SIESTA Deployment Options

SIESTA School, 2021

Vladimir Dikan vdikan@icmab.es

SIESTA Deployment Options

Author: Vladimir Dikan

Table of Contents

SIESTA Deployment Options

- Ready-to-Use Options
- Source Code Compilation
- Scripted Installations

https://docs.siesta-project.org

As you might know (or will realize during the school), SIESTA has quite a number of capabilities and operation modes. And and as many other HPC codes, it relies on quite a number of dependencies and build options, that sometimes give their users hard times with configuration of their research environments.

Below is a review of some aspects of compilation and deployment of SIESTA. Some pre-configured options are discussed, followed by an overview of SIESTA's general Makefile template and

Ready-to-Use Options

Quantum Mobile

What is Quantum Mobile

Quantum Mobile is a Virtual Machine for computational materials science.

Quantum Mobile provides a uniform environment for quantum-mechanical materials simulations. Simulation codes are set up and ready to be used. Users can run their calculations directly or automate workflows and provenance tracking.

Open source throughout.

Based on Ubuntu Linux

Pre-built images

Available for Linux, MacOS or Windows computers, using VirtualBox. Or deploy on cloud services like OpenStack or Amazon Elastic Compute Cloud using ansible.

Simulation codes pre-installed

Abinit, BigDFT, CP2K, Fleur, Quantum ESPRESSO, Siesta, Wannier90, Yambo, together with AiiDA, JupyterLab, and the AiiDAlab Jupyter environment.

Tools pre-installed

atomistic (xcrysden, jmol, cif2cell, ase, pymatgen, seekpath, spglib, pycifrw), visualization (grace, gnuplot, matplotlib, bokeh, jupyter), simulation environment (slurm, OpenMPI, FFT/BLAS/LAPACK, gcc, gfortran, singularity).

Modular setup

with individually tested ansible roles. Build your own flavour tailored to your use case.

QuantumMobile VM

https://quantum-mobile.readthedocs.io/

TAG

master

Last pushed **a month ago** by vdikan

docker pull vdikan/siesta-dist:master

DIGEST

eee62cd76e63

OS/ARCH

linux/amd64

COMPRESSED SIZE

185.66 MB

TAG

psml

Last pushed a month ago by vdikan

DIGEST

a7e09b3984f7

OS/ARCH

linux/amd64

COMPRESSED SIZE

186.83 MB

TAG

elsi

Last pushed a month ago by vdikan

DIGEST

6dcbf229a52b

OS/ARCH

linux/amd64

COMPRESSED SIZE

193.45 MB

TAG

4.1-b4

Last pushed **a month ago** by vdikan

docker pull vdikan/siesta-dist:4.1-b4

DIGEST

b2b17b466a1d

OS/ARCH

linux/amd64

COMPRESSED SIZE

170.26 MB

Containers: Docker

https://hub.docker.com/

```
docker pull vdikan/siesta-dist:master
docker run --interactive --tty -w /app \
       -v "$(pwd):/app" vdikan/siesta-dist:master
```

Cloud Library

Cloud Library is the official image registry provided by **Sylabs.io**. Users can share Singularity images through the Cloud Library, as well as pull/push SIF™ images through Singularity CLI. Email any feature requests or feedback to **support@sylabs.io**.

Containers: Singularity

https://cloud.sylabs.io/library

From an existing Docker image:

```
singularity build siesta-dist.sif \
              docker://vdikan/siesta-dist:master

singularity run siesta-dist.sif
```

Source Code Compilation

Staging the build directory

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

```
# 1. obtain src archive
wget https://gitlab.com/siesta-project/siesta/-/\
archive/v4.1.5/siesta-v4.1.5.tar.gz
tar -xzf siesta-v4.1.5.tar.gz
cd siesta-v4.1.5/
# 2. create distinct build directory for siesta
cd Obj/
sh ../Src/obj_setup.sh
# 3. bootstrap arch.make inside build directory
cp ARCH-EXPERIMENTAL/master-raw.make ./arch.make
$EDITOR arch.make  # configure siesta build
# 4. compile siesta
make
# 5. optionally, build the utilities
cd Util && ./build_all.sh
```

arch.make configuration

Inside the **Obj/arch.make** file:

- define a set of external dependencies
- provide linking paths and symbols for dependencies
- configure compilers

```
# Make sure you have the appropriate library symbols
# (Either explicitly here, or through shell variables, perhaps
#  set by a module system)
# Define also compiler names and flags
#------------------------------------------------------
XMLF90_ROOT=$(shell spack location -i xmlf90)
PSML_ROOT=$(shell spack location -i libpsml)
GRIDXC_ROOT=$(shell spack location -i libgridxc@thtr)
LIBXC_ROOT=$(shell spack location -i libxc)
#ELSI_ROOT=
#ELPA_ROOT=
#ELPA_INCLUDE_DIRECTORY=
#FLOOK_ROOT=
#------------------------------------------------------
NETCDF_ROOT=$(shell spack location -i netcdf-c)
NETCDF_FORTRAN_ROOT=$(shell spack location -i netcdf-fortran)
#HDF5_LIBS=-L/apps/HDF5/1.10/GCC/OPENMPI/lib -lhdf5_hl -lhdf5 -lsz -lz
SCALAPACK_LIBS=-L$(shell spack location -i netlib-scalapack)/lib -lscalapack
LAPACK_LIBS = -L$(shell spack location -i openblas)/lib -lopenblas
#FFTW_ROOT=/apps/FFTW/3.3/GCC/OPENMPI
# Needed for PEXSI (ELSI) support
#LIBS_CPLUS=-lstdc++ -lpthread
#------------------------------------------------------
#
# FC_PARALLEL=$(shell spack location -i openmpi)/bin/mpif90
#
FC_SERIAL=gfortran
FPP = $(FC_SERIAL) -E -P -x c
FFLAGS = -g -O2 -fPIC -ftree-vectorize
FFLAGS_DEBUG= -g -O0 -fPIC -ftree-vectorize
#RANLIB=echo
#
# Alternatively, prepare a fortran.mk file with compiler definitions,
```

Siesta Dependencies

https://gitlab.com/siesta-project/libraries

Common	SIESTA-specific
MPI, BLAS, LAPACK	libfdf
ELSI + ext. solvers	xmlf90
NetCDF	libPSML
libXC	libGridXC
flook	flos

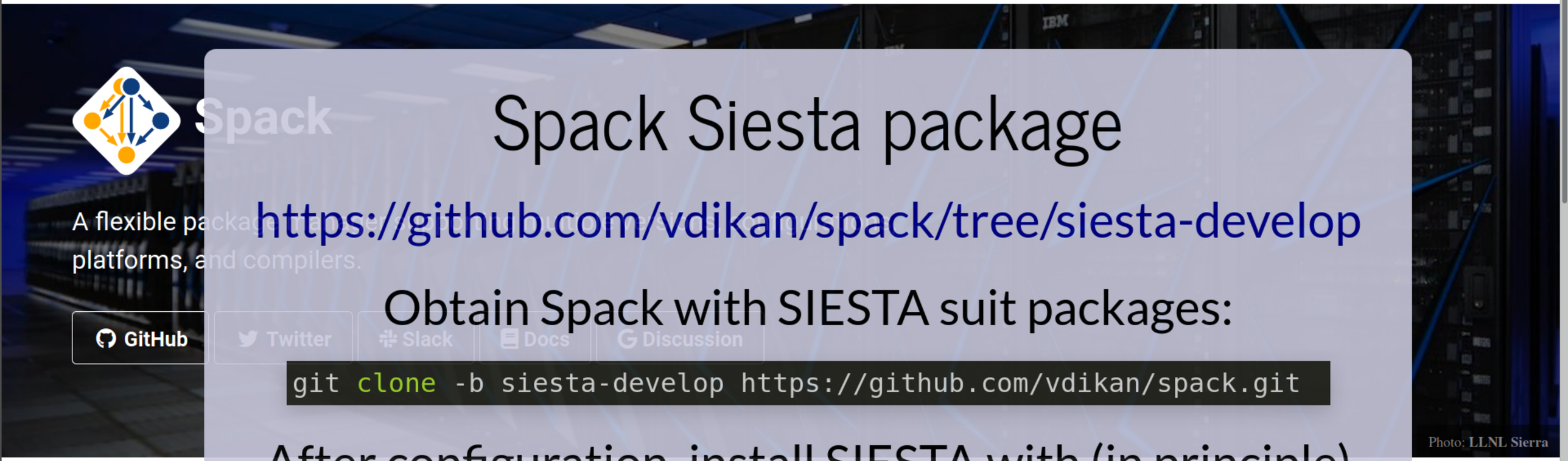
Scripted Installations

Siesta-Install-Scripts

https://gitlab.com/mailhexu/siesta-install-scripts/-/tree/ubuntu

```
git clone https://gitlab.com/mailhexu/siesta-install-scripts.git
cd siesta-install-scripts/
# 1. address README-s in subdirectories
# 2. inspect ./Tarballs/download.sh
./Tarballs/download.sh
cp cp Config/gnu/* Tarballs/  # copy Makefile for GNU
cp Config/siesta.common.arch.make Tarballs/
# 3. inspect and adapt Makefiles
# 4. inspect and edit ./do_all.sh
./do_all.sh
```

Collection of scripts to build Siesta (MaX version 1)



Spack Siesta package

https://github.com/vdikan/spack/tree/siesta-develop

Obtain Spack with SIESTA suit packages:

```
git clone -b siesta-develop https://github.com/vdikan/spack.git
```

After configuration, install SIESTA with (in principle) one-line spec command:

```
spack install siesta@master +utils ^openmpi +cxx +cxx_exceptions
```

Welcome to Spack!

Spack is a package manager for supercomputers, Linux, and macOS. It makes installing scientific software easy. Spack isn't tied to a particular language; you can build a software stack in Python or R, link to libraries written in C, C++, or Fortran, and easily swap compilers or target specific microarchitectures. Learn more here.

Recent Posts

Thank You!

Documentation:

- https://docs.siesta-project.org/projects/siesta/
- https://siesta-project.org/siesta/
- SIESTA Manual
- Matter Modeling StackExchange: `siesta`

Contact Me:

- Event Official Channel
- Email: vdikan@icmab.es