```
root@test:/opt/image-builder/src/minimal# bash base.sh ubuntu/22.04/ /dev/sdX ubuntu/22.04/download/
```

b. openEuler 22.03 镜像的命令如下所示：

```
root@test:/opt/image-builder/src/minimal# bash base.sh openEuler/22.03/ /dev/sdX openEuler/22.03/download/
```

**注意，上面的命令中的 `/dev/sdX` 需要换成 TF 卡对应的磁盘编号，请不要照抄。**

10) 正常运行完后会打印如下的信息，然后就可以退出 TF 卡，然后将 TF 卡插入开发板中启动运行了（注意 `ubuntu/22.04/download` 或 `openEuler/22.03/download/` 文件夹的内容请不要删除，目前脚本还无法通过网络来自动下载制作最小镜像需要的部分软件包，只能用已经缓存好的）。

```
[2024-02-03 15:49:41] [MINIMAL] Minimal image build successful!
```

11) 注意，制作好的最小镜像第一次启动时会自动重启一次。请等待自动重启完成后，再使用串口登录系统进行其他操作。

12) Ubuntu 和 openEuler 中都有一个 `cfg.json` 文件来控制脚本运行哪些步骤。默认是所以步骤都运行的（如下所示，都为 `y`）。在制作最小镜像的过程中，每运行完一个步骤，就会将对应步骤后面的 `y` 修改为 `n`。等所有步骤都运行完成后，再将所有的 `n` 重新修改为 `y`。当中间的某步出错退出后，再次执行脚本时，会从前面中断

的那步继续运行。

```
root@test:/opt/image-builder/src# cat minimal/ubuntu/22.04/cfg.json
{
.....

"function": {
  "get_base_image": "y",
  "get_npu_driver": "y",
  "get_hdk": "y",
  "get_file_system": "y",
  "write_to_device": "y",
  "post_process": "y",
  "opi_func": "y"
}
}
```

## 6.5. 制作完整镜像的方法

1) 首先请按照[制作最小镜像的方法](#)一小节的说明制作好最小镜像，然后启动最小镜像并使用 **root** 用户登录串口命令行。如果没有环境自己制作最小镜像，可以直接下载 Orange Pi 提供的最小镜像（即 **minimal 版本的镜像**）文件，然后烧录到 32GB 或 32GB 以上容量的 TF 卡中使用。

2) 然后请确保开发板能正常上网。

3) 最小镜像（即 **minimal 版本的镜像**）中已经包含了制作完整镜像需要的脚本和部分软件包了，他们存放的路径如下所示：

```
/opt/complete/
```

4) 然后在 Ubuntu 22.04 电脑中执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

5) 然后进入 **complete** 中。

```
root@orangepiaipro:~# cd /opt/complete
root@orangepiaipro:/opt/complete# ls
```



```
base.sh download openEuler ubuntu
```

6) 然后使用如下命令进行完整镜像的制作。

a. Ubuntu 镜像的命令如下所示：

```
root@orangepiaipro:/opt/complete# bash base.sh -v ubuntu/22.04/ download/
```

b. OpenEuler 镜像的命令如下所示：

```
root@orangepiaipro:/opt/complete# bash base.sh -v openEuler/22.03/ download/
```

7) Ubuntu 和 openEuler 中都有一个 **cfg.json** 文件来控制脚本运行哪些步骤。默认是所以步骤都运行的（如下所示，都为 **y**）。在制作完整镜像的过程中，每运行完一个步骤，就会将对应步骤后面的 **y** 修改为 **n**。等所有步骤都运行完成后，再将所有的 **n** 重新修改为 **y**。当中间的某步出错退出后，再次执行脚本时，会从前面中断的那步继续运行。

```
root@test:/opt/image-builder/src/complete# cat ubuntu/22.04/cfg.json
{
.....

"function": {
  "pre_process": "y",
  "apt_install": "y",
  "install_miniconda": "y",
  "python_pip_install": "y",
  "install_cann": "y",
  "install_mxvision": "y",
  "install_acllite": "y",
  "add_local_desktop": "y",
  "add_remote_desktop": "y",
  "opi_func": "y",
  "post_process": "y"
},
.....
}
```

## 6.6. 制作压缩扩容镜像的方法

1) 首先将制作好的完整镜像的 TF 卡插入读卡器，然后将读卡器插入电脑的 USB 接口中。

2) 然后在 Ubuntu 22.04 电脑中执行如下命令，切换至 root 用户。

```
test@test:~$ sudo -i
```

3) 然后将 **emmc-head** 命令依赖的两个库文件拷贝到**/usr/lib64** 下面（如果前面已经做了这步操作，可以跳过），步骤如下所示：

a. 首先打开开发板的资料下载页面：

```
http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-AIpro.html
```

b. 然后选择 Linux 源码。



c. 然后下载依赖的库文件。



d. 再将下载好的库文件拷贝到 Ubuntu 22.04 系统的**/usr/lib64** 目录下。

```
root@test:~# cp library/* /usr/lib64/
```

e. 然后运行下 **emmc-head** 命令，如果输出和下面一样，说明库文件安装正确。

```
root@test:~# cd /opt/image-builder/src/compress
```



```

root@test:/opt/image-builder/src/compress# ./download/emmc-head --help
Usages: emmc-head firmware_path boot_a_devname boot_b_devname [force_recover]
The following files must be contained in firmware_path:
Image, itrustee.img, dt.img, initrd.
boot_a_devname: A Partition boot device name, for example, eMMC:mmcblk0p2, SD:mmcblk1p2
boot_b_devname: B Partition boot device name, for example, eMMC:mmcblk0p3, SD:mmcblk1p3
force_recover: force recover flag.

Example: /var/davinci/driver/emmc-head ./firmware /dev/mmcblk0p2 /dev/mmcblk0p3

```

4) 然后进入 **image-builder** 的 **compress** 目录中。

```

root@test:~# cd /opt/image-builder/src/compress/
root@test:/opt/image-builder/src/compress# ls
base.sh config.ini download E2E_samples_download_tool.sh openEuler ubuntu

```

5) 然后使用 **fdisk -l** 命令查看下 TF 卡的磁盘编号，如 **/dev/sdb**。

```

root@test:/opt/image-builder/src/compress# fdisk -l
.....
Disk /dev/sdb: 29.72 GiB, 31914983424 bytes, 62333952 sectors
Disk model: MassStorageClass
.....

```

6) 然后就可以开始导出 TF 卡中的镜像，命令如下所示：

a. 导出 Ubuntu 22.04 镜像的命令如下所示：

```

root@test:/opt/image-builder/src/compress# bash base.sh -c ubuntu/22.04 /dev/sdX linux.img

```

b. 导出 openEuler 22.03 镜像的命令如下所示：

```

root@test:/opt/image-builder/src/compress# bash base.sh -c openEuler/22.03/ /dev/sdX linux.img

```

**注意，上面的命令中的 /dev/sdX 需要换成 TF 卡对应的磁盘编号，请不要照抄。**

7) 当看到下面的输出时，说明镜像导出并压缩完成。

```

[2024-02-03 16:28:57] [COMPRESS] sd card compress success!

```

8) 导出的 Linux 镜像文件如下所示：

- a. **linux.img**: Linux 镜像文件
- b. **linux.img.xz**: 压缩后的 Linux 镜像文件



```
root@test:/opt/image-builder/src/compress# ls linux.img*  
linux.img linux.img.xz
```

9) 如果不需要压缩 Linux 镜像文件，可以将命令中的 **-c** 选项去掉。

a. 导出 Ubuntu 22.04 镜像的命令如下所示：

```
root@test:/opt/image-builder/src/compress# bash base.sh ubuntu/22.04 /dev/sdX linux.img
```

b. 导出 openEuler 22.03 镜像的命令如下所示：

```
root@test:/opt/image-builder/src/compress# bash base.sh openEuler/22.03/ /dev/sdX linux.img
```

## 7. 附录

### 7.1. 用户手册更新历史

版本	日期	更新说明
v0.1	2024-02-03	初始版本
v0.2	2024-02-27	1. AI CPU 和 control CPU 的设置方法
v0.3	2024-03-18	1. 更新开发板 v1.3 版本的图片 2. 更新耳机音频的测试方法 3. 设置 Swap 内存的方法 4. 测试 MindSpore 的方法

### 7.2. 镜像更新历史

日期	更新说明
2024-02-03	opiaipro_ubuntu22.04_desktop_aarch64_20240202.img.xz opiaipro_ubuntu22.04_minimal_aarch64_20240203.img.xz opiaipro_openEuler23.04_desktop_aarch64_20240203.img.xz  * 初始版本
2024-02-27	opiaipro_ubuntu22.04_minimal_aarch64_20240227.img.xz  * 打开关机按键的配置，解决关机按键没有反应的问题  opiaipro_ubuntu22.04_desktop_aarch64_20240227.img.xz  * 打开关机按键的配置，解决关机按键没有反应的问题 * 禁止休眠 * 预装 docker * 预装 torch_npu  opiaipro_openEuler23.04_desktop_aarch64_20240203.img.xz



	<ul style="list-style-type: none"><li>* 打开关机按键的配置，解决关机按键没有反应的问题</li><li>* 禁止休眠</li><li>* 预装 torch_npu</li></ul>
2024-03-18	<p>opiaipro_ubuntu22.04_desktop_aarch64_20240318.img.xz</p> <ul style="list-style-type: none"><li>* 更新音频测试程序</li><li>* 预装 MindSpore</li></ul>