



SIESTA: building, deployment and execution

Dr. José María Escartín Esteban

15th November 2024

SIESTA was built and deployed on MareNostrum 5 for the School

```
$ ssh nct01300@glogin1.bsc.es
...
[nct01300@glogin1 ~]$ source /gpfs/projects/nct_315/SIESTA/siestarc.sh
...
[nct01300@glogin1 ~]$ siesta -v
Executable      : siesta
Version         : 5.2.0-dirty
Architecture    : x86_64
Compiler version: Intel-2021.10.0.20230609
Compiler flags  : -O2 -ip -xHost -fp-model=strict -prec-div -prec-sqrt
Parallelisations: MPI, OpenMP
* OpenMP version: 201611
NetCDF support
NetCDF-4 support
NetCDF-4 MPI-IO support
Lua support
Native ELPA support
ELSI support. Solvers:
  ELPA (internal)
  NTPoly
  OMM
  PEXSI
DFT-D3 support
Wannier90 wrapper support
```

Next week: real life

```
$ ssh nct01300@glogin1.bsc.es
nct01300@glogin1.bsc.es's password:
Permission denied, please try again.
nct01300@glogin1.bsc.es's password:
Permission denied, please try again.
nct01300@glogin1.bsc.es's password:
nct01300@glogin1.bsc.es: Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
```

```
$ source /gpfs/projects/nct_315/SIESTA/siestarc.sh
bash: /gpfs/projects/nct_315/SIESTA/siestarc.sh: No such file or directory
```

```
$ siesta -v
bash: siesta: command not found
```



Question 0:

How do I get to run SIESTA on a computer?

The full seven steps

- 1 **Check** the hardware requirements.
- 2 **Prepare** the build environment.
- 3 **Download** the SIESTA source code.
- 4 **Build** SIESTA, its utilities, and any needed dependencies.
- 5 **Test** the binaries.
- 6 **Deploy** the files you built (binaries, libraries, etc.).
- 7 **Run** the executables in the correct environment.

Question 1:

Can I run SIESTA on any computer?

Hardware requirements

- SIESTA can work on a broad variety of computer architectures, from a Raspberry PI to massively parallel supercomputers.
- Recommended minimum requirements:
 - 1 GB RAM (although small atomic structures can be studied with less).
 - 2 GB disk storage, in particular if you want to run the SIESTA test suite.
- SIESTA will likely work on any CPU, but bear in mind that most development and testing is done on x86_64 (Intel/AMD) and ARMv8 architectures.

Question 2:

What software do I need to build SIESTA?

Building software requirements

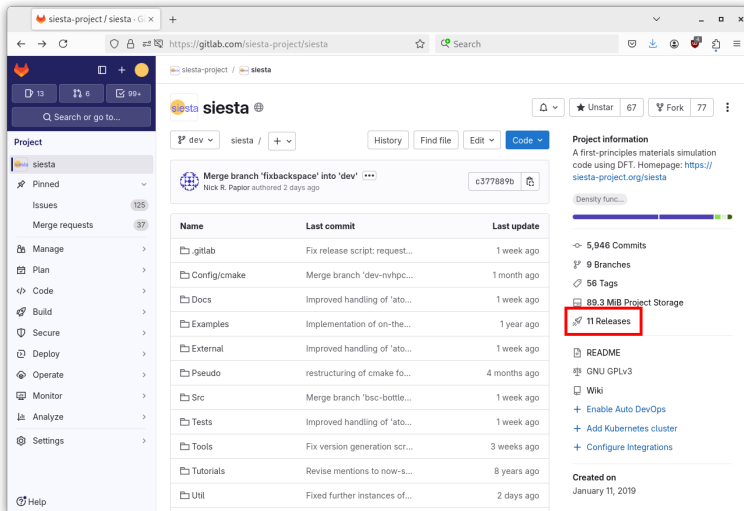
- CMake version ≥ 3.20 (released in March 2021).
- A Fortran compiler with full support of the Fortran 2003 standard and partial support of the Fortran 2008 standard (gfortran, ifort, CRAY Fortran, etc.).
Note: mixing old compilers with new hardware is usually a bad idea.
- A C/C++ companion compiler.
- If you want to build a parallel version of SIESTA, you will also need a MPI distribution (including development files).
- If you want to offload to a Nvidia (/AMD) GPU, you will need the corresponding CUDA (/HIP) distribution.
- If you want to build the SIESTA user manual, you will also need a $\text{\LaTeX}$ distribution.
- If you want to automatically download the source code of dependencies, a non-antique version of git.

Question 3:

Where can I find the authoritative source code of SIESTA?

GitLab: SIESTA

<https://gitlab.com/siesta-project/siesta/>



The screenshot shows the GitLab web interface for the `siesta-project / siesta` repository. The left sidebar contains navigation links for Project, Pinned, Issues (125), Merge requests (37), Manage, Plan, Code, Build, Secure, Deploy, Operate, Monitor, Analyze, Settings, and Help. The main content area displays the repository name, a merge button for branch 'fixbackspace' into 'dev', and a table of files and their commit history. The right sidebar provides project information, including commit count (5,946), branches (9), tags (56), project storage (89.3 MiB), and releases (11, which is highlighted with a red box). It also includes links to README, Wiki, and DevOps configurations.

Project information

A first-principles materials simulation code using DFT. Homepage: <https://siesta-project.org/siesta>

Density func...

5,946 Commits

9 Branches

56 Tags

89.3 MiB Project Storage

11 Releases

README

GNU GPLv3

Wiki

[+ Enable Auto DevOps](#)

[+ Add Kubernetes cluster](#)

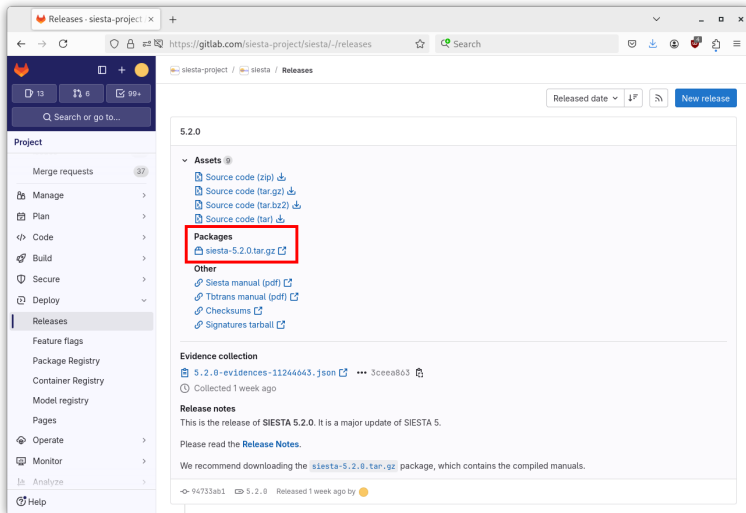
[+ Configure Integrations](#)

Created on
January 11, 2019

Name	Last commit	Last update
.gitlab	Fix release script: request...	1 week ago
Config/cmake	Merge branch 'dev-nvhpc...	1 month ago
Docs	Improved handling of 'ato...	1 week ago
Examples	Implementation of on-the...	1 year ago
External	Improved handling of 'ato...	1 week ago
Pseudo	restructuring of cmake fo...	4 months ago
Src	Merge branch 'bsc-bottle...	1 week ago
Tests	Improved handling of 'ato...	1 week ago
Tools	Fix version generation scr...	3 weeks ago
Tutorials	Revise mentions to now-s...	8 years ago
Util	Fixed further instances of...	2 days ago

GitLab: SIESTA releases

`https://gitlab.com/siesta-project/siesta/-/releases/`



Which version of SIESTA should I use?

We strongly recommend that you use the current **stable release** of SIESTA.



The screenshot shows a web browser window with the address bar displaying `https://siesta-project.org/siesta/CodeAccess/`. The page features the SIESTA logo (an orange circle with the word 'siesta' in blue) and a navigation menu with links: Home, What is SIESTA?, Getting the code, Documentation, Support, News, Events, The Team, and For Developers. The main heading is 'Getting SIESTA'. Below it, a paragraph states: 'The current stable release of SIESTA is **SIESTA 5.2.0**, the first major revision of SIESTA 5.0'. The text 'SIESTA 5.2.0' is highlighted with a red rectangular box. Following this, there are three sections: 'Downloading and installing from source code' with a link to 'GitLab releases page', 'SIESTA binary installation' with a link to 'Conda forge', and 'SIESTA License'.

Getting SIESTA

Home What is SIESTA? Getting the code Documentation Support News Events The Team For Developers

Getting SIESTA

The current stable release of SIESTA is **SIESTA 5.2.0**, the first major revision of SIESTA 5.0

Downloading and installing from source code

You can download SIESTA releases from the [GitLab releases page](#).

Information on how to install SIESTA is available on [the installation section of the documentation site](#).

SIESTA binary installation

Compiled versions of SIESTA are readily available for download from [Conda forge](#), see [Installing Siesta with conda](#) for instructions.

SIESTA License

Should I use SIESTA stable even if I am an advanced user?

Advanced users that want to try arbitrary branches or versions of SIESTA should really avoid the GitLab download button, and interact with the repository using git (git version ≥ 2.13).

These non-release versions are generally **unsupported** by the SIESTA development team.

Question 4:
How do I build SIESTA?

Building SIESTA 5

Download tarball:

```
$ wget https://gitlab.com/siesta-project/siesta/-/releases/5.2.0/downloads/siesta-5.2.0.tar.gz
```

Extract files:

```
$ tar -xvzf siesta-5.2.0.tar.gz
```

Enter source directory:

```
$ cd siesta-5.2.0
```

Initialize build directory:

```
$ cmake -S. -B_build
```

Build:

```
$ cmake --build _build -j 4
```


Basic building options

- Specify Fortran compiler

```
FC=gfortran cmake ...
```

- Specify Fortran compiler flags

```
cmake -DFortran_FLAGS='-O3 -march=native'
```

- Explicitly enable/disable MPI (default: ON if MPI compiler found, otherwise OFF):

```
cmake ... -DSIESTA_WITH_MPI=ON|OFF
```

- Explicitly enable/disable OpenMP (default: OFF):

```
cmake ... -DSIESTA_WITH_OPENMP=ON|OFF
```

- Specify toolchain file (some samples in Config/cmake/toolchains/, but this directory will probably be superseded soon by the Build Tools repository

<https://gitlab.com/siesta-project/ecosystem/build-tools>):

```
cmake ... -DSIESTA_TOOLCHAIN=/path/to/toolchain/file
```

Check the **SIESTA manual** for details about all the building options.

Question 5:

How do I test my SIESTA executable?

Testing SIESTA

- The `ctest` command provided by CMake allows to test SIESTA and its dependencies. See `Tests/README` for more details.
- List (`-N`) all the execution tests [i.e., excluding (`-E`) the verification tests]:

```
$ cd _build
$ ctest -N -E verify
Test project /home/jme52/SIESTA/siesta-5.2.0/_build
Test #1: fdf-sample-test
Test #2: fdf-units-test
Test #3: fdf-ambiguous-units-test
Test #4: XMLF90_SAX_Features
...
Test #27: siesta-00.BasisSets-charge_confinement
Test #28: siesta-00.BasisSets-ghost_atom
Test #29: siesta-01.PseudoPotentials-psf
Test #30: siesta-01.PseudoPotentials-full.psml
...
```

Total Tests: 147

- A less comprehensive set of tests can be chosen by selecting (`-L`) the simple ones:
`$ ctest -N -L simple -E verify`

Testing SIESTA

Expected output when you execute the tests:

```
$ ctest -L simple -E verify
Test project /home/jme52/SIESTA/siesta-5.2.0/_build
   Start  21: siesta-00.BasisSets-default_basis
1/27 Test  #21: siesta-00.BasisSets-default_basis ..... Passed    1.32 sec
   Start  29: siesta-01.PseudoPotentials-psf
2/27 Test  #29: siesta-01.PseudoPotentials-psf ..... Passed    0.78 sec
   Start  30: siesta-01.PseudoPotentials-full.psml
3/27 Test  #30: siesta-01.PseudoPotentials-full.psml ..... Passed    0.78 sec
   Start  44: siesta-03.SpinOrbit-FePt-onsite
4/27 Test  #44: siesta-03.SpinOrbit-FePt-onsite ..... Passed    2.84 sec
   Start  50: siesta-05.Bands-ge_bands
5/27 Test  #50: siesta-05.Bands-ge_bands ..... Passed    0.85 sec
...
100% tests passed, 0 tests failed out of 27
```

Label Time Summary:

```
simple      = 49.67 sec*proc (27 tests)
```

```
Total Test time (real) = 49.70 sec
```

Question 6:
How do I deploy SIESTA?

Tell cmake where to install SIESTA, and install it there:

```
$ cmake -S. -B_build -DCMAKE_INSTALL_PREFIX=/path/to/installation
$ cmake --build _build -j 4
$ cmake --install _build
$ ls /path/to/installation
    bin  include  lib64  share
```

Then make your environment aware of this installation:

```
$ cat siestarc.sh
#!/bin/sh
```

```
LD_LIBRARY_PATH="/path/to/installation/lib:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH
```

```
PATH="/path/to/installation/bin:$PATH"
export PATH
```

Question 7:
How do I execute SIESTA?

Execution of SIESTA

Serial:

```
$ siesta < input.fdf > output.txt
```

MPI:

```
$ mpirun -np 12 siesta < input.fdf > output.txt
```

OpenMP:

```
$ OMP_NUM_THREADS=4 mpirun -np 12 siesta < input.fdf > output.txt
```

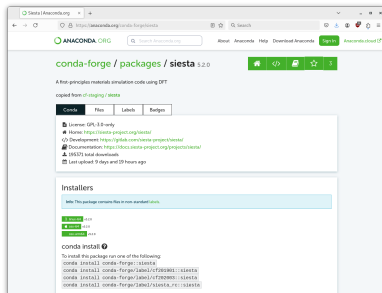
Read the documentation of your HPC facility and your MPI manual for advanced options (pinning, etc.)

Question 8:

Check, prepare, download, build, test, deploy, and run: can I take any shortcuts?

Conda

- Conda is a package manager.
- Conda-Forge is community repository of recipes and packages.
- SIESTA available on Conda-forge: <https://anaconda.org/conda-forge/siesta>



```
$ conda install -c conda-forge siesta=5.2.0=*openmpi*
```

- Strength: easy access to SIESTA: somebody else built the package, so you can directly deploy.
- Weakness: a Conda package may run on a range of CPUs $\implies$ package not optimized for your particular instruction set $\implies$ SIESTA not as performant as it could be.

If you are running SIESTA on a HPC facility, check first if SIESTA is already installed!

We are working on SIESTA modules for all EuroHPC partitions.

Question 9:

Are there other methods or frameworks to build and/or deploy?

- spack: Recipes in the Build Tools repository:
<https://gitlab.com/siesta-project/ecosystem/build-tools>
- EasyBuild: WIP for SIESTA 5.

Thank you for your attention.