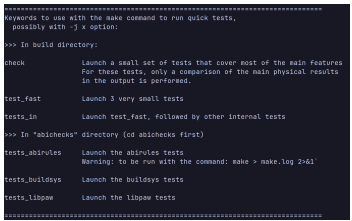
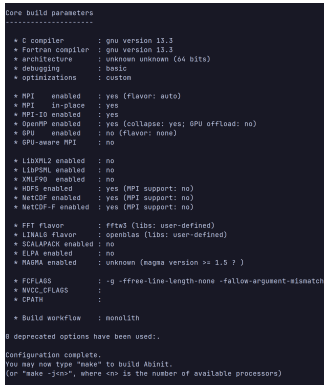
Compiling and installing Abinit

Fabien BRIEUC

Abinit School 2026

2 Février 2026





Introduction

How to compile ABINIT on a personal computer or for high-performance computing

Introduction

How to compile ABINIT on a personal computer or for high-performance computing

Installing ABINIT on..

-  Linux.. ✅
-  MacOS.. ✅
-  Windows.. ❌

Introduction

How to compile ABINIT on a personal computer or for high-performance computing

Installing ABINIT on..

-  Linux.. ✅
-  MacOS.. ✅
-  Windows.. ❌

Pre-compiled packages are available for:

-  Ubuntu using apt

```
> sudo apt install abinit
```

-  MacOS using Homebrew (or MacPorts)

```
> brew tap abinit/tap
> brew install abinit
```

Introduction

How to compile ABINIT on a personal computer or for high-performance computing

Installing ABINIT on..

-  Linux.. ✅
-  MacOS.. ✅
-  Windows.. ❌

Pre-compiled packages are available for:

-  Ubuntu using apt

```
> sudo apt install abinit
```

-  MacOS using Homebrew (or MacPorts)

```
> brew tap abinit/tap
> brew install abinit
```

Compile ABINIT from sources

- Install the required libraries
- Get the ABINIT source code
- Compile ABINIT
- Check the compilation

All the information presented here is available (with more details) in the documentation:
https://docs.abinit.org/installation/

Prerequisites – Compilers and required external libraries

1. C & Fortran compiler (mandatory)
   GNU **gcc**/**gfortran**, intel, nvidia nvhpc..
2. Python interpreter (mandatory)
3. MPI libraries (recommended)
   openmpi, mpich..
4. Input/Output libraries
   - HDF5 (mandatory)
     MPI version required for parallel IO
   - NetCDF + NetCDF-Fortran (mandatory)
5. Linear Algebra libraries
   Multi-threaded versions are recommended
   - BLAS + LAPACK (mandatory)
     openblas, netlib, mkl
6. FFT library (mandatory)
   FFTW, mkl..
7. Exchange-correlation library LibXC (mandatory)

Ubuntu using apt

with openmpi, openblas and fftw3

```
> sudo apt install build-essential gfortran

> sudo apt install python3-dev

> sudo apt install mpi-default-dev libopenmpi-dev

> sudo apt install libhdf5-dev

> sudo apt install libnetcdf-dev libnetcdff-dev

> sudo apt install libopenblas-dev \
libopenblas-openmp-dev

> sudo apt install libfftw3-dev

> sudo apt install libxc-dev
```

Compiling ABINIT: basic steps (using autotools)

1. Download the latest release version on github at
   https://github.com/abinit/abinit/releases/tag/10.4.7

```
> wget https://forge.abinit.org/abinit-10.4.7.tar.gz
> tar -zxf abinit-10.4.7.tar.gz
> cd abinit-10.4.7
```

2. Use the **configure** script to generate *makefiles*
   Use prefix to indicate a specific path for installation

```
> ./configure --prefix=$HOME/abinit
```

3. Compile by running **make**
   Use the option -j <nproc> to compile in parallel on <nproc> processes

```
> make -j
```

4. Install in the directory specified by prefix [optional]

```
> make install
```

Compiling ABINIT: taking control (using autotools)

Everything is set up at the configure step

Example where the mandatory LibXC library was not found

```
+----------------+--------+--------+--------+--------+
|Feature         |Enabled |Init    |Working |Fallback|
+----------------+--------+--------+--------+--------+
|abinit_common   |no      |def     |unknown |no      |
|bigdft          |no      |def     |unknown |disabled|
|cuda            |no      |def     |        |no      |
|fft             |yes     |def     |yes     |no      |
|fftw3           |no      |def     |unknown |no      |
|gpu             |no      |def     |no      |no      |
|hdf5            |yes     |def     |yes     |disabled|
|kokkos          |no      |def     |unknown |no      |
|levmar          |no      |def     |unknown |no      |
|libpaw          |no      |def     |unknown |no      |
|libpsml         |no      |def     |unknown |disabled|
|libxc           |yes     |def     |no      |disabled|
|libxml2         |no      |def     |unknown |no      |
|linalg          |yes     |def     |yes     |disabled|
|mpi             |yes     |def     |yes     |no      |
|netcdf          |yes     |def     |yes     |disabled|
|netcdf_fortran  |yes     |pkg     |yes     |disabled|
|papi            |no      |def     |unknown |no      |
|pfft            |no      |def     |unknown |no      |
|rocm            |no      |def     |        |no      |
|triqs           |no      |def     |unknown |no      |
|wannier90       |no      |def     |unknown |disabled|
|xmlf90          |no      |def     |unknown |disabled|
|yaml            |no      |def     |unknown |no      |
+----------------+--------+--------+--------+--------+


+-----------------------------------------------------------------------+
| WARNING : LibXC is not WORKING !                                      |
+-----------------------------------------------------------------------+
| Please specify the installation directory of libXC                    |
+-----------------------------------------------------------------------+



+-----------------------------------------------------------------------+
| Problems must be solved before continuing                             |
+-----------------------------------------------------------------------+
```

Compiling ABINIT: taking control (using autotools)

Everything is set up at the configure step

Example where the mandatory LibXC library was not found

Solution: Explicitly give the location of the libxc library using

```
> ./configure --prefix=$HOME/abinit --with-libxc=$HOME/lib/libxc-7.0
```

and add the location in the `LD_LIBRARY_PATH` environment variable.

```
+-----------------+--------+--------+--------+--------+
|Feature          |Enabled |Init    |Working |Fallback|
+-----------------+--------+--------+--------+--------+
|abinit_common    |no      |def     |unknown |no      |
|bigdft           |no      |def     |unknown |disabled|
|cuda             |no      |def     |        |no      |
|fft              |yes     |def     |yes     |no      |
|fftw3            |no      |def     |unknown |no      |
|gpu              |no      |def     |no      |no      |
|hdf5             |yes     |def     |yes     |disabled|
|kokkos           |no      |def     |unknown |no      |
|levmar           |no      |def     |unknown |no      |
|libpaw           |no      |def     |unknown |no      |
|libpsml          |no      |def     |unknown |disabled|
|libxc            |yes     |def     |no      |disabled|
|libxml2          |no      |def     |unknown |no      |
|linalg           |yes     |def     |yes     |disabled|
|mpi              |yes     |def     |yes     |no      |
|netcdf           |yes     |def     |yes     |disabled|
|netcdf_fortran   |yes     |pkg     |yes     |disabled|
|papi             |no      |def     |unknown |no      |
|pfft             |no      |def     |unknown |no      |
|rocm             |no      |def     |        |no      |
|triqs            |no      |def     |unknown |no      |
|wannier90        |no      |def     |unknown |disabled|
|xmlf90           |no      |def     |unknown |disabled|
|yaml             |no      |def     |unknown |no      |
+-----------------+--------+--------+--------+--------+


+----------------------------------------------------------------------+
| WARNING : LibXC is not WORKING !                                     |
+----------------------------------------------------------------------+
| Please specify the installation directory of libXC                   |
+----------------------------------------------------------------------+



+----------------------------------------------------------------------+
| Problems must be solved before continuing                            |
+----------------------------------------------------------------------+
```

Compiling ABINIT: taking control (using autotools)

Everything is set up at the configure step

Now the configure step is able to finish..

..but you should still check the output

```
Core build parameters
---------------------

 * C compiler        : gnu version 13.3
 * Fortran compiler  : gnu version 13.3
 * architecture      : unknown unknown (64 bits)
 * debugging         : basic
 * optimizations     : custom

 * MPI    enabled    : yes (flavor: auto)
 * MPI    in-place   : yes
 * MPI-IO enabled    : yes
 * OpenMP enabled    : no (collapse: ignored; GPU offload: ignored)
 * GPU    enabled    : no (flavor: none)
 * GPU-aware MPI     : no

 * LibXML2 enabled   : no
 * LibPSML enabled   : no
 * XMLF90  enabled   : no
 * HDF5 enabled      : yes (MPI support: no)
 * NetCDF enabled    : yes (MPI support: no)
 * NetCDF-F enabled  : yes (MPI support: no)

 * FFT flavor        : fftw3 (libs: user-defined)
 * LINALG flavor     : openblas (libs: user-defined)
 * SCALAPACK enabled : no
 * ELPA enabled      : no
 * MAGMA enabled     : unknown (magma version >= 1.5 ? )

 * FCFLAGS           : -g -ffree-line-length-none -fallow-argument-mismatch

 * NVCC_CFLAGS       :
 * CPATH             :

 * Build workflow    : monolith

0 deprecated options have been used:.

Configuration complete.
You may now type "make" to build Abinit.
(or "make -j<n>", where <n> is the number of available processors)
```

Compiling ABINIT: taking control (using autotools)

Everything is set up at the configure step

Now the configure step is able to finish..
..but you should still check the output

OpenMP (shared memory parallelism) is not enabled !

Solution: Tell the configure script to activate openmp

```
> ./configure --prefix=$HOME/abinit --with-libxc=$HOME/lib/libxc-7.0 \
--enable-openmp
```

```
Core build parameters
---------------------

  * C compiler       : gnu version 13.3
  * Fortran compiler  : gnu version 13.3
  * architecture      : unknown unknown (64 bits)
  * debugging         : basic
  * optimizations     : custom

  * MPI    enabled    : yes (flavor: auto)
  * MPI    in-place   : yes
  * MPI-IO enabled    : yes
  * OpenMP enabled    : yes (collapse: yes; GPU offload: no)
  * GPU    enabled    : no (flavor: none)
  * GPU-aware MPI     : no

  * LibXML2 enabled   : no
  * LibPSML enabled   : no
  * XMLF90  enabled   : no
  * HDF5 enabled      : yes (MPI support: no)
  * NetCDF enabled    : yes (MPI support: no)
  * NetCDF-F enabled  : yes (MPI support: no)

  * FFT flavor        : fftw3 (libs: user-defined)
  * LINALG flavor     : openblas (libs: user-defined)
  * SCALAPACK enabled : no
  * ELPA enabled      : no
  * MAGMA enabled     : unknown (magma version >= 1.5 ? )

  * FCFLAGS           : -g -ffree-line-length-none -fallow-argument-mismatch
  * NVCC_CFLAGS       :
  * CPATH             :

  * Build workflow    : monolith

0 deprecated options have been used:.

Configuration complete.
You may now type "make" to build Abinit.
(or "make -j<n>", where <n> is the number of available processors)
```

Compiling ABINIT: taking control (using autotools)

Everything is set up at the configure step

Now the configure step is able to finish..
..but you should still check the output

OpenMP (shared memory parallelism) is not enabled !

Solution: Tell the configure script to activate openmp

```
> ./configure --prefix=$HOME/abinit --with-libxc=$HOME/lib/libxc-7.0 \
--enable-openmp
```

The configure script can take a lot of options and it is usually easier to use a configuration file and simply type

```
> ./configure --with-config-file=<config-filename.ac9>
```

Examples of such files can be found in the ABINIT sources in `doc/build/config-examples`

```
Core build parameters
---------------------

  * C compiler       : gnu version 13.3
  * Fortran compiler  : gnu version 13.3
  * architecture     : unknown unknown (64 bits)
  * debugging        : basic
  * optimizations    : custom

  * MPI    enabled   : yes (flavor: auto)
  * MPI    in-place  : yes
  * MPI-IO enabled   : yes
  * OpenMP enabled   : yes (collapse: yes; GPU offload: no)
  * GPU    enabled   : no (flavor: none)
  * GPU-aware MPI    : no

  * LibXML2 enabled  : no
  * LibPSML enabled  : no
  * XMLF90  enabled  : no
  * HDF5 enabled     : yes (MPI support: no)
  * NetCDF enabled   : yes (MPI support: no)
  * NetCDF-F enabled : yes (MPI support: no)

  * FFT flavor       : fftw3 (libs: user-defined)
  * LINALG flavor    : openblas (libs: user-defined)
  * SCALAPACK enabled : no
  * ELPA enabled     : no
  * MAGMA enabled    : unknown (magma version >= 1.5 ? )

  * FCFLAGS          : -g -ffree-line-length-none -fallow-argument-mismatch
  * NVCC_CFLAGS      :
  * CPATH            :

  * Build workflow   : monolith

0 deprecated options have been used:.

Configuration complete.
You may now type "make" to build Abinit.
(or "make -j<n>", where <n> is the number of available processors)
```

Compiling ABINIT: taking control (using autotools)

Example of a standard ac9 configuration file

- Set up compilers and optimization flags
  mpicc and mpif90 for parallel build
- Enable parallelism
  MPI and openMP
- Define linear algebra libraries
  BLAS and LAPACK
- Define Fast Fourier Transform (FFT) libraries
- Define LibXC library
- Define IO libraries
  HDF5, netCDF, netCDF-Fortran

Hands-on: Write your own ac9 file and compile ABINIT

```
# Configuration file for Abinit
# using openmpi, openblas, fftw3
# tested on ubuntu 24.04 and abinit 10.4.7

# Libraries and include directories in ubuntu
LIBDIR="/usr/lib/x86_64-linux-gnu"
INCDIR="/usr/include/x86_64-linux-gnu"

# Installation directory
PREFIX="${HOME}/tuto-abischool26/abinit-10.4.7"

# Compilers & Flags
FC=mpifort
CC=mpicc
with_optim_flavor="custom"
with_debug_flavor="basic"
FCFLAGS_EXTRA="-Wall -Wextra"
FCFLAGS_OPTIM="-O3 -march=native"

# MPI & OpenMP
with_mpi="yes"
enable_mpi_io="yes"
enable_mpi_in_place="yes"
enable_openmp="yes"

# Linear algebra (OpenBLAS)
with_linalg_flavor="openblas"
LINALG_LIBS="-L${LIBDIR}/openblas-openmp -lopenblas"
LINALG_FCFLAGS="-I${INCDIR}/openblas-openmp"

# FFTW3
with_fft_flavor="fftw3"
FFTW3_LIBS="-L${LIBDIR} -lfftw3 -lfftw3f -lfftw3_threads -lfftw3f_threads -lfftw3_omp -lfftw3f_omp"

# LibXC
with_libxc="${LIBDIR}"

# HDF5
# Serial version: parallel I/O not needed.
with_hdf5="${LIBDIR}/hdf5/serial/"
H5CC="/usr/bin/h5cc"

# NetCDF
NETCDF_LIBS="-L${LIBDIR} -lnetcdf"

# NetCDF-Fortran
NETCDF_FORTRAN_LIBS="-L${LIBDIR} -lnetcdff"
```

Compiling ABINIT

Now we run `make -j <nprocs>` and wait for a bit..
If everything goes fine, we finally get

```
================================================================================
Keywords to use with the make command to run quick tests,
  possibly with -j x option:

>>> In build directory:

check               Launch a small set of tests that cover most of the main features
                    For these tests, only a comparison of the main physical results
                    in the output is performed.

test_fast           Launch 3 very small tests

tests_in            Launch test_fast, followed by other internal tests

>>> In "abichecks" directory (cd abichecks first)

tests_abirules      Launch the abirules tests
                    Warning: to be run with the command: make > make.log 2>&1`

tests_buildsys      Launch the buildsys tests

tests_libpaw        Launch the libpaw tests

================================================================================
```

At the end the build system tells you to run some tests.

You should always check that your newly compiled ABINIT *is working fine!*

Testing ABINIT

At least run the basic tests

```
> make check
```

```
Suite            failed  passed  succeeded  skipped  disabled  run_etime  tot_etime
tutomultibinit        0       0          0        1         0       0.00       0.00
v10                   0       0          1        0         0       8.18       8.20
v2                    0       0          2        0         0       6.61       6.65
v3                    0       0          2        0         0       6.98       7.16
v5                    0       1          0        0         0       2.46       2.46
v6                    0       0          1        0         0       1.46       1.47
v67mbpt               0       0          1        0         0      35.56      35.61
v7                    0       0          2        0         0       5.82       5.85
v8                    0       0          1        0         0       7.34       7.38
v9                    0       0          1        0         0       2.48       2.49
wannier90             0       0          0        1         0       0.00       0.00

Completed in 77.72 [s]. Average time for test=6.41 [s], stdev=9.06 [s]
Summary: failed=0, succeeded=11, passed=1, skipped=2, disabled=0
```

Testing ABINIT

At least run the basic tests

```
> make check
```

Or even better run the full test suite using the `runtests.py` python script

```
> python3 doc/tests/runtests.py -j <nprocs>
```

Be careful it takes time !
1h30m on my laptop using 8 processes

Failure rate: 4/975 - 0.41%

Suite	failed	passed	succeeded	skipped	disabled	run_etime	tot_etime
atdep	0	0	38	0	0	1874.62	1887.64
atompaw	0	0	0	2	0	0.00	0.01
bigdft	0	0	0	19	0	0.00	0.06
bigdft_paral	0	0	0	4	0	0.00	0.01
built-in	0	0	5	2	0	18.13	18.16
etsf_io	0	0	8	0	0	72.19	72.55
fast	0	0	11	0	0	122.27	123.66
gpu	0	0	0	8	0	0.00	0.01
gpu_kokkos	0	0	0	1	0	0.00	0.00
gpu_omp	0	0	0	49	0	0.00	0.07
gwr_suite	0	0	0	11	0	0.00	0.01
hpc_gpu_omp	0	0	0	13	0	0.00	0.02
libxc	0	9	28	0	0	1680.93	1685.94
mpiio	0	0	1	16	0	4.43	4.50
paral	0	8	22	127	0	1580.23	1584.69
psml	0	0	0	14	0	0.00	0.01
rttddft_suite	0	0	6	0	0	352.17	353.00
seq	0	0	0	18	0	0.00	0.02
tutoatdep	0	0	5	0	0	56.10	56.58
tutomultibinit	0	0	6	4	0	1241.36	1248.94
tutoparal	0	0	4	24	0	514.01	514.49
tutoplugs	0	0	0	8	0	0.00	0.01
tutorespfn	3	7	19	3	0	6155.61	6161.65
tutorial	0	13	58	1	0	5189.30	5194.61
unitary	0	3	22	13	0	574.81	575.29
v1	0	2	72	0	0	670.00	675.58
v10	0	1	24	7	0	1753.93	1756.73
v2	0	17	62	0	0	742.12	747.88
v3	0	12	67	0	0	1496.92	1507.31
v4	0	16	45	0	0	962.25	969.35
v5	1	18	53	0	0	2806.09	2814.08
v6	0	9	51	0	0	1740.27	1750.70
v67mbpt	0	8	17	0	0	884.80	890.07
v7	0	17	48	0	0	3513.00	3526.64
v8	0	15	51	5	0	5562.36	5577.29
v9	0	20	85	3	0	4780.73	4801.10
vdwxc	0	0	0	1	0	0.00	0.00
wannier90	0	0	0	10	0	0.00	0.01

Completed in 5652.47 [s]. Average time for test=45.30 [s], stdev=101.88 [s]
Summary: failed=4, succeeded=800, passed=175, skipped=363, disabled=0

Additional information

- A new version of the build system using the more modern `CMAKE` tool has been developed. More information about this new build system is accessible here: `https://docs.abinit.org/installation/#how-to-build-abinit-with-cmake`
- If you wish to compile ABINIT for GPUs the procedure is almost exactly the same. But during the configure step you need to:
  - Use a compiler that is compatible with the GPUs you will use (for instance nvhpc for NVIDIA GPUs)
  - Enable GPUs by adding `enable_mpi_gpu_aware=yes` in your configuration file
  - Add a few specific options to define which GPU implementation to use and the GPU architecture

  Example of configuration files for GPUs are accessible in `doc/build`
- In some cases, you won't have access to the library you need in that case your last resort is to compile it yourself.
- In that case, you can have a look at the abinit-fallbacks project on github `https://github.com/abinit/abinit-fallbacks` that helps you build all the dependencies of ABINIT.

Hands-on session

Write your own ac9 file and compile ABINIT

- You can do it on your own laptop or on the cluster (recommended!).
- Connect to the INTI cluster and get the files for the practical session

```
> ssh -Y stagXX@inti.ocre.cea.fr
> cd $CCCWORKDIR
> cp -r $ABISCHOOL/handson_compil .
> cd handson_compil
```

You should work on the work directory ($CCCWORKDIR) not your home !

- On the cluster, the environment is managed using modules.
  More information available on the howto_inti document. *Read this document !*

```
> evince howto_inti.pdf &
```

Hands-on session

Write your own ac9 file and compile ABINIT

- Start an interactive session and connect to the node

```
> start_session
> connect
```

- Go to the right directory (if you are not already there)

```
> cd $CCCWORKDIR/handson_compil
```

Important

Always work on the work directory ($CCCWORKDIR)

- You can find this presentation with the following information for the hands-on session here

```
> evince abischool_compil.pdf &
```

Hands-on session

- You should first extract the sources of ABINIT

```
> tar -zxf abinit-10.4.7.tar.gz
> cd abinit-10.4.7
```

- Creating a separate directory to build in is cleaner

```
> mkdir build
> cd build
```

- Now copy the model configuration file (inti-intel-rome.ac9) that you will have to edit

```
> cp ../../inti-intel-rome.ac9 .
```

Hands-on session

We will compile using the following

- Compiler: Intel 19.1
- Parallelism:
  - MPI: OpenMPI 4.1.4
  - OpenMP
- Input/Output:
  - HDF5 1.8
  - NetCDF 4.6
  - NetCDF-Fortran 4.4
- Linear Algebra: Intel MKL 20.0 (BLAS+SCALAPACK)
- FFT: Intel MKL 20.0 (DFTI)
- LibXC 6.1

All these dependencies should be automatically loaded by the abischool module.
You can check that after loading the modules you have the right version of the compiler with

```
> mpif90 -v
```

Hands-on session

The modules set up a lot of environment variables that define the location of the libraries.
You can check what the module is doing using `module show`.
For instance for libxc

```
> module show libxc/6.1.0
```

All the `set-env` instructions are creating environment variables. You can see that several variables have been created including `LIBXC_ROOT`, `LIBXC_LIBDIR`, `LIBXC_LDFLAGS`..
You should use some of these variables to define the location of the libraries in the ac9 file.

Once you have entirely filled the configuration file, you can run the configure step

```
> ../configure --with-config-file="inti-intel-rome.ac9"
```

Hands-on session

Inspect the output carefully and check that

- you have the expected compilers
- MPI is enabled with MPI in-place and MPI-IO
- OpenMP is enabled
- HDF5, NetCDF and NetCDF-F are enabled with MPI-support
- you have the right FFT flavor
- you have the right LINALG flavor
- SCALAPCK is enabled

If everything looks good just start the compilation with

```
> make -j
```

Unfortunately, the compilation takes time ($\sim$ 20 min)..

Hands-on session

Once it is done and hopefully finishes without any problem.
You can finally run the small test suite using

```
> make check
```

and make sure that they succeed.

If so, well done you most probably have correctly compiled ABINIT :-)