

从零开始，快速上手Duo 开发板攻略

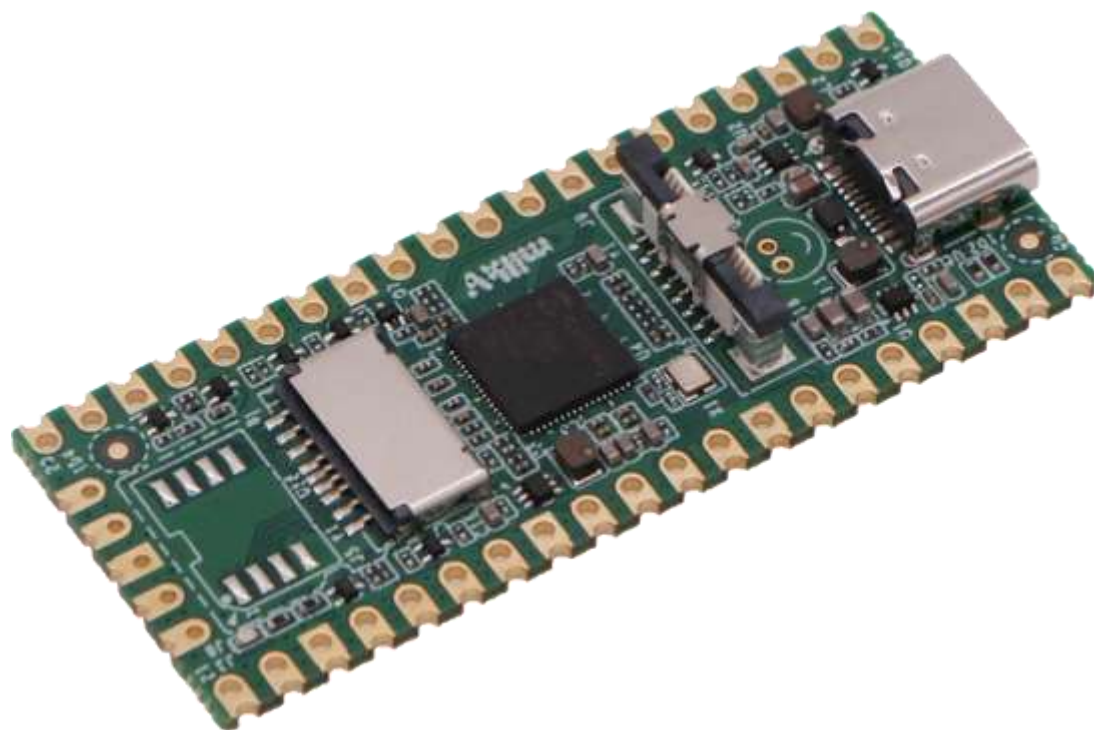
汇报人：张馥媛

汇报时间：2023.12.15

目录

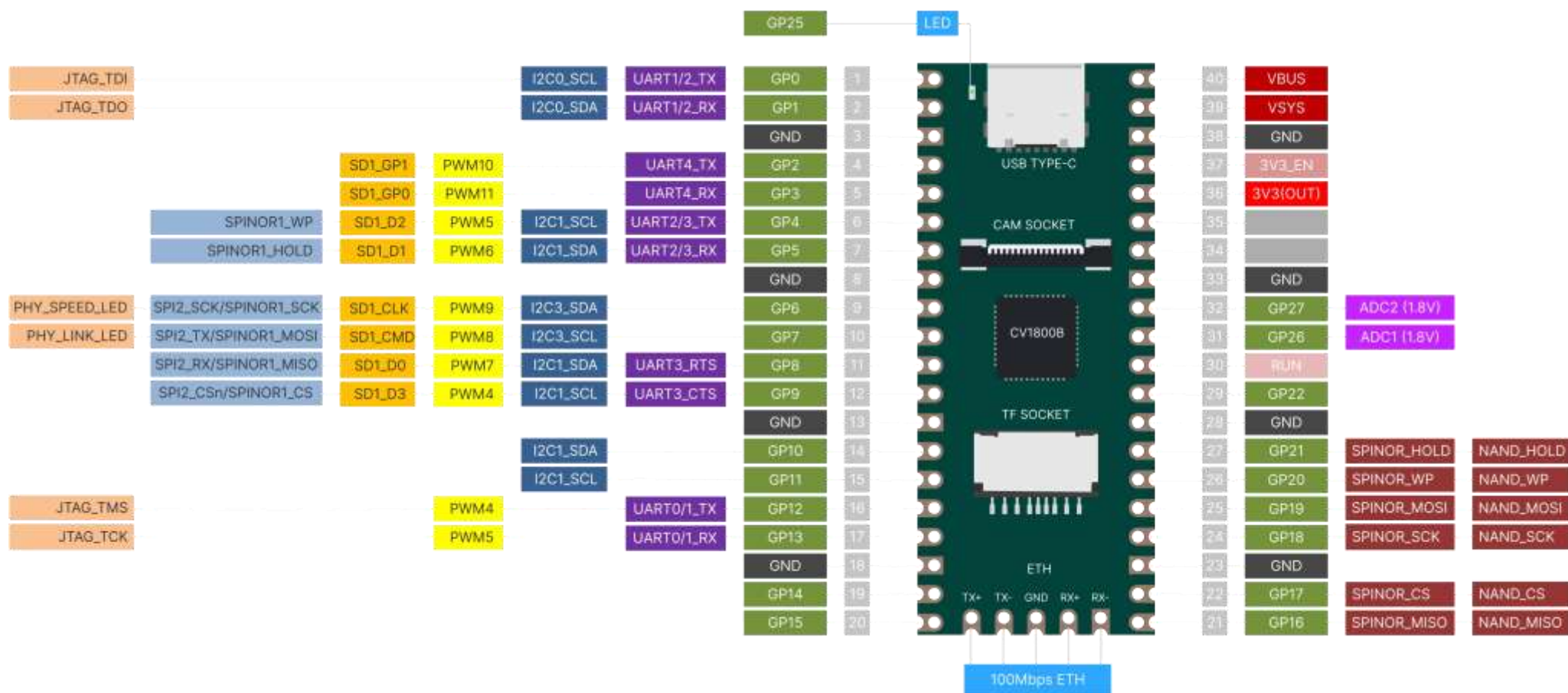
- Duo简介
- 登录Duo
- TPU
- 摄像头

Duo简介



Milk-V Duo	规格
处理器	CVITEK CV1800B (C906@1Ghz + C906@700MHz)
内存	DDR2 64MB
Storage	1x Mirco SD slot,1x SD NAND solder pad
USB	1x Type-C for data and Power,1x USB2 solder pad
摄像	1x 16P FPC connector (MIPI CSI 2-lane)
芯片	up to 26 Pins available for general purpose I/O (GPIO)
尺寸	21mm*51mm

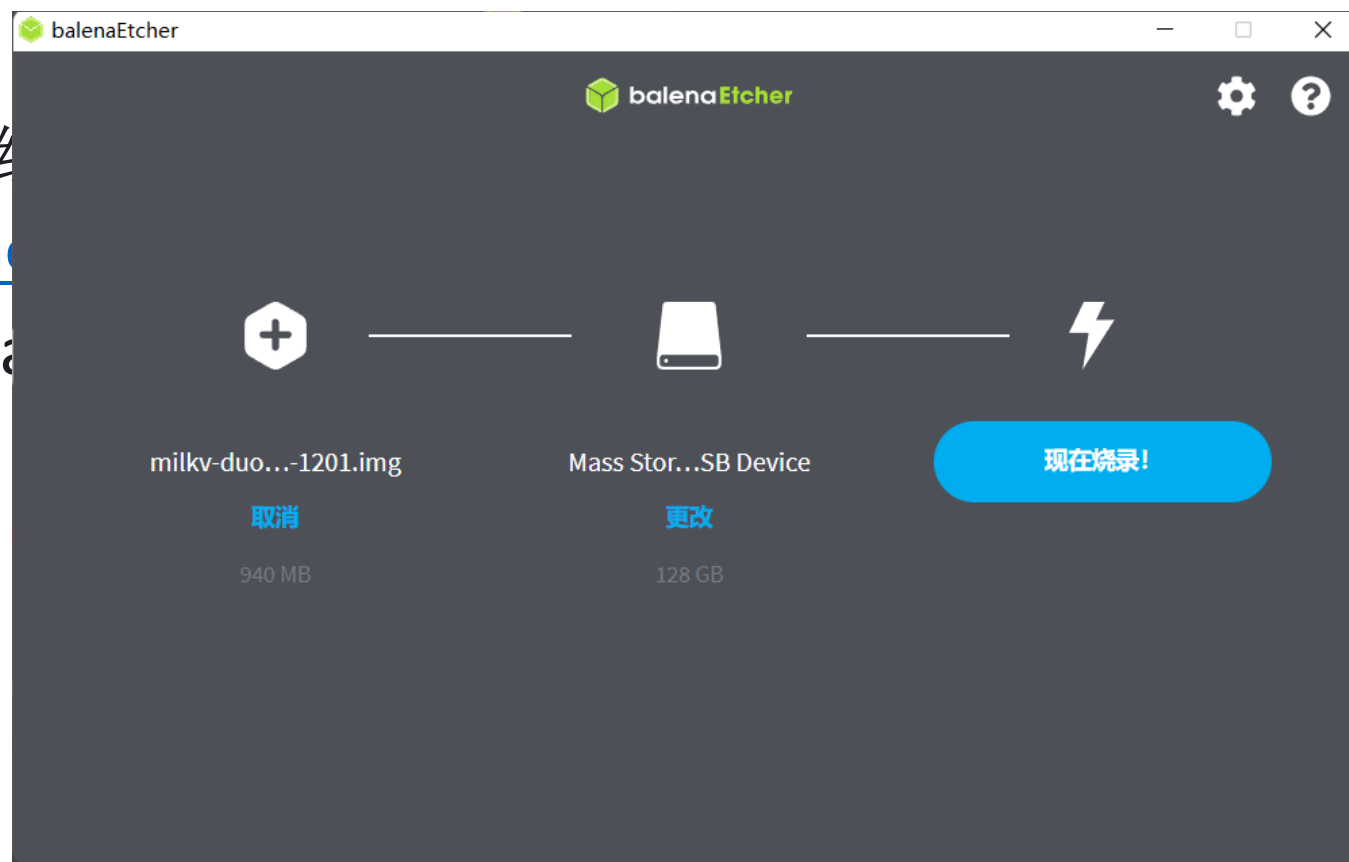
Duo简介



登录Duo

准备：Duo+microSD+Type-C

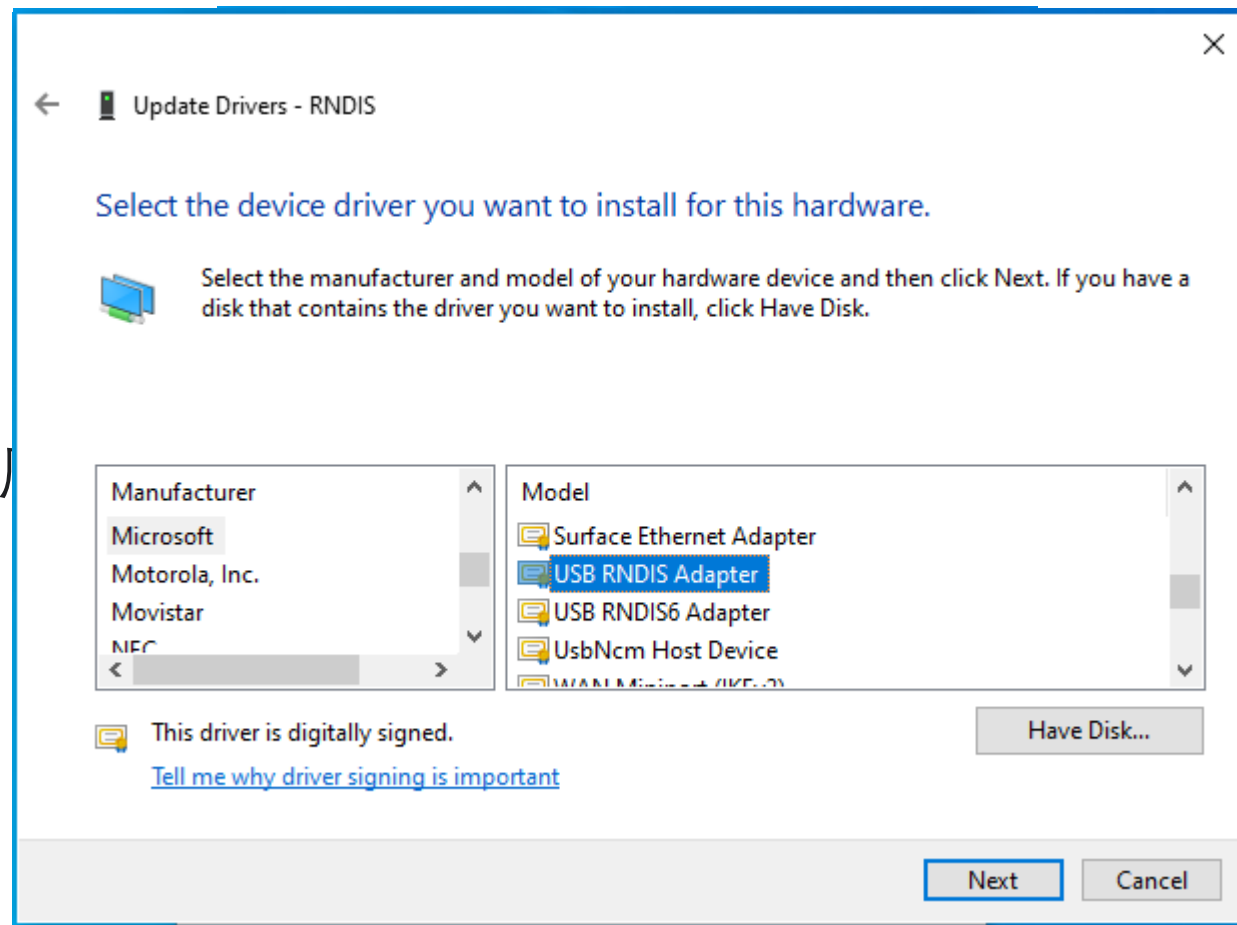
- 1. 下载镜像和工具
 - 从[官方镜像和SDK](#)下载系统
 - 下载Flash工具 [balenaEtcher](#)
- 2. 烧录镜像（以balena



登录Duo

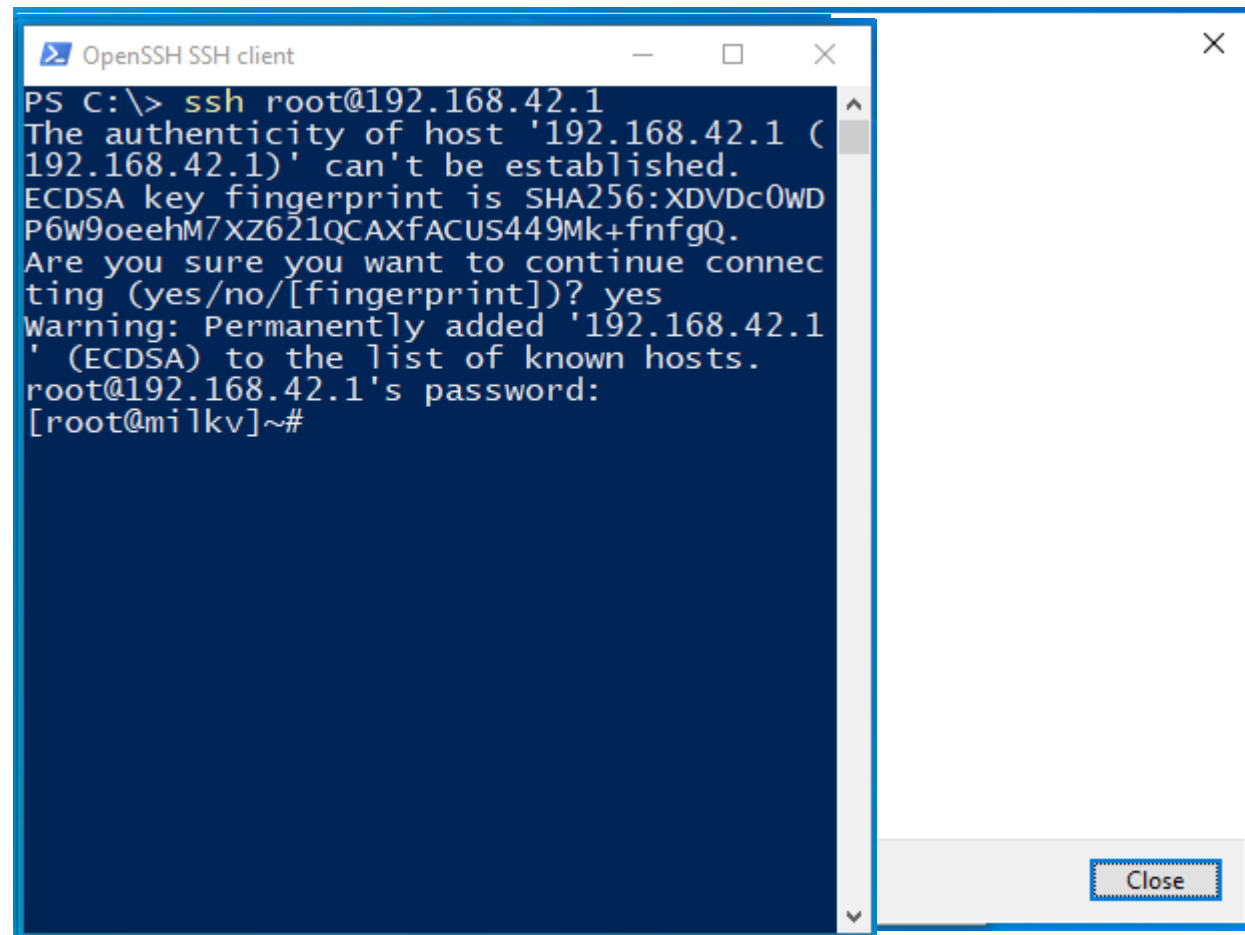
• 3. USBnet 设置

- 通过Type-C线将Duo与电脑连接
- “RNDIS” 设备出现在设备管理器中
- 选择 "RNDIS "并右键单击以更新驱动程序
- 选择 "Browse my computer for drivers“
- 选择 "Let me pick from a list of available drivers on my computer“
- 选择 "Network adapters“
- Manufacturer/Model: Microsoft/USB RNDIS Adapter



登录Duo

- 忽略警告信息
- 驱动程序更新成功
- 检查 "USB RNDIS Adapter"
- 找到IP并使用ping来测试网络
- 4.SSH
 - 打开终端，输入 `ssh root@192.168.42.1`, 并回答是
 - 输入密码 `milkv`
 - 登录成功



```
OpenSSH SSH client
PS C:\> ssh root@192.168.42.1
The authenticity of host '192.168.42.1 (192.168.42.1)' can't be established.
ECDSA key fingerprint is SHA256:XDVDc0WD P6w9oeehM7XZ621QCAXfACUS449Mk+fnfgQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.42.1' (ECDSA) to the list of known hosts.
root@192.168.42.1's password:
[root@milkv]~#
```

TPU

- **准备： Duo+microSD+Type-C**

Duo 的 CPU CV1800B 集成了 CVITEK TPU，用于智能检测

TPU 是深度学习神经网络的 AI 加速引擎，可用于加速图像分类、物体检测、人脸检测与识别、分割、LSTM 等。TPU 的主要功能是分担 CPU 工作，加速计算机视觉和语音相关操作

TPU

- 基于 YOLOv5 的目标

- 1.配置 Docker 开发环

- 下载 Docker Desktop for

- 控制面板 —— 程序和功

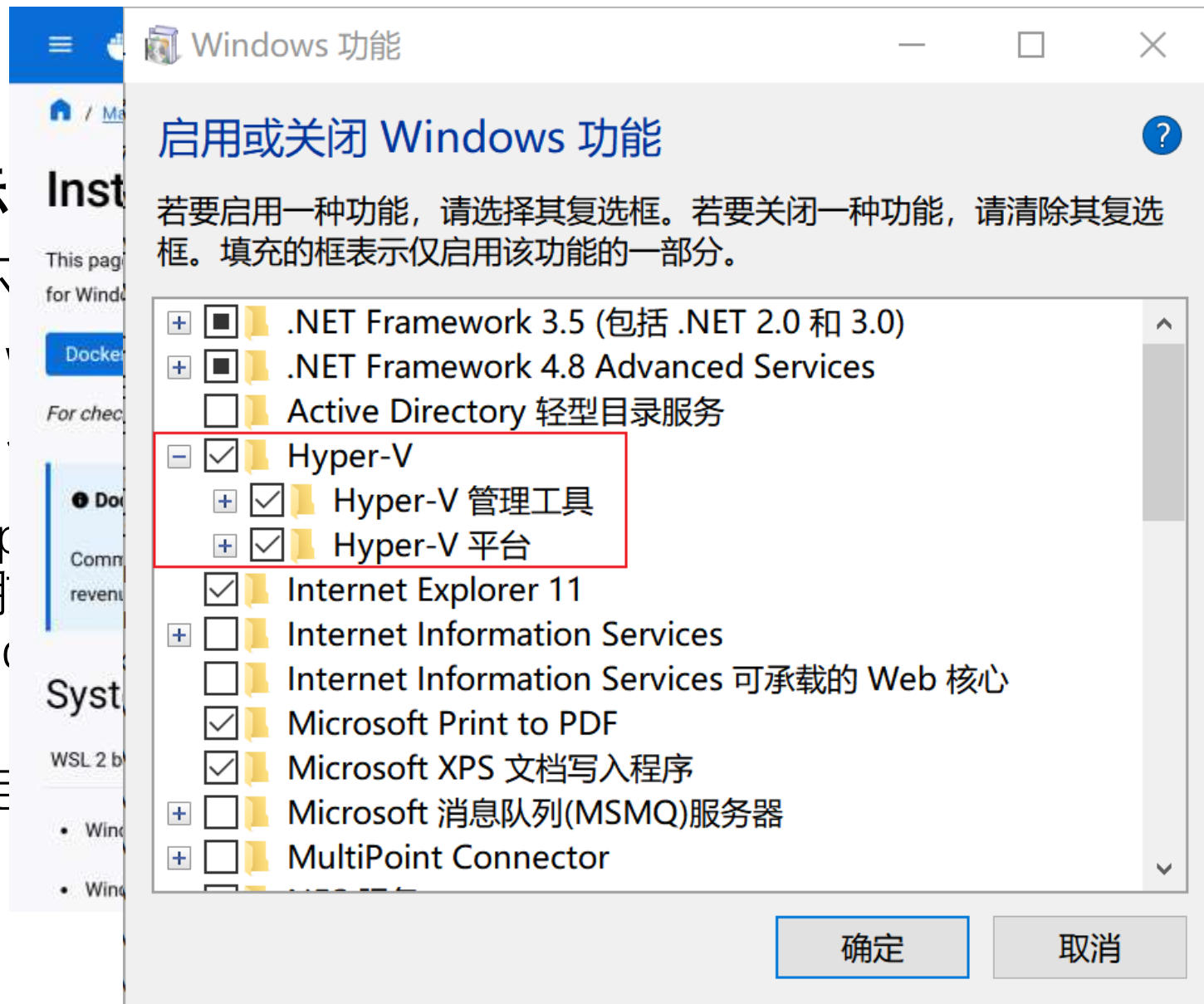
- 找到 Hyper-V，勾选 Hyper

- 统文件配置完成后重启电脑

- 然后即可安装下载好 Docker

- 的后端进行相应的勾选

- 安装完成后，需要重启电



TPU

-拉取开发所需 Docker 镜像

从 Docker hub 获取镜像文件

```
docker pull sophgo/tpuc_dev:v3.1
```

-启动 Docker 容器

在 Windows 终端中执行

```
docker run --privileged --name <container_name> -v /workspace -it sophgo/tpuc_dev:v3.1
```

其中, `<container_name>` 为自己定义的容器名, 比如叫 DuoTPU

```
docker run --privileged --name DuoTPU -v /workspace -it sophgo/tpuc_dev:v3.1
```

TPU

-获取开发工具包并添加环境变量

在 Docker 终端中下载 TPU-MLIR 模型转换工具包

```
git clone https://github.com/milkv-duo/tpu-mlir.git
```

在 Docker 终端中, 用 `source` 命令添加环境变量 In the Docker terminal, use the `source` command to add environment variables

```
# source ./tpu-mlir/envsetup.sh
```

TPU

- 2.准备原始模型文件

[-基于 YOLOv5 的目标检测 | Milk-V \(milkv.io\)](#)

- 3.在 Docker 中准备工作目录

创建并进入 `yolov5n_torch` 工作目录, 注意是与 `tpu-mlir` 同级的目录, 并将模型文件和图片文件都放入该目录下

```
# mkdir yolov5n_torch && cd yolov5n_torch
```

新建一个 Windows 终端, 将 `yolov5n_jit.pt` 从 windows 拷贝到 Docker 中

```
docker cp <path>/yolov5-master/yolov5n_jit.pt <container_name>:/workspace/yolov5n_torch/yolov5n_jit.pt
```

TPU

其中, `<path>` 为 windows 系统中 yolov5 开发工具包所在的文件目录, `<container_name>` 为容器名, 比如

```
docker cp C:\Users\Carbon\Duo-TPU\yolov5-master\yolov5n_jit.pt DuoTPU:/workspace/yolov5n_torch/yolov5n_jit.pt
```

再回到 Docker 终端下, 将图片文件放入当前目录(`yolov5n_torch`)下

```
# cp -rf ${TPUC_ROOT}/regression/dataset/COCO2017 .  
# cp -rf ${TPUC_ROOT}/regression/image .
```

这里的 `${TPUC_ROOT}` 是环境变量, 对应 `tpu-mlir` 目录, 是在前面配置 Docker 开发环境中 `source ./tpu-mlir/envsetup.sh` 这一步加载的

创建并进入 `work` 工作目录, 用于存放编译生成的 `MLIR`、`cvimodel` 等文件

```
# mkdir work && cd work
```

TPU

- 4. YOLOv5n-TORCH 模型转换
-TORCH 模型转换成 MLIR

```
# model_transform.py \  
--model_name yolov5n \  
--model_def ../yolov5n_jit.pt \  
--input_shapes [[1,3,640,640]] \  
--pixel_format "rgb" \  
--keep_aspect_ratio \  
--mean 0,0,0 \  
--scale 0.0039216,0.0039216,0.0039216 \  
--test_input ../image/dog.jpg \  
--test_result yolov5n_top_outputs.npz \  
--output_names 1219,1234,1249 \  
--mlir yolov5n.mlir
```

TPU

-MLIR 转 INT8 模型

在转 INT8 模型之前需要先生成校准表，这里用现有的 100 张来自 COCO2017 的图片举例，执行 calibration

```
# run_calibration.py yolov5n.mlir \  
  --dataset ../COCO2017 \  
  --input_num 100 \  
  -o ./yolov5n_cali_table
```

TPU

-MLIR 量化成 INT8 非对称 cvimodel

```
# model_deploy.py \  
--mlir yolov5n.mlir \  
--quantize INT8 \  
--calibration_table ./yolov5n_cali_table \  
--chip cv180x \  
--test_input ../image/dog.jpg \  
--test_reference yolov5n_top_outputs.npz \  
--compare_all \  
--tolerance 0.96,0.72 \  
--fuse_preprocess \  
--debug \  
--model yolov5n_int8_fuse.cvimodel
```


TPU

- 5. 在 Duo 开发板上进行验证

-获取 tpu-sdk

在 Docker 终端下切换到 `/workspace` 目录

```
cd /workspace
```

下载 tpu-sdk

```
git clone https://github.com/milkv-duo/tpu-sdk.git
```

TPU


-将开发工具包和模型文件拷贝到 Duo 开发板上

在 duo 开发板的终端中，新建文件目录 `/mnt/tpu/`

```
# mkdir -p /mnt/tpu && cd /mnt/tpu
```

在 Docker 的终端中，将 `tpu-sdk` 和模型文件拷贝到 Duo 开发板上

```
# scp -r /workspace/tpu-sdk root@192.168.42.1:/mnt/tpu/  
# scp /workspace/yolov5n_torch/work/yolov5n_int8_fuse.cvimodel root@192.168.42.1:/mnt/tpu/tpu-sdk
```



TPU

-设置环境变量

在 Duo 开发板的终

```
# cd /mnt/tpu/  
# source ./env
```

-进行目标检



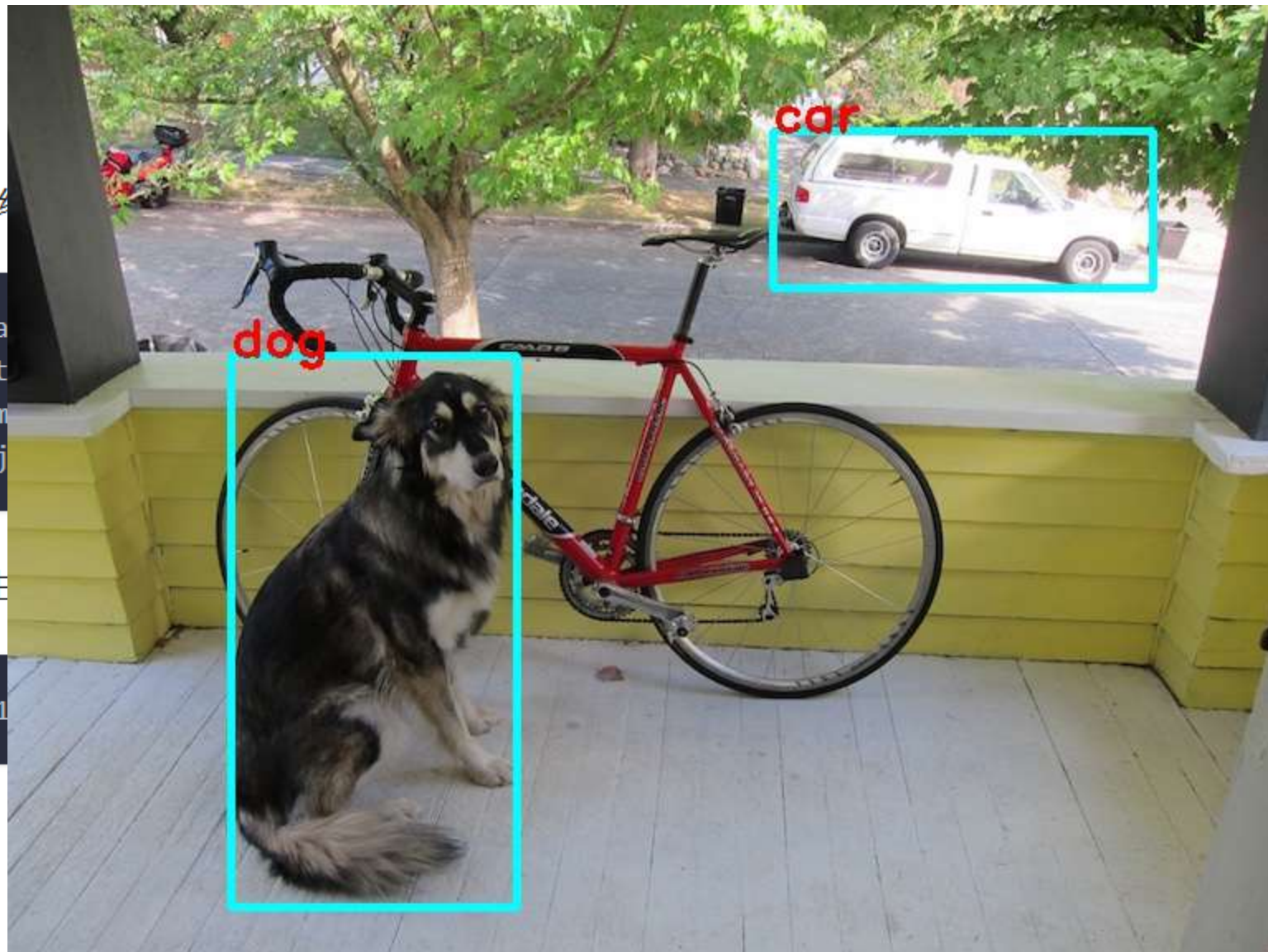
TPU

在 Duo 开发板的终

```
# ./samples/sa  
./yolov5n_int  
./samples/sam  
yolov5n_out.j
```

运行成功后，会生

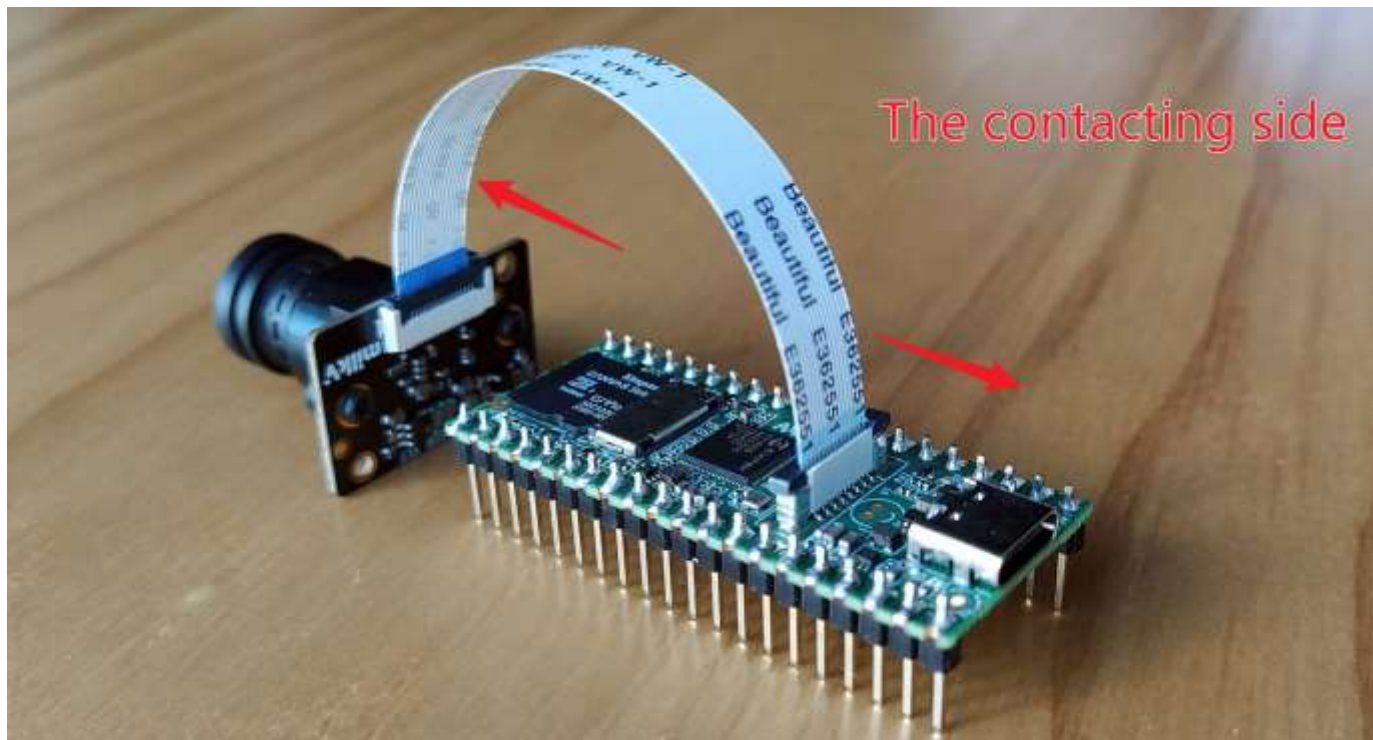
```
scp root@192.1
```



摄像头

准备： Duo+microSD+Type-C+CAM-GC2083

- CAM-GC2083 配备了 格科微 的 GC2083 CMOS 图像传感器，分辨率高达 200 万像素。它与 Milk-V Duo 板上的 16P MIPI CSI 接口兼容。



摄像头

- 显示摄像头推流的画面

文末链接下载测试文件包 `duo_camera_test_v0.2.tar.gz` 并通过scp或其他方式传到Duo上

```
scp duo_camera_test_v0.2.tar.gz root@192.168.42.1:/root/
```

登陆到Duo终端

```
ssh root@192.168.42.1
```

解压测试包

```
tar xzf duo_camera_test_v0.2.tar.gz -C /
```

摄像头

进入测试程序目录

```
cd /mnt/data/install/
```

执行测试程序推流

```
./CviIspTool.sh 64M
```

正常情况下，终端最后会看到如下日志

```
VPSS init with src (1920, 1080) dst (1920, 1080).  
CVI_VPSS_CreateGrp:0, s32Ret=0  
rtsp://127.0.1.1:8554/stream0  
prio:0  
CVI_RTSP_SERVICE_CreateFromJsonFile[./cfg_64M.json]  
[REMOTE] cvi_raw_dump_run,158: raw dump ready...  
waiting for connect...
```



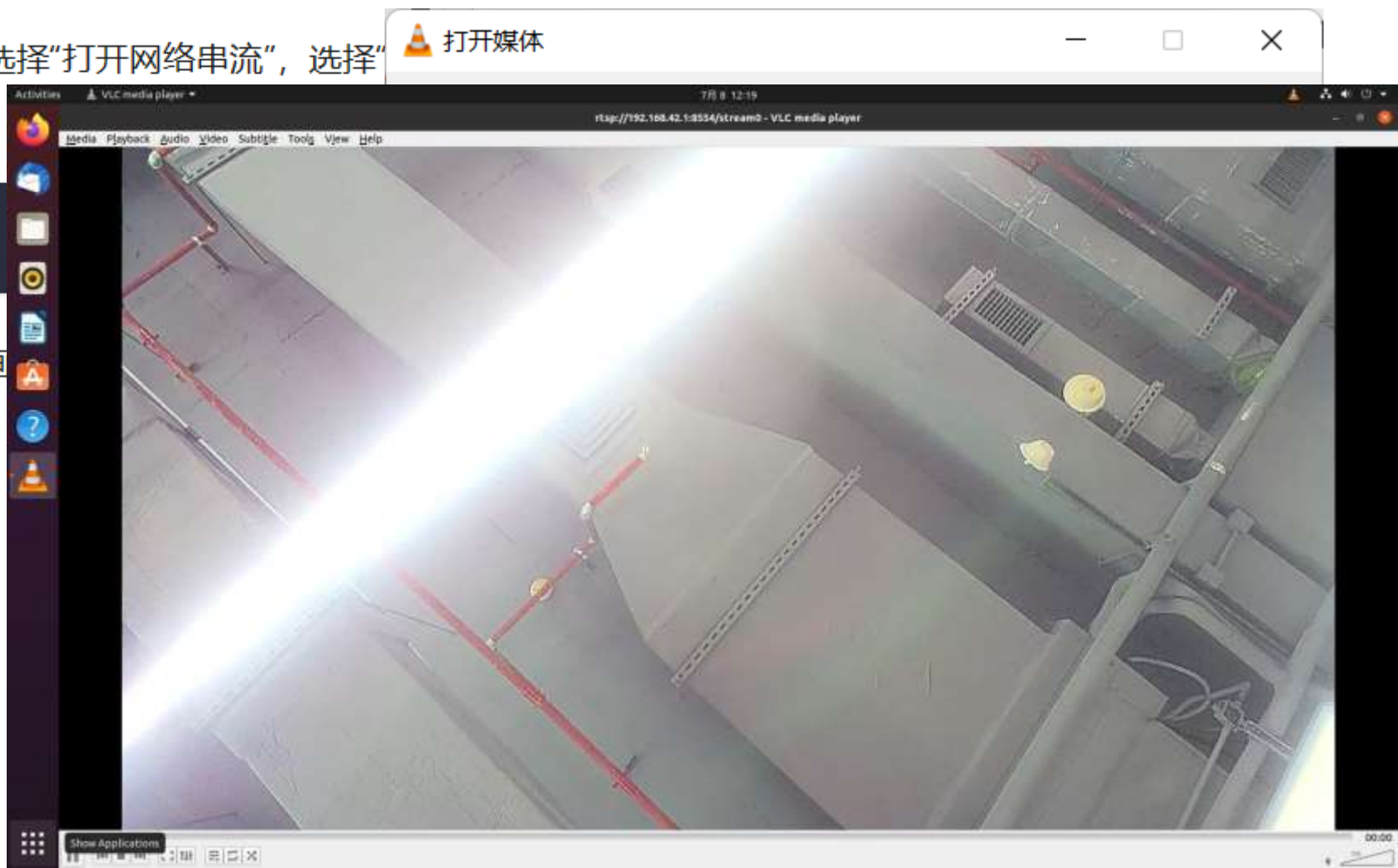
摄像头

注意rtsp:开头的链接，把IP改成Duo的IP就是我们要在VLC中拉流的地址了

在PC上打开VLC播放器，菜单“媒体”中选择“打开网络串流”，选择“入

`rtsp://192.168.42.1:8554/stream0`

再点“播放”，就可以看到摄像头推流的画



参考资料

- [启动 Duo | Milk-V \(milkv.io\)](#)
- [设置工作环境 | Milk-V \(milkv.io\)](#)
- [配置 Docker 开发环境 | Milk-V \(milkv.io\)](#)
- [基于 YOLOv5 的目标检测 | Milk-V \(milkv.io\)](#)
- [CAM-GC2083 | Milk-V \(milkv.io\)](#)

谢谢

