

RK1808开发板

☰ Tags	公司:上海电力
🕒 Created time	@June 14, 2023 4:30 PM
🕒 Last edited time	@July 4, 2023 6:02 PM

主板：RK3568J

AI计算棒：TB-RK1808M0

Knowledge (From Ted Wang)

[Reference](#)

[开发板的基本认识](#)

[查询当前开发板的操作系统](#)

[识别开发板上RK1808AI计算棒 \(算卡\)](#)

[TB-RK1808M0不支持PCIe x1模式](#)

[RK1808开发板两种模式介绍](#)

[被动模式](#)

[主动模式](#)

[npu_transfer_proxy](#)

[rknn.list_devices\(\) 使用rknn list出来NPU设备](#)

[开发板\(RK3568J主板 附带 TB-RK1808M0\)：硬软件说明及查询](#)

[总结](#)

[安装兼容版本的RKNN和npu驱动：](#)

[检查RK3568J 主板 系统间各固件版本](#)

[检查RK1808M0 计算棒 系统间各固件版本](#)

[Debian Linux 配置网络接口卡 - IP 地址和网络掩码](#)

[设置路由转发](#)

[配置域名解析](#)

[查看路由规则](#)

[查看内核的编译时间](#)

[进入RK1808M0计算棒](#)

[验证](#)

[RK1808官方inference例子](#)

[找错helmet_yolov5_rknn_deploy](#)

[环境配置\(vmware@10.187.109.80\)](#)

[环境配置的过程](#)

[mobilenet_v1 转到开发板推理测试 成功✔](#)

[官网yolov5\(onnx → rknn to 然后再去inference\) 成功✔](#)

[依赖包的版本 \(PC: ubuntu20.04\)](#)

[helmet_yolov5s 从onnx格式模型转为rknn格式模型](#)

[Reference](#)

Knowledge (From Ted Wang)


RK1808开发板 自己带ssh
ssh `linaro@10.117.202.132`
密码 `linaro`



RK1808开发板学习

RK1808 计算棒使用（主动模式）_Top嵌入式的博客-CSDN博客

使用 RK1808 计算棒运行神经网络，分担系统计算压力

 https://blog.csdn.net/qq_45396672/article/details/123075716

需要丢文件到1808：

scp xxx `toybrcik@192.168.180.0:/home/`

算卡的网口需要和主板的网口在一个网段，才能互相ping 到

Reference

<https://t.rock-chips.com/wiki.php?filename=板级指南/TB-RK1808S0>

<https://t.rock-chips.com/wiki.php?filename=板级指南/TB-RK1808M0>

```
ifconfig enx10dcb6gf2f29 192.168.180.1
```

```
ifconfig enx10dcb6gf2f29 up
```

我们在使用RK1808 算卡时候遇到一些疑问，

当前用RK1808M0 的算卡，接在 **RK3568J**的主板上，走PCIE的总线

我们虚拟网口已经可以被识别，1808 板子的 192.168.180.8 也可以被访问。

但是被动模式安装调用pcie加速卡的方式，实际的情况就是rknn 调用list_devices()没有返回结果

算卡的storage不大，内存不大。

开发板的基本认识

查询当前开发板的操作系统

```
linaro@linaro-alip:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description:    Debian GNU/Linux 10 (buster)
Release:        10
Codename:       buster
```

识别开发板上RK1808AI计算棒 (算卡)

在 RK1808 开发板上，可以使用 `lsusb` 命令查看 USB 设备列表，确认是否存在名为 "rknn" 的 NPU 设备。也可以使用 `npu-smi` 命令来检查 NPU 设备的状态和信息。

1. 首先，确保你的 RK1808 开发板已经正确连接到电脑或显示器，并处于开机状态。
2. 插入 RK1808 人工智能计算棒到 RK1808 开发板上接口。
3. 在你的电脑上打开终端或命令提示符，确保你的电脑可以通过开发板的 端口与 RK1808 开发板进行通信。
4. 运行以下命令来检测 RK1808 人工智能计算棒是否被识别：

```
linaro@linaro-alip:~$ lsusb
Bus 006 Device 002: ID 05e3:0620 Genesys Logic, Inc. USB3.2 Hub
Bus 006 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 005 Device 004: ID 413c:2107 Dell Computer Corp. Dell USB Entry Keyboard
Bus 005 Device 003: ID 046d:c063 Logitech, Inc. DELL Laser Mouse
Bus 005 Device 002: ID 05e3:0610 Genesys Logic, Inc. 4-port hub
Bus 005 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 002: ID 2207:0018 Fuzhou Rockchip Electronics Company TB-RK1808M0
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
linaro@linaro-alip:~$
```

插入TB-RK1808人工智能计算棒，PC上会显示U盘设备，U盘设备的目录结构如下：



```
linaro@linaro-alip:~$ du -sh /media/linaro/B362-5C8B/
5.0M    /media/linaro/B362-5C8B/
linaro@linaro-alip:~$ du -sh /media/linaro
5.0M    /media/linaro
linaro@linaro-alip:~$ du -sh /media
5.0M    /media
linaro@linaro-alip:~$
```

TB-RK1808M0不支持PCIe x1模式

因此数码开关要向下切换为USB3.0，这样通过 lsusb可以查看到人工智能计算棒。

请注意，**TB-RK1808M0通常使用USB接口进行连接**，而不是PCIe x1模式。

RK1808开发板两种模式介绍

被动模式开发流程简介	主动模式开发流程简介
被动模式下：RK1808 人工智能计算棒是一个 通用 AI加速器	主动模式下，RK1808人工智能计算棒是一个 专用 AI应用模块。
上位机 通过RKNN-Toolkit 将模型及前处理后的数据 传输 给RK1808 人工智能计算棒， RK1808 人工智能计算棒完成推理 ，并把 结果返回 上位机，上位机进行后处理以及显示等操作。	RK1808人工智能计算棒 内部默认已安装rknn-toolkit和rknn-api ，上位机（也称宿主机）无需安装rknn-toolkit和rknn-api， 模型及算法固化在RK1808人工智能计算棒中 ，上位机只需通过USB口向计算棒输入数据（例如图片和视频流），RK1808人工智能计算棒自动完成数据的前处理、推理、后处理，然后把处理结果通过USB口输出给上位机

当前python3.7, 对应的RKNN安装包为: rknn_toolkit_lite2-1.4.0-cp37-cp37m-linux_aarch64.whl`

- rknn for 1808已经改名为rknn-toolkit-lite。方便低内存低容量设备（例如1808）仅推理使用，依赖少。
- from rknnlite.api import RKNNLite as RKNN
from rknn.api import RKNN # 不是import rknn, 应为 import rknnlist

被动模式

在被动模式下，RK1808 开发板处于等待外部调用的状态，**不会主动发送设备信息**。因此，通过 `list_devices()` 方法**可能无法直接查找到** RK1808 设备。

现在的平台是 **RK3568J**

目前以下平台及系统支持被动模式 From link <https://t.rock-chips.com/wiki.php?filename=板级指南/TB-RK1808S0>：发现

- PC端Linux系统、Windows系统、Mac OS
- RK3399、RK3399Pro平台Linux系统、Android系统
- 其他不支持被动模式的平台及系统，用户可以使用主动模式，主动模式请参考Wiki教程--TB-RK1808S0-主动模式开发

因此当前**RK3568J 平台应该是不支持被动模式！**

RK1808M0计算棒在RK3568的平台上是不支持被动模式!

主动模式

npu_transfer_proxy

```
linaro@linaro-alip:~$ npu_transfer_proxy devices
```

```
List of ntb devices attached
```

```
TM018084201100315 4c362350 USB_DEVICE
```

在 TB-RK1808M0 开发板上，**npu_transfer_proxy** 主要用于将主机上的数据传输到 NPU 进行推理计算，或者将 NPU 推理结果传输回主机。

npu_transfer_proxy 提供了一个**轻量级的数据传输框架**，它与 Rockchip NPU 驱动程序协同工作，将数据有效地传输到 NPU 上进行处理，并将结果返回给主机。通过使用 npu_transfer_proxy，开发者可以方便地在主机和 NPU 之间传输输入数据和输出结果，从而实现在 TB-RK1808M0 上进行 NPU 推理计算的功能。

失败的 😞

```
linaro@linaro-alip:~$ npu_transfer_proxy
I NPUTransfer: Starting NPU Transfer Proxy, Transfer version 2.0.0 (8f9ebbc@2020-04-03T09:12:43), devid = TM018084201100054, pid = 2189:1093

linaro@linaro-alip:~$ TM018084201100054 device
-bash: TM018084201100054: 未找到命令
```

成功的 😊

```
linaro@linaro-alip:~$ npu_transfer_proxy
I NPUTransfer: Starting NPU Transfer Proxy, Transfer version 2.0.0 (8f9ebbc@2020-04-03T09:12:43), devid = TM018084201100054, pid = 2306:1093
linaro@linaro-alip:~$ npu_transfer_proxy devices
List of ntb devices attached
TM018084201100315 4c362350 USB_DEVICE
linaro@linaro-alip:~$
```

rknn.list_devices() 使用rknn list出来NPU设备

RKNN版本：**rknn_toolkit-1.7.3-cp37-cp37m-linux_aarch64.whl**

```
linaro@linaro-alip:~/rknn$ python3
Python 3.7.3 (default, Oct 31 2022, 14:04:00)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from rknnlite.api import RKNNLite as RKNN
>>> rknn = RKNN()
>>> rknn.list_devices()
*****
None devices connected.
*****
([], [])
# 没有返回结果
```

正常应该是如下结果

返回的设备列表信息如下（这里有两个计算棒，它们的连接模式都是 ntb）：

```
all device(s) with adb mode:
[]
all device(s) with ntb mode:
['TB-RK1808S0', '515e9b401c060c0b']
```

注：使用多设备时，需要保证它们的连接模式都是一致的，否则会引起冲突，导致设备连接失败。

本机上安装的rknn版本是

rknn-toolkit-lite2 1.4.0

开发板(RK3568J主板 附带 TB-RK1808M0)：硬软件说明及查询

总结

- 主板型号: TB-RK3568J
- 人工智能计算棒：RK1808M0
- 系统信息: debian10
- **RK3568J主板**上开发库以及部署工具版本：
 - python版本: 3.7

从**RK3568J**的主板上，查询到的系统固件各组件之间的对应关系：

RKNN Toolkit (rknn-toolkit-lite)	rknn_server	NPU 驱动	librknn_runtime
1.4.0	1.3.0	无	1.3.0

彼此版本不匹配 且 当前RK3568J主板上没有AI计算棒，因此肯定查询不到NPU。

- **TB-RK1808M0** 人工智能计算棒 开发库以及部署工具版本:

从**RK3568J**的主板上ssh登陆到RK1808M0计算棒，查询到的系统固件各组件之间的对应关系：

RKNN Toolkit (rknn-toolkit-lite)	rknn_server	NPU 驱动	librknn_runtime
1.4.0	1.5.0	6.4.3.5.293908	1.5.0

```
>>> from rknnlite.api import RKNNLite as RKNN
>>> rknn = RKNN()
```

```
>>> rknn.list_devices()
```

可能是由于RKNN和NPU版本不兼容导致无法通过 `list_devices()` 找到NPU设备。

和系统固件各组件之间的对应关系如下：

RKNN Toolkit	rknn_server	NPU 驱动	librknn_runtime
1.0.0	0.9.6/0.9.7	6.3.3.3718	0.9.8/0.9.9
1.1.0	0.9.8	6.3.3.03718	1.0.0
1.2.0	0.9.9	6.4.0.213404	1.1.0
1.2.1	1.2.0	6.4.0.213404	1.2.0
1.3.0	1.3.0	6.4.0.227915	1.3.0
1.3.2	1.3.2	6.4.0.7915	1.3.2
1.4.0	1.4.0	6.4.0.27915	1.4.0
1.6.0	1.6.0	6.4.3.5.293908	1.6.0
1.6.1	1.6.1	6.4.3.5.293908	1.6.1
1.7.0	1.7.0	6.4.6.5.351518	1.7.0
1.7.1	1.7.1	6.4.6.5.351518	1.7.1

CSDN @myqpy

安装兼容版本的RKNN和Npu驱动：

在使用RKNN API进行部署之前，需要使用RKNN Toolkit将原始模型转化成rknn模型。

- RK1808/RK1806/RV1109/RV1126/RK3399Pro 使用：<https://github.com/rockchip-linux/rknn-toolkit>

rknn-toolkit 要和 rknpu 匹配

RKNN Toolkit (rknn-toolkit-lite)	rknn_server	NPU 驱动	librknn_runtime
1.7.1	1.7.1	6.4.3.5.293908	1.5.0
<code>rknn_toolkit_lite-1.7.0-cp37-cp37m-linux_aarch64.whl</code>	<code>drivers/linux-aarch64/usr/bin</code>		

```
# 更新RK1808的驱动 为 1.7.1版本：

# 1. 适应于RK1808平台的是rknn-toolkit 和 rknpu

# 2. 下载 rknpu-1.7.1.zip 解压缩并 scp到 toybrick@192.168.180.1(这是AI计算棒) 上

# 3. 选择更新成 NPU full driver, RK1808对应的驱动是 drivers/linux-aarch64

# 4. 把相关的驱动拷贝到对应的目录就可以
cp drivers/npu_ko/galcore.ko /lib/modules/galcore.ko

cp drivers/linux-aarch64/ /
```

检查RK3568J 主板 系统间各固件版本

```
# execute these commands on RK1808

# 查询 NPU 驱动版本
dmesg | grep -i galcore # None

# 查询 rknn_server 版本
$ strings /usr/bin/rknn_server | grep build
1.3.0 (121b661 build: 2022-04-29 11:11:47)
.note.gnu.build-id

# 查询 librknn_runtime 版本
# librknn_api是对librknn_runtime的封装，主要是为了减少对其他so的编译依赖，功能上并没有区别。检查驱动版本时，一般以librknn_runtime.so为准。

strings /usr/lib/librknn_runtime.so | grep version
# 或者
$ strings /usr/lib/librknn_api.so | grep version
librknnrt version: 1.3.0 (9b36d4d74@2022-05-04T20:17:01)
rknn_query, info_len(%d) != sizeof(rknn_sdk_version)(%d)!
model version < 3, cannot query RKNN_QUERY_NATIVE_NHWC_OUTPUT_ATTR
unsupported file format version
Invalid RKNN model version
RKNN Model Information: version: %d, toolkit version: %s, target: %s, target platform: %s, framework name: %s, framework layout: %s
failed to check rknpu hardware version: %x
The rknn model does not match the RKNPU hardware version!
RKNN Model version: %d.%d.%d not match with rknn runtime version: %d.%d.%d
Generated from compiler version:
(compiler version:
RKNN Driver Information: version: %d.%d.%d
Mismatch driver version, %s requires driver version >= %d.%d.%d, but you have driver version: %d.%d.%d which is in compatible!
Current driver version: %d.%d.%d, recommend to upgrade the driver to the new version: >= %d.%d.%d
wrong version
incompatible version
.gnu.version
.gnu.version_r
```

总结起来，

NPU驱动是底层的硬件驱动程序，负责与NPU设备通信；

rknn-toolkit是模型转换、编译和部署的工具集；

librknn_runtime是Rockchip NPU的运行时库，用于加载和执行RKNN模型；

而rknn_server是一个使用librknn_runtime库的模型推理服务器程序。


它们之间协同工作，完成了从模型转换到模型推理的整个流程。

检查RK1808M0 计算棒 系统间各固件版本

在Debian 10环境下更新瑞芯微rk1808计算棒

在linux环境下更新瑞芯微rk1808计算棒_myqpy的博客-CSDN博客


瑞芯微rk1808计算棒升级

 <https://blog.csdn.net/quququququq/article/details/130552698>

Debian Linux 配置网络接口卡 - IP 地址和网络掩码

Debian Linux 配置网络接口卡 - IP 地址和网络掩码

Debian Linux 配置网络接口卡 - IP 地址和网络掩码 要在 Debian Linux 系统上配置网络接口卡 (NIC)，您可以使用 ifconfig 和 ip 命令。要查看 NIC 的当前配置，请使用 ifconfig 命令：ifconfig 这将显示每个配置的 NIC 的名称、IP 地址和网络掩码。要为 NIC 分配静态 IP 地址和网络掩码，请使用具有以下语法的 ifconfig

 https://www.onitroad.com/jc/linux/faq/debian-linux-configure-network-interface-cards_-_ip-address-and-netmasks.html

给Fuzhou Rockchip NPU设备虚拟网口配置IP

```
# 查看 NIC 的当前配置，请使用 ifconfig命令
sudo ifconfig
# 或者
ip addr

# 要为 NIC 分配静态 IP 地址和网络掩码
sudo ifconfig enx10dcb69f302e 192.168.180.1 netmask 255.255.255.0 up

# 应用更改并启动接口
sudo ifconfig enx10dcb69f2f29 up

# 稍等一分钟再次查看 NIC，理应看到 enx10dcb69f2f29 已经分配了IP
sudo ifconfig
```

设置路由转发

(在当前用户的环境变量 vi ~/.bashrc里面加上路径，就可以直接运行iptables了，不需要加sudo了)

```
export PATH=$PATH:/sbin:/bin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin
```

source ~/.bashrc)

```
#设置本地ipv4转发
sudo echo "net.ipv4.ip_forward = 1" >> /etc/sysctl.conf
sudo sysctl -p
#路由转发

# check alternatives to iptables-legacy
```

```
$ update-alternatives --config iptables

sudo iptables -t nat -L # List the rules in a chain or all chains
# sudo iptables -F # Delete all rules in chain or all chains
# sudo iptables -t nat -F

sudo iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
#其中eth1 需要修改为对应的以太网标识码
# 完成上述命令后, 通过 sudo iptables -t nat -L 查看可以看到多了一行: MASQUERADE all -- anywhere anywhere
```

1. 查看nat tables

可是我按照toybrick 文档配置虚拟网卡, IP为192.168.180.1

```
linaro@linaro-alip:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination

Chain INPUT (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source                destination
```

2. 通过 iptables-legacy 设置路由

```
sudo apt-get update
sudo apt-get install iptables-persistent

# 你可以尝试使用iptables-legacy命令来代替iptables, 以确保使用传统的iptables而不是nf_tables。在Debian 10中, 可以通过以下命令安装
iptables-legacy

root@linaro-alip:/home/linaro# iptables-legacy -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination

Chain INPUT (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source                destination

sudo iptables-legacy -t nat -A POSTROUTING -o eth1 -j MASQUERADE

root@linaro-alip:/home/linaro# iptables-legacy -t nat -A POSTROUTING -o eth1 -j MASQUERADE
root@linaro-alip:/home/linaro#
root@linaro-alip:/home/linaro# iptables-legacy -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination

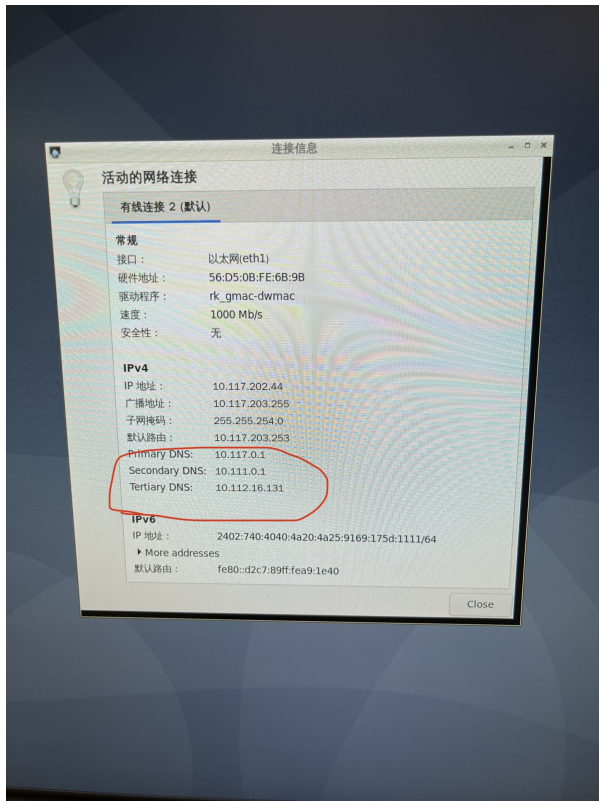
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source      destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source      destination
MASQUERADE all  --  anywhere    anywhere
```

配置域名解析

在公司里面只能用公司自己的域名进行解析，查看开发板的网络如下图，然后进入计算棒中 vi /etc/resolv.conf 添加 `nameserver 10.117.0.1` 完成之后，ping baidu可以ping通。



```
toybrick@debian10:~$ cat /etc/resolv.conf
nameserver 10.117.0.1
nameserver 10.111.0.1
nameserver 8.8.8.8
```

```
toybrick@debian10:~$ ping www.baidu.com
PING www.a.shifen.com (110.242.68.4) 56(84) bytes of
data:
64 bytes from 110.242.68.4 (110.242.68.4): icmp_seq=
1 ttl=51 time=10.3 ms
64 bytes from 110.242.68.4 (110.242.68.4): icmp_seq=
2 ttl=51 time=10.7 ms
```

查看路由规则

```
root@linaro-alip:/home/linaro# ip route
default via 10.117.203.253 dev eth1 proto dhcp metric 101
10.117.202.0/23 dev eth1 proto kernel scope link src 10.117.202.132 metric 101
192.168.180.0/24 dev enx10dcb69f2f29 proto kernel scope link src 192.168.180.1

toybrick@debian10:~$ ip route
default via 192.168.180.1 dev usb0
192.168.180.0/24 dev usb0 proto kernel scope link src 192.168.180.8
```

查看linux 内核 和 iptables 版本

```
# debian 10 当前 kernel 内核对版本
linaro@linaro-alip:~$ uname -r
4.19.232

# iptables 版本
$ sudo iptables-legacy -V
iptables v1.8.2 (legacy)

$ update-alternatives --config iptables
There are 2 choices for the alternative iptables (providing /usr/sbin/iptables).

   Selection    Path                                Priority  Status
   -----
   0             /usr/sbin/iptables-nft             20      auto mode
   * 1           /usr/sbin/iptables-legacy          10      manual mode
   2             /usr/sbin/iptables-nft             20      manual mode

$ sudo update-alternatives --set iptables /usr/sbin/iptables-legacy

$ sudo update-alternatives --set ip6tables /usr/sbin/ip6tables-legacy

$ sudo update-alternatives --set iptables /usr/sbin/iptables-nft
$ sudo update-alternatives --set ip6tables /usr/sbin/iptables-nft
```

查看内核的编译时间

第二次内核编译时间：[Mon Jun 19 13:44:50](#)

```
linaro@linaro-alip:~$ cat /proc/version
Linux version 4.19.232 (wxp@shfae-Lenovo) (gcc version 6.3.1 20170404 (Linaro GCC 6.3-2017.05), GNU ld (Linaro_Bin
utils-2017.05) 2.27.0.20161019) #35 SMP Mon Jun 19 13:44:50 CST 2023
linaro@linaro-alip:~$ uname -a
Linux linaro-alip 4.19.232 #35 SMP Mon Jun 19 13:44:50 CST 2023 aarch64 GNU/Linux
```

第三次内核编译时间：[Tue Jun 20 13:47:08](#)

```
root@linaro-alip:/home/linaro# cat /proc/version
Linux version 4.19.232 (wxp@shfae-Lenovo) (gcc version 6.3.1 20170404 (Linaro GCC 6.3-2017.05), GNU ld (Linaro_Bin
utils-2017.05) 2.27.0.20161019) #36 SMP Tue Jun 20 13:47:08 CST 2023
root@linaro-alip:/home/linaro# uname -a
Linux linaro-alip 4.19.232 #36 SMP Tue Jun 20 13:47:08 CST 2023 aarch64 GNU/Linux
```

进入RK1808M0计算棒

从设置路由转发的步骤，可以发现，由于内核中缺少路由转发的模块，因此无法完成路由转发的功能。

现在 仅可以 以 RK3568J 主板作为跳板机，登陆 RK1808M0 计算棒，但是RK1808M0计算棒 没有联网功能。

```
linaro@linaro-alip:~$ ssh toybrick@192.168.180.8
toybrick@192.168.180.8's password:
Linux debian10.toybrick 4.4.194 #7 SMP PREEMPT Tue Oct 27 15:35:44 CST 2020 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

```

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Oct 23 00:13:22 2019 from 192.168.180.1

# 在 RK1808M0计算棒上
# rknn-toolkit 版本是 1.4.0
toybrick@debian10:~$ pip list | grep rknn
rknn-toolkit-lite 1.4.0

# 在 RK1808M0计算棒上
# 查询 NPU 驱动版本: 查到NPU驱动版本为 Galcore version 6.4.3.5.293908
toybrick@debian10:~$ dmesg | grep -i galcore
[ 6.846591] galcore: loading out-of-tree module taints kernel.
[ 6.850197] galcore: npu init.
[ 6.852894] galcore: start npu probe.
[ 6.853839] Galcore version 6.4.3.5.293908
[ 6.853847] Galcore options:
[ 6.862079] Galcore Info: ContiguousBase=0x32e00000 ContiguousSize=0x400000
[ 6.866980] Galcore Info: MMU mapped core 0 SRAM[0] hardware virtual address=0x400000 size=0x80000
[ 6.867020] Galcore Info: MMU mapped core 0 SRAM[1] hardware virtual address=0x480000 size=0x80000
[ 6.867035] Galcore Info: MMU mapped external shared SRAM[0] CPU view base=0xfec10000 GPU view base=0xfec10000
GPU virtual address=0xfec10000 size=0x1f0000
[ 6.873233] galcore ffb0000.npu: Init npu devfreq
[ 6.873454] galcore ffb0000.npu: leakage=23
[ 6.873642] galcore ffb0000.npu: Failed to get pvtm
[ 6.874103] galcore ffb0000.npu: avs=0
[ 6.878526] galcore ffb0000.npu: l=0 h=2147483647 hyst=5000 l_limit=594000000 h_limit=0 h_table=0

# 在 RK1808M0计算棒上
# 查询 rknn_server 版本 为 1.5.0
toybrick@debian10:~$ strings /usr/bin/rknn_server | grep build
1.5.0 (83d70a9 build: 2020-09-30 16:25:49)
.note.gnu.build-id

# 在 RK1808M0计算棒上
# 查询 librknn_runtime 版本 为 1.5.0
toybrick@debian10:~$ strings /usr/lib/librknn_runtime.so | grep version
rknn_get_sdk_version
pre_compile_version
optimization_level_version
op_version
I [%s:%d]custom op version: %d
E [%s:%d] rknn_query, info_len(%d) != sizeof(rknn_sdk_version)(%d)!
librknn_runtime version 1.5.0 (cc8a066 build: 2020-09-30 16:23:55 base: 1126)
W [%s:%d]Unsupported evis version
//int8 version
E [%s:%d]unsupport kernel size:%d/dilation:%d/evis version:%d
.gnu.version
.gnu.version_r

```

综上所述：

从RK3568J的主板上ssh登录到RK1808M0计算棒，查询到的系统固件各组件之间的对应关系：

RKNN Toolkit (rknn-toolkit-lite)	rknn_server	NPU 驱动	librknn_runtime
1.4.0	1.5.0	6.4.3.5.293908	1.5.0

```

>>> from rknnlite.api import RKNNLite as RKNN
>>> rknn = RKNN()
>>> rknn.list_devices()

```

可以发现 可能 是由于RKNN版本不兼容导致无法通过 `list_devices()` 找到NPU设备。

验证

RK1808官方inference例子

Index of /rk1808/

<https://repo.rock-chips.com/rk1808/>

`mobilenet_v1/` 验证成功 😊

```
toybrick@debian10:~/mobilenet_v1$ python3 test_rknn.py
rknn load model result output: 0
#####
--> Init runtime environment
init runtime environment: 0
#####
done
--> Running model
mobilenet_v1
-----TOP 5-----
[156]: 0.892578125
[155]: 0.061737060546875
[205]: 0.00876617431640625
[188]: 0.00791168212890625
[263]: 0.005245208740234375

inference time: 0.020009279251098633
done
toybrick@debian10:~/mobilenet_v1$ python3
```

`yolo3/` 验证成功 😊

`helmet_yolov5_rknn_deploy` 失败 😞

```
rknn load model result output: 0
#####
1 --> Init runtime environment
E Catch exception when init runtime!
*****
None devices connected.
*****
E []
E Traceback (most recent call last):
  File "/home/toybrick/.local/lib/python3.7/site-packages/rknnlite/api/rknn_lite.py", line 152, in init_runtime
    self.rknn_runtime.build_graph(self.rknn_data, self.load_model_in_npu)
  File "rknnlite/api/rknn_runtime.py", line 306, in rknnlite.api.rknn_runtime.RKNNRuntime.build_graph
  File "rknnlite/api/rknn_model.py", line 132, in rknnlite.api.rknn_model.RKNNModel.parse_data_struct.error: unpack_from requires a buffer of at least 8 bytes
init runtime environment: -1
```

找错helmet_yolov5 rknn deploy

为什么没有成功呢？

☐ 模型转换时候的环境问题，依赖包的版本？

- RK1808 芯片要求 `rknn_toolkit_lite`；RK3568是 `rknn_toolkit_lite2`
- `rknn_toolkit_lite-1.4.0-cp36-cp36m-linux_x86_64.whl`
- 验证方法：用官网现有的例子 https://repo.rock-chips.com/rk1808/mobilenet_v1/去一个配置好的环境下去做转换

☐ *.pt → *.rknn 模型转换时没有加target，如下？

```
rknn.config(...)
target_platform = 'rk1808',
...
```

- 验证方法：用官网现有的yolov5例子(会带有target)转换后再次实验

环境配置(vmware@10.187.109.80)

环境配置的过程

- `libsqlite3` installed
- `sudo apt install libsqlite3-dev` `ModuleNotFoundError: No module named '_sqlite3'` `from _sqlite3 import *`
- Python 3.6 installed

Linux 安装Python3.6

如果你的机器上已经存在Python2.6的话不用删除，因为有很多程序是依赖Python2.6。第一步：我们先安装Python3.6所依赖的包：第二步：下载Python...

<https://www.jianshu.com/p/5ff67140c404>



`/usr/local/python3/bin/python3`

```
# Linux install Python3.6.14

# 从官网下载https://www.python.org/downloads/
wget https://www.python.org/ftp/python/3.6.14/Python-3.6.14.tgz
tar -xzf Python-3.6.14.tgz

# build
cd Python-3.6.14
sudo ./configure --prefix=/usr/local/python3 --enable-loadable-sqlite-extensions
sudo make install

Python 3.6 installed
```

• virtualenv installed

```
# install virtualenv
sudo /usr/local/python3/bin/python3 -m pip install virtualenv
virtualenv rknnenv
$ python
Python 3.6.14 (default, Jul 3 2023, 03:42:21)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license"
>>>
```

• rknn-toolkit-1.7.1 Installed

<https://github.com/rockchip-linux/rknn-toolkit/releases/tag/v1.7.1>

`rknn_toolkit-1.7.1-cp36-cp36m-linux_x86_64.whl`

mobilenet_v1 转到开发板推理测试 成功✓

mobilenet_v1 模型转换需要的依赖包

- `pip3 install tensorflow==1.14.0` (mobilenet_v1 模型转换需要) `tensorflow-1.14.0-cp36-cp36m-manylinux1_x86_64.whl`
- 在该环境中生成 `mobilenet_v1_nimbus.rknn` to do model inference

官网yolov5(onnx → rknn to 然后再去inference) 成功✓

依赖包的版本 (PC: ubuntu20.04)

```
(rknnenv) vmware@sc2-10-187-109-80:~/workspace/software$ python -V
Python 3.6.14
(rknnenv) vmware@sc2-10-187-109-80:~/workspace/software$ pip -V
pip 21.3.1 from /home/vmware/workspace/rknnenv/lib/python3.6/site-packages/pip (python 3.6)
(rknnenv) vmware@sc2-10-187-109-80:~/workspace/software$
(rknnenv) vmware@sc2-10-187-109-80:~/workspace/software$ pip list
Package            Version
-----
absl-py            1.4.0
astor              0.8.1
certifi           2023.5.7
chardet           3.0.4
click             8.0.4
dataclasses       0.8
decorator         5.1.1
dill              0.2.8.2
Flask             1.0.2
flatbuffers       1.10
future            0.18.3
gast              0.5.4
google-pasta     0.2.0
grpcio            1.48.2
h5py              2.8.0
idna              2.8
importlib-metadata 4.8.3
itsdangerous      2.0.1
Jinja2            3.0.3
joblib            1.1.1
Keras-Applications 1.0.8
Keras-Preprocessing 1.1.2
lmdb              0.93
Markdown          3.3.7
MarkupSafe        2.0.1
networkx          1.11
numpy             1.16.3
onnx              1.6.0
onnx-tf           1.2.1
onnxruntime       1.10.0
opencv-python    4.4.0.46
Pillow            5.3.0
pip               21.3.1
ply               3.11
protobuf         3.11.2
psutil            5.6.2
PyYAML            6.0
requests          2.22.0
rknn-toolkit      1.7.1
ruamel.yaml       0.15.81
scikit-learn     0.24.2
scipy             1.5.4
setuptools        59.6.0
six               1.16.0
sklearn           0.0
tensorboard       1.14.0
```

```

tensorflow          1.14.0
tensorflow-estimator 1.14.0
termcolor           1.1.0
threadpoolctl       3.1.0
torch               1.6.0
typing              3.7.4.3
typing_extensions   4.1.1
urllib3             1.25.11
Werkzeug            2.0.3
wheel               0.37.1
wrapt               1.15.0
zippp               3.6.0
(rknnenv) vmware@sc2-10-187-109-80:~/workspace/software$

```

helmet_yolov5s 从onnx格式模型转为rknn格式模型

```

# onnx -> rknn
# https://github.com/harperjuanl/helmet_yolov5_rknn_deploy/blob/main/onnx_to_rknn/examples/onnx/yolov5s/onnx2rknn.py
rknn.config(mean_values=[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]],
             std_values=[[255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0, 255.0]],
             batch_size=opt.batch_size,
             target_platform = 'rk1808') # Necessary to add target=rk1808

```

Reference

<https://github.com/MichaIng/DietPi/issues/1012>

主动模式：把计算棒当开发版，直接在计算棒上运行程序，和上位机通过计算棒之间的虚拟网卡socket编程交互。

被动模式：把计算棒当计算资源，在上位机开发程序，通过调用rknn接口让计算棒进行运算。

- rk1808 mobilenet_v1 例子：https://repo.rock-chips.com/rk1808/mobilenet_v1/
- 官方rockchip github yolov5例子：<https://github.com/rockchip-linux/rknn-toolkit/tree/master/examples/onnx/yolov5>
- 我们需要搞定的 helmet_yolov5s例子：
https://github.com/harperjuanl/helmet_yolov5_rknn_deploy/tree/main
- 配置好的环境：ssh vmware@10.187.109.80 密码是1，配置了一个虚拟python3.6环境: [source](#)
`/home/vmware/workspace/rknnenv/bin/activate`
例子代码的位置：`/home/vmware/workspace/example-rknn`