

1 Before starting

1.1 Source codes

I will describe a full compilation of all libraries needed to run SIESTA-2.0 in parallel so before everything be shure to have already the following packages :

- Intel C/C++ and Fortran77/**90** compilers : **icc** et **ifort**
- MPICH/MPICH2
- BLAS, LAPACK, (ATLAS)
- BLACS, SCALAPACK

1.2 Environment parameters

Be shure to work in a good environment, check that you will have the needed tools in your PATH, LD_LIBRARY_PATH and MANPATH (it's help too).

A part of my .bashrc file follows :

...

MPI=mpich or mpich2

Man pages

MANPATH=\$HOME/appz/intel/fc/9.0/man:\$HOME/appz/intel/cc/9.0/man

MANPATH=\$MANPATH:\$HOME/appz/**\$MPI**/man

MANPATH=\$MANPATH:/opt/pbs/man:/usr/share/man:/usr/X11R6/man

MANPATH=\$MANPATH:/usr/kerberos/man:/usr/man:/usr/local/man

Applications

PATH=\$HOME/scripts/bin:\$HOME/appz/bin

PATH=\$PATH:\$HOME/appz/**\$MPI**/bin:/usr/pbs/bin

PATH=\$PATH:\$HOME/appz/intel/fc/9.0/bin:\$HOME/appz/intel/cc/9.0/bin

PATH=\$PATH:/usr/local/bin:/bin:/usr/bin

PATH=\$PATH:/usr/X11R6/bin:/usr/pbs/bin:/usr/kerberos/bin

Library

LD_LIBRARY_PATH=\$HOME/appz/intel/fc/9.0/lib:\$HOME/appz/intel/cc/9.0/lib

LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:\$HOME/appz/lib :

LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:\$HOME/appz/**\$MPI**/lib

LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/usr/lib:/lib:/usr/local/lib:/usr/pbs-2.3.16/lib

...

Remarque : If you want to compile the code and libraries as I did in your \$HOME directory, and if there is already a version either of the compiler or the

library install on your system be shure not to let the usual system PATH and LD_LIBRARY_PATH in your env.

2 Installation

2.1 Compilers

All codes and lib will be compiled using the same compiler I used the following :

- ifort : Version 9.0 Build 20051201Z Package ID : l_fc_c_9.0.031
- icc : Version 9.0 Build 20051201Z Package ID : l_cc_c_9.0.030

So install them (if not already done) before everything, both fortran and c are used thereafter. You have to used both C and Fortran Intel compilers because of a condition of high compatibility of the C and Fortran compilers during mpich and mahtematic lib build. You can compile mpich using Ifort and gcc/g++ but the parallel Siesta compilation may clash and anyway the parallel version will not work (for me it did not work).

2.2 MPI

2.2.1 MPICH

Used compilation options :

```
|% tar -zxf mpich.tar.gz
|% cd mpich-1.2.7p1
|% ./configure --prefix=$HOME/appz/mpich -f90=ifort -f77=ifort -cc=icc -fflags="-O3 -tpp7" -cflags="-O3 -tpp7" -c++=icc
|% make
|% make install
```

You may have trouble during C++ test at the end of install process ... It does not affect the installation of mpich.

2.3 MPICH2

Used compilation options :

```
|% tar -zxf mpich2-1.0.3.tar.gz
|% cd mpich2-1.0.3
|% ./configure --prefix=$HOME/appz/mpich2 FC=ifort F90=ifort F77=ifort CC=icc FFLAGS="-O3 -tpp7" CFLAGS="-O3 -tpp7" CXX=icc
|% make
```

```
]% make install
```

3 Mathematics library

On my sytem I did not build SIESTA 2.0 sucefully using Intel mkl or cmkl lib, so I did an "*Ab-initio*" build of the mathematic lib.

Here you will have to choose a Fortran and a C/C++ compiler, be shure to check the Fortran/C compatibility option, I did not compile Siesta 2.0 successfuly using ifort and gcc, only using ifort and icc both Intel compilers.

3.1 BLAS and LAPACK

Used compilation options :

```
]% tar -zxf lapack.tgz
```

```
]% cd LAPACK
```

Edit INSTALL/make.inc.'Your SYST' and change the following options :

```
FORTTRAN = ifort
```

```
OPTS = -O3 (-tpp7 -xW) your sytem options
```

```
DRVOPTS = $(OPTS)
```

```
NOOPT = -O0
```

```
LOADER = ifort
```

Save the file as "make.inc"

```
]% make blaslib
```

```
]% make lapacklib
```

2 files have been created : blas_LINUX.a et lapack_LINUX.a

You have now to share them in an accessible dir of your LD_LIBRARY_PATH

In my case : \$HOME/appz/lib

3.2 BLACS

Used compilation options :

```
]% tar -zxf mpiblacs.tgz
```

```
]% cd BLACS
```

Edit BMAKES/Bmake.MPI-"Your SYST" and change the following options :

```
BTOPdir = $HOME/tmp/BLACS
```

```
MPIdir = $HOME/appz/$MPI (In my case)
```

```
MPIdev = ch_p4
```

```
MPIplat = LINUX
```

```
MPILIBdir = $(MPIdir)/lib/$(MPIplat)/$(MPIdev)
```

```
MPIINCdir = $(MPIdir)/include
```

MPILIB = \$(MPIdir)/lib/"libmpich.a" or "libmpi.a" (mpich or openmpi)

INTERFACE = -DAdd_

F77 = mpif77

F77NO_OPTFLAGS = -O0

F77FLAGS = \$(F77NO_OPTFLAGS) -O3 (-tpp7 -xW) your sytem options

F77LOADER = \$(F77)

F77LOADFLAGS =

CC = mpicc

CCFLAGS = -O3 (-tpp7 -xW) your sytem options

CCLOADER = \$(CC)

CCLOADFLAGS =

]% make mpi

In the LIB dir 3 files have been created blacsCinit_MPI-LINUX-0.a, blacsF77init_MPI-LINUX-0.a and blacs_MPI-LINUX-0.a You have now to share them in an acces-
sible dir of your LD_LIBRARY_PATH

In my case : \$HOME/appz/lib

3.3 SCALAPACK

Used compilation option :

]% tar -zxf scalapack.tgz

]% cd scalapack-1.7.3

Edit file INSTALL/SLmake."Your SYST" and change the following options :

home = \$(HOME)/tmp/scalapack-1.7.3

BLACSdir = \$(HOME)/appz/lib

SMPLIB = \$(HOME)/appz/\$MPI/lib/ "libmpich.a" or "libmpi.a"

BLACSFINIT = \$(BLACSdir)/blacsF77init_MPI-LINUX-0.a

BLACSCINIT = \$(BLACSdir)/blacsCinit_MPI-LINUX-0.a

BLACSLIB = \$(BLACSdir)/blacs_MPI-LINUX-0.a

F77 = mpif77

CC = mpicc

NOOPT = -O0

F77FLAGS = \$(NOOPT) -O3 (-tpp7 -xW) your sytem options

CCFLAGS = -O3 (-tpp7 -xW) your sytem options

SRCFLAG =

F77LOADER = \$(F77)

```
CCLOADER = $(CC)
F77LOADFLAGS =
CCLOADFLAGS =
```

```
CDEFS = -DAdd_ -DNO_IEEE $(USEMPI)
```

```
BLASLIB = $(HOME)/appz/lib/blas_LINUX.a
```

Save modification in file SLmake.inc

```
]% make
```

File libscalapack.a has been created you have now to share it in an accessible dir
of your LD_LIBRARY_PATH

In my case : \$HOME/appz/lib

4 SIESTA

I just put here a copy of my arch.make :

```
#
# This file is part of the SIESTA package.
#
# Copyright (c) Fundacion General Universidad Autonoma de Madrid :
# E.Artacho, J.Gale, A.Garcia, J.Junquera, P.Ordejon, D.Sanchez-Portal
# and J.M.Soler, 1996-2006.
#
# Use of this software constitutes agreement with the full conditions
# given in the SIESTA license, as signed by all legitimate users.
#
.SUFFIXES :
.SUFFIXES : .f .F .o .a .f90 .F90
```

```
SIESTA_ARCH=i686-pc-linux-gnu-Intel
```

```
FPP=
FPP_OUTPUT=
FC=mpif90
RANLIB=ranlib
```

```
SYS=nag
```

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

```
FFLAGS= -w -O3 (-tpp7) your sytem options
FPPFLAGS= -DMPI -DFC_HAVE_FLUSH -DFC_HAVE_ABORT
LDLFLAGS= -Vaxlib -static
```

```
ARFLAGS_EXTRA=
```

```
FCFLAGS_fixed_f=
FCFLAGS_free_f90=
FPPFLAGS_fixed_F=
FPPFLAGS_free_F90=
FFLAGS_DEBUG= -g
```

```
HOME_LIB=/home/master/leroux/appz/lib
BLAS_LIBS=$(HOME_LIB)/blas_LINUX.a
LAPACK_LIBS=$(HOME_LIB)/lapack_LINUX.a
BLACS_LIBS=$(HOME_LIB)/blacsCinit_MPI-LINUX-0.a $(HOME_LIB)/blacsF77init_MPI-
LINUX-0.a $(HOME_LIB)/blacs_MPI-LINUX-0.a
SCALAPACK_LIBS=$(HOME_LIB)/libscalapack.a
```

```
NETCDF_LIBS=
```

```
LIBS= $(GUIDE) $(SCALAPACK_LIBS) $(BLACS_LIBS) $(LAPACK_LIBS)
$(BLAS_LIBS) $(NETCDF_LIBS)
```

```
#SIESTA needs an F90 interface to MPI
#This will give you SIESTA's own implementation
#If your compiler vendor offers an alternative, you may change
#to it here.
MPI_HOME=$(HOME)/appz/$(MPI)
MPI_INTERFACE=libmpi_f90.a
MPI_INCLUDE=$(MPI_HOME)/include
MPI_LIB=$(MPI_HOME)/lib
DEFS_MPI=-DMPI
```

```
#Dependency rules are created by autoconf according to whether
#discrete preprocessing is necessary or not.
```

```
.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) \$<

I hope It will help you ...

Sébastien Le Roux
Doctorant - PhD candidate
LPMC UMR 5617
CC003
Université de Montpellier II
Place E. Bataillon
34095 MONTPELLIER cedex 05
E mail : leroux@lpmc.univ-montp2.fr
Tel : +33(0)4.67.14.41.21
Fax : +33(0)4.67.14.41.90