密级状态：绝密(　　)　　秘密(　　)　　内部(　)　　公开(√)

RKNN SDK 快速上手指南

（技术部，图形计算平台中心）

文件状态： [　] 正在修改 [√] 正式发布	当前版本：	1.4.0
	作　　者：	NPU
	完成日期：	2022-8-30
	审　　核：	熊伟
	完成日期：	2022-8-30

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd

更新记录

版本	修改人	修改日期	修改说明	核定人
1.3.0b0	林双杰	2022-4-23	初始版本	熊伟
1.3.3b0	林双杰	2022-6-20	1.修改下载的网盘地址 2.修改 lib 库路径设置问题	熊伟
1.4.0	林双杰	2022-08-29	更新版本到 1.4.0	熊伟

目 录

1 主要说明

此文档向零基础用户详细介绍如何快速在 ROCKCHIP 芯片的 EVB 板子上使用 RKNN-Toolkit2 和 RKNPU2 工具转换 yolov5s.onnx 模型为 yolov5s.rknn 模型并进行板端推理。

支持的平台：RV1106、RV1103。

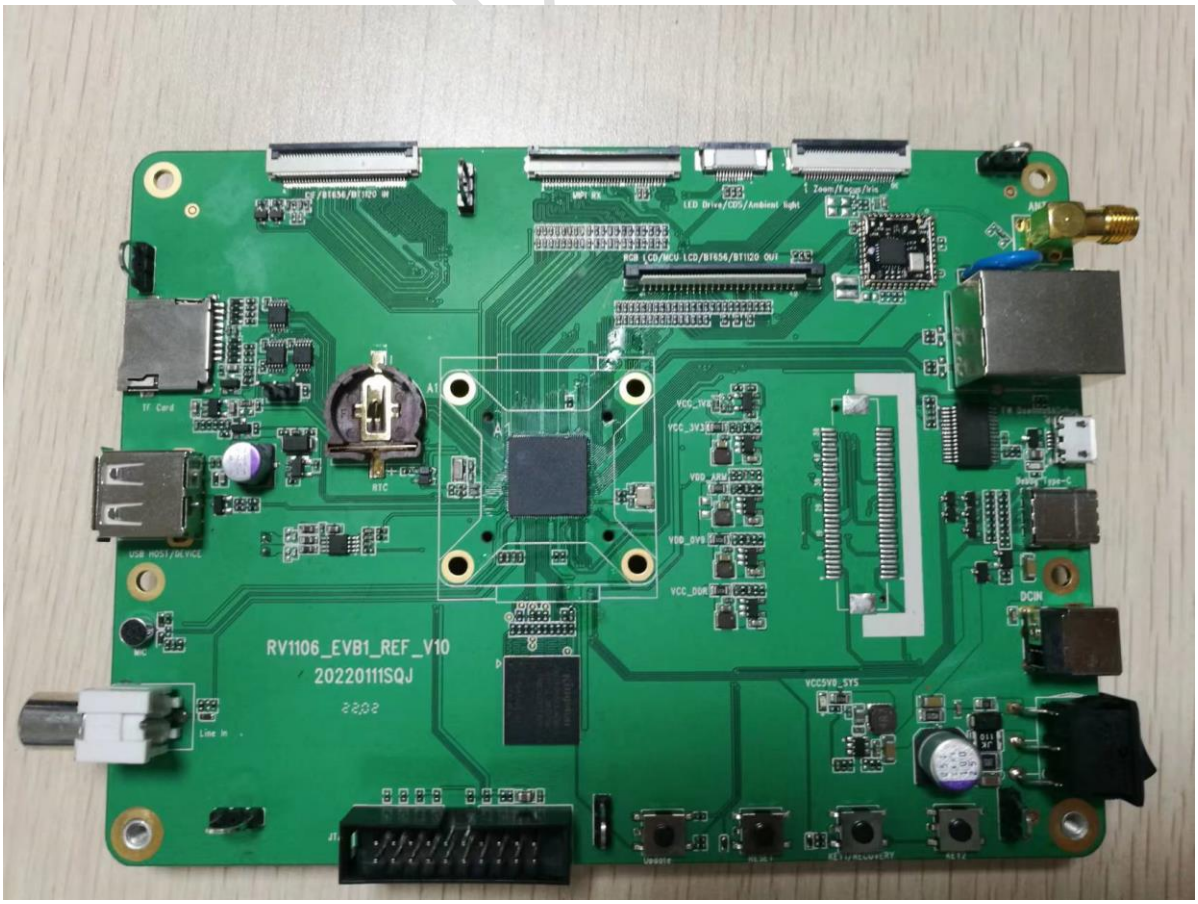
RKNPU2 工程下载地址：https://github.com/rockchip-linux/rknpu2

RKNN-Toolkit2 工程下载地址：https://github.com/rockchip-linux/rknn-toolkit2

2 准备工具

1. 一台操作系统 Ubuntu18.04 / Ubuntu20.04 的电脑。
2. 一块 EVB 板子（RV1106、RV1103）

RV1106



3．一条连接板子和电脑的数据线

RV1106：双头 USB 线

4．一个电源适配器

RV1106：输出 12V-2A

3 快速入门使用 RKNN-Toolkit2 和 RKNPU2

3.1 安装 RKNN-Toolkit2

本章节介绍两种安装使用 RKNN-Toolkit2 的方法，“通过 pip install 安装”和“通过 Docker 镜像安装”，用户可自行选择安装方式。如果不是 Ubuntu18.04 / Ubuntu20.04 系统的电脑推荐使用“通过 Docker 镜像安装”方式，已集成所有所需的安装包依赖，简单可靠。

以下操作以 Ubuntu18.04 和 Python3.6 为例。

3.1.1 通过 Docker 镜像安装并推理

1. 电脑若没有安装 Docker 工具，请先按照此安装教程（https://mirrors.tuna.tsinghua.edu.cn/help/docker-ce/）安装 Docker 工具,再进行下一步。
2. 打开一个终端命令行窗口，cd 进入 RKNN-Toolkit2 工程的 docker 文件夹，根据工程的保存路径修改 cd 命令中的路径

   cd <输入 RKNN-Toolkit2 工程其中 docker 文件夹的路径>

   命令：

   ```
   cd ~/Projects/rknn-toolkit2-1.x.x/docker
   ls
   ```

   查看到当前目录下有一个 docker 镜像文件 rknn-toolkit2-1.x.x-cp36-docker.tar.gz。
3. 加载 docker 镜像

   ```
   docker load --input rknn-toolkit2-1.x.x-cp36-docker.tar.gz
   ```

4. 查看当前所有的 docker 镜像

   命令：

   ```
   docker images
   ```

   能查询到 REPOSITORY 为 rknn-toolkit2，TAG 为 1.x.x-cp36 则表示加载成功。

5. 运行 docker 容器

命令：

```
docker run -t -i --privileged -v /dev/bus/usb:/dev/bus/usb \
-v ~/Projects/rknpu2/examples/RV1106_RV1103/rknn_yolov5_demo:/rknn_yolov5_demo \
rknn-toolkit2:1.x.x-cp36 /bin/bash
```

将目录映射进 Docker 环境可通过附加“-v <host src folder>:<image dst folder>”。

绿色部分为 rknpu2 工程中 examples/RV1106_RV1103/rknn_yolov5_demo 本地文件夹路径（根据本机路径修改）映射到 docker 容器中/rknn_yolov5_demo 文件夹。

成功进入 docker 容器后，命令 ls 能查看到文件夹 rknn_yolov5_demo 则表示映射成功。

6. 在 docker 容器中进入 rknn_yolov5_demo/convert_rknn_demo/yolov5 目录

命令：

```
cd rknn_yolov5_demo/convert_rknn_demo/yolov5
```

7. 转换 yolov5s.onnx 为 rknn 模型

```
python3 onnx2rknn.py
```

```
--> Loading model
W load_onnx: It is recommended onnx opset 12, but your onnx model opset is 11!
W load_onnx: Model converted from pytorch, 'opset_version' should be set 12 in torch.onnx.export for successful convert!
             More details can be found in examples/pytorch/torch2onnx
done
--> Building model
Analysing : 100%|████████████████████████████████████████| 162/162 [00:00<00:00, 3820.70it/s]
Quantizating : 100%|████████████████████████████████████| 162/162 [00:00<00:00, 531.97it/s]
W remove_dataconvert: The default input dtype of 'images' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '334' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '353' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '372' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
done
--> Export RKNN model: ././/yolov5s-640-640.rknn
done
```

此脚本直接生成 RV1106 平台部署的 rknn 模型，如果需要进行仿真实现，参考 rknn-toolkit2/examples/onnx/yolov5 下的 test.py 中的仿真实现代码。

3.1.2 通过 pip install 安装并推理

1. 打开一个终端命令行窗口，安装 Python3.6 和 pip3

命令：

```
sudo apt-get install python3 python3-dev python3-pip
```

2. 安装所需的依赖包

命令：

```
sudo apt-get install libxslt1-dev zlib1g-dev libglib2.0 libsm6 \
libgl1-mesa-glx libprotobuf-dev gcc
```

3. 进入 Toolkit2 工程文件夹，根据工程的保存路径修改 cd 命令中的路径

cd <输入 Toolkit2 工程的路径>

命令：

```
cd ~/rknn-toolkit2-1.x.x
```

4. 安装必要相应版本的依赖包

命令：

```
pip3 install -r doc/requirements_cp36-1.x.x.txt
```

备注：

1) 若在安装过程中出现“匹配不到 XX 版本”的错误，可能由于是 pip 版本太低造成。可先执行以下升级 pip 版本命令，pip 升级至版本 21.3.1，再执行重新执行上述安装命令。

```
python3 -m pip install --upgrade pip
```

5. 安装 RKNN-Toolkit2（Python3.6 for x86_64）

命令：

```
pip3 install \
package/rknn_toolkit2-1.x.x_xxxxxxxx-cp36-cp36m-linux_x86_64.whl
```

6. 检查 RKNN-Toolkit2 是否安装成功

命令：

```
python3
from rknn.api import RKNN
```

```
Python 3.6.9 (default, Dec  8 2021, 21:08:43)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from rknn.api import RKNN
>>>
```

若没有出现错误，说明安装成功。同时按住 Ctrl+D 退出 Python3。

7. cd 进入 rknpu2/examples/RV1106_RV1103/rknn_yolov5_demo/convert_rknn_demo/yolov5 目录

```
cd rknpu2/examples/RV1106_RV1103/rknn_yolov5_demo/convert_rknn_demo/yolov5
```

8. 转换 yolov5s.onnx 为 rknn 模型

命令：

```
python3 onnx2rknn.py
```

```
--> Loading model
W load_onnx: It is recommended onnx opset 12, but your onnx model opset is 11!
W load_onnx: Model converted from pytorch, 'opset_version' should be set 12 in torch.onnx.export for successful convert!
             More details can be found in examples/pytorch/torch2onnx
done
--> Building model
Analysing : 100%|████████████████████████████████████████| 162/162 [00:00<00:00, 3820.70it/s]
Quantizating : 100%|████████████████████████████████████| 162/162 [00:00<00:00, 531.97it/s]
W remove_dataconvert: The default input dtype of 'images' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '334' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '353' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
W remove_dataconvert: The default output dtype of '372' is changed from 'float32' to 'int8' in rknn model for performance!
                      Please take care of this change when deploy rknn model with Runtime API!
done
--> Export RKNN model: ././/yolov5s-640-640.rknn
done
```

此脚本直接生成 RV1106 平台的 rknn 模型，如果需要进行仿真实现，参考 rknn-toolkit2/examples/onnx/yolov5 下的 test.py 中的仿真实现代码。

3.2 RKNPU2 的编译及使用方法

此章节以 rknn_yolov5_demo 在 RV1106 linux arm32 位平台上运行为例，介绍如何使用 RKNPU2。

3.2.1 下载编译所需工具

下载完成后进行解压无需安装，记录文件夹的绝对路径。

板子为 linux 系统则需下载 gcc 交叉编译器

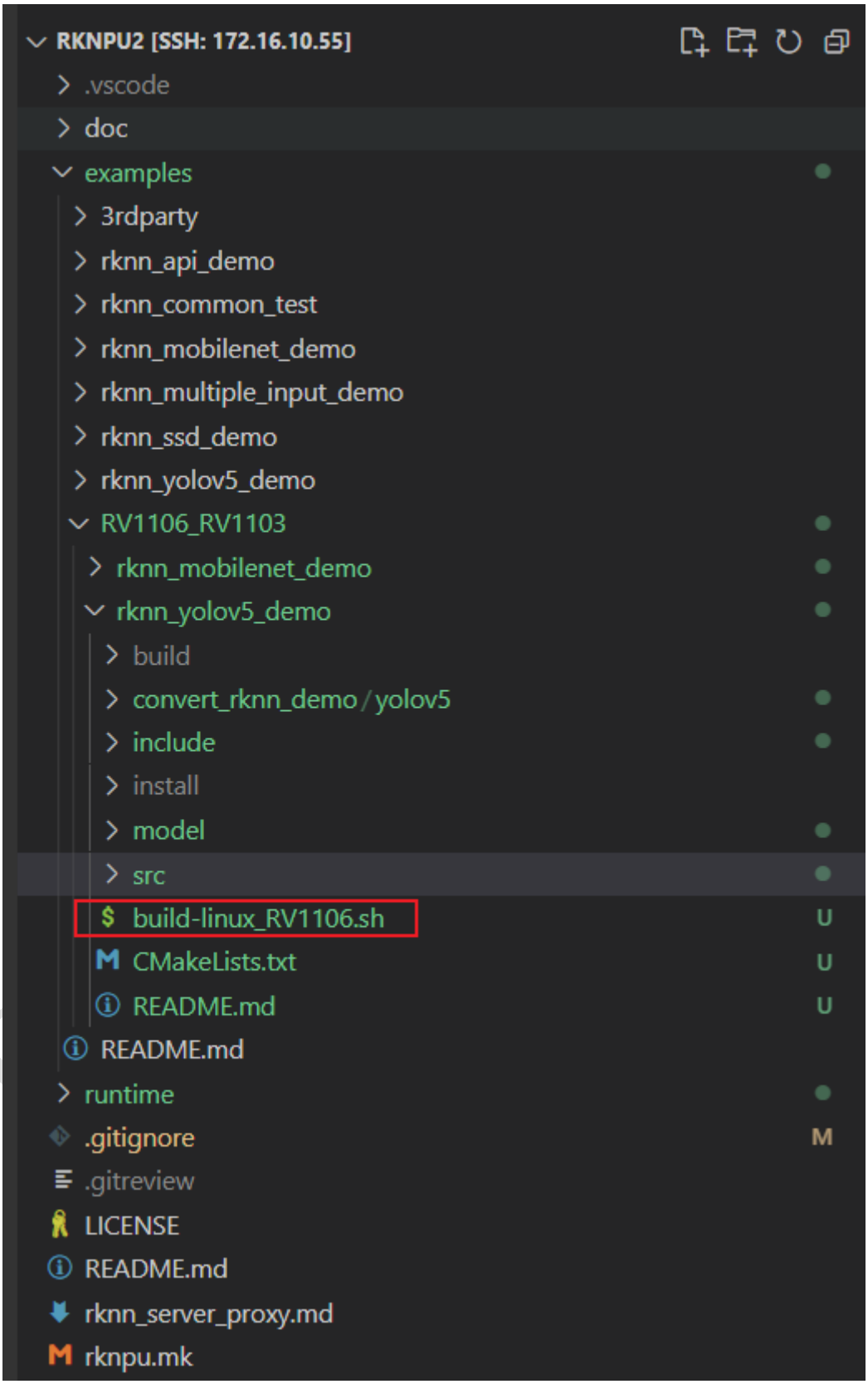
推荐版本 arm-rockchip830-linux-uclibcgnueabihf，百度网盘下载地址:

链接：https://eyun.baidu.com/s/3qZxSDNQ

提取码：rknn

3.2.2 demo 的编译工具路径设置

打开编译脚本 examples/RV1106_RV1103/rknn_yolov5_demo/build-linux_RV1106.sh


RKNPU2 [SSH: 172.16.10.55]
.vscode
doc
examples
3rdparty
rknn_api_demo
rknn_common_test
rknn_mobilenet_demo
rknn_multiple_input_demo
rknn_ssd_demo
rknn_yolov5_demo
RV1106_RV1103
rknn_mobilenet_demo
rknn_yolov5_demo
build
convert_rknn_demo / yolov5
include
install
model
src
build-linux_RV1106.sh
CMakeLists.txt
README.md
README.md
runtime
.gitignore
.gitreview
LICENSE
README.md
rknn_server_proxy.md
rknpu.mk

1) Linux 系统

设置 RK_RV1106_TOOLCHAIN 为本地电脑 arm-rockchip830-linux-uclibcgnueabihf 的路径并保存。

```
#!/bin/bash
set -e

if [ -z $RK_RV1106_TOOLCHAIN ]; then
  echo "Please set the RK_RV1106_TOOLCHAIN environment variable!"
  echo "example:"
  echo "  export RK_RV1106_TOOLCHAIN=<path-to-your-dir/arm-rockchip830-linux-uclibcgnueabihf>"
  exit
fi

# for arm
GCC_COMPILER=$RK_RV1106_TOOLCHAIN

ROOT_PWD=$( cd "$( dirname $0 )" && cd -P "$( dirname "$SOURCE" )" && pwd )

```

3.2.3 更新 RKNN 模型

把章节 3.1 转换后的 RV1106 平台模型 yolov5s-640-640.rknn 复制到 rknpu2/examples/RV1106_RV1103/rknn_yolov5_demo/model/RV1106/目录下。

3.2.4 编译 rknn_yolov5_demo

1) 在终端命令窗口进入 rknn_yolov5_demo 文件夹

命令：

```
cd examples/RV1106_RV1103/rknn_yolov5_demo/
```

2) 运行 build-linux_RK1106.sh 脚本编译程序

命令：

```
./ build-linux_RV1106.sh
```

备注：

1) 编译 RV1106 仅支持 arm linux 编译。详情可参考 /rknpu2/examples/RV1106_RV1103/rknn_yolov5_demo/README.md。

2) 若在编译时出现 cmake 错误，可执行以下命令安装 cmake 后再运行编译脚本。

```
sudo apt install cmake
```

3.2.5 在板端运行 rknn_yolov5_demo

1) 把编译好的程序和所需的文件 install/rknn_yolov5_demo_Linux 文件夹上传到板子的/data/文件夹下

命令：

```
adb root
adb push install/rknn_yolov5_demo_Linux    /data/
```

2) 进入板子系统

命令：

```
adb shell
```

3) cd 进入程序所在的目录

命令：

```
cd /data/rknn_yolov5_demo_Linux/
```

4) 设置库文件路径（**特别注意：路径一定要为绝对路径，不能使用相对路径**）

命令：

```
export LD_LIBRARY_PATH=/data/rknn_yolov5_demo_Linux/lib
```

5) 运行程序识别相应的图片中物体的类别

用法 Usage: ./rknn_yolov5_demo <rknn model> <jpg>

命令：

```
./rknn_yolov5_demo ./model/RV1106/yolov5s-640-640.rknn ./model/bus.jpg
```

```
# ./rknn_yolov5_demo model/RV1106/yolov5s-640-640.rknn  model/bus.jpg 1
rknn_api/rknnrt version: 1.2.6b3 (8b1c09099@2022-04-23T15:40:12), driver version: 0.7.0
model input num: 1, output num: 3
input tensors:
  index=0, name=images, n_dims=4, dims=[1, 640, 640, 3], n_elems=1228800, size=1228800, fmt=NHWC, type=UINT8, qnt_type=AFFINE, zp=-128, scale=0.003922
output tensors:
  index=0, name=334, n_dims=5, dims=[1, 16, 1, 6400, 16], n_elems=1638400, size=1638400, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=77, scale=0.080445
  index=1, name=353, n_dims=5, dims=[1, 16, 1, 1600, 16], n_elems=409600, size=409600, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=56, scale=0.080794
  index=2, name=372, n_dims=5, dims=[1, 16, 1, 400, 16], n_elems=102400, size=102400, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=69, scale=0.081305
custom string:
Begin perf ...
   0: Elapse Time = 63.60ms, FPS = 15.72
output origin tensors:
  index=0, name=334, n_dims=4, dims=[1, 255, 80, 80], n_elems=1632000, size=1632000, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=77, scale=0.080445
  index=1, name=353, n_dims=4, dims=[1, 255, 40, 40], n_elems=408000, size=408000, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=56, scale=0.080794
  index=2, name=372, n_dims=4, dims=[1, 255, 20, 20], n_elems=102000, size=102000, fmt=NCHW, type=INT8, qnt_type=AFFINE, zp=69, scale=0.081305
model is NHWC input fmt
loadLabelName ./model/coco_80_labels_list.txt
person @ (474 251 559 523) 0.996514
person @ (112 238 208 521) 0.992214
bus @ (101 141 559 445) 0.976798
person @ (211 242 285 509) 0.976798
```

4 参考文档

有关 RKNN-Toolkit2 更详细的用法和接口说明，请参考《Rockchip_User_Guide_RKNN_Toolkit2_CN.pdf》手册。

有关 RKNPU API 更详细的用法和接口说明，请参考《Rockchip_RKNPU_User_Guide_RKNN_API_V1.4.0_CN.pdf》手册。

5 附录

5.1 查看和设置开发板的 CPU、DDR 和 NPU 频率

通常，板子上的各个单元的频率是动态调频，这种情况下测试出来的模型性能会有波动。为了防止性能测试结果不一致，在性能评估时，建议固定板子上的相关单元的频率再做测试。相关单元的频率查看和设置命令如下：

5.1.1 CPU 定频命令

1) 查看 CPU 频率

```
cat /sys/kernel/debug/clk/clk_summary | grep arm
```

2) 固定 CPU 频率（**不可设置**）

5.1.2 DDR 定频命令

1) 查看 DDR 频率

```
cat /sys/kernel/debug/clk/clk_summary | grep ddr
```

2) 固定 DDR 频率（**不可设置**）

5.1.3 NPU 定频命令

1) 查看 NPU 频率（**需固件支持**）

对于 RV1106:

```
cat /sys/kernel/debug/clk/clk_summary | grep npu
```

2) 固定 NPU 频率（**不可设置**）

5.2 命令 adb devices 查看不到设备

1. 检查连线是否正确、重插拔或更换电脑 USB 插口

2. 在本地电脑和 docker 容器中使用 USB 连接的板子时，同一时间只能有一端使用 adb server。

   故若在一端执行命令（adb devices）时查看不到设备，可在另一命令端执行命令

```
adb kill-server
```

终止外部 adb service，再返回原先命令终端窗口执行命令（adb devices）查看设备。

3. 出现以下错误时，是未安装 adb。需要执行安装命令安装 adb。

```
Command 'adb' not found, but can be installed with:
sudo apt install adb
```

   命令：

```
sudo apt install adb
```