# Arm RAN Acceleration Library

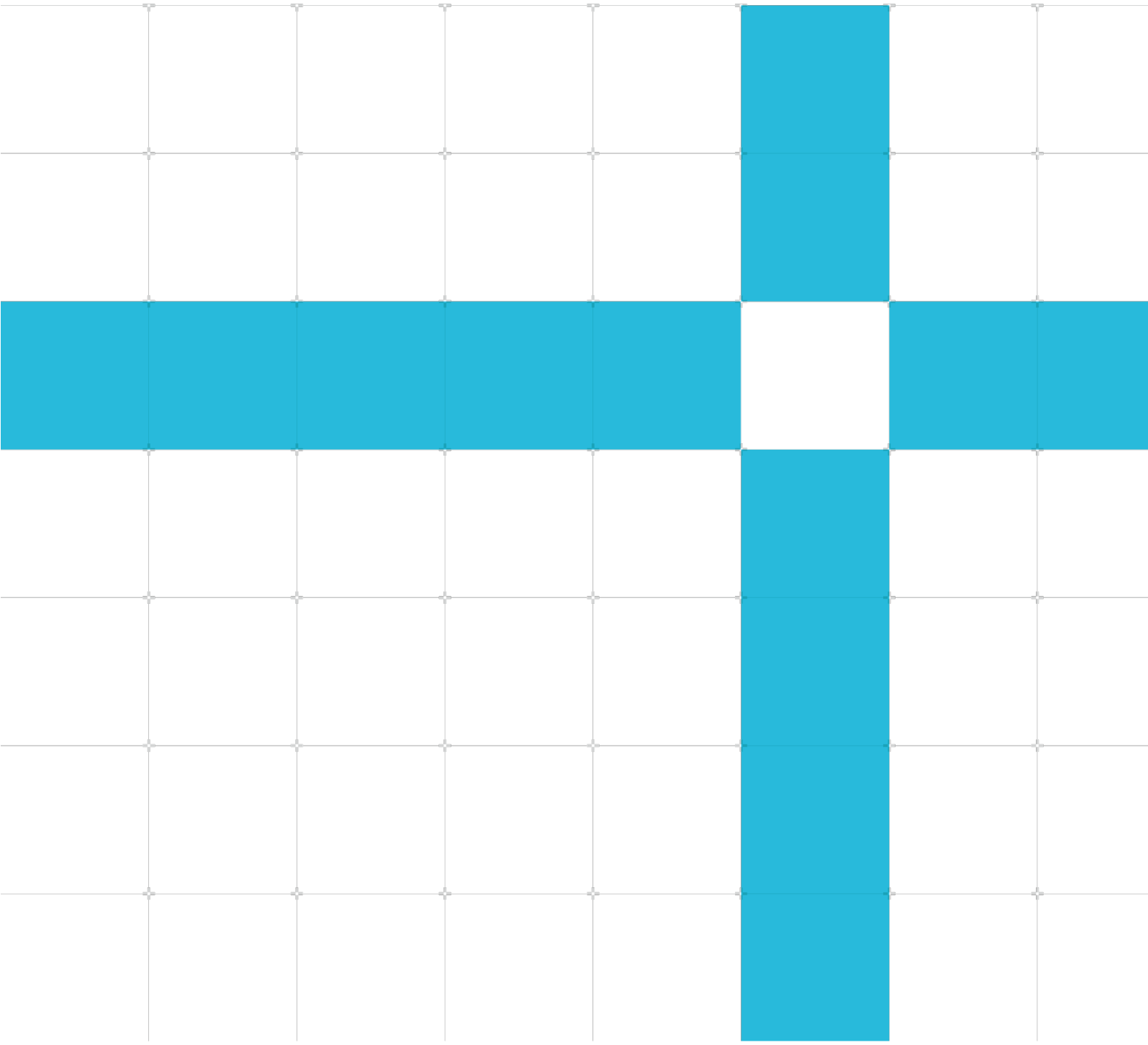
Product revision: 23.07

# Release Note

Non-Confidential

Copyright © 2020, 2023 Arm Limited (or its affiliates).
All rights reserved.

Document ID:
107561



# Arm RAN Acceleration Library

## Release Note

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.

## 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, 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.

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 affiliates) 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
**https://www.arm.com/company/policies/trademarks**.

Copyright © 2020, 2023 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.

## Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on Arm RAN Acceleration Library, create a ticket on **https://support.developer.arm.com**.

To provide feedback on the document, fill the following survey:
**https://developer.arm.com/documentation-feedback-survey**.

## Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

To report offensive language in this document, email **terms@arm.com**.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# Contents



Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# 1 Release overview

The following sections describe the product that this release note describes and its quality status at time of release.

Use of Arm RAN Acceleration Library is subject to a BSD-3-Clause license, the text of which can be found in the `license_terms` folder of your product installation. We will receive inbound contributions under the same license.

## 1.1 Product description

The Arm RAN Acceleration Library (ArmRAL) contains a set of functions for accelerating telecommunications applications such as, but not limited to, 5G Radio Access Networks (RANs).

The Arm RAN Acceleration Library 23.07 package provides a library that is optimized for Arm AArch64-based processors.

Arm RAN Acceleration Library provides:

- Vector functions

- Matrix functions

- Lower PHY support functions

- Upper PHY support functions

- DU-RU Interface support functions

Arm RAN Acceleration Library includes functions that operate on 16-bit signed integers and 32-bit floating-point values.

## 1.2 Release status

This is the 23.07 release of Arm RAN Acceleration Library.

These deliverables are being released under the terms of the agreement between Arm and each licensee (the "Agreement"). All planned verification and validation is complete.

The release is suitable for volume production under the terms of the Agreement.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

## 1.3 Licensing Information

Use of Arm RAN Acceleration Library is subject to a BSD-3-Clause license, the text of which can be found in the `license_terms` folder of your product installation. We will receive inbound contributions under the same license.

If you require a different license than BSD-3-Clause for compatibility with your end product, please get in contact.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# 2 Release contents

Arm RAN Acceleration Library releases contain documentation and source files.

The following sections describe:

- Downloading and unpacking the product.
- The contents of this release.
- Any changes since the previous release.
- Any known issues and limitations that exist at the time of this release.

## 2.1 Downloading and unpacking

You can either clone the source as a git repository from Arm's Gitlab, or you can download Arm RAN Acceleration Library as a tarball of source from the Arm Developer website and then unpack the contents.

To clone the Arm RAN Acceleration Library repository via SSH:

```
git clone git@git.gitlab.arm.com:networking/ral.git
```

To clone the Arm RAN Acceleration Library repository via HTTPS:

```
git clone https://git.gitlab.arm.com/networking/ral.git
```

To download the tarball and unpack the contents:

1. Go to https://developer.arm.com/solutions/infrastructure/developer-resources/5g/ran/download.
2. Complete the form and click Submit. The package downloads.
3. Locate the downloaded .tar.gz file.
4. Copy the .tar.gz file to the directory where these files are to be built.
5. Extract the tar file contents using a tar utility:
   ```
   tar zxvf arm-ran-acceleration-library-23.07-aarch64.tar.gz
   ```

## 2.2 Deliverables

The downloaded product includes the deliverables listed in this section.

- Arm RAN Acceleration Library 23.07

- Release Notes (this document)

- Documentation

Product documentation is available on the Arm Developer website at:

https://developer.arm.com/documentation/102249/2307

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential


Documentation, errata and release notes might change between product releases. For the latest documentation bundle, check the product download page.


Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of this document when used with any other PDF reader.
A suitable PDF reader can be downloaded from Adobe at **http://www.adobe.com**.

## 2.3 Differences from previous release

The following subsections describe differences from the previous release of Arm RAN Acceleration Library.

### 2.3.1 Additions and functionality changes

Describes new features or components added, or any technical changes to features or components, in this release.

- Added function to compute the regularized pseudo-inverse of a single complex 32-bit floating-point matrix (`armral_cmplx_pseudo_inverse_direct_f32`)

- Added function to compute the multiplication of a complex 32-bit floating-point matrix with its conjugate transpose (`armral_cmplx_mat_mult_aah_f32`)

- Added function to compute the complex 32-bit floating-point multiplication of the conjugate transpose of a matrix with a matrix (`armral_cmplx_mat_mult_ahb_f32`)

- Added variants of existing functions which take a pre-allocated buffer rather than performing memory allocations internally. For functions where the buffer size is not easily calculated from the input parameters, helper functions to calculate the required size have been provided

  - Regularized pseudo-inverse of a single complex 32-bit floating-point matrix (`armral_cmplx_pseudo_inverse_direct_f32_noalloc`)
  - Singular value decomposition of a single complex 32-bit floating-point matrix (`armral_svd_cf32_noalloc` and `armral_svd_cf32_noalloc_buffer_size`)
  - Convolutional decoding (`armral_tail_biting_convolutional_decode_block_noalloc` and `armral_tail_biting_convolutional_decode_block_noalloc_buffer_size`)
  - Polar decoding CRC checks (`armral_polar_crc_check_noalloc` and `armral_polar_crc_check_noalloc_buffer_size`)
  - Polar rate recovery (`armral_polar_rate_recovery_noalloc`)

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

- Polar rate matching (`armral_polar_rate_matching_noalloc`)
- Polar CRC attachment (`armral_polar_crc_attachment_noalloc` and `armral_polar_crc_attachment_noalloc_buffer_size`)
- LDPC block encoding (`armral_ldpc_encode_block_noalloc` and `armral_ldpc_encode_block_noalloc_buffer_size`)
- LDPC rate recovery (`armral_ldpc_rate_recovery_noalloc`)
- LDPC rate matching (`armral_ldpc_rate_matching_noalloc`)
- LDPC block decoding (`armral_ldpc_decode_block_noalloc` and `armral_ldpc_decode_block_noalloc_buffer_size`)
- Turbo rate recovery (`armral_turbo_rate_recovery_noalloc` and `armral_turbo_rate_recovery_noalloc_buffer_size`)
- Turbo rate matching (`armral_turbo_rate_matching_noalloc` and `armral_turbo_rate_matching_noalloc_buffer_size`)
- Turbo block encoding (`armral_turbo_encode_block_noalloc`)
- Turbo block decoding (`armral_turbo_decode_block_noalloc` and `armral_turbo_decode_block_noalloc_buffer_size`)

### 2.3.2 Performance improvements

- Added Neon optimized implementations of the following routines:

  - Batched complex 32-bit floating-point matrix-vector multiplication (`armral_cmplx_mat_vec_mult_batch_f32`). The product of the number of matrices and the number of vectors per matrix needs to be a multiple of 12

- Added SVE2 optimized implementations of the following routines:

  - Complex 32-bit floating-point general matrix inverse for matrices of size `2x2`, `3x3` and `4x4` (`armral_cmplx_mat_inverse_f32`)

- Performance improvements for Neon implementations of the following routines:

  - 8-bit block float compression (`armral_block_float_compr_8bit`)
  - Mu Law compression (`armral_mu_law_compr_8bit`, `armral_mu_law_compr_9bit`, and `armral_mu_law_compr_14bit`)

- Performance improvements for SVE2 implementations of the following routines:

  - 9-bit block scaling decompression (`armral_block_scaling_decompr_9bit`)
  - 14-bit block scaling decompression (`armral_block_scaling_decompr_14bit`)
  - Mu Law compression (`armral_mu_law_compr_8bit`, `armral_mu_law_compr_9bit`, and `armral_mu_law_compr_14bit`)

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

- 8-bit and 12-bit block float compression (`armral_block_float_compr_8bit` and `armral_block_float_compr_12bit`)

### 2.3.3 Changes to simulation programs

- Moved the definition of the symbol rate out of the `ebn0_to_snr` routine (`simulation/awgn/awgn.cpp`) so that it is now a parameter that gets passed in by each of the simulation programs

- Updated the `convolutional_awgn` simulation program to use OpenMP (`simulation/convolutional_awgn/convolutional_awgn.cpp`)

- Updated simulation programs to accept a path to write graphs to, instead of auto generating filenames

- Added the maximum number of iterations to the output of the Turbo simulation program (`simulation/turbo_awgn/turbo_error_rate.py`)

- Updated formatting of labels in graph legends

### 2.3.4 Resolved issues

- Removed bandwidth scaling in all simulation programs so that the maximum spectral efficiency doesn't exceed the number of bits per symbol. This means that for coding rates larger than about 1/3 the BER graphs no longer show results that are better than the Shannon limits

- The convolutional decoding algorithm (`armral_tail_biting_convolutional_decode_block`) now returns correct results for input lengths greater than 255

- The test file for convolutional decoding (`test/ConvCoding/decoding/main.cpp`) is updated so that the tests pass as expected for input lengths which are not a multiple of 4

- Neon block float decompression routines (`armral_block_float_decompr_8bit`, `armral_block_float_decompr_9bit`, `armral_block_float_decompr_12bit`, and `armral_block_float_decompr_14bit`) now truncate values before storing rather than rounding them. This means the Neon implementations of these routines now have the same behavior as the SVE implementations

- Neon block scaling decompression routines (`armral_block_scaling_decompr_8bit`, `armral_block_scaling_decompr_9bit`, and `armral_block_scaling_decompr_14bit`) now truncate values before storing rather than rounding them. This means the Neon implementations of these routines now have the same behavior as the SVE implementations

## 2.4 Known limitations

There are no known issues with this release.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# 3 Support

If you have any issues with the installation, content or use of this release, create a ticket on **https://support.developer.arm.com**. Arm will respond as soon as possible.


Support for this release of the product is only provided by Arm to partners who have a current support and maintenance contract for the product.

## 3.1 Tools

The following points list the tools that are required to build or run Arm RAN Acceleration Library:

- A recent version of a C/C++ compiler, such as GCC. Arm RAN Acceleration Library has been tested with GCC 7.5.0, 8.5.0, 9.5.0, 10.4.0, 11.3.0, 12.3.0, and 13.1.0.


If you are cross-compiling, you need a cross-toolchain compiler that targets AArch64. You can download open-source cross-toolchain builds of the GCC compiler on the Arm Developer website:
https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-a/downloads
The variant to use for an AArch64 GNU/Linux target is `aarch64-none-linux-gnu`.

- A recent version of CMake (version 3.3.0, or higher).

In addition to the preceding requirements:

- To run the benchmarks, you must have the Linux utility tool `perf` installed and a recent version of Python 3. Arm RAN Acceleration Library has been tested with Python 3.8.5.
- To build a local version of the documentation, you must have Doxygen installed. Arm RAN Acceleration Library has been tested with Doxygen version 1.8.13.
- To generate code coverage HTML pages, you must have `gcovr` installed. The library has been tested with `gcovr` version 4.2.


Arm RAN Acceleration Library runs on AArch64 cores, however to use the convolutional encoder, CRC, and sequence generator functions, you must run on a core that supports the AArch64 PMULL extension. If your machine supports the PMULL extension, `pmull` is listed under the "Features" list given in the `/proc/cpuinfo` file.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# 4 Release history

A full release history (with release notes) for Arm RAN Acceleration Library is available on the Arm Developer website:

https://developer.arm.com/downloads/-/arm-ran-acceleration-library/previous-releases-of-the-arm-ran-acceleration-library

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

# 5 Conventions

The following subsections describe conventions used in Arm documents.

## 5.1 Glossary

The Arm Glossary is a list of terms that are used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the Arm Glossary for more information: **https://developer.arm.com/glossary**.

Copyright © 2020, 2023 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential