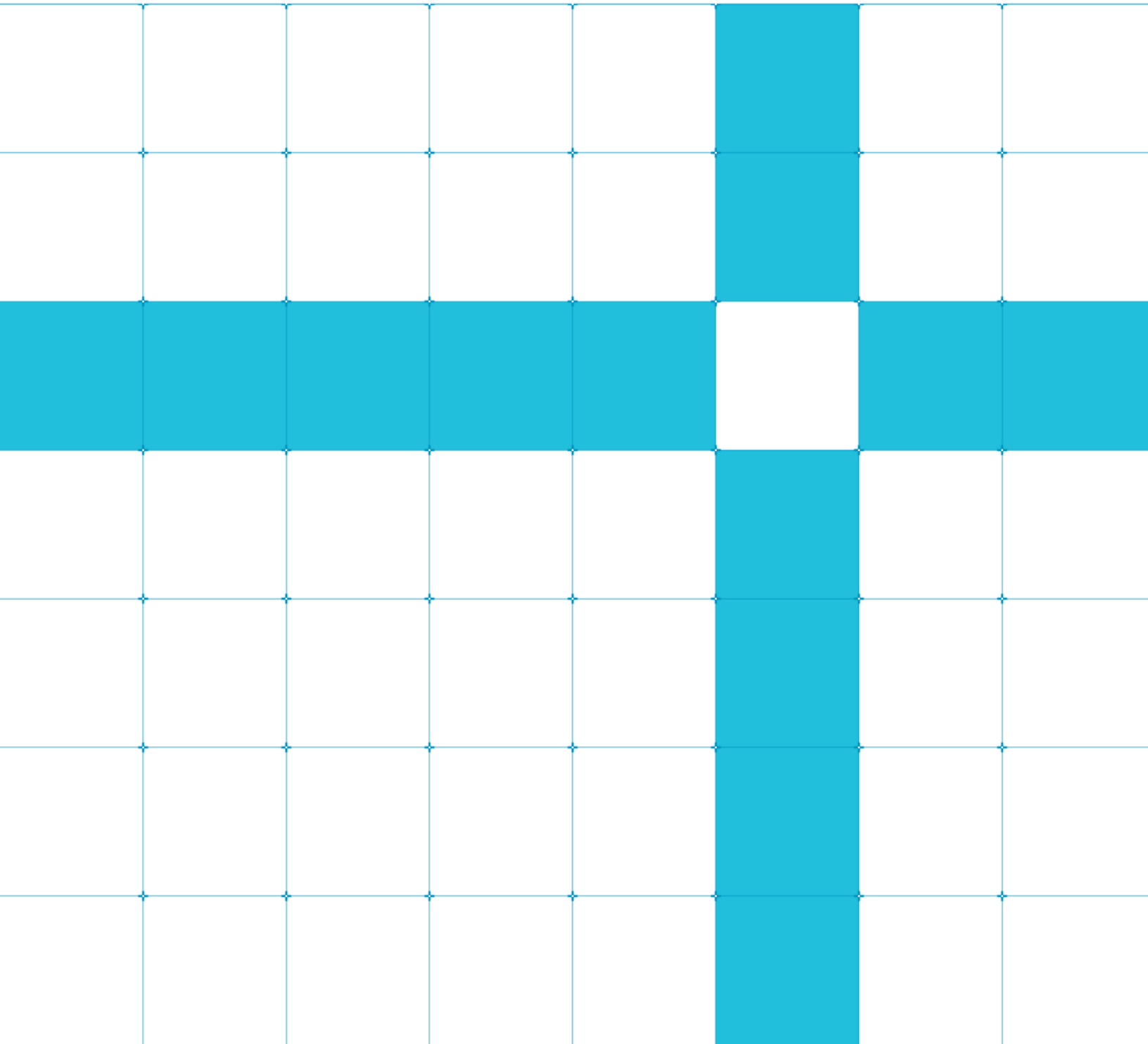




# Configure the Arm NN SDK for TensorFlow

Version 1.7



# Configure the Arm NN SDK build environment for TensorFlow

Copyright © 2018-2020 Arm Limited (or its affiliates). All rights reserved.

## Release Information

### Document History

Version	Date	Confidentiality	Change
1.0	13 April 2018	Non-Confidential	First release.
1.1	03 August 2018	Non-Confidential	Minor enhancements.
1.2	04 September 2018	Non-Confidential	Minor enhancements.
1.3	30 October 2018	Non-Confidential	Minor enhancements.
1.4	11 March 2019	Non-Confidential	Minor enhancements.
1.5	07 June 2019	Non-Confidential	Minor enhancements to the Before you begin section.
1.6	18 November 2019	Non-Confidential	Adds the command for cloning the Tensorflow source code.
1.7	21 February 2020	Non-Confidential	Adds instructions on including standalone dynamic backend tests. Also, updates the git checkout command in the Setting up the build dependencies section.

## Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at <http://www.arm.com/company/policies/trademarks>.

Copyright © 2018-2019 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

## Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

## Product Status

The information in this document is Final, that is for a developed product.

## Web Address

<http://www.arm.com>

# Contents

<b>1 Overview</b> .....	<b>5</b>
<b>2 Before you begin</b> .....	<b>6</b>
<b>3 Downloading the Arm NN SDK and Compute Library files from GitHub</b> .....	<b>7</b>
<b>4 Setting up the build dependencies</b> .....	<b>8</b>
<b>5 Building the environment</b> .....	<b>9</b>
<b>6 Testing the build</b> .....	<b>10</b>
<b>7 Other resources</b> .....	<b>12</b>

# 1 Overview

Before you run code using Arm NN, you must first set up and configure the build environment and the other required tools on your board or platform. You must then test that your build has completed successfully. This guide walks you through that process.

The configuration process is:

1. Download the Arm NN SDK and **Compute Library** files from GitHub.
2. Set up the build dependencies.
3. Build the environment.
4. Test that the build is successful.

## 2 Before you begin

Your platform or board must have:

- An **ARMv7-A** or **ARMv8-A CPU**, and optionally an **Arm Mali GPU** using the OpenCL driver.
- At least 4GB of RAM.
- At least 1GB of free storage space.

Before you configure and build your environment, you must install the following tools on your platform or board:

- A Linux distribution.
- Git.
- SCons. Arm has tested SCons 2.4.1 (Ubuntu) and 2.5.1 (Debian). Other versions might work.
- CMake. Arm has tested CMake 3.5.1 (Ubuntu) and 3.7.2 (Debian). Other versions might work.

These instructions assume that you use Ubuntu 16.04 or Debian 9.0, but should work on most Linux distributions. Arm tests the Arm NN SDK on Ubuntu and Debian.

# 3 Downloading the Arm NN SDK and Compute Library files from GitHub

To download the Arm NN SDK and the Compute Library, you must clone them from their GitHub repositories using Git.

To clone the Arm NN SDK files, open a terminal or bash screen on your platform or board and enter: `git clone https://github.com/Arm-software/armnn`.

To clone the **Compute Library** files, open a terminal or bash screen on your platform or board and enter: `git clone https://github.com/ARM-software/ComputeLibrary`.

Make a note of where you store these Git repositories, because you will need these locations when you build the environment.

# 4 Setting up the build dependencies

The Arm NN SDK relies on some software, which you must set up and install.

To set up the build dependencies:

1. Download and build **Boost**. Arm has tested version 1.64. Some other versions might work. For instructions to do this, see the [Boost getting started guide](#). When you build Boost, include the following flags:  
`link=static cxxflags=-fPIC --with-filesystem --with-test --with-log --with-program_options --prefix=path/to/installation/prefix`
2. Clone and build protobuf. Arm has tested version 3.5.0. Some other versions might work. To clone version 3.5.0, enter:  
`git clone -b v3.5.0 https://github.com/google/protobuf.git`

Build protobuf using the C++ installation instructions that you can find on the [Protobuf GitHub](#).

Make a note of the location of the installation, you will need it during the TensorFlow build process.

2. Clone the TensorFlow source code from GitHub with this command:

```
git clone https://github.com/tensorflow/tensorflow.git
cd tensorflow/
git checkout 590d6eef7e91a6a7392c8ffffb7b58f2e0c8bc6b
```



# 5 Building the environment

To use the Arm NN SDK, you must build the environment on your platform or board.

To build the environment:

1. Compile the Compute Library using SCons. To do this, change your directory to the Compute Library Git repository on your machine, and enter:

```
scons arch=arm64-v8a extra_cxx_flags="-fPIC" benchmark_tests=0 validation_tests=0 for ARMv8-A or
scons extra_cxx_flags="-fPIC" benchmark_tests=0 validation_tests=0 for ARMv7-A.
```

If you want to enable benchmark tests, set `benchmark_tests` to 1. If you want to enable validation tests, set `validation_tests` to 1.

You can enable support for **NEON** on the CPU, and support for **OpenCL** on an Arm Mali GPU if you have one.

If you want to support OpenCL for your Arm Mali GPU, add these arguments to the SCons command:

```
opencl=1 embed_kernels=1
```

If you want to support NEON, add this argument to your SCons command:

```
neon=1
```

2. Change the directory to <the installation root>.
3. Generate C++ sources and headers using the protobuf compiler. To do this, enter:

```
pushd tensorflow
${ARMNN_DIR}/armnn/scripts/generate_tensorflow_protobuf.sh <directory where the generated
files will be placed> <the protobuf install directory>
popd
```

4. Configure the Arm NN SDK build using CMake. To do this, change your directory to the Arm NN directory and enter:

```
mkdir build
cd build
cmake .. -DARMCOMPUTE_ROOT=<the location of your Compute Library source files directory> -
DARMCOMPUTE_BUILD_DIR=<the location of your Compute Library source files directory>/build -
DBOOST_ROOT=<directory used with the prefix flag> -DTF_GENERATED_SOURCES=<the output
directory from step 3> -DBUILD_TF_PARSER=1 -DPROTOBUF_ROOT=<the location of your protobuf
install directory>
```

If you are supporting NEON, add this argument to the CMake command:

```
-DARMCOMPUTENEON=1
```

If you are supporting OpenCL, add this argument to the CMake command:

```
-DARMCOMPUTECL=1
```

If you want to include Arm NN reference support, add this argument to the CMake command:

```
-DARMNNREF=1
```

If you want to include standalone sample dynamic backend tests, add the argument to enable the tests and the dynamic backend path to the CMake command:

```
-DSAMPLE_DYNAMIC_BACKEND=1 -DDYNAMIC_BACKEND_PATHS=<the location of the sample dynamic backend>
```

Also, after building Arm NN, build the standalone sample dynamic backend using the guide in the following path: `$BASEDIR/armnn/src/dynamic/README.md#standalone-dynamic-backend-build`.

5. Enter `make`.

## 6 Testing the build

To check that your build of the Arm NN SDK is working correctly, you can run the Unit Tests. To do this, change to the Arm NN build directory and enter `./UnitTests`.

If the tests are successful, the output from the tests ends with `*** No errors detected`.

If some of the tests are unsuccessful, go back through the steps and check that all the commands have been entered correctly.

Now that you have built your environment, you are ready to start programming.

# 7 Other resources

You might find the following resource helpful:

- [Arm NN SDK TensorFlow supported operators.](#)