



# Install Arm Compiler for Linux

Version 1.0

## Non-Confidential

Copyright © 2022 Arm Limited (or its affiliates).  
All rights reserved.

## Issue 00

102621\_0100\_00\_en



# Install Arm Compiler for Linux

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.

## Release information

### Document history

Issue	Date	Confidentiality	Change
0100-00	31 May 2022	Non-Confidential	Initial release

## 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.

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 © 2022 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|version 21.0)

## 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 the product, 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.

This document includes language that can be offensive. We will replace this language in a future issue of this document.

To report offensive language in this document, email [terms@arm.com](mailto:terms@arm.com).

# Contents

1. Introduction.....	6
2. Licensing information.....	7
3. Download.....	8
4. Install Arm Compiler for Linux.....	9
5. Configuration.....	12
6. Get started with the tools.....	16
7. Uninstall.....	17
8. Next steps.....	18

# 1. Introduction

This tutorial describes how to download, install, setup your environment, and get started with Arm Compiler for Linux.

The Arm Compiler for Linux package includes the Arm C/C++ Compiler, Arm Fortran Compiler, and Arm Performance Libraries.

Note: Arm Compiler for Linux runs on 64-bit Arm hardware, it is not a cross-compiler. For more information about compatibility, see [Supported Platforms](#).

## 2. Licensing information

Use of Arm Compiler for Linux is subject to the terms and conditions of the applicable End User License Agreement (“EULA”).

You do not require a license to use Arm Compiler for Linux.

## 3. Download

Download the Arm Compiler for Linux package from the [Downloads](#) page.



## 4. Install Arm Compiler for Linux

Follow these instructions to install the Arm Compiler for Linux package:

- [Before you begin](#)
- [Procedure](#)
- [\(Optional\) Enable command line completion](#)

### Before you begin

If any of the following tools are not already installed by your Linux distribution, you must install them before installing Arm Compiler for Linux:

- Python (version 2.7 or later)
- C Libraries:
  - SUSE and RHEL: `glibc-devel`
  - Ubuntu: `libc6-dev`

You must have at least 2 GB of free hard disk space to both download and unpack the Arm Compiler for Linux package. You must also have an additional 6 GB of free space to install the package.

### Procedure

1. Extract the downloaded package:

```
tar -xvf <package_name>.tar
```

Replace `<package_name>` with the full name of the downloaded package.

2. Change to the extracted package directory:

```
cd <package_name>
```

3. Install the package:

- To install the package using the default options (requires superuser permissions), run:

```
sudo ./<package_name>.sh
```

The packages are installed to `/opt/arm/<package_name>`.

- To install the package using a set of options, run:

```
sudo ./<package_name>.sh <options>
```



sudo permissions are not required if you install the package to a location where you have write permissions using `--install-to` (see table below).

Where `<options>` can be one or more of:

Option	Description
<code>-a, --accept</code>	Automatically accept the EULA (the EULA still displays).
<code>-l, --list-packages</code>	List the installer packages
<code>-i, --install-to &lt;install_location&gt;</code>	Install to the given directory, <code>&lt;install_location&gt;</code> .  Use this option if you do not have <code>sudo</code> rights to install to <code>/opt/arm</code> or another central location.
<code>-s, --save-packages-to &lt;install_location&gt;</code>	Extracts the <code>.deb</code> or <code>.rpm</code> files to the given directory, <code>&lt;install_location&gt;</code> .
<code>-f, --force</code>	Force an install attempt to a non-empty directory.
<code>-h, --help</code>	Display this table in the form of a help message in the terminal.

The packages are installed to `<install_dir>/<package_name>`, where `<install_dir>` defaults to `/opt/arm` if not explicitly provided using the `--install-to` option.

- The installer displays the EULA and prompts you to agree to the terms. To agree, type 'yes' at the prompt.



Arm recommends that you rebuild your MPI implementation (Open MPI, MVAPICH, MPICH) after each installation of a new version of Arm Compiler for Linux. This ensures that the Fortran interface incorporates any changes to the `armflang` module format, and that the correct run-time libraries are used.

### (Optional) Enable command line completion



The following information only applies if: a) a users' system does not have the 'bash-autocomplete' package installed, or b) users do not use bash shells, but want to use the command line complete functionality that the `bash-autocomplete.sh` script allows.

If 'bash-completion' is installed, you can load the `'acfl/<package-version>'` environment module and use the command line completion tool immediately.

Arm Compiler for Linux versions 20.2+ include a tool that allows bash terminal users to autocomplete their command line using the Tab key on their keyboard. For compile commands, it allows users to press Tab to auto-complete the `armflang|armclang|armclang++` commands, supported compiler options, and option arguments. For example, to list the supported arguments

for `-ffp-contract=` with `armclang`, users can type the following command line into your terminal and press Tab where indicated by `[TAB]`:

```
armclang -ffp-contract=[TAB]
```

The arguments supported by `-ffp-contract=` return:

```
fast off on
```

Auto-completion relies on the support in a users' shell, and on the completion functions for the compiler (which make use of the `<install-dir>/share/utils/bash-autocomplete.sh` file in the Arm Compiler for Linux package). If, after you have loaded the compiler modulefile, the command line completion is not working, consider the following points:

- To enable bash users to use this functionality, System Administrators should ensure that the 'bash-completion' package is installed and available for all users to use.
- To enable users of other shell types to use this functionality, System Administrators must rewrite the `_clang()` function defined in `<install-dir>/share/utils/bash-autocomplete.sh` using an appropriate syntax for your users' shell, then register the `_clang()` function with the completion system for that shell. To automatically register autocompletion scripts when the '`acfl/<package-version>`' environment module is loaded, System Administrators might also want to modify the '`acfl/<package-version>`' module.

## 5. Configuration

There are two types of configuration you might want to complete before using Arm Compiler for Linux:

- [Environment configuration](#): Dynamically modify your environment on Linux to easily use the tools.
- [\(Optional\) Tool configuration: Arm Compiler for Linux](#): Change some of the default functionality of the tools using configuration files.

For more information, see the [Environment configuration](#) and [Tool configuration](#) sections below.

### Environment configuration

[Environment Modules](#), and the [Lmod Environment Module](#), systems enable you to dynamically modify your user environment on Linux. They are especially useful in managing different versions of software packages. Use the following instructions to configure your Linux environment to use the Arm Compiler for Linux.

#### Before you begin

- Ensure your system has Environment Modules or the Lmod Environment Modules system installed.

To install Environment Modules on Linux, CentOS, and RHEL distributions, install the `environment-modules` package:

```
sudo yum install environment-modules
```

To install Environment Modules on Ubuntu or other systems use `apt-get`:

```
sudo apt-get install environment-modules
```



You might need to open a new shell before attempting to load Environment Modules.

---

To learn how to install the Lmod Environment Modules system, see the [Lmod Environment Modules documentation](#).

## Procedure

---



This procedure assumes that you have installed Arm Compiler for Linux to the default location `/opt/arm`.

---

If you have installed the packages to a different location, replace `/opt/arm` with that location in the following steps:

1. To see which environment modules are available, run:

```
module avail
```



If you do not see any Arm Compiler for Linux modules, add the module for the missing tool using `module add </path/to/install-dir>`, or configure the `MODULEPATH` environment variable to include the installation directory:

---

```
export MODULEPATH=$MODULEPATH:</path/to/modulefile>
```

For a default Arm Compiler for Linux installation, `</path/to/modulefile>` is `/opt/arm/modulefiles`.

2. Load the environment modules:
  - a. Arm C/C++/Fortran Compiler:

```
module load acfl/<package-version>
```

Where '`<package-version>`' is equivalent to '`<major-version>.<minor-version>{.<patch-version>}`'

- b. If you are using Arm Performance Libraries with Arm C/C++/Fortran Compiler, you do not need to load the Arm Performance Libraries modulefile. Instead, when you compile your code, include the '`-armpl`' options.
- 



If you are using Arm Performance Libraries with GCC, you must load both the GCC and the Arm Performance Libraries modulefiles:

---

```
module load gnu/<package-version>  
module load armpl/<package-version>
```

The '`armpl/<package-version>`' will only become available to load after you load the modulefile for the packaged GCC compiler modulefile, '`gnu/<package-version>`'.

3. Check your environment by examining the `PATH` variable. It should contain the appropriate `bin` directories for the tools loaded in the previous step. For example, examining the `PATH` variable for an Arm Compiler for Linux installation gives:

```
echo $PATH
/opt/arm/arm-linux-compiler-<version>_Generic-AArch64_<OS>-<OS-version>_aarch64-
linux/bin:<path/to...
```

4. You can also use the `which` command to check that, for example, the Arm C/C++ Compiler `armclang` command is available:

```
which armclang
/opt/arm/arm-linux-compiler-<version>_Generic-AArch64_<OS>-<OS-version>_aarch64-
linux/bin/armclang
```

For more information about the environment variables and modules provided with Arm's suite of Server and HPC tools, see our [Environment variables reference for Arm Server and HPC tools](#) topic.



Note

- Ensure that you only load the appropriate module for the package you are using. For example, when compiling and using Arm Performance Libraries, do not load both the GCC and Arm C/C++/Fortran Compiler modules. Similarly, do not load multiple modules corresponding to multiple versions of the same packages. In these instances, the most recently loaded module will take priority and will be the environment variables that remain set.
- If in doubt as to which environment modules are loaded, use the `module list` command and look for multiple modules loaded for the same package or purpose. Unload modules as appropriate.
- By default, Arm Compiler for Linux modulefiles are configured to find dependent Arm Compiler for Linux modulefiles at a location that is relative to the installed `'/modulefiles/'` directory. If you move or copy the `'/modulefiles/'` directory to a new location, the modulefile dependency directories (`'/moduledeps/'` and `'/module_globals/'`), must also be moved to maintain their same relative position (to `'/modulefiles/'`). When trying to locate binaries, broken relative links between modulefile locations will cause the modulefiles to fail. To learn more about how to move Arm Compiler for Linux modulefiles, see the [Error moving Arm Compiler for Linux modulefiles](#) section in the [Arm Fortran Compiler](#) or [Arm C/C++ Compiler Developer](#) and Reference guides.

---

## (Optional) Tool configuration: Arm Compiler for Linux

---



Note

This is an optional step. Arm Compiler for Linux does not require you to configure the tools. However, if you want to specify a set of default options to all invocations of the compiler, this step describes how to configure the tools.

---

You can configure Arm Compiler for Linux using a configuration file.

A configuration file allows a set of compiler options to be added to all invocations of the compiler. The configuration file can be copied or modified to suit your needs.

A template configuration file `example.cfg` can be found in `<install-directory>/share/config`. The template also includes more detailed information about the syntax, order of precedence, language-specific options, linker-specific options, and some example options.

## How to use the configuration file

The configuration file can be manually invoked using the `--config` option. For example:

```
[armclang|armclang++|armflang] --config /path/to/this/<filename>.cfg <option>
```

For a more permanent solution, Arm Compiler for Linux supports three environment variables, which can be set to the path of a configuration file. The configuration files provided are used automatically, for each of `armclang`, `armclang++`, or `armflang`. All three variables can be set to the same configuration file. Alternatively, to set language-specific options, create language-specific files:

- For `armclang`: `ARM_COMPILER_CC_CONFIG`
- For `armclang++`: `ARM_COMPILER_CXX_CONFIG`
- For `armflang`: `ARM_COMPILER_FC_CONFIG`

To set these environment variables, open your Arm Compiler for Linux environment modulefile for editing, and add:

- For `armclang`:

```
setenv ARM_COMPILER_CC_CONFIG /path/to/c_options.cfg
```

- For `armclang++`:

```
setenv ARM_COMPILER_CXX_CONFIG /path/to/c_options.cfg
```

- For `armflang`:

```
setenv ARM_COMPILER_FC_CONFIG /path/to/fortran_options.cfg
```

replacing `/path/to/` with the path for your system.

For more information, see the header text in the `example.cfg` template file.

## 6. Get started with the tools

Learn how to compile and link your applications with the Arm Compiler for Linux tools:

<a href="#">Get started with Arm Fortran Compiler</a> - Learn how to compile your Fortran code
<a href="#">Get started with Arm C/C++ Compiler</a> - Learn how to compile your C/C++ code
<a href="#">Get started with Arm Performance Libraries</a> - Learn how to select the optimal library for your system



## 7. Uninstall

Describes how to uninstall the Arm Compiler for Linux package.

For convenience, the Arm Compiler for Linux package includes a script that can be used to uninstall it, `uninstall.sh`. This script is located in:

```
<install-dir>/arm-compiler-for-linux-<release>/uninstall.sh
```

Run this script to uninstall all the packages supplied as part of this suite.

This script attempts to uninstall all the components supplied as part of Arm Compiler for Linux. However, if other packages outside of this product depend on the GCC component, GCC will not be uninstalled.

Notes:

For versions of Arm Compiler for Linux 19.3 or earlier:

- Components depended on by other packages must be manually uninstalled.
- If you use the `--install-to` option provided by the installer, you will need to uninstall the packages by manually removing their directories.

## 8. Next steps

To find out more information about the Arm Compiler for Linux tools, and to find documentation and other resources, see the component web pages:

- [Arm C/C++ Compiler](#)
- [Arm Fortran Compiler](#)
- [Arm Performance Libraries](#)

To learn more about how to use Arm Compiler for Linux to port your applications to Arm-based, or Arm SVE-based, hardware, see our porting guides:

- [Porting and Optimizing HPC Applications for Arm](#)
- [Porting and Optimizing HPC Applications for Arm SVE](#)