昇腾 310B 使用手册

目录

1. 开发板参数介绍

1.1. 芯片规格

特征	规格
昇腾 AI 处理器	昇腾 310 系列 AI 处理器 1 个 DaVinciV300 AI core（主频 500MHz） 4 个 TAISHANV200M 处理器核（主频 1.0GHz）
AI 算力 a	半精度（FP16）：4 TFLOPS 整数精度（INT8）：8 TOPS
内存	类型：LPDDR4X 速率：3200Mbps 位宽：64bit 容量：4GB 支持 ECC
存储	内置 SPI flash 提供一个 Micro SD 卡接口，类型为 SD 3.0，向下兼容 SD 2.0 标准 推荐使用 SD 3.0 接口标准的 Micro SD 卡 提供一个 M.2 Key M 连接器，可扩展 M.2 2242/2280 形态 SSD，支持 NVMe
编解码能力	支持 H.264/H.265 Decoder 硬件解码，20 路 1080P (1920 x 1080) 30FPS，YUV420 支持 H.264/H.265 Decoder 硬件解码，2 路 4K (3840 x 2160) 75FPS，YUV420 支持 H.264/H.265 Encoder 硬件编码，12 路 1080P (1920 x 1080) 30FPS，YUV420 支持 H.264/H.265 Encoder 硬件编码，2 路 4K (3840 x 2160) 50FPS，YUV420 JPEG 解码能力 1080P (1920 x 1080) 512FPS，编码能

特征	规格
	力 1080P (1920 x 1080) 256FPS，最大分辨率：16384x 16384，最小分辨率 32x32

1.2. 硬件外设接口

特征	规格
USB 接口	2 个 USB3.0 Host 接口、1 个 TYPEC 3.0 USB OTG
HDMI 接口	2 个 HDMI2.0 接口
电源	12V/2A 电源输入
按键	一个复位按键、一个开机按键、一个升级按键
40Pin 扩展口	用于扩展 UART、IIC、SPI、GPIO 等接口
以太网接口	两个千兆以太网接口

1.3. 开发板示意图

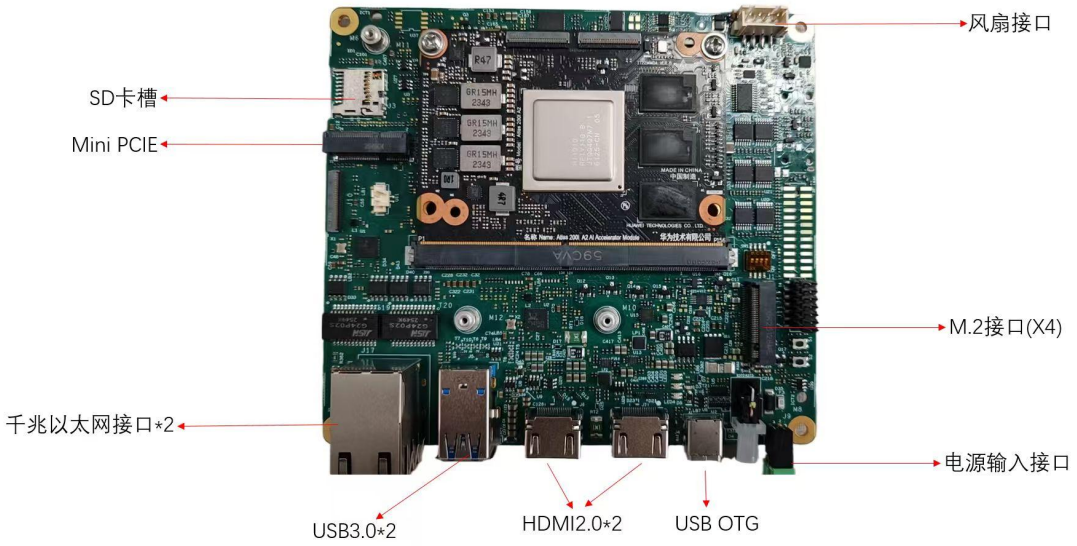

图 1-1 开发板示意图

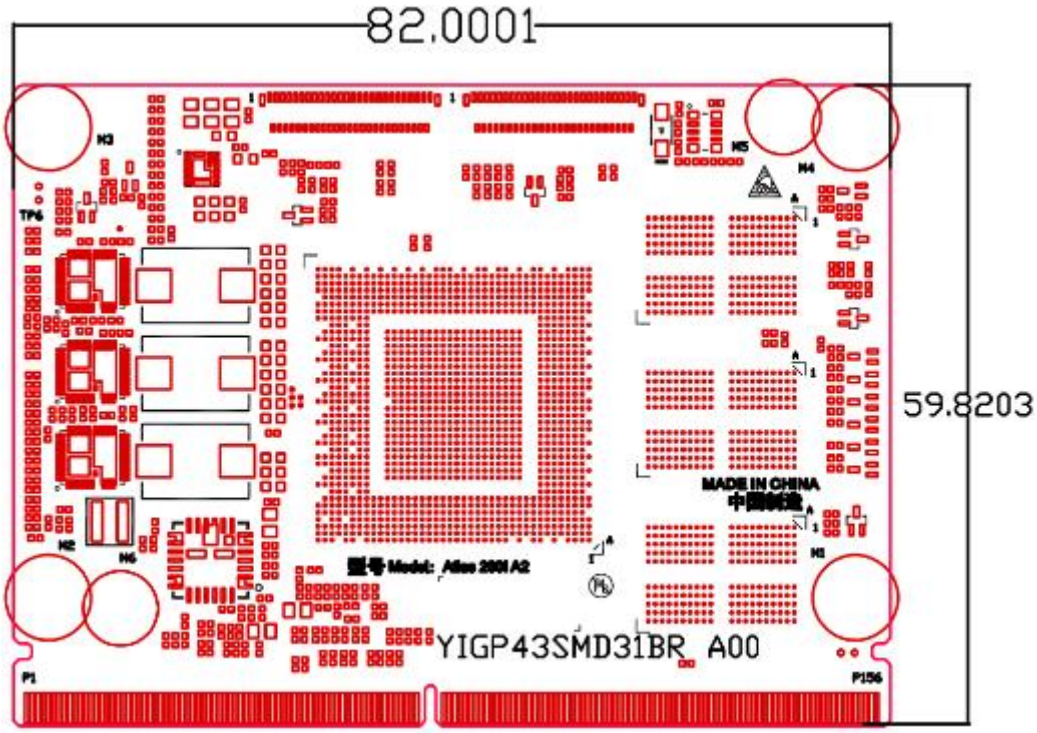

图 1-2 核心板尺寸图

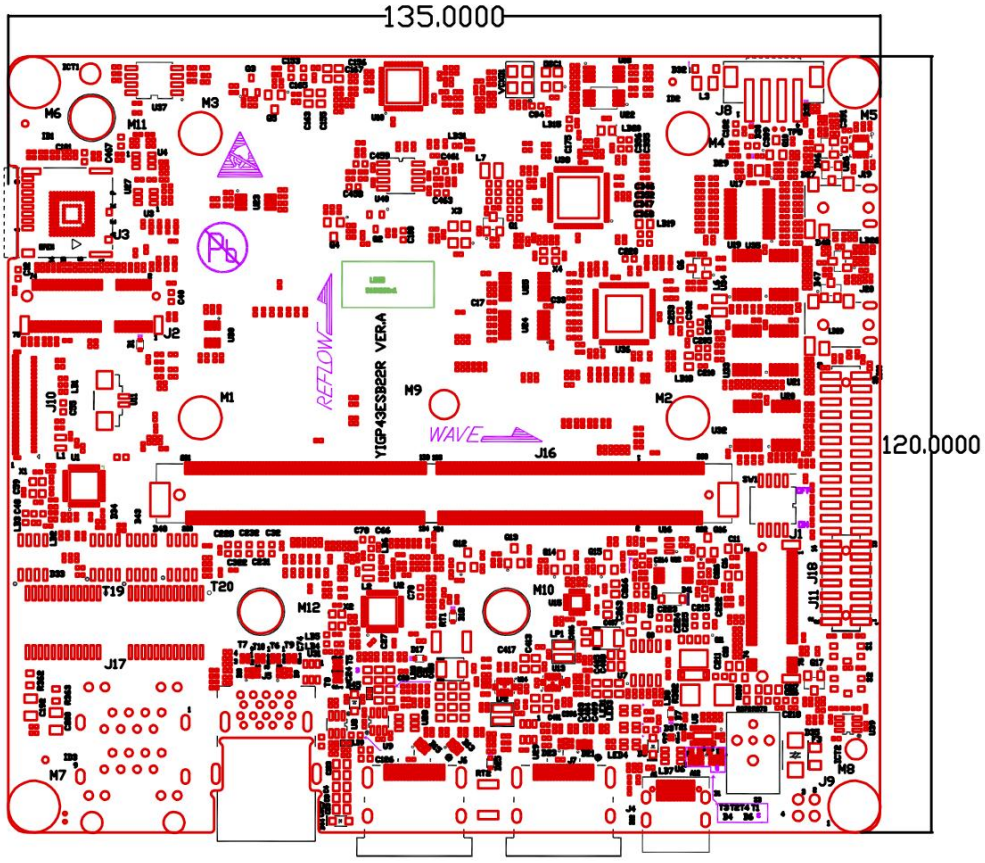

图 1-3 底板尺寸图

2. 开发板使用介绍

同时最好准备一台 Ubuntu22.04 的 PC，用于交叉编译。如果没有，也可以用虚拟机。

2.1. 配件准备

(1) TF 卡，最小 64GB 的 class10 级或以上的高速 TF 卡，不然可能无法正常启动。推荐用 64GB/128GB 的 TF 卡。

(2) 读卡器，用来烧录镜像到 TF 卡。

(3) HDMI 线，用于连接屏幕和开发板。

(4) 网线，用于连接电脑和开发板的网口。

(5) Typec 接口数据线，用于 Typec USB 接口 Device 等功能。

(6) 12V/2A 输入电源

(7) USB 键盘、鼠标、拓展坞、USB 摄像头等。

2.2. 制卡

需要将我们的系统镜像烧录到 TF 卡中，Windows 推荐使用 **Win32Diskimager** 来完成镜像烧录。用 **SD Card Formatter** 来格式化 TF 卡。

目前推荐使用官方制卡工具制卡。具体参考下发链接:

制卡指南

2.3. 使用注意

需要将资料包里的 patch 里的文件上传到板子中。

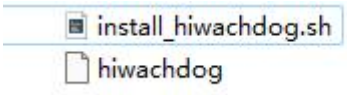

图 2-1

将上述文件通过 SSH 传到板子中。之后运行下列指令:

```
chmod +x install_hiwachdog.sh
sudo bash install_hiwachdog.sh
reboot
```

```
(base) root@davinci-mini:~# cd /opt/
(base) root@davinci-mini:/opt# chmod +x install_hiwachdog.sh
sudo bash install_hiwachdog.sh
reboot
[WARN] 清理旧服务...
[INFO] 安装 hiwachdog 到 /usr/local/bin/hiwachdog
[INFO] 创建 systemd 服务
Created symlink /etc/systemd/system/multi-user.target.wants/hiwachdog.service → /etc/systemd/system/hiwachdog.service.
[INFO] ====== 部署完成 ======
● hiwachdog.service - Hi Watchdog Service
     Loaded: loaded (/etc/systemd/system/hiwachdog.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2022-04-08 03:30:07 CST; 17ms ago
    Process: 4576 ExecStartPre=/bin/sleep 1 (code=exited, status=0/SUCCESS)
   Main PID: 4583 (hiwachdog)
      Tasks: 4 (limit: 7998)
     Memory: 1.2M
     CGroup: /system.slice/hiwachdog.service
             ├─4583 /usr/local/bin/hiwachdog
             ├─4586 sh -c "gpio_operate set_direction 4 0 1"
             └─4587 gpio_operate set_direction 4 0 1

[INFO] 查看日志:  tail -f /var/log/hiwachdog.log
[INFO] 查看状态:  systemctl status hiwachdog
[INFO] 停止服务:  systemctl stop hiwachdog
[INFO] 重启服务:  systemctl restart hiwachdog

Remote side unexpectedly closed network connection


Session stopped
    - Press <Return> to exit tab
    - Press R to restart session
    - Press S to save terminal output to file
```

图 2-2

安装成功后，执行 reboot 后会重启系统。等待板子重新启动后执行:

tail -f /var/log/hiwachdog.log

```
tail -f /var/log/hiwachdog.log(base) root@davinci-mini:~# tail -f /var/log/hiwachdog.log
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
Set gpio pin value successed.
```

图 2-3

看到日志输出上述信息即表示成功，可以正常开发

2.4. 串口调试

2.5. 串口访问

使用 USB 转 TTL 串口线连接电脑与 310B 的串口调试接口（在 40PIN IO 上）。

在 Windows 设备管理器中确认识别到端口（如 COM3）。

2.6. MobaXterm 配置

启动软件：打开 MobaXterm，点击左上角的 Session。

选择类型：在弹出的窗口中选择 Serial。

参数设置：

Serial port：下拉选择对应的 COM 口（如果不确定，请检查设备管理器）。

Speed (Baudrate)：设置为 115200（这是 310B 默认的波特率）。

Data bits / Stop bits / Parity：保持默认值（8, 1, None）。

建立连接：点击 OK。此时窗口应变为黑色背景，按下 Enter 键，系统会弹出登录提示。

图 2.4-1 串口配置示意图

用户名:root

密码:Mind@123

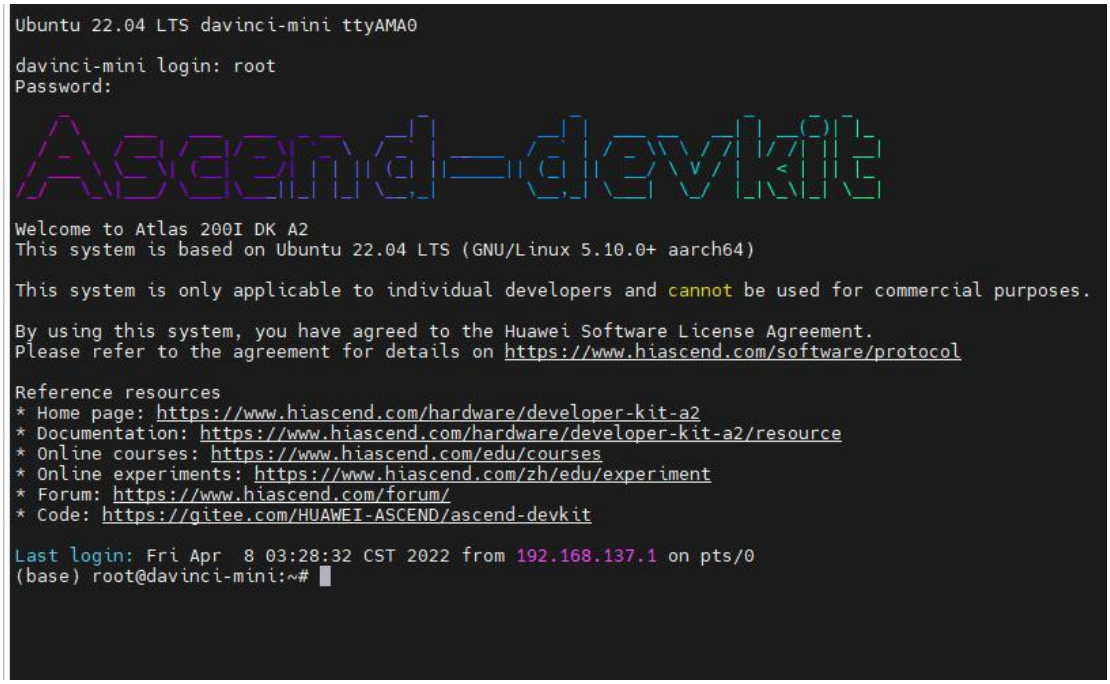

图 2.4-2 串口登录示意图

3. 网口 SSH 登录

3.1. 通过 RJ45 千兆网口

ETH1 是 DHCP(需要接路由器、或电脑共享网络)
ETH0 是固定静态 IP，192.168.2.2
准备工作：
ETH1 (DHCP)： 将网口连接到路由器，或连接到已开启“Internet 共享”的电脑网口。你需要先在路由器后台或通过扫描工具找到分配给设备的 IP 地址。
ETH0 (静态 IP)： 将电脑网口与设备 ETH0 直接相连。需手动将电脑网卡的 IP 设置为 192.168.137.x 网段（例如 192.168.137.10），掩码 255.255.255.0。
MobaXterm 操作步骤：
打开 MobaXterm，点击左上角的 Session 按钮。在弹出窗口选择 SSH 协议。
Remote host(远程主机): 输入 192.168.137.100（若使用 ETH1）
Specify username(指定用户名): 勾选此项并输入用户名 root。
端口:默认 22
点击 OK。

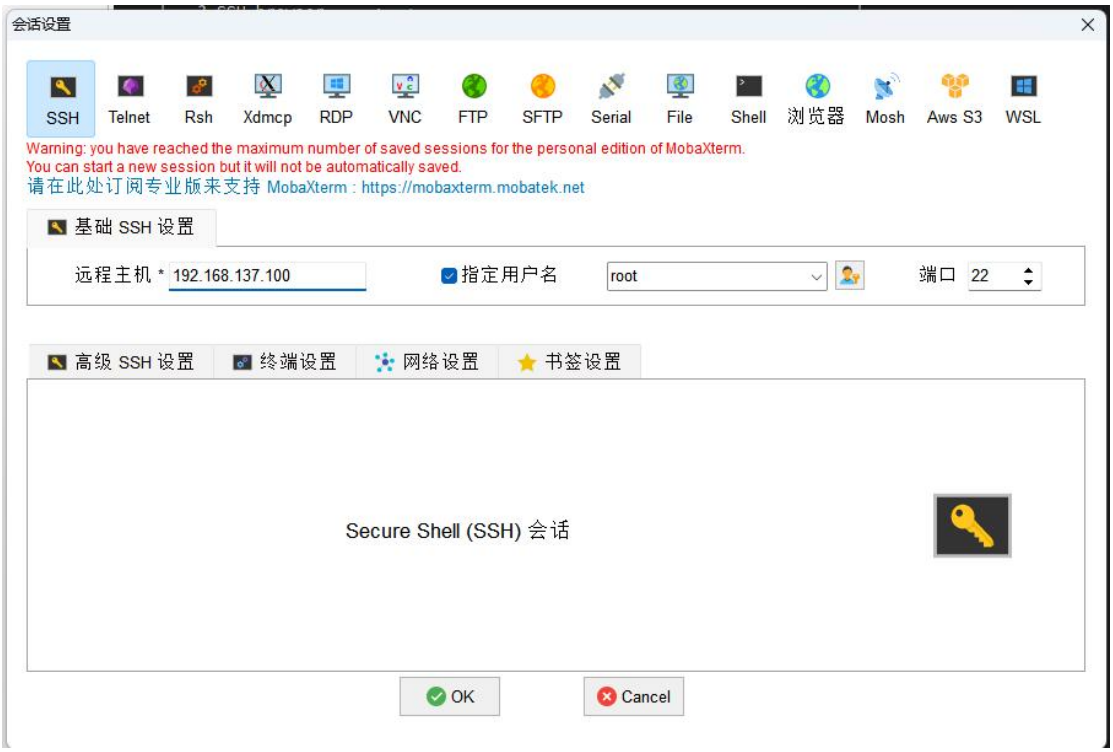

图 3-1 设置示意图

第一次访问要在黑色终端界面输入密码（输入时字符不显示，输完回车即可）。
密码:Mind@123
输入完成后，访问进去，即可看到下面的界面

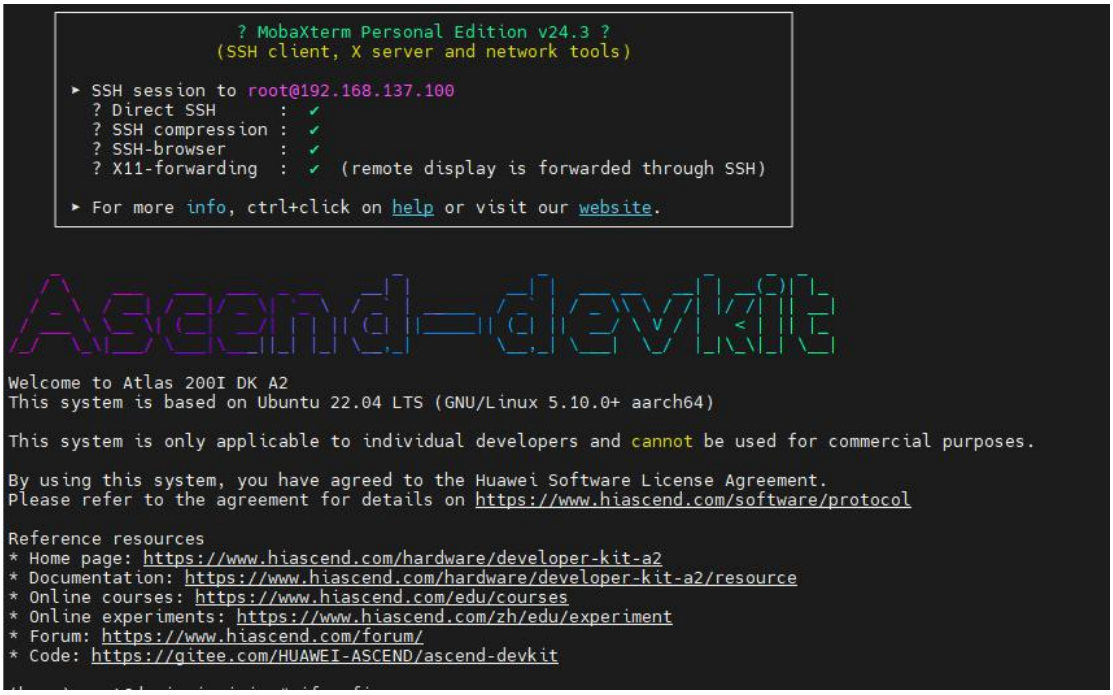

图 3-2 登陆界面示意图

3.2. 通过 USB TypeC 接口访问

该网口是固定静态 IP 地址:192.168.0.2,操作步骤保持一致，IP 改成 192.168.0.2 即可。

图 3-3 Typec 网口连接示意图

密码同样为 Mind@123。

3.3. 310B 联网说明

3.3.1. 共享网络

共享网络后，电脑的 IP 地址会固定为 192.168.137.1。

开启共享：

右键点击该网卡 -> 属性 -> 共享 选项卡。

勾选 “允许其他网络用户通过此计算机的 Internet 连接来连接”。

在“家庭网络连接”下拉菜单中，选择连接 310B 的那个虚拟网卡（通常显示为以太网 x）。

点击“确定”。此时系统通常会提示：“IP 地址将被设置为 192.168.137.1”。

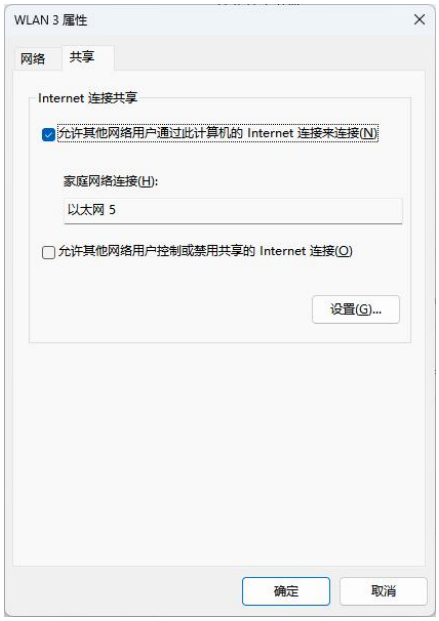

图 3-4 共享网络

笔者电脑是以太网 5。

设置后可以试试

ping www.baidu.com

```
(base) root@davinci-mini:/home# ping www.baidu.com
PING www.wshifen.com (103.235.46.102) 56(84) bytes of data.
64 bytes from 103.235.46.102 (103.235.46.102): icmp_seq=1 ttl=47 time=84.2 ms
64 bytes from 103.235.46.102 (103.235.46.102): icmp_seq=2 ttl=47 time=54.5 ms
```

图 3-5 ping 示意图

能出现回复即表示成功联网。

3.3.2. 连接路由器

将 ETH1(DHCP)的网口接路由器即可

如果使用静态 IP 的网口，要先查看路由器现在分配的 IP 地址是多少，然后修改静态 IP 地址保证和路由器的地址在同一网段。

比如:路由器分配的 IP 的网段是 192.168.3.xx,那开发板的静态 IP 就设为 192.168.3.50(同一网段即可，一般可以 2~254)。

验证步骤同 4.3.1。

4. HDMI 接口使用

将 HDMI0 接口连接 HDMI 显示屏即可看到桌面输出。

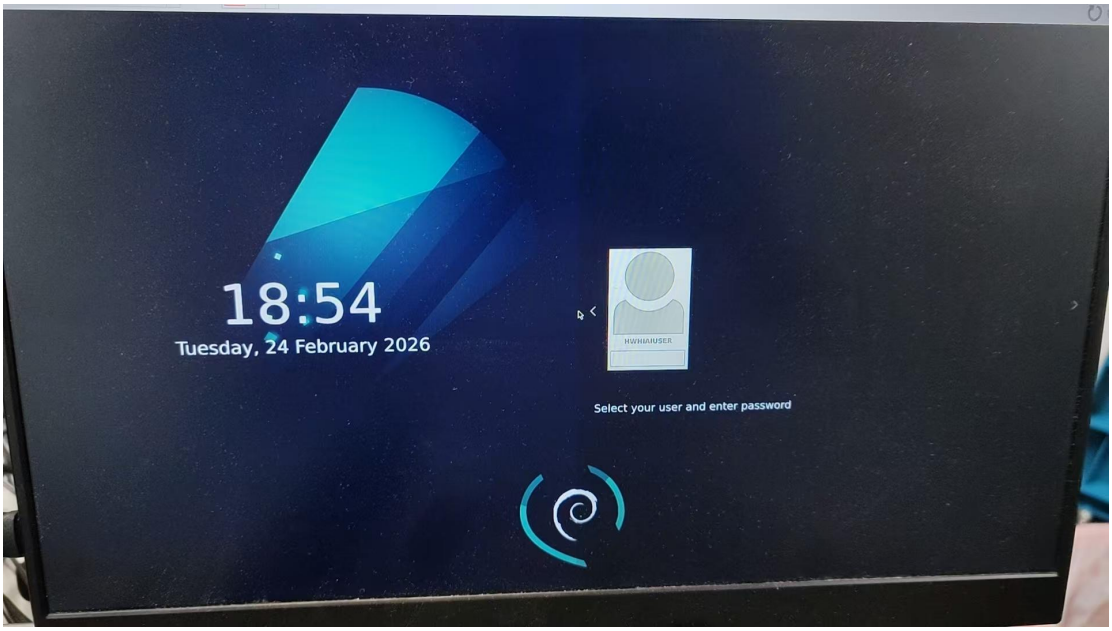

图 4-1 桌面输出显示

HDMI1 接口需要参考官方提供的 DEMO 来完成输出，需要重新编译内核、设备树等。参考如下:

https://gitee.com/ascend/samples/tree/master/cplusplus/level1_single_api/6_media/1_hdmi/hdmi_gitee

注意:当图形桌面系统关闭时，HDMI0 和 HDMI1 可以用于 NVR 二次开发。

5. USB 接口使用

板卡配备 2 个 USB HOST 接口，可以用来接 USB 键盘、鼠标、麦克风、摄像头、音响等。

5.1. 鼠标、键盘、拓展坞

任意 HOST 口均可以接入鼠标键盘和拓展坞，插入即可使用。

5.2. USB 摄像头显示

HOST 接口可以接入 1080p、720p 得 USB 摄像头。

提供一个 py 代码进行测试。

在资料包中:usb_camera_display

图 5-1 示意图

将里面的 main.py 代码上传到开发板上。

之后安装相关 python 依赖(注意要给 310B 联网):

sudo apt-get update

sudo apt-get install -y libgl1-mesa-glx libglib2.0-dev libgtk2.0-dev pkg-config

pip uninstall -y opencv-python-headless

pip install opencv-python

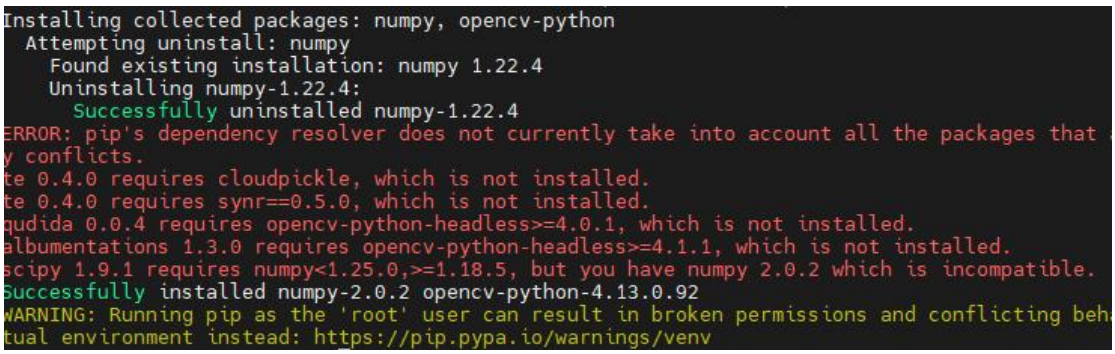

图 5-2 回显示意图

Error 可以忽略。出现和上述一样得回显即可。

之后运行

python main.py

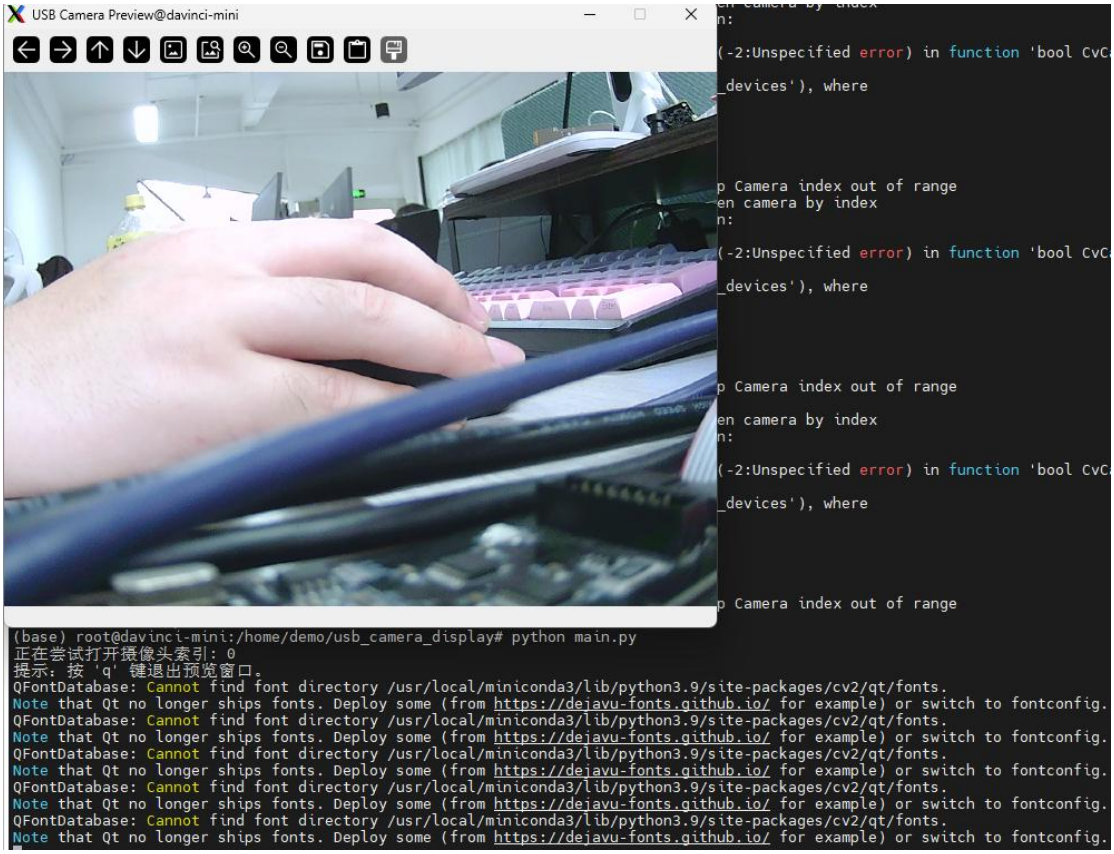

图 5-3 输出示意图

此时能看到摄像头画面即测试成功。

5.3. 音响、麦克风

5.3.1. USB 麦克风(录音设备)

仅 root 用户可以访问

如果普通用户要访问的话，要添加到对应用户组，以 HwHiAiUser 为例

usermod -a -G audio HwHiAiUser

执行以下命令查询设备 ID：

arecord –l

```
(base) root@davinci-mini:/home/demo/usb_camera_display# arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: Microphone [USB Microphone], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

图 5-4 回显示意图

当 card 为 i，device 为 j 时，ID 编号表示为“hw:i,j”，如回显所示内容，设备的 ID 为“hw:0,0”

执行以下命令进行录音，使用设备为“hw:0,0”，通道数为 1，采样率 16000，格式 S16_LE，录音文件名为 test.wav。

arecord -D hw:0,0 -c 1 -r 16000 -f S16_LE test.wav

```
(base) root@davinci-mini:/home/demo/usb_camera_display# arecord -D hw:0,0 -c 1 -r 16000 -f S16_LE test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
Warning: rate is not accurate (requested = 16000Hz, got = 44100Hz)
        please, try the plug plugin
^CAborted by signal Interrupt...
```

图 5-5 回显示意图

crtl+c 停止录音

通道数和采样率视设备而定，常见采样率包括 44100、48000、16000，常见通道数为 1、2.

5.3.2. USB 音响设备

执行以下命令查询设备 ID：

aplay -l

```
(base) root@davinci-mini:/home/demo/usb_camera_display# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: UACDemoV10 [UACDemoV1.0], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

图 5-6 示意图

当 card 为 i，device 为 j 时，ID 编号表示为“hw:i,j”，如回显所示内容，设备的 ID 为“hw:1,0

执行以下命令进行播放，使用设备为“hw:0,0”，录音文件名为 test.wav。

aplay -D plughw:0,0 test.wav

```
(base) root@davinci-mini:/home/demo/usb_camera_display# aplay -D plughw:0,0 test.wav
Playing WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Mono
```

图 5-7 回显示意图

ctrl+c 停止播放

6. 体验 AI 用例

系统内置 jupyter，其中包含许多用 python 实现的 AI 用例，具体使用方法参考官方文档说明。

为方便新手开发者进行应用开发和程序运行，镜像中已包含 jupyter lab 软件（可视化代码演示、数据分析工具），可为用户提供一个图形化运行推理样例的界面，本节介绍基于远程登录模式如何登录 jupyter lab 并运行一个 ResNet50 样例。

6.1. 本机显示模式

进入“notebooks”目录，在命令终端窗口输入如下命令（可以拷贝）： cd /home/HwHiAiUser/samples/notebooks

图 6-1 示意图

修改 jupyter lab 启动脚本（start_notebook.sh）中 jupyter lab 启动 IP 地址，执行 vi start_notebook.sh 命令修改启动 IP 地址。在键盘按 i 键进入编辑模式，将脚本中以下加粗的 IP 地址修改为 127.0.0.1。

```
. /usr/local/Ascend/ascend-toolkit/set_env.sh
export PYTHONPATH=/usr/local/Ascend/thirdpart/aarch64/acllite:$PYTHONPATH
if [ $# -eq 1 ];then
    jupyter lab --ip $1 --allow-root --no-browser
else
    jupyter lab --ip 127.0.0.1 --allow-root --no-browser
fi
```

图 6-2 修改示意图

修改完成后，按 Esc 键退出编辑模式，按键盘输入:wq 保存文件。

如果本机显示模式下，如果开发者套件同时有接口连接 PC，可以将 jupyter lab 启动 IP 地址修改为开发者套件连接 PC 的接口 IP 地址，用户可以实现同时通过本机显示模式和远程登录模式登录开发者套件。

接下来执行./start_notebook.sh 命令启动 jupyter lab。

```
[I 2023-05-09 08:02:59.683 ServerApp] nbclassic | extension was successfully loaded.
[I 2023-05-09 08:02:59.685 ServerApp] Serving notebooks from local directory: /home/HwHiAiUser/samples/notebooks
[I 2023-05-09 08:02:59.685 ServerApp] Jupyter Server 1.23.6 is running at:
[I 2023-05-09 08:02:59.685 ServerApp] http://127.0.0.1:8888/lab?token=a046a76dc21f1504f271c16278ed62ed7fb014aaf38ee807
[I 2023-05-09 08:02:59.685 ServerApp]  or http://127.0.0.1:8888/lab?token=a046a76dc21f1504f271c16278ed62ed7fb014aaf38ee807
[I 2023-05-09 08:02:59.685 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 2023-05-09 08:02:59.701 ServerApp]
```

图 6-3 回显示意图

如果出现上述回显，则表示启动正常。

如果出现失败，请查阅下面链接，请查阅常见报错解决。

在回显信息页面按住键盘“Ctrl”键并使用鼠标左键单击步骤 4 回显中加粗的网址链接如：http://127.0.0.1:8888/lab?token=a046a76dc21f1504f271c16278ed62ed7fb014aaf38ee807），进入 jupyter lab 界面，即可运行开发者套件预置的 Python 推理样例。

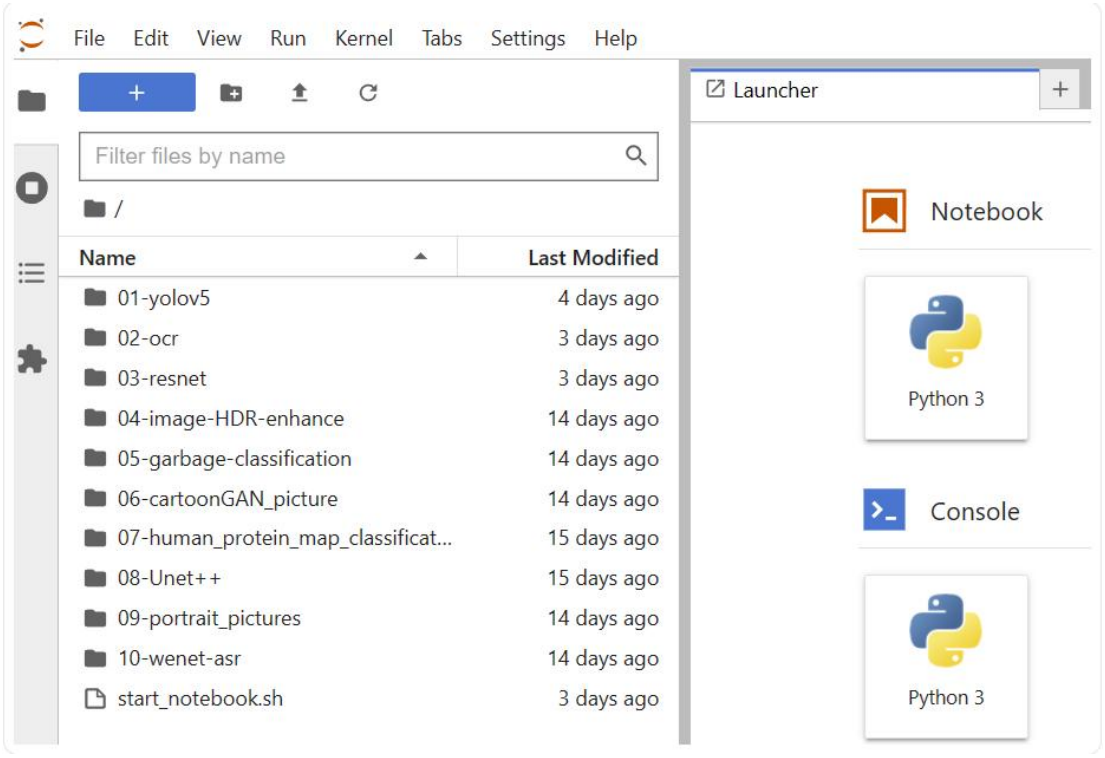

图 6-4 样例示意图

接下来运行 yolov5 实现目标检测。

在 jupyter lab 可视化界面目录中双击“01-yolov5”模型样例目录。

双击打开“main.ipynb”脚本，进入样例运行页面。

图 6-5 示意图

当前目标检测样例支持图片、视频、USB 摄像头目标检测，用户可修改代码块中 infer_mode，默认值：video，可选值：image、video、camera。

```
infer_mode = 'video'

if infer_mode == 'image':
    img_path = 'world_cup.jpg'
    infer_image(img_path, model, labels_dict, cfg)
elif infer_mode == 'camera':
    infer_camera(model, labels_dict, cfg)
elif infer_mode == 'video':
    video_path = 'racing.mp4'
    infer_video(video_path, model, labels_dict, cfg, output_path='output.mp4')
```

图 6-6 示意图

单击 ⏩ 按钮运行样例，单击“Restart”按钮，运行完成后等待若干秒时间，查看推理结果。

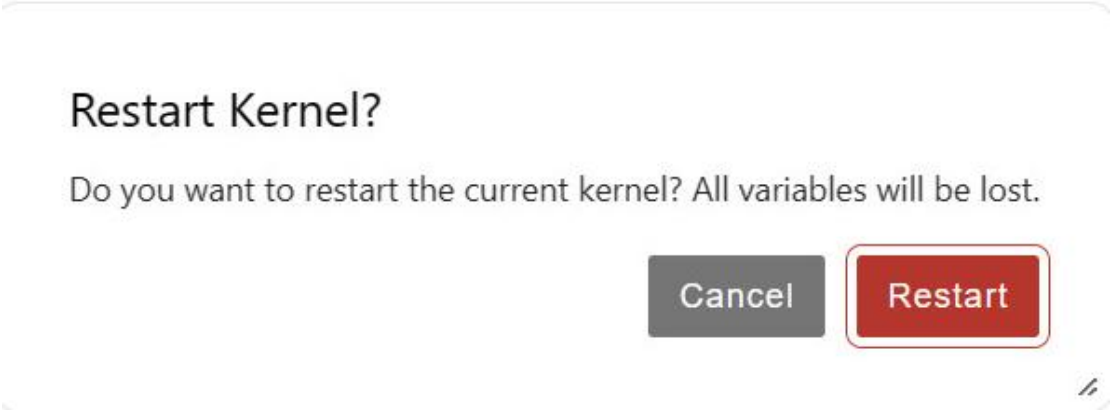

图 6-7 示意图

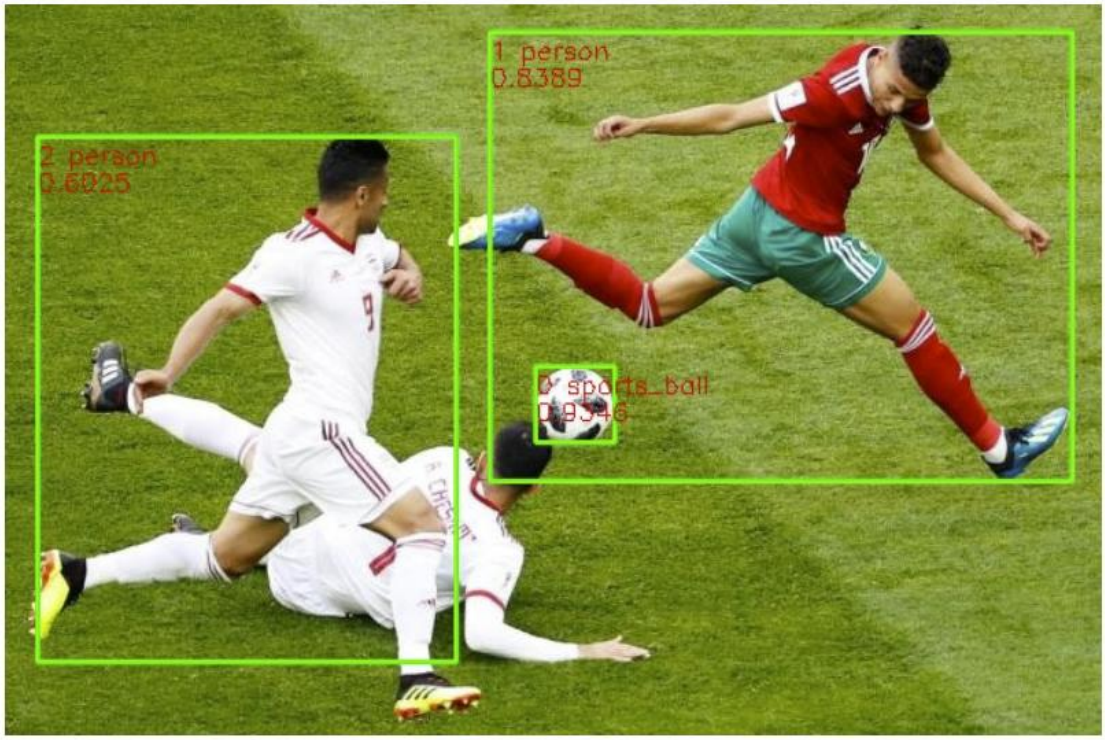

图 6-8 图像检测结果

6.2. 远程登录模式

修改 jupyter lab 启动脚本（start_notebook.sh）中 jupyter lab 启动 IP 地址，执行 vi start_notebook.sh 命令修改启动 IP 地址。在键盘按 i 键进入编辑模式，将脚本中以下加粗的 IP 地址修改为 192.168.137.100。该 IP 是实际以太网接口的 IP，笔者这里是 192.168.137.100，请根据实际进行修改。

```
. /usr/local/Ascend/ascend-toolkit/set_env.sh
export PYTHONPATH=/usr/local/Ascend/thirdpart/aarch64/acllite:$PYTHONPATH
if [ $# -eq 1 ];then
    jupyter lab --ip $1 --allow-root --no-browser
else
    jupyter lab --ip 192.168.137.100 --allow-root --no-browser
fi
```

图 6-9 示意图

其余步骤与本机显示一致。

6.3. USB 摄像头 YOLOV5 检测

资料包: usb_camera_yolov5_display 将该文件夹上传到 310B 上。

/home/demo/usb_camera_yolov5_display/

名称	大小 (KB)	最后修改	所有者	群组
..				
coco_names.txt	1	2022-04-08...	root	root
det_utils.py	8	2022-04-08...	root	root
main.py	5	2022-04-08...	root	root
yolo.om	15 299	2022-04-08...	root	root
yolov5s.onnx	28 650	2022-04-08...	root	root

将该文件夹上传到 310B 上。首先确保已经跑通案例:USB 摄像头显示，如果没跑过请先查 6.2 章节跑通基础的 USB 摄像头显示。

之后输入命令降级 numpy

pip install numpy==1.24.4 --install

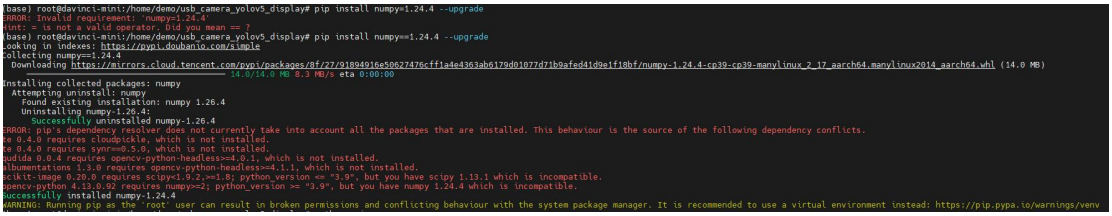

图 6-10 回显示意图

能出现上述 Successfully 的回显即表示成功。

python main.py

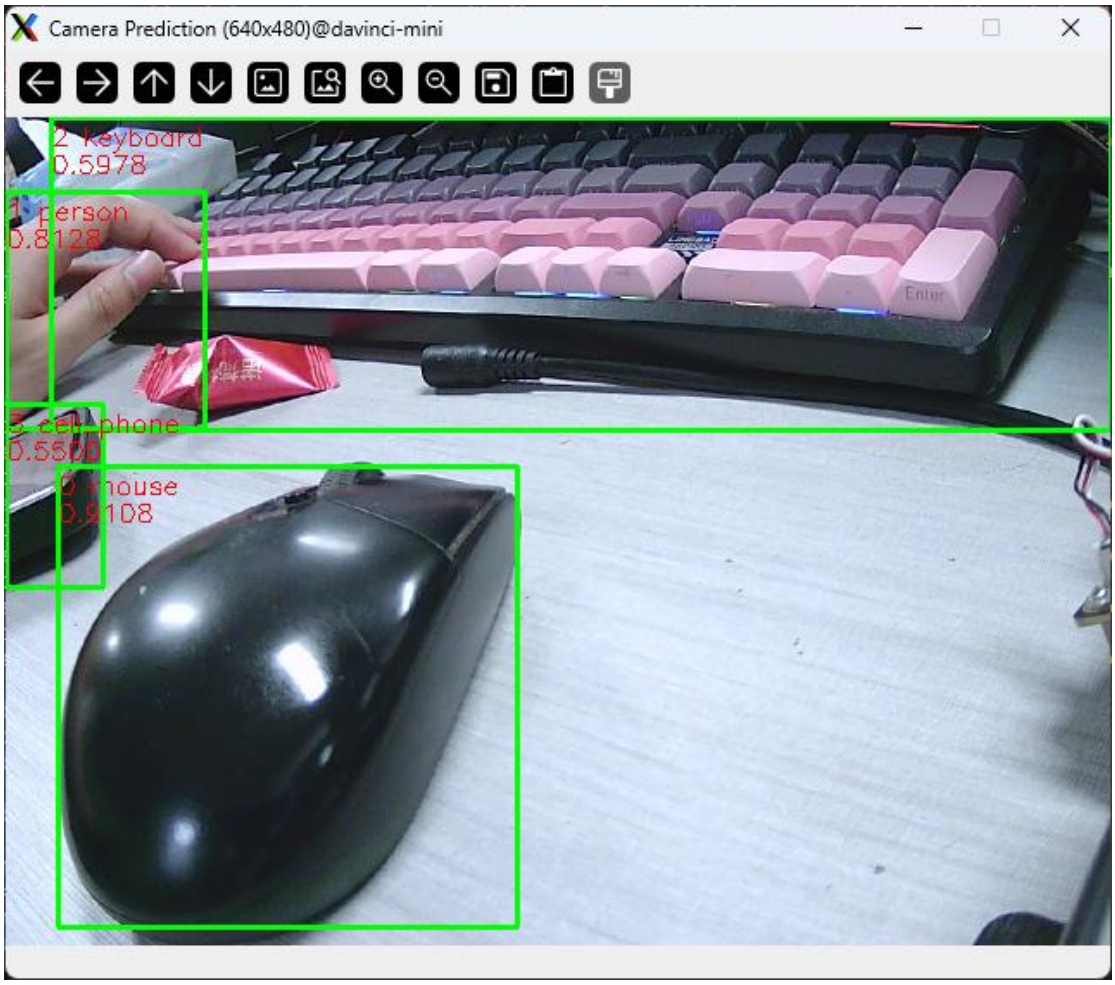

图 6-11 示意图

能显示出图像表示运行成功。

SDK 源码包使用

资料包提供 Ascend310B-source.tar.gz，即昇腾 310B 的完整 SDK 源码包，当要二次开发的时候，用户需要用到该 SDK 源码包。

需要将该源码包上传至 Ubuntu 电脑中，以下用 22.04 做说明。

同时资料包也提供交叉编译工具链:toolchain.tar.gz。同样要上传到 Ubuntu 电脑中

6.4. 配置交叉编译工具链

执行如下命令，进入到“/opt/compiler”目录。

首先将 toolchain.tar.gz 上传到服务器中

之后执行：

tar -xvf toolchain.tar.gz -C ./

解压交叉编译工具。

根据当前目录，执行 export，导出环境变量。

export PATH=/home/wb/200k_310b/Ascend310B-sdk/toolchain/bin:$PATH

注意，这是笔者的路径，请根据你实际的路径进行修改。

执行如下命令，查看交叉编译工具链版本。

aarch64-target-linux-gnu-gcc -v

显示有版本信息，则表明安装工具链成功。具体版本号请以实际情况为准。

gcc version 7.3.0 (Do-Compiler V100R003C01B001)

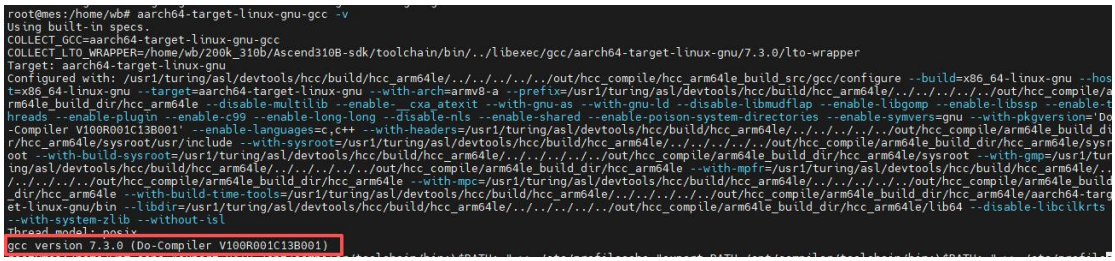

图 0-1 示意图

该方法只对当前终端生效，一旦关闭当前终端或者切换别的终端，该配置失效。

6.5. 编译内核

先将 Ascend310B-source.tar.gz 上传到 Ubuntu 电脑上，然后解压。

tar -xzvf Ascend310B-source.tar.gz

同时要先安装依赖:

apt-get install -y python3 make gcc unzip pigz bison flex libncurses-dev squashfs-tools bc device-tree-compiler libssl-dev cmake rpm2cpio

做好这些工作后，cd Ascend310B-source，进入源码目录。

注意一定要先导出交叉编译工具链配置，并再编译源码。

此时执行:

bash build.sh kernel

编译过程中会弹出内核配置界面

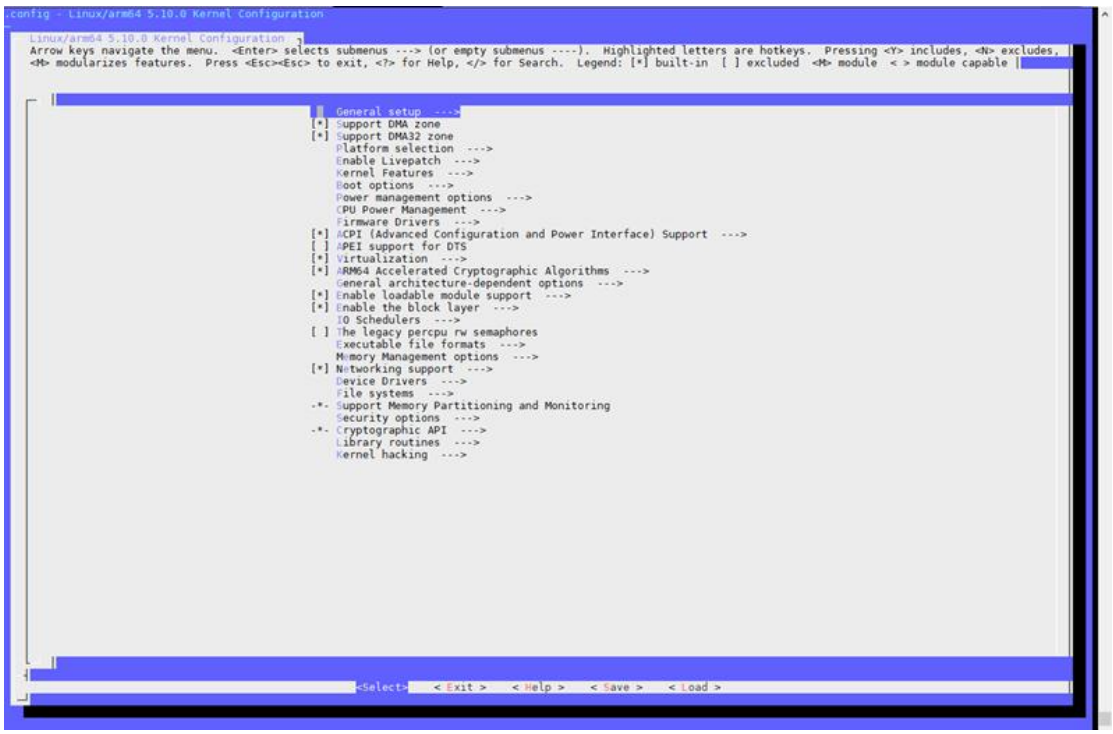

图 0-2 内核配置图

默认不改即可。

等待一段时间出现下面的回显，表示内核编译成功。

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

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

6.6. 编译生效内核 Image

编译步骤等同 8.2 编译内核。

Ascend310B-source/output/Image

将该 Image 文件上传到 310B 开发板中。

之后执行 dd 命令，更新 Image

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

6.7. 编译生效内核 DTB 文件

开发板使用的 DTS 文件如下所示:

/Ascend310B-sdk/Ascend310B-source/dtb/dts/hi1910b/hi1910BL/

hi1910B-default.dts

用户要修改可以修改上述 dts 文件

然后

bash build.sh dtb

此时会生成 dt.img，该文件存放在 output 文件夹里，将该文件上传到开发板。

执行命令:

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

6.8. 交叉编译应用程序

当编译应用程序的时候，往往在 310B 上编译速度会很慢，所以需要用 x86 电脑来交叉编译应用程序，从而提高效率。

这里以输出 hello world 作为例子。需要编写 Makefile 以及 C 代码程序。

```
# 定义交叉编译器路径（使用你提供的路径）
CC = /home/wb/200k_310b/Ascend310B-sdk/toolchain/bin/aarch64-target-linux-gnu-gcc

# 编译选项
# -Wall: 显示所有警告
# -O2: 优化代码
CFLAGS = -Wall -O2

# 目标文件名
TARGET = hello_ascend

# 默认构建目标
all: $(TARGET)

# 链接规则
$(TARGET): hello.o
  $(CC) $(CFLAGS) -o $(TARGET) hello.o
  @echo "Build complete! Use 'file $(TARGET)' to check the architecture."

# 编译规则
hello.o: hello.c
  $(CC) $(CFLAGS) -c hello.c -o hello.o

# 清理规则
clean:
  rm -f *.o $(TARGET)

.PHONY: all clean
```

图 0-3 Makefile 示意

注意配置交叉编译工具链的路径，要和自己的路径对应。

```
#include <stdio.h>

/**
 * Ascend 310B Cross-Compilation Example
 * 这是一个简单的 C 语言程序，用于验证交叉编译工具链是否配置成功。
 */

int main(int argc, char *argv[]) {
    printf("============================================\n");
    printf("  Hello, Ascend 310B!\n");
    printf("  This program was cross-compiled on X86.\n");
    printf("  Target Architecture: AArch64 (ARMv8)\n");
    printf("============================================\n");

    return 0;
}
```

图 0-4 hello.c 示意

这是 hello.c 的代码，整体非常简单，用来验证交叉编译。接下来 CD 到例程目录，然后执行 make

```
root@mes:/home/wb/200k_310b/hello_world# make
/home/wb/200k_310b/Ascend310B-sdk/toolchain/bin/aarch64-target-linux-gnu-gcc -Wall -O2 -c hello.c -o hello.o
/home/wb/200k_310b/Ascend310B-sdk/toolchain/bin/aarch64-target-linux-gnu-gcc -Wall -O2 -o hello_ascend hello.o
Build complete! Use 'file hello_ascend' to check the architecture.
```

图 0-5 make

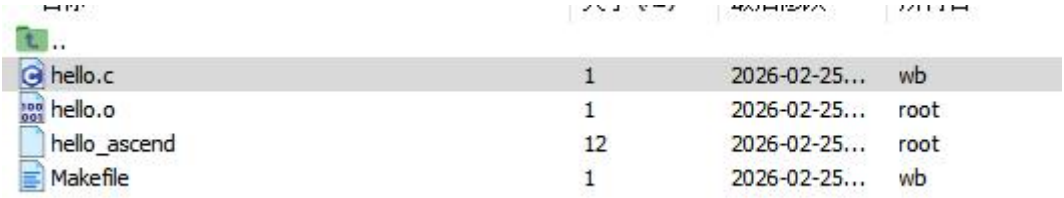

图 0-6 示意图

如果 make 正常，可以看到上述 hello.o 和 hello_ascend 文件。其中 hello_ascend 为可执行文件，此时将该文件通过 ssh 等方法传到板卡中。

图 0-7 示意图

笔者将该文件上传到 310B 的 home 文件夹下，同时执行:

chmod -R 777 hello_ascend 赋予权限

接下来

./hello_ascend

图 0-8 输出示意图

如果能出现上述打印信息，表示交叉编译正常。

7. AI 工具链配置

7.1. 安装 toolkit 工具

资料包提供了:Ascend-cann-toolkit_8.0.1_linux-x86_64.run

将该工具链上传到 Ubuntu 电脑后执行:

chmod +x Ascend-cann-toolkit_8.0.1_linux-x86_64.run

./Ascend-cann-toolkit_8.0.1_linux-x86_64.run --install

如果用户未指定安装路径，则软件会安装到默认路径下，默认安装路径如下。root 用户：“/usr/local/Ascend”，非 root 用户：“${HOME}/Ascend”，${HOME}为当前用户目录。

安装成功后，配置环境变量。

source ${HOME}/Ascend/ascend-toolkit/set_env.sh

```
# 安装toolkit包时配置
source ${HOME}/Ascend/ascend-toolkit/set_env.sh
# 其中<arch>请替换为实际架构
export LD_LIBRARY_PATH=${HOME}/Ascend/ascend-toolkit/latest/<arch>-linux/devlib/:$LD_LIBRARY_PATH
```

图 7-1 示意图

cd ${HOME}/Ascend/ascend-toolkit/latest/<arch>-linux

cat ascend_toolkit_install.info

查看相关版本信息。

为了后续板子上也能正常适配环境，需要在板子上同步运行:

source /usr/local/Ascend/ascend-toolkit/set_env.sh

export LD_LIBRARY_PATH=/lib64:$LD_LIBRARY_PATH

注意 export 是临时导出环境变量，一旦切换终端或者重启，该环境变量消失。

8. 官方参考

官方 Gitee 仓库：

https://gitee.com/ascend/samples