**Выполнил:**

Александр

Тюльпанов

дд.мм.гггг

**Сборка и установка пакетов под архитектуру aarch64**

**0. Небольшое предисловие**

Здесь я опишу несколько комманд, такие как скопировать по ssh на целевую платформу и работа с docker контейнерами.

Копирование по ssh:

scp путь\_до\_файла логин@ip\_адресс:путь\_назначения\_на\_платформе

Запуск docker контейнера в фоне:

docker run --name <ID образа> имя\_образа /bin/true

Запуск docker контейнера с определенной командой:

docker run -it имя\_контейнера комманда

Копирование из docker контейнера:

docker cp <ID образа>:/work/имя\_файла имя\_файла

**1.OpenVINO (Для Intel Compute Stick)**

**1.1 Требуемые пакеты (Их поставить просто через pip):**

1) tensorflow-aarch64>=1.2.0

3) networkx>=1.11

4) numpy>=1.12.0

5) protobuf==3.6.1

6) test-generator==0.1.1

7) defusedxml>=0.5.0

**1.2 Inference-engine**

OpenVINO официально не поддерживает arm64,только arm32. Но это не беда, так как arm64 умеет запускать arm32 приложения, поэтмоу сначала мы соберем кросс-компиляцией весь OpenVINO, а затем перекинем его на arm64 платформу (в моем случае – orangepi). Для кросс-компиляции будем использовать docker.

1) Скачайте репозиторий:

git clone <https://github.com/au4lab/orangePi_neuralNetwork.git>

cd orangePi\_neuralNetwork

2) Там будет папка cross\_OpenVINO. Запустите образ docker

docker image build -t ie\_cross\_armhf cross\_openVINO/

Дождитесь сборки. Все собранные файлы лежат в директори /work/

Сейчас мы собрали inference-engine, что включает для себя все нужные библиотеки и файлы. Затем через ssh закидываем файл на arm64 платформу.

**1.3 Model optimizer**

Для конвертации популярных форматов обученных моделей нам потребуется собрать model optimizer.Если вам не требуются модели Mxnet, то шаг 1.3.1 можно пропустить, так как простым способом типа pip на arm64 Mxnet не поставить. А для сборки напрямую на платформе не хватает мощности. Поэтому используем кросс-компиляцию.

**1.3.1 Mxnet**

В уже скачанном вами репозитории есть папка cross\_mxnet.

cd cross\_mxnet/

docker build -f Dockerfile.build.armv8 .

Дождитесь сборки. Все собранные файлы лежат в директори /work/

Затем перекиньте файлы на arm64 платформу через ssh

**2. Что делать с файлами на arm64 платформе**

Сначала скопируйте их в любое удобное место, у меня это была директория Documents/. Затем нужно установить нужные пакеты, для того, чтобы приложения для архитектуры armhf (он же arm32) могли запускаться на aarch64.

dpkg --add-architecture armhf

apt-get update

apt-get install libc6:armhf libstdc++6:armhf

Теперь, чтобы поставить какой-либо пакет для arm32 нам нужно после его названия писать :armhf. Также для нормальной работы OpenVINO с камерой нам понадобиться opencv 4.0, но собирать нам ее уже не надо будет, так как у intel есть Raspi Toolkit – еще один набор с нужными библиотеками (в том числе и opencv 4.0).

**3 Создание окружения для компиляции программ с поддержкой openVINO**

Так как собирать программы прямо на платформе – идея не самая хорошая, так как компиляторы для arm64 не всегда могуть ужиться с компиляторами для arm32. Да, есть способы, типа chroot. Но компилировать прямо на целевой платформе нет смысла из-за соображений производительности. К тому же как правило на релизной версии тулчейнов и копиляторов не содердится из-за соображений экономии памяти. Поэтому будем использовать docker. В скачанном вами репозитории есть папка openVINO\_env. Проделайте следующее:

docker build --rm -f "openVINO\_env/Dockerfile" -t openvino\_env:latest openVINO\_env

Дождитесь сборки. Затем скопируйте архив. К тому же в данном контейнере вы получили окружение, где можно компилировать для openVINO.

**3. ROS**

Собирать будем из исходников

sudo apt-get install python-rosdep python-rosinstall-generator python-wstool python-rosinstall build-essential

sudo rosdep init

rosdep update

**Создадим рабочую директорию**

mkdir ~/ros\_catkin\_ws

cd ~/ros\_catkin\_ws

Ставить будем Bare-bone версию, так как графические пакеты не нужны

rosinstall\_generator ros\_comm --rosdistro kinetic --deps --wet-only --tar > kinetic-ros\_comm-wet.rosinstall

wstool init -j8 src kinetic-ros\_comm-wet.rosinstall

Ставим нужные зависимости

rosdep install --from-paths src --ignore-src --rosdistro kinetic -y

Собираем

./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release

После сборки может быть ошибка при запуске ROS комманд – permission denied. Чтобы это исправить:

sudo chmod -R 777 /home/orangepi/ros\_catkin\_ws/

sudo chmod -R 777 /home/orangepi/.ros/

Для загрузки setup`a:

source ~/ros\_catkin\_ws/install\_isolated/setup.bash