Siesta 4.1.5

ウェブページ

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

バージョン

4.1.5 (+ELPA 2021.05.002)

ビルド環境

- Intel Compiler 19.1.2 (Intel Parallel Studio 2020 Update 2)
- Intel MKL 2020.0.2 (Intel Parallel Studio 2020 Update 2)
- Intel MPI 2018.0.4 (Intel Parallel Studio 2018 Update 4)

ビルドに必要なファイル

- siesta-v4.1.5.tar.gz
- arch.make

```
.SUFFIXES:
.SUFFIXES: .f .F .o .c .a .f90 .F90

SIESTA_ARCH = rccs-intel20-mkl

CC = mpiicc
FPP = $(FC) -E -P -x c
FC = mpiifort
FC_SERIAL = ifort
FFLAGS = -O2 -fPIC -xHost -fp-model source

AR = ar
ARFLAGS_EXTRA =
RANLIB = ranlib

SYS = nag

SP_KIND = 4
DP_KIND = 8
KINDS = $(SP_KIND) $(DP_KIND)

DEFS_PREFIX =

LDFLAGS =
FCFLAGS_fixed_f = -fixed
FCFLAGS_free_f90 = -free
FPPFLAGS_fixed_F = -fixed
FPPFLAGS_free_F90 = -free

BLAS_LIBS = -mkl=sequential
LAPACK_LIBS = -mkl=sequential
SCALAPACK_LIBS = -L$(MKLROOT)/lib/intel64 -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lpthread -lm
-ldl

COMP_LIBS =

NETCDF_ROOT = /local/apl/lx/siesta415/exts
NETCDF_INCFLAGS = -I$(NETCDF_ROOT)/include
NETCDF_LIBS = -L$(NETCDF_ROOT)/lib -lnetcdff -lnetcdf

MPI_INTERFACE = libmpi_f90.a
MPI_INCLUDE = .
```

```
FPPFLAGS = $(DEFS_PREFIX)-DFC_HAVE_ABORT -DSIESTA__ELPA -DMPI -DCDF -DFC_HAVE_ABORT -DFC_HAVE_FLUSH

LIBS = $(NETCDF_LIBS) $(SCALAPACK_LIBS) $(LAPACK_LIBS) $(MPI_LIBS) $(COMP_LIBS)

FFLAGS_DEBUG = -g -O1 -fp-model source

# ELPA
ELPA_ROOT = /local/apl/lx/siesta415/exts
ELPA_INCFLAGS = -DSIESTA__ELPA -I$(ELPA_ROOT)/include/elpa-2021.05.002/modules
ELPA_LIBS = -L$(ELPA_ROOT)/lib -lelpa

LIBS += $(ELPA_LIBS)
FPPFLAGS += $(ELPA_INCFLAGS)

atom.o: atom.F
	$(FC) -c $(FFLAGS_DEBUG) $(INCFLAGS) $(FPPFLAGS) $(FPPFLAGS_fixed_F) $<
state_analysis.o: state_analysis.F
	$(FC) -c $(FFLAGS_DEBUG) $(INCFLAGS) $(FPPFLAGS) $(FPPFLAGS_fixed_F) $<

.c.o:
	$(CC) -c $(CFLAGS) $(INCFLAGS) $(CPPFLAGS) $<
.F.o:
	$(FC) -c $(FFLAGS) $(INCFLAGS) $(FPPFLAGS) $(FPPFLAGS_fixed_F)  $<
.F90.o:
	$(FC) -c $(FFLAGS) $(INCFLAGS) $(FPPFLAGS) $(FPPFLAGS_free_F90) $<
.f.o:
	$(FC) -c $(FFLAGS) $(INCFLAGS) $(FCFLAGS_fixed_f)  $<
.f90.o:
	$(FC) -c $(FFLAGS) $(INCFLAGS) $(FCFLAGS_free_f90)  $<
```

- elpa-2021.05.002.tar.gz
- netcdf-c-4.8.1.tar.gz
- netcdf-fortran-4.5.3.tar.gz

ビルド手順

ELPA-2021.05.002

```
#!/bin/sh

VERSION=2021.05.002
INSTDIR=/local/apl/lx/siesta415/exts
WORKDIR=/work/users/${USER}

BASEDIR=/home/users/${USER}/Software/ELPA/${VERSION}
TARBALL=${BASEDIR}/elpa-${VERSION}.tar.gz

PARALLEL=12

#-------------------------------------------------------------------------------
umask 0022
ulimit -s unlimited

module purge
module load intel/19.1.2
module load mpi/intelmpi/2018.4.274
module load mkl/2020.0.2

export LANG=C
export LC_ALL=C

export FC=mpiifort
export CC=mpiicc
export CXX=mpiicpc
export FCFLAGS=-I${MKLROOT}/include/intel64/lp64/
```

```
# mkl_link_tool -libs -c intel_f -p no --cluster_library=scalapack
export LDFLAGS="-L${MKLROOT}/lib/intel64 -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lpthread -lm
-ldl"

cd ${WORKDIR}
if [ -d elpa-${VERSION} ]; then
  mv elpa-${VERSION} elpa-erase
  rm -rf elpa-erase &
fi
tar zxf ${TARBALL}
cd elpa-${VERSION}

./configure --prefix=${INSTDIR}
make -j ${PARALLEL}
make check
make install
```

Siesta

(先に上の ELPA を導入しておく)

```
#!/bin/sh

VERSION=4.1.5
INSTDIR=/local/apl/lx/siesta415
WORKDIR=/work/users/${USER}
BASEDIR=/home/users/${USER}/Software/Siesta/${VERSION}
TARBALL=${BASEDIR}/siesta-v${VERSION}.tar.gz
ARCHMAKE=${BASEDIR}/arch.make

NETCDF_C_VERSION=4.8.1
NETCDF_F_VERSION=4.5.3
BASEDIR_NETCDF=/home/users/${USER}/Software/NETCDF
TARBALL_NETCDF_C=${BASEDIR_NETCDF}/c${NETCDF_C_VERSION}/netcdf-c-${NETCDF_C_VERSION}.tar.gz
TARBALL_NETCDF_F=${BASEDIR_NETCDF}/f${NETCDF_F_VERSION}/netcdf-fortran-${NETCDF_F_VERSION}.tar.gz

PARALLEL=12 # NOTE: parallel make cannot be used for siesta

#-------------------------------------------------------------------------------
umask 0022
ulimit -s unlimited

module purge
module load intel/19.1.2
module load mpi/intelmpi/2018.4.274
module load mkl/2020.0.2

export LANG=C
export LC_ALL=C
export FC=ifort
export CC=icc

cd ${WORKDIR}
if [ -d netcdf-c-${NETCDF_C_VERSION} ]; then
  mv netcdf-c-${NETCDF_C_VERSION} netcdf-c-erase
  rm -rf netcdf-c-erase &
fi
tar zxf ${TARBALL_NETCDF_C}
cd netcdf-c-${NETCDF_C_VERSION}

./configure --prefix=${INSTDIR}/exts
make -j${PARALLEL}
make -j${PARALLEL} check
make install

export PATH="${PATH}:${INSTDIR}/exts/bin"
```

```
export CPATH="${CPATH}:${INSTDIR}/exts/include"
export LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:${INSTDIR}/exts/lib"
export LIBRARY_PATH="${LIBRARY_PATH}:${INSTDIR}/exts/lib"

cd ${WORKDIR}
if [ -d netcdf-fortran-${NETCDF_F_VERSION} ]; then
  mv netcdf-fortran-${NETCDF_F_VERSION} netcdf-fortran-erase
  rm -rf netcdf-fortran-erase &
fi
tar zxf ${TARBALL_NETCDF_F}
cd netcdf-fortran-${NETCDF_F_VERSION}

./configure --prefix=${INSTDIR}/exts
make -j${PARALLEL}
make -j${PARALLEL} check
make install

cd ${INSTDIR}
if [ -d siesta-v${VERSION} ]; then
  mv siesta-v${VERSION} siesta-erase
  rm -rf siesta-erase
fi
tar zxf ${TARBALL}
mv siesta-v${VERSION}/* .
rmdir siesta-v${VERSION}

# hidoiyo...
echo >> Tests/OMM_h2o/OMM_h2o.fdf
echo >> Tests/OMM_si/OMM_si.fdf

mkdir bin # install dir

cd Obj
../Src/obj_setup.sh
cp ${ARCHMAKE} ./arch.make

# build transiesta
cd ${INSTDIR}/Obj && make transiesta
# build siesta
make clean-transiesta && make

# utils
cd ${INSTDIR}/Util
echo "m_cite.o: version.o" >> Gen-basis/Makefile
sh build_all.sh

# test siesta & transiesta
cd ${INSTDIR}/Obj/Tests
make MPI="mpirun -np 2" SIESTA="${INSTDIR}/Obj/siesta" check >& make_check.log
make MPI="mpirun -np 2" TS="${INSTDIR}/Obj/transiesta" tests-ts >& make_check_ts.log
cd ../
mv siesta ${INSTDIR}/bin
mv transiesta ${INSTDIR}/bin
```

テスト

ELPA

```
# TOTAL: 100
# PASS:  39
# SKIP:  54
# XFAIL: 0
# FAIL:  7
# XPASS: 0
# ERROR: 0
```

失敗した 7 つのテストについては、gfortran を使った場合(この時も MKL 使用)も同様にエラー。Intel コンパイラのバージョンを変えても変化無し。

NetCDF

- tst_charvlenbug.c: build に失敗

Siesta

- ar2_vdw: 軽微な数値エラー
- dipole_correction: 軽微な数値エラー
- fe_clust_noncollinear: 軽微な(?)数値エラー
- fe_cllust_noncollinear-gga: 軽微な(?)数値エラー
- fe_nocol_kp: 軽微な数値エラー
- gate_G_charge: 軽微な数値エラー
- ge111: 軽微な数値エラー
- graphite_c6_full: 少なくとも最終的には軽微な数値エラー？
- h2o_orderN: 軽微な(?)数値エラー
- mix_broyden: 軽微な数値エラー
- mix_pulay: 軽微な(?)数値エラー
- ptcda-au: 軽微な数値エラー
- si001-diags_mrrr: 実行エラー (MRRR 未導入のため?)
- sic-slab: 軽微な数値エラー
- sih_fire: 最終的には軽微な(?)数値エラー
- sinw_2: 軽微な数値エラー
- var_cell: 軽微な(?)数値エラー
- wannier: 実行エラー (wannier90 未導入のため?)
- zmatrix: ?出力内容がそもそも違う?

メモ

- Intel 2020 の MPI を使うと挙動がおかしい点があるため回避
- GCC よりも Intel コンパイラを使った方が全体的に速い
- Siesta の Util についてはビルドをチェックする方法がわからず
- バイナリは Obj/ から bin/ 以下に移動しています