TensorFlowOnACL

User Manual

2017-12-25

Reversion Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Change Description | Author |
| 2017-12-25 | 0.1.0 | Initial | Yuming Cheng  Yu Wang |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**catalog**

[1 Purpose 3](#_Toc502358879)

[2 Terminology 3](#_Toc502358880)

[3 Environment 3](#_Toc502358881)

[3.1 Hardware Platform 3](#_Toc502358882)

[3.2 Software platform 4](#_Toc502358883)

[4 Install Guide 4](#_Toc502358884)

[4.1 Download code 4](#_Toc502358885)

[4.2 Compiled Environment Prepared 4](#_Toc502358886)

[4.2.1 Setup build server 4](#_Toc502358887)

[4.2.2 Install crossbuild tools 5](#_Toc502358888)

[4.2.3 Install bazel (release 0.7.0 or above) 5](#_Toc502358889)

[4.2.4 Setup RK3399 device 5](#_Toc502358890)

[4.2.5 Compile ACL on build server 6](#_Toc502358891)

[4.2.6 Compile Tensorflow Example 6](#_Toc502358892)

[4.3 Run bazel build command 7](#_Toc502358893)

[4.4 Compile Unit test 8](#_Toc502358894)

[4.5 Run Tests 9](#_Toc502358895)

[5 Configuration Guide 10](#_Toc502358896)

[5.1 Enable ACL In Compile Time 10](#_Toc502358897)

# Purpose

This guide help developers to use the code of TensorFlowOnACL (TensorFlow+ACL) to improve the performance of their applications based on the TensorFlow framework.

# Terminology

* **ACL:** Arm Compute Library[Arm Computer library](https://community.arm.com/graphics/b/blog/posts/arm-compute-library-for-computer-vision-and-machine-learning-now-publicly-available)
* **TensorFlow**：A deep learning library or framework. [http://Tensorflow.berkeleyvision.org/](http://caffe.berkeleyvision.org/)
* **TensorFlowOnACL:** optimized Tensorflow on Arm platform with ACL.
* **ACL/GPU**: In the below tables, it is specialized to mean using GPU by Arm Compute Library to test. (Mali: GPU from Arm)
* **ACL/Neon**: In the below tables, it is specialized to mean using Neon by Arm Compute Library to test. (Neon: ARM coprocessor supporting SIMD)
* **NCHW, NHWC**: Tensor format for input/output activations used in convolution operations.

The mnemonics specify the meaning of each tensor dimension sorted from largest to smallest memory stride. N = Batch, H = Image Height, W = Image Width, C = Number of Channels. NHWC is the default format in TensorFlow. NCHW often improves performance on GPUs.

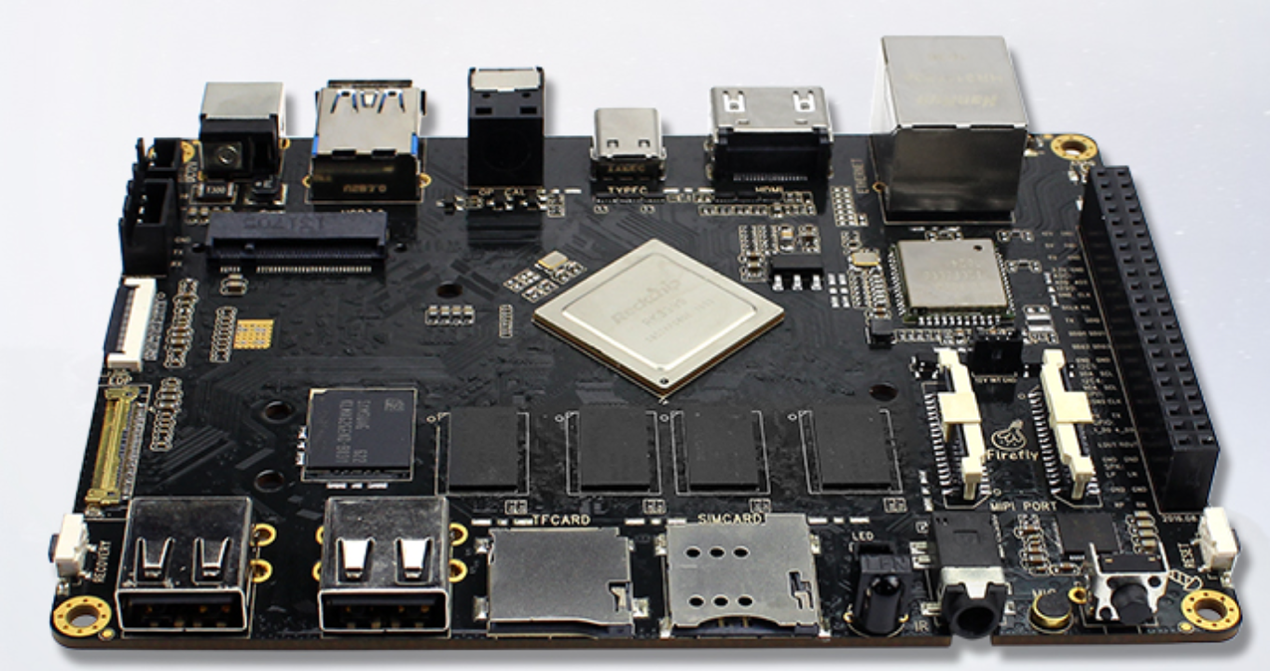
* **OIHW, HWIO**: Tensor format for convolutional filters. The mnemonics specify the meaning of each tensor dimension sorted from largest to smallest memory stride. H = Kernel Height, W = Kernel Width, I = Input Channels, O = Output Channels. HWIO is the default filter format in TensorFlow. OIHW often improves performance on GPUs.
* **1st :** The first test loop**;** In the test applications ”classification\_profiling” and “classification\_ profiling\_gpu” include all the process
* **2nd~11th :** the 2nd to 11th test loops, unlike the first test loop, aren’t guaranteed to use all the allocation and config processes.
* TPI : The total time for per inference

# Environment

## Hardware Platform

SoC : Rockchip RK3399

* GPU : Mali T864 (800MHz)
* CPU : Dual-core Cortex-A72 up to 2.0GHz (real frequency is 1.8GHz); Quad-core Cortex-A53 up to 1.5GHz (real frequency is 1.4GHz)



## Software platform

Operating System : Ubuntu 16.04

# Install Guide

## Download code

git clone <https://github.com/ARM-software/ComputeLibrary.git>

git clone <https://github.com/OAID/TensorflowOnACL.git>

## Compiled Environment Prepared<https://bazel.build/versions/master/docs/install.html>

### Setup build server

1. Update build server to add arm64 architecture and backup apt sources.list.

sudo dpkg --add-architecture arm64

sudo mv /etc/apt/sources.list /etc/apt/sources.list.bak

1. create file tsinghua.list in /etc/apt/sources.list.d/ with following content.

deb [arch=amd64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ xenial main restricted universe multiverse

deb [arch=amd64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ xenial-updates main restricted universe multiverse

deb [arch=amd64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ xenial-security main restricted universe multiverse

deb [arch=amd64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ xenial-backports main restricted universe multiverse

deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports/ xenial main restricted universe multiverse

deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports/ xenial-updates main restricted universe multiverse

deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports/ xenial-security main restricted universe multiverse

deb [arch=arm64] https://mirrors.tuna.tsinghua.edu.cn/ubuntu-ports/ xenial-backports main restricted universe multiverse

### Install crossbuild tools

sudo apt-get update

sudo apt-get install gcc-aarch64-linux-gnu g++-aarch64-linux-gnu scons rsync

sudo apt-get install python-numpy python-dev python-pip python-wheel

sudo apt-get install libpython2.7-dev:arm64

### Install bazel (release 0.7.0 or above)

Tensorflow uses bazel to build. If bazel is not installed on your system, install it now by following guide at <https://docs.bazel.build/versions/master/install.html>.

### Setup RK3399 device

sudo apt-get update

sudo apt-get install rsync

sudo apt-get install python-numpy python-dev python-pip python-wheel

sudo pip install pip enum34 mock

### Compile ACL on build server

cd ~/project/ComputeLibrary

mkdir build

aarch64-linux-gnu-gcc opencl-1.2-stubs/opencl\_stubs.c -Iinclude -shared -o build/libOpenCL.so

scons Werror=1 -j8 debug=0 asserts=1 neon=1 opencl=1 embed\_kernels=1 os=linux arch=arm64-v8a

### Compile Tensorflow Example

1. Update cross build setting for ACL

Modify following 3 lines in “TensorflowOnAcl/tools/aarch64\_compiler/CROSSTOOL” to ACL path in build server.

35: linker\_flag: "-L/home/cym/project/ComputeLibrary/build"

36: linker\_flag: "-L/home/cym/project/ComputeLibrary/build/opencl-1.2-stubs/"

44: cxx\_builtin\_include\_directory: "/home/cym/project/ComputeLibrary"

1. Run Tensorflow configure

cd ~/project/TensorflowOnAcl

./configure

WARNING: Running Bazel server needs to be killed, because the startup options are different.

You have bazel 0.7.0 installed.

Please specify the location of python. [Default is /usr/bin/python]:

Found possible Python library paths:

/usr/local/lib/python2.7/dist-packages

/usr/lib/python2.7/dist-packages

Please input the desired Python library path to use. Default is [/usr/local/lib/python2.7/dist-packages]

Do you wish to build TensorFlow with jemalloc as malloc support? [Y/n]:

jemalloc as malloc support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Google Cloud Platform support? [Y/n]: n

No Google Cloud Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Hadoop File System support? [Y/n]: n

No Hadoop File System support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Amazon S3 File System support? [Y/n]: n

No Amazon S3 File System support will be enabled for TensorFlow.

Do you wish to build TensorFlow with XLA JIT support? [y/N]: n

No XLA JIT support will be enabled for TensorFlow.

Do you wish to build TensorFlow with GDR support? [y/N]: n

No GDR support will be enabled for TensorFlow.

Do you wish to build TensorFlow with VERBS support? [y/N]: n

No VERBS support will be enabled for TensorFlow.

Do you wish to build TensorFlow with OpenCL SYCL support? [y/N]: n

No OpenCL SYCL support will be enabled for TensorFlow.

Do you wish to build TensorFlow with CUDA support? [y/N]: n

No CUDA support will be enabled for TensorFlow.

Do you wish to build TensorFlow with MPI support? [y/N]: n

No MPI support will be enabled for TensorFlow.

Please specify optimization flags to use during compilation when bazel option "--config=opt" is specified [Default is -march=native]:

Add "--config=mkl" to your bazel command to build with MKL support.

Please note that MKL on MacOS or windows is still not supported.

If you would like to use a local MKL instead of downloading, please set the environment variable "TF\_MKL\_ROOT" every time before build.

Configuration finished

## Run bazel build command

NOTE ： needs to change ComputeLibrary path

cd ~/project/TensorflowOnAcl

bazel build -c opt \

--incompatible\_load\_argument\_is\_label=false \

--cxxopt=-fexceptions --copt="-I/home/cym/project/ComputeLibrary/" \

--copt="-I/home/cym/project/ComputeLibrary/include" \

--cpu=aarch64 \

--crosstool\_top=//tools/aarch64\_compiler:toolchain \

--host\_crosstool\_top=@bazel\_tools//tools/cpp:toolchain \

--copt="-DUSE\_ACL=1" --copt="-DTEST\_ACL=1" \

--verbose\_failures //tensorflow/examples/label\_image/...

If there is a following build error :

/home/xxxx/.cache/bazel/\_bazel\_xxxx/xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx/external/nsync/BUILD:402:13: Configurable attribute "copts" doesn't match this configuration (would a default condition help?).

Conditions checked:

@nsync//:android\_arm

...

@nsync//:msvc\_windows\_x86\_64.

Solution:

Open “/home/xxx/.cache/bazel/\_bazel\_xxxx/xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx/external/nsync/BUILD”, Add a default conditions there as marked as red line below:

NSYNC\_OPTS\_GENERIC = select({

# Select the CPU architecture include directory.

# This select() has no real effect in the C++11 build, but satisfies a

# #include that would otherwise need a #if.

":gcc\_linux\_x86\_64\_1": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

":gcc\_linux\_x86\_64\_2": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

":gcc\_linux\_aarch64": ["-I" + pkg\_path\_name() + "/platform/aarch64"],

":gcc\_linux\_ppc64": ["-I" + pkg\_path\_name() + "/platform/ppc64"],

":clang\_macos\_x86\_64": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

":ios\_x86\_64": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

":android\_x86\_32": ["-I" + pkg\_path\_name() + "/platform/x86\_32"],

":android\_x86\_64": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

":android\_armeabi": ["-I" + pkg\_path\_name() + "/platform/arm"],

":android\_arm": ["-I" + pkg\_path\_name() + "/platform/arm"],

":android\_arm64": ["-I" + pkg\_path\_name() + "/platform/aarch64"],

":msvc\_windows\_x86\_64": ["-I" + pkg\_path\_name() + "/platform/x86\_64"],

"//conditions:default": [],

}) + [

## Compile Unit test

Type the following commands :

cd ~/project/TensorflowOnAcl

bazel build -c opt \

--incompatible\_load\_argument\_is\_label=false \

--cxxopt=-fexceptions \

--copt="-I/home/cym/project/ComputeLibrary/" \

--copt="-I/home/cym/project/ComputeLibrary/include" \

--cpu=aarch64 \

--crosstool\_top=//tools/aarch64\_compiler:toolchain \

--host\_crosstool\_top=@bazel\_tools//tools/cpp:toolchain \

--copt="-DUSE\_ACL=1" --copt="-DTEST\_ACL=1" –verbose\_failures \

//tensorflow/python/kernel\_tests:<TEST\_CASE>

TEST\_CASE list:

acl\_conv\_ops\_test

acl\_lrn\_op\_test

acl\_relu\_op\_test

acl\_matmul\_op\_test

acl\_softmax\_op\_test

acl\_cwise\_ops\_test

acl\_pooling\_ops\_test

acl\_softplus\_op\_test

## Run Tests

Run Tensorflow label\_image :

cd ~/project/TensorflowOnAcl

rsync -avL –progress \

bazel-bin/tensorflow/examples/label\_image/label\_image\* \

firefly@192.168.3.211:/home/firefly/

ssh firefly@192.168.3.211

mkdir -p tensorflow/examples/label\_image/data/

curl -L \

"https://storage.googleapis.com/download.tensorflow.org/models/inception\_v3\_2016\_08\_28\_frozen.pb.tar.gz" | \

tar -C tensorflow/examples/label\_image/data -xz

copy “tensorflow/examples/label\_image/data/grace\_hopper.jpg” from TensorflowAcl source code tree and run following command in RK3399.

./label\_image

output message :

2017-12-28 10:18:43.381239: I tensorflow/examples/label\_image/main.cc:250] military uniform (653): 0.734324

2017-12-28 10:18:43.382520: I tensorflow/examples/label\_image/main.cc:250] mortarboard (668): 0.0430416

2017-12-28 10:18:43.382583: I tensorflow/examples/label\_image/main.cc:250] academic gown (401): 0.0220961

2017-12-28 10:18:43.382630: I tensorflow/examples/label\_image/main.cc:250] pickelhaube (716): 0.00787005

2017-12-28 10:18:43.382720: I tensorflow/examples/label\_image/main.cc:250] bow tie (458): 0.00745112

Run Unit test :

cd ~/project/TensorflowOnAcl

rsync -avL --progress \

bazel-bin/tensorflow/python/kernel\_tests/<TEST\_CASE>\* \ 192.168.3.211:/home/firefly/test/

Example:

rsync -avL --progress \

bazel-bin/tensorflow/python/kernel\_tests/acl\_conv\_ops\_test\* \

192.168.3.211:/home/firefly/test/

output message:

./acl\_conv\_ops\_test

testInceptionFwd\_0 [4, 5, 5, 124] [1, 1, 124, 12] [4, 5, 5, 12] 1

testInceptionFwd\_1 [4, 8, 8, 38] [1, 1, 38, 38] [4, 8, 8, 38] 1

testInceptionFwd\_2 [4, 8, 8, 38] [1, 1, 38, 38] [4, 8, 8, 38] 1

...

Testing InceptionFwd %s ([4, 8, 8, 176], [1, 1, 176, 19], 1, 'SAME')

Testing InceptionFwd %s ([4, 8, 8, 176], [1, 1, 176, 19], 1, 'SAME')

......

----------------------------------------------------------------------

Ran 62 tests in 2.735s

OK

# Configuration Guide

## Enable ACL In Compile Time

* Enable ACL functions by “-DUSE\_ACL=1” in bazel build command
* Disable it without define “USE\_ACL” in build command.

The TensorflowOnACL disable ACL by default.