**SCONE Installation guide for macOS**

Please make sure you follow the instructions listed below carefully, as these will be uploaded to the SCONE documentation regarding installation for macOS.

1. Make sure you have the latest version of macOS (presently macOS Sequoia 15.0.1; if you do not have this version, please update it in: System Settings → General → Software Update).
2. Install Xcode from the App Store. Xcode contains crucial headers which are read and interpreted when compiling software containing C/C++ languages. Once installed, launch Xcode so that it can complete its initialisation. A dialog will be presented indicating which Simulator runtimes are built-in, and which Simulator runtimes you may download. Choose ‘Continue’ to finish setting up Xcode.
3. Open a new Terminal window. If Terminal is not docked, you may find it by opening a new Finder window, then going to Applications → Utilities.
4. Install Homebrew by typing the following command in your terminal window:

/bin/bash -c "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh>)"

1. Once Homebrew is installed, type the following command in your terminal window. This will install the latest versions of all the packages required to correctly set up and run SCONE:

brew install gcc cmake python git openblas lapack libomp

1. Close your Terminal window. Open a new Finder window and navigate to your Home directory (⌘ + ⇧ + h). Display hidden files (⌘+ ⇧ + .); find the ‘.zprofile’ file, open it and insert any of the following lines which are not already present (note, this depends on whether you have a Mac running on an Intel CPU or an ARM – Apple Silicon – chip):
* Intel:

# Setting PATH for Python 3.13. The original version is saved in .zprofile.pysave.

PATH="/Library/Frameworks/Python.framework/Versions/3.13/bin:${PATH}"

export PATH

# Set shell environment for Homebrew.

eval "$(/usr/local/bin/brew shellenv)"

# Export pFUnit installation folder.

export PFUNIT\_DIR=~/pFUnit/build/

# Export environmental variables required by pFUnit.

export F90=gfortran

export F90\_VENDOR=GNU

# Export OpenMP root path and flags.

export OpenMP\_ROOT=$(brew --prefix)/opt/libomp

export LDFLAGS="-L/usr/local/opt/libomp/lib"

export CPPFLAGS="-I/usr/local/opt/libomp/include"

* ARM:

# Setting PATH for Python 3.13. The original version is saved in .zprofile.pysave.

PATH="/Library/Frameworks/Python.framework/Versions/3.13/bin:${PATH}"

export PATH

# Set shell environment for Homebrew.

eval "$(/opt/homebrew/bin/brew shellenv)"

# Export pFUnit installation folder.

export PFUNIT\_DIR=~/pFUnit/build/

# Export environmental variables required by pFUnit.

export F90=gfortran

export F90\_VENDOR=GNU

# Export OpenMP root path and flags.

export OpenMP\_ROOT=$(brew --prefix)/opt/libomp

export LDFLAGS="-L/opt/homebrew/opt/libomp/lib"

export CPPFLAGS="-I/opt/homebrew/opt/libomp/include"

1. Save the changes you made in your ‘.zprofile’ file and close it. You may now hide hidden files (⌘ + ⇧ + .).
2. Open a new Terminal window. By default it should open in your Home directory, but if not navigate to it by entering:

cd

1. Download the pFUnit repository from Git, enter the source code repository and create a ‘build’ directory by typing the following command:

git clone https://github.com/Goddard-Fortran-Ecosystem/pFUnit.git

cd pFUnit

mkdir build

cd build

1. Before proceeding, **make sure that the default C compiler is Apple Clang** by entering the following command:

gcc --version

If it is not, then you have an alias (symlink) which is pointing to another C compiler. In this case, you have two options:

* Remove the alias, which will default the C compiler back to Apple Clang for all future compilations. To do so, open a new Finder window then open the ‘Go to Folder’ prompt by pressing (⇧ + ⌘ + g) and entering /usr. Navigate to /local/bin, locate the ‘gcc’ alias and delete it. Once this is done, you may revert to your Terminal window and type:

gcc --version

to ensure that the default C compiler is Apple Clang. Now initialise CMake (you should still be in the ‘build’ folder on your Terminal) by typing:

cmake ./..

* Initialise CMake by specifying which C compiler to use. In your Terminal window enter the following:

cmake -D CMAKE\_C\_COMPILER=CLANG ./..

1. Compile tests and install by typing:

make tests

make install

1. Create a new folder (for instance, on your Desktop), name it however you want it (eg, Academic) and navigate to it from your Terminal.
2. Download SCONE from the following test repository, enter the source directory and create a ‘Build’ folder inside it by entering in your Terminal:

git clone https://github.com/CambridgeNuclear/SCONE.git

cd SCONE

mkdir Build

* If your default C compiler is Apple Clang, initialise CMake using the following commands:

cmake -E chdir ./Build cmake ./..

* If not, then run:

cmake -E chdir ./Build cmake -D CMAKE\_C\_COMPILER=CLANG ./..

1. Now compile SCONE by typing the following command:

Make -C Build

1. Once compilation is complete, execute unit and integration tests to verify that they all pass by running:

./Build/unitTests

./Build/integrationTests