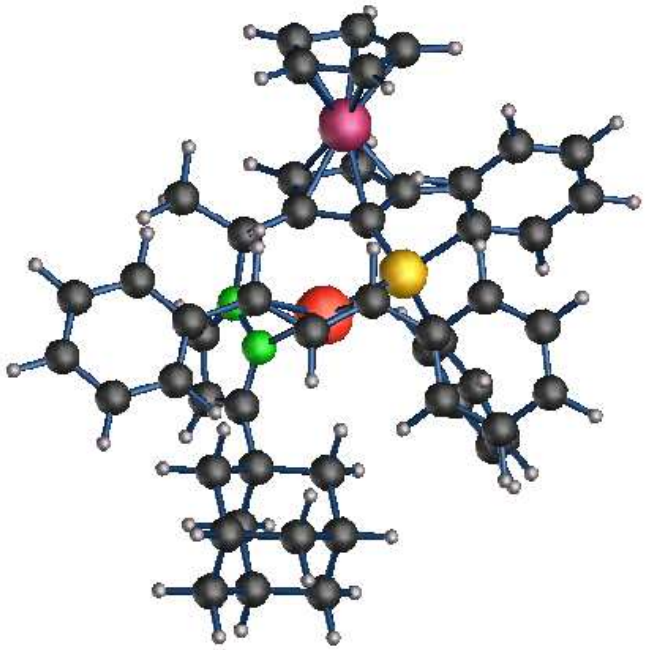
CP-PAW
Installation Guide



Peter E. Blöchl, Clausthal University of Technology
(December 9, 2007)

Contents

Chapter 1

Preface

This installation guide goes back to a version of Clemens Först, who was the first to set up a rather comfortable installation procedure. After Clemens left our group, I changed the installation procedure. In this process I also undid some goodies that Clemens introduced, because I was a complete newcomer to configure scripts.

After changing the installation in 2007, it was necessary to write a new installation guide. The first step was taken by Axel Ehrich. Then I took over.

As of now the Guide is still written partly in German. The translation is still in progress.

I want to thank everybody, who contributed to the installation procedure and this guide.

Peter Blöchl

Chapter 2

Installation Guide

2.1 Obtain the PAW distribution

The distribution of the CP-PAW code can be obtained from the CP-PAW web page

`https://orion.pt.tu-clausthal.de/paw`

following the link “Download”. Access to the PAW distribution is restricted to users with a valid license. You can apply for a license on the same web-page. With a valid password you can obtain the codes and the so-called “SETUP-files”, which are needed to perform PAW calculations.

After you have given you Login name and Password you have reached the “CP-PAW Download Page”. Following the link “Sources” on that page you will find the actual distributions. We will denote it in the following as ***paw-distribution***. You should always use the newest version. Even though it is called Beta, it is currentlt our most safe version. The reason for naming it Beta was to leav the option for a more stringent level.)

The ***paw-distribution*** is a gzipped tar file. You can see the contents using the command

`tar tvfz` ***paw-distribution*** `.tgz`

The archive is expanded into the current directory exactly the way it is listed here with the command

`tar xvfz` ***paw-distribution*** `.tgz`

The directory created this way, whihc may be named “paw-beta” will be denoted as ***paw-directory***.

On other linux systems it may necessary to change the extension of the file from `.tgz` to `.tar.gz`. Then one can unzip the file with the command

`gunzip` ***paw-distribution***.tar.gz

The unzipped file is expanded with the command.

`tar xvf` ***paw-distribution***.tar

The directory ***paw-directory*** should look like this:

```
configure
configure.ac
dx
Makefile_bare.in
Makefile_targets.in
parameters
parms.example
parms.g95
README
src
```

2.1.1 Download the Setup files

Once the PAW distribution has been obtained, download the so-called setup files from the PAW Download page. This Tar file is similarly expanded into a directory ***paw-setupdir***. These data files will be needed later to execute the CP-PAW code.

2.2 Installation of Compiler und Libraries

One needs

1. a Fortran90 compiler,
2. a numerical library for linear algebra, which corresponds to BLAS. Examples are BLAS, ACML, MKL, ESSL.
3. a numerical library for linear algebra, which corresponds to LAPACK. Examples are LAPACK, ACML, MKL, ESSL.
4. a numerical library for Fast Fourier Transforms (FFT) such as FFTW, ACML, MKL, ESSL.
5. An implementation of the Message Passing Interface MPI. Examples are MPICH and MVAPICH.
6. depending on the, compiler a separate utility library is required.

These items shall be installed. The reader should consult section 3, which contains additional information.

2.3 Adapt the parameter file

Now the distribution needs to be configured. The configuration creates a set of make files, which are adapted to your system and which take care of the final installation. More information about configuring in general can be found in section 6.20.

The configure script needs a parameter file "***parmfile***", which needs to be adapted. One should start with one of the examples listed in the appendix B. The reader should then proceed to section 4, which contains a detailed description of the parameters.

Note, that the format of the parameter file may not yet be final. Please consult the current installation file.

2.4 Konfigurieren

Once the parameter file has been set up the configuration is done with the command

`./configure --with-parmfile=`***parmfile***

which is executed in the ***paw-directory***. This process will create a set of make files and a subdirectory tree

paw-directory`/bin/`***arch***

within ***paw-directory***. The name ***arch*** should be unique and allows to maintain several installations simultaneously.

A successful configuration is indicated by a line

```
----done!---configuration completed successfully!--------------
```

A typical printout of a configuration can be found in appendix A.

2.5 Make

Next, one executes the command

```
make
```

in the ***paw-directory***. This compiles the sources and prepares the executables.

In the beginning it may be wise to do the installation in small steps so that Problems are observed early. The following sequence may be sensible

```
make docs
make none
make fast
make fast_parallel
make tools
make
```

The documentation is found in ***paw-directory***`/doc`. The binaries are, for example, ***paw-directory/arch***`/paw_fast.x`.

It is possible that optimization creates erroneous results. Therefore the results of `/paw_fast.x` should be compared once with `/paw_fast.x`. The latter does not contain optimization flags.

This completes the installation.

Chapter 3

Required software

- In section 3.1 we will provide information on the installation of a FORTRAN compiler.
- In section 3.3 we will describe the installation of numerical libraries and the Message Passing Interface.
- In section 3.2 we will discuss the Utility library.

In addition to the above one will need

- a GNU C-preprocessor `cpp`
- the GNU make tool. (Under the AIX operating system , the GNU Make is named `gmake`. The default `make` of AIX will not work with our make files.)

3.1 FORTRAN Compiler

PAW is written in FORTRAN90. Later compiler versions such as FORTRAN95 and probably FORTRAN2003 can be used as well.

In teh following, we will mention some compilers that we have gained experience with.

3.1.1 G95

- License: Open source (GNU Licence)
- Source: *http://www.g95.org*
- Problems:
  - We found it necessary to link the g2c library with G95. I found a statement saying that this is no more required, though.

3.1.2 PGI Fortran Compiler

- Suppplier: The Portland Group
- Source: *http://www.pgroup.com/*
- Lizence: commercial
- Problems: non known Problems
- Remarks:
  - The distribution of the PGI compiler already contains precompiled librarues such as LAPACK, ACML, BLAS, MPICH.

3.1.3 IFORT

- Supplier: Intel
- Source: Register on the page *https://welcome.intel.com/Login.aspx*. Then search for “Intel Fortran Compiler”.
- Problems:
  - using the interprocedural optimization IPO the IFORT can produce very large code. This caused the system to exceed the stack size. The result was completely unpredictable behavior. If this is a problem increase the stack size with `ulimit -s unlimited`. The stack size can be controlled with `ulimit -a`.
- Remarks:
  - Very fast.

3.1.4 XLF Compiler

- Supplier: IBM
- Source: *http://www-306.ibm.com/software/awdtools/fortran/xlfortran/features/xlf-linux.html*
- Lizence: commercial
- Remark: Only for the Power architecture of IBM
- Problems: no known problems

3.1.5 GFORTRAN

- Supplier: Free Software Foundation
- Source*http://gcc.gnu.org/fortran/*
- Problems: We did not succed to compile PAW with gfortran.

3.1.6 Absoft Fortran

- Supplier: Absoft
- http://www.absoft.com/
- Problems: We did not succed to compile PAW with gfortran.

3.2 Utility Library

There are a few routines that are not part of teh fortran standard but which are supplied by all compilers in similar form as library. Sometimes they are treated like instrinsic functions as Fortran extensions without the need to link a library.

Die Utility Bibliothek is named differently by different supplierst. IBM calls it “Service and Utility Library”, Intel calls it “Portability Routines”. Traditional it is called Utility Library U77, where 77 is related to the Fortran77 Standard.

Even though the library calls are nearly identical, they differ in the kind parameter of the arguments. In order to avoid the corresponding problems the CP-PAW code contains interfaces to the utility library for the individual compilers. The specific interfaces are selected via the C-preprocessor, if the corresponding parameters have been set in the paameter file.

Consult the user guide of the corresponding compiler to find out if If a utility library needs to be linked.

3.3 Numerical Libraries

CP-PAW depends on numerical libraries for the following three purposes. Often they are combined in asingle library.

- Basic Linear Algebra Subroutines (BLAS): mathematical library for elementary vector-matrix operations such as vector and matrix multiplications.
- Lineare Algebra PACKage (LAPACK): more complex matrix operations such as eigenvalue solvers.
- Fast Fourier Transformation Library (FFT): As the name says: Fast Fourier transformations

These libraries are to a large part responsible for the efficiency of the CP-PAW calculations.

Für die Linearen Algebra Routinen gibt es Standardpakete, nämlich das Linear Algebra PACKage (LAPACK) und die Basic Linear Algebra Subprograms (BLAS) . Die Beschreibung der LAPACK-Routinen findet man unter

`http://www.netlib.org/lapack/`

und die von BLAS unter

`http://www.netlib.org/blas/`

Diese stellen sozusagen einen Standard für entsprechende Bibliotheken dar.

Diese Pakete sind zum Teil in speziellen Bibliotheken enthalten.

- IBM Engineering and Scientific Software Library (ESSL)
- AMD Core Math Library (ACML)
- Intel Math Kernel Library (MKL)

Es ist ratsam, nicht die mit dem Operating System mitgeliefereten Bibliotheken zu nutzen, sondern diese möglichst selber auf dem Zielsystem zu Kompilieren. Dies erzeugt schnellere Binaries.

Wir legen selbst erzeugte Bibliotheken in einer Directory `/home/tools` auf dem jeweiligen Rechner ab. Dies hat Vorteile bei der Installation von Parallelrechnern, bei denen die Bibliotheken nur auf dem Frontendrechner vorgehalten werden.

LAPACK	BLAS	FFT
LAPACK	ATLAS	FFTW
ACML	ACML	FFTW
ACML	ACML	ACML
MKL	MKL	FFTW(MKL)
MKL	MKL	FFTW
ESSL	ESSL	ESSL

3.3.1 ATLAS-BLAS

- Supplier: Open Source
- Name: ATLAS=Automatically Tuned Linear Algebra Software
- License: Open Source
- Source *http://math-atlas.sourceforge.net/*
- Funktions: BLAS
- Installation:
  - After unpacking the ATLAS distribution type make and follow the instructions. Always use the default value ([y] or [n] by just typing ENTER) until you arrive at

    ```
    use express setup? [y]:
    ```

Enter `no` and proceed taking the defaults if you like so. Use `f90` as FORTRAN77 compiler (just needed to compile the wrappers). As `F77 FLAGS` use

```
-YEXT_NAMES=LCS -YEXT_SFX=_ -O
```

to ensure, that the linking works out.
Again take the default values until you reach the

```
Enter C Flags (CCFLAGS) [-fomit-frame-pointer -O3 -funroll-all-loops]:
```

prompt. Just use `-fomit-frame-pointer -O` and proceed accepting the defaults.
If you have compiled an ATLAS BLAS for different architectures (e.g. Pentium and ATHLON), the corresponding libraries will be in different subdirectories of the ATLAS distribution. You find these subdirectories in `ATLAS/lib`. If there is just one, the configure script will chose it automatically.

3.3.2 LAPACK

- Name: LAPACK=Linear Algebra PACKage
- Source: http://www.netlib.org/lapack/
- Funktions: LAPACK
- Reamrk: This library is not recommended. It si better to use the libraries specifically tuned to your hardware.
- Installation:
  1. Download *lapack-lite-3.1.1.tgz* von http://www.netlib.org/lapack/index.html
  2. gunzip and untar the file
  3. Copy and edit the file LAPACK/make.inc.example to LAPACK/make.inc.
  4. Edit the file LAPACK/Makefile (see http://www.netlib.org/lapack/lawn81/node13.html)
  5. type make

3.3.3 FFTW

- Supplier: Massachusetts Institute of Technology (MIT)
- Name: FFTW=Fastest Fourier Transform in the West
- Source: `http://www.fftw.org/download.html`
- Reamrk
  - The versions FFTW2 und FFTW3 are not compatible. Currently we only support FFTW2.
- Installation:
  1. download FFTW 2.1.5 from `http://www.fftw.org/download.html`.
  2. Archiv mit `tar -xvzf Dateiname` entpacken
  3. Den Befehl `./configure --enable-i386-hacks` ausführen. The option `--enable-i386-hacks` takes advantage if rge gcc/x86 specific performance hacks. (Beim core2duo muss die Option `--enable-i386-hacks` weggelassen werden!)
  4. Folgende Zeile in der *fftw/config.h* ändern (am Ende der Datei):

     ```
     \#define F77_FUNC_(name,NAME) name ## __
     ```

     dort den letzten Unterstrich entfernen, so dass daraus die folgende Zeile wird:

     ```
     \#define F77_FUNC_(name,NAME) name ## _
     ```
  5. Den Befehl *make* ausführen

3.3.4 ACML

- Name: ACML=AMD Core Math Library
- Supplier: AMD
- Source: `http://developer.amd.com/acml.jsp`
- Contains: FFT, LAPACK, BLAS
- Installation: Untar the archive after downaloading and follow the installation instructions.

3.3.5 MKL

- Supplier: Intel
- Name: MKL=Math Kernel Library
- Source: `http://www.intel.com/cd/software/products/asmo-na/eng/307757.htm`
- Installation:
  1. Register at `https://welcome.intel.com/Login.aspx`
  2. http://www.intel.com/cd/software/products/asmo-na/eng/307757.htm
- When using the MKL, the library “pthread” must be linked. (See also appendix 6.6)

FFTW der MKL

Die MKL enthält ein Interface für FFTW Aufrufe, und stellt damit auch die Routinen für Fouriertransformationen zu Verfügung.

Um die FFTW von der MKL nutzen zu können, muss man in das Unterverzeichnis *interfaces/fftw2xf* der MKL wechseln und dort das Kommando *make* ausführen. Benötigt wird dafür der Intel Fortran Compiler und der Intel C Compiler. Es wird dann die Bibliothek *libfftw2xf_intel.a* im entsprechenden *mkl-libs* Verzeichnis erzeugt, welche in PAW eingebunden werden kann, indem man die fftw Pfade im Parameterfile anpasst. Desweiteren muss unter den Punkten *LIBS_SCALAR* und *LIBS_PARALLEL* der Wert *fftw* durch *fftw2xf_intel* ersetzt werden.

3.3.6 ESSL

- Supplier: IBM
- Source: `http://www-03.ibm.com/systems/p/software/essl/index.html`
- Anmerkung:
  - Nur für IBM Hardware. Nur für AIX operating system oder Linux für Power achitekture.
  - ESSL verwendet nicht dieselben Calling sequences wie LAPACK und BLAS. PAW besitzt aber spezielle Interfaces für ESSL.
- Covers: FFT, LAPACK, BLAS

3.4 Message Passing Interface

Das MPI (Message Passing Interface) ist ein Protokoll, um ein verteiltes (paralleles) Rechnen zu ermöglichen. Das MPI muss so compiliert sein, das seine Funktiosaufrufe mit einem Underscore funktionieren!

Die meisten Hardware Hersteller bieten eigene MPI Implementierungen an. Es gibt aber auch freie MPI implementierungen wie MPICH und Open MPI*http://www.open-mpi.org/*. Unsere Erfahrungen beschränken sich auf MPICH bzw MVAPICH, sowie die MPI von IBM.

3.4.1 MPICH

MPICH ist eine freie Implementierung des MPI für Ethernet.

- Hersteller: Argonne National Laboratory and Missisippi State University
- Source http://www-unix.mcs.anl.gov/mpi/mpich1/
- Anmerkungen:
  - Inzwischen existitert eine neue Implementierung MPICH2, die heute empfohlen wird. Sie unterstützt Cluster aus Single und SMP Knoten. MPICH2 kann unter *http://www-unix.mcs.anl.gov/mpi/mpich2/index.htm* heruntergeladen werden.
- Installation:
  1. Download der neuesten version unter

     http://www-unix.mcs.anl.gov/mpi/mpich1/
  2. Nach dem Entpacken kann man mit folgendem Script MPI für PAW compilieren:

     ```
     #!/bin/sh
     export F90=g95
     export F90FLAGS=-fno-second-underscore
     export FC=g95
     export FFLAGS=-fno-second-underscore
     export FLINKER=g95
     export RSHCOMMAND=ssh
     ./configure --enable-f77
     ```

     Für Pathscale

     ```
     #!/bin/sh
     export CC=pathcc
     export FC=pathf90
     export F90=pathf90
     export F90FLAGS=-fno-second-underscore
     export FFLAGS=-fno-second-underscore
     export FLINKER=pathf90
     export RSHCOMMAND=ssh
     ./configure -c++=pathcc -opt=-O3 --enable-f90 --enable-f90modules \
                 --with-romio --disable-weak-symbols
     ```
  3. *make* ausführen

3.4.2 MVAPICH

Freie MPI Implementierung für ein Infiniband Netzwerk. Sie basiert auf MPICH.

- Hersteller: Ohio State University
- Source *http://mvapich.cse.ohio-state.edu/*

Chapter 4

The Parameter File

The parameter file is needed for the configuration of the compilation process of CP-PAW. The parameter file selects the compiler, its options, the numerical libraries, etc.

An example is given in the following section 4.1 and discussed later.

4.1 Example for a Parameter File

```
#################################################################################
##_____________________________architecture (arbitrary name)_____________________________##
ARCH="g95_guam"
##_____________________________flag for uppercase or lower case module file names____##
TUPPERCASEMOD="F"
##_____________________________flag for parallelization_____________________________##
TPARALLEL="T"
##_______special rules for the configure script and f90 preprocessor__________##
SPECIAL="none"
##_____________________DIRECTORIES containing LIBRARIES_____________________##
BLASDIR=""
LAPACKDIR="/home/tools/libs/acml3.0.0_64/gnu64/lib/"
FFTDIR="/home/tools/libs/fftw-2.1.5_no-second-underscore/"
MPIDIR="/home/tools/libs/mpich-1.2.6/"
##_____________________________ include file for fftw_____________________________##
FFT_HEADER="${FFTDIR}/fortran/fftw_f77.i"
##_____________________________________ include file for mpi "mpif.f90"___________##
MPI_HEADER="${MPIDIR}/include/mpif.h"
##_______F90 compiler and linker for scalar executables_____________________##
COMPILER_SCALAR="g95 -fno-second-underscore "
##_______F90 compiler and linker for parallel executables___________________##
COMPILER_PARALLEL="g95 -fno-second-underscore "
##_____________________standard compiler flags_____________________________##
FCFLAGS_NONE="-c "
#_____________________compiler flags for optimization_____________________##
FCFLAGS_OPT="-c -O3 -fshort-circuit -funroll-loops -fomit-frame-pointer -msse2"
##_____________________compiler flags for  profiling_____________________##
FCFLAGS_PROF="-c -pg -O3 -fshort-circuit -funroll-loops  -msse2"
#_____________________compiler flags for debugging_____________________##
FCFLAGS_DBG="-c -g -std=f95 -Wall -ftrace=full  -fimplicit-none -fbounds-check"
#_____________________flags for linking_____________________________##
LDFLAGS_SCALAR="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR}  \
                -L${FFTDIR}/fftw/.libs/"
#_____________________flags for linking_____________________________##
```

```
LDFLAGS_PARALLEL="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
                  -L${FFTDIR}/fftw/.libs/ -L${MPIDIR}/mpe/lib"
#_____________________________external libraries (sequential)_____________________________##
LIBS_SCALAR="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c"
#_____________________external libraries (parallel)_____________________________________##
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c -Wl,-dy -lmpe \
               -Wl,-dy -llmpe"
#_____________________________preprocessor variables_____________________________________##
CPPFLAGS="-DCPPVAR_COMPILER_G95 -DCPPVAR_FFT_FFTW \
          -DCPPVAR_LAPACK_LAPACK -DCPPVAR_BLAS_BLAS"
#_____________________________file extension_____________________________________________##
FEXT="f90"
##############################################################################################
```

4.2 Explanation of the Variables in the Parameter File

4.2.1 ARCH

The value is a directory name, which will be created as

paw-directory`/bin/`***arch***

This directory will contain all executables. It will also contain subdirectories with the object files, module files,etc.

4.2.2 TUPPERCASEMOD

Different compilers name the module files either with uppercase letters or lowercase letters. By specifying `TUPPERCASEMOD='T'` for uppercase letters or `TUPPERCASEMOD='F'` for lowercase letters the make files are able to detect the correct dependencies. An incorrect setting will result in unnecessary compilation of files that are unaffected by a certain change of the source code.

To find out about the proper setting, compile first with an arbitrary value. Then inspect the module files created with the command

`ls` ***paw-directory***`/bin/`***arch***`/Objects/fast/*.mod`

and adapt the parameter file correspondingly.

4.2.3 TPARALELL

This logical variable specifies if an executable for parallel computers shall be built. A requirement is a corresponding MPI library. Specify `TPARALELL='F'` to create only sequential libraries.

4.2.4 SPECIAL

Leave the setting `SPECIAL=' '`. It is a flag for special rules in the make files.

4.2.5 BLASDIR,LAPACKDIR,FFTDIR,MPIDIR

This set of variables specifies the path for the various libraries. These variables can be used within the parameter file. They are not used themselfes by the make files.

The reason for this construction is that configure does not recognize intermediate variables in the parameter file. The variables defined here will be specified in the make files, so that their values can be used.

4.2.6 FFT_HEADER

Path to the include file `fftw_f77.i` for the FFTW library. This file contains constants , that need passed to the FFTW routines.

If FFTW2 is used as described in section 3.3.3, no change is needed.

4.2.7 MPI_HEADER

Path to the include file `mpif.h` for MPI. This file must be supplied by the MPI Distribution. No change is required if MPICH is used.

Was ist wenn das file nicht existiert, aber auch nicht benötigt wird?

Newer versions such as MPICH-2 offer a module file, which is more consistent with the Fortran90 standard. This file includes explicit interfaces for the Fortran calls to MPI. Unfortunately, only a small subset of the interfaces require by CP-PAW are supported. This is the reason to use the more conventional method of an include file to set the MPI-specific parameters.

4.2.8 COMPILER

This variable specifies the command to call the compiler for the sequential and the parallel executables.

Compiler options that are used generally can be integrated here with the compiler as in the present example: The G95 compiler used in this example requires the option *-fno-second-underscore* in order to link the libraries properly.

4.2.9 FCFLAGS_NONE, FCFLAGS_OPT, FCFLAGS_DBG, FCFLAGS_PROF

Here the compiler options are specified. Different sets of compiler options are used to create different executables.

- FCFLAGS_NONE: The most simple set of compiler options. The executables are to explore the dependency on the compiler options. In particular to test the results of a highly optimized version.

sequential executable	***paw-directory***`/bin/`***arch***`/paw.x`
parallel executable:	***paw-directory***`/bin/`***arch***`/ppaw.x`
sequential object directory:	***paw-directory***`/bin/`***arch***`/Objects/none`
parallel object directory:	***paw-directory***`/bin/`***arch***`/Objects/none_parallel`

- FCFLAGS_OPT: This set of compiler options should be the highest level of optimization. The resulting exectubles shall be the ones to be used during normal production.

sequential executable	***paw-directory***`/bin/`***arch***`/paw_fast.x`
parallel executable:	***paw-directory***`/bin/`***arch***`/ppaw_fast.x`
sequential object directory:	***paw-directory***`/bin/`***arch***`/Objects/fast`
parallel object directory:	***paw-directory***`/bin/`***arch***`/Objects/fast_parallel`

- FCFLAGS_DBG: The is teh set of compiler option for debugging. Besides the parameter -g it should include all reasonable tests such as array-bound checking. No optimization shall be selected.

sequential executable	***paw-directory***`/bin/`***arch***`/paw_dbg.x`
parallel executable:	***paw-directory***`/bin/`***arch***`/ppaw_dbg.x`
sequential object directory:	***paw-directory***`/bin/`***arch***`/Objects/dbg`
parallel object directory:	***paw-directory***`/bin/`***arch***`/Objects/dbg_parallel`

- FCFLAGS_PROF: This is the set of compiler options for profiling. The parameters should include the parameter "-pg" and all optimizations.

sequential executable	***paw-directory***`/bin/`***arch***`/paw_prof.x`
parallel executable:	***paw-directory***`/bin/`***arch***`/ppaw_prof.x`
sequential object directory:	***paw-directory***`/bin/`***arch***`/Objects/prof`
parallel object directory:	***paw-directory***`/bin/`***arch***`/Objects/prof_parallel`

The compiler options depend on the compiler. Some, use fro optimizations, also depend on the computer architecture and the CPU.

4.2.10 LDFLAGS_SCALAR,LDFLAGS_PARALLEL

LDFLAGS_PARALLEL is used only, if TPARALLEL=''T''.

These are the parameter sets for the loader. In particular one specifies the search paths for the libraries. The libraries may differ for the sequential and the parallel executable.

4.2.11 LIBS_SCALAR,LIBS_PARALLEL

LIBS_PARALLEL is used only, if TPARALLEL=''T''.

This variable specifies the libraries that should be linked.

With -Wl,-dn and -Wl-dy one specifies whether the libraries are linked statically or dynamically. A statically linked library -lfoo points to libfoo.a A dynamically linked library -lfoo points to libfoo.so The default should be to link dynamically. However some libraries cannot be linked dynamically. If the executable is to be osed on a different computer, static linking is mandatory.

The library g2c is needed by the G95 compiler. The library is contained foer SUSE Linux in the package *compat-g77*.

The MKL library also requires the libraries "libguide" and "libpthread". pthread is a native linux library used by libguide. libguide provides multithreading support within MKL. It is important that pthread is specified as last item in the link line.

```
-lmkl_lapack -lmkl_em64t -lguide -lpthread
```

When linking the atlas blas library, one does need to specify -lf77blas -latlas. ATLAS is a C-library and f77blas is a Fortran interface to the C-subroutines.

4.2.12 CPPFLAGS

Flags for the C-preprozessor. Using these flags, the C-preprozessor selects certain code segments. They are mostly used to select the interfaced to external libraries. The interfaces are located in ***PAW-directory***/src/paw_library.f90.

- The variable -DCPPVAR_COMPILER_*foo* selects the interface to the Utility library. The Utility library contains Fortran interfaces to system routines, which are written in C and which are part of the operating system Allowed values are
  - CPPVAR_COMPILER_G95
  - CPPVAR_COMPILER_IFC
  - CPPVAR_COMPILER_IFC7
  - CPPVAR_COMPILER_ABSOFT
  - CPPVAR_COMPILER_XLF
  - CPPVAR_COMPILER_PGI
  - CPPVAR_COMPILER_PATHSCALE
- The variable CPPVAR_FFT_foo selects the interfaces to the Fourier transform library. Allowed values are:
  - CPPVAR_FFT_FFTW
  - CPPVAR_FFT_ESSL
  - CPPVAR_FFT_ACML
- The variable CPPVAR_LAPACK_foo selects the interfaces to the LAPACK routines or equivalent library routines.

  Allowed values for CPPVAR_LAPACK_foo are:
  - CPPVAR_LAPACK_ESSL
  - CPPVAR_LAPACK_LAPACK: Default behavior. The LAPACK interfaces are used.

- The variable CPPVAR_BLAS_foo selects the interfaces to BLAS routines or equivalenter library routines. Allowed values for CPPVAR_BLAS_foo are:
  - CPPVAR_BLAS_ESSL
  - CPPVAR_BLAS_BLAS: Default behavior. The BLAS interfaces are used.

4.2.13 FEXT

Extension expected by the compiler for the source code files. It is usually “f90” or “f95”.

Chapter 5

Liste der Targets des Make files

Das Makefile erkennt die folgenden Targets:

- none Erzeugt ein sequentielles Binary ohne Optimierungen. Hiermit sollte beim erstmalligen installieren angefangen werden.
- dbg
- fast Mit dem Kommando *make fast* werden im Verzeichnis *bin/$arch* die Objekte und die Programmdateien für ein sequentielles Binary erzeugt.
- prof
- none_parallel
- dbg_parallel
- fast_parallel
- prof_parallel
- clean
- clean_none
- clean_dbg
- clean_fast
- clean_prof
- tools
- docs

Chapter 6

Problems und other Remarks

6.1 Stack-size exceeded

Unpredictable behavior can occur if the stack-size is exceeded. In the bash shell the stack size can be increased using the command

```
ulimit -s unlimited
```

This is apparently a problem of the operating system. It can be caused by certain optimizations such as inlining. The latter increases the code size.

6.2 No core dump

If the code crashes without creating a core dump, the limit for the core file size must be increased using the command

```
ulimit -c unlimited
```

6.3 Second underscore

The compiler translates Fortran names such as subroutine names or variable names into internal symbols. Typically the compiler appends a single underscore to each name, in order to distinguish them from other names. However, some compilers deviate from this standard, if the fortran name already contains one underscore. In the latter case two underscores are attached instead of one. This is the so-called “second-underscore” convention.

> “G95 follows the f2c convention of adding an underscore to public names, or two underscores if the name contains an underscore.” Diese Konvention kann üblicherweise durch Compilerflags geändert werden.

In the following table we denote the “second-underscore” as “su” and the “no-second-underscore” as “nsu”.

Fortran Name	Symbol su	Symbol nsu
`abc`	`abc_`	`abc_`
`a_b_c`	`a_b_c__`	`a_b_c_`
`abc_`	`abc___`	`abc__`
`a_b_c_`	`a_b_c___`	`a_b_c__`

Apparently there are problems if one links a library, that has been created with a different underscoring convention than the rest of the code.

Many external libraries are written in C and have a fortran wrapper. It is important that this wrapper has been compiled with thes same underscoring convention.

In order to explore the symbols that have been created and to check if they are matching, one can inspect the symbol tables using the command, as described in section 6.4.

6.4 Symbol tables of object files and libraries

In order to explore which library routines are called by an object file and which are available in a library one can inspect the symbol tables.

The commmand

```
nm foo.o
```

prints the symbol table of the object file foo.o. Similarly one can inspect a library with

```
nm foo.a
```

6.5 Runtime error in viacheck.c, code=VAPI_RETRY_EXC_ERR

The routine viacheck.c is part of MVAPICH, the MPI implementation for Infiniband networks. The error means that the Infiniband Reliable Connection retry Coundf was exceeded. This may occur if there is a bad cable or port on the hardware. It may also occur if the code undergoes a segmentation fault, so that the job is not stopped on all nodes. On those nodes the job fails, because it does not receive the required response.

6.6 Missing library pthread

`libpthread` is a system library required by the MKL library.

Under the SUSE-Linux we had problems with the pthread library. The static SUSE pthread library (`/usr/lib/libpthread.a`) is buggy. The problem is usually solved by linking the pthread library dynamically, that is with "`-Wl,-dy -lpthread`". Another workaround has been to use the pthread library from Debian or Redhat. This library is copied to, for example, `/usr/lib/libpthread-debian.a`, and then linked using `-lpthread-debian` instead of `-lpthread`.

6.7 Missing library g2c

If the linker complains about "undefined references" in "`xerbla.o`" it is a sign that the g2c library is missing or not linked properly.

"`g2c`" is the GNU Fortran-to-C converter, which is the basis of the G77 compiler. In older times, this converter was called "`f2c`" (libf2c). The library "`g2c` is usually part of "`g77`" or the "`gfortran`" package.

I found the following remark on the WWW.

"Library g2c is the Fortran 77 shared library needed to run Fortran 77 dynamically linked programs. The library is no longer needed with the new family of Fortran (native) compilers like g95. To correct the error, we searched first of all for the shared version of g2c in /usr/lib*:"

6.8 MPI: rsh versus ssh

The parallelization requires a method to communicate between different machines. Two possibilities exist, namely via rsh and ssh. rsh is a bit faster and ssh is a lot safer. Therefore the use of ssh is strongly recommended.

6.9 Dynamic versus static linking

Libraries can be linked dynamically or statically. A statically linked library is completely integrated with the binary. If the binary is to be used on another computer, it is important to link the libraries statically.

The code size is smaller when the libraries are linked dynamically, that is during runtime. In that case only an interface to a shared library is integrated into the binary. If one copiesthe executable to another machine, it is important that the shared libraries are available and identical to those on the original machine.

The preferred mode is dynamic linking.

It is possible to convert a static library into a dynamic library. Use teh command

```
ls -z allextract *.a *.so
```

6.10 Multiple definition of ...

- (To `paw_...` This problem should be absent in the current implementation. The possible cause is that paw routines are linked twice, once directly and once as part of the paw librart libpaw.a. This may happen when linking the PAW tools.
- Do not link both, blas and mkl (or blas and acml). Both contain the BLAS routines

6.11 Cannot find lf77blas

Set the correct path to the library libf77blas.a. It is part of the ATLAS package.

6.12 PMPI_Allreduce

This problem is related to the PATHSCALE compiler:

If the linker complains that it does not find routines starting with PMPI it helps if one also links the library `libpmpich`. The library commands are the "`-lfmpich -lmpich -lpmpich`". (Solution obviously for MPICH only.)

6.13 Linker flag -I does not work

Some time ago the ABSOFT compiler did not accept the -l flag to set the search path include files. The current installation procedure takes care of this.

6.14 cannot find -lvapi

vapi is a library used by the infiniband drivers. (Infiniband is a network protocoll that can be used by MPI).

The solution is to link vapi dynamically. `-Wl,-dy -lvapi`.

6.15 Cannot find library gcc_s

This problem is related to the PATHSCALE compiler:

Simply add "`-lgcc_s`" to the list of libraries.It may be necessary to supply also the path. On my system itr is in "`-L/usr/lib64/gcc/x86_64-suse-linux/4.1.2/`".

6.16 P4_GLOBMEMSIZE

For a parallel job, I obtain the following error message:

```
p1_6605: (2.488281) xx_shmalloc: returning NULL; requested 4767920 bytes
p1_6605: (2.488281) p4_shmalloc returning NULL; request = 4767920 bytes
You can increase the amount of memory by setting the environment variable
P4_GLOBMEMSIZE (in bytes); the current size is 4194304
p1_6605:  p4_error: alloc_p4_msg failed: 0
```

Include a statement into your .bashrc file to increase the variable `P4_GLOBMEMSIZE`

```
export P4_GLOBMEMSIZE=16777216
```

This seems to be the maximum error one can use under linux.

6.17 Resources exhausted

If many parallel jobs are run ok and suddenly parallel jobs crash run the following script.

```
#/bin/sh
ipcs -m | awk '/^ *0x/ {print $2 }' | xargs -n 50 ipcrm shm
ipcs -s | awk '/^ *0x/ {print $2 }' | xargs -n 50 ipcrm sem
```

6.18 FFTW

FFTW version 3 (FFTW3) is not compatible with older versions (FFTW2). FFTW3 cannot yet be used with PAW.

6.19 Informationen über das System

Um zum die richtigen Compiler und Bibliotheken herunterzuladen und für bestimmte Lizenzen, sind einige Informationen uber das aktuelle System notwendig.

6.19.1 Rechnername

- "`hostname`" provides the computer name.
- "`hostname -i`" provides the computer name. provides the IP addresses
- "`hostname -d`" provides the internet domain name.

6.19.2 MAC Adress

Die Media Access Control (MAC) Adresse oder Ethernet ID wird manchmal benötigt um eine Lizenz speziell für einen Rechner ausstellen zu können. Sie ist eine eindeutige Nummer für jede Netzwerkkarte. Die MAC Adresse besteht aus 6 Byte und wird häufig hexadezimal geschrieben, z.B. `08:00:20:AE:FD:7E`.

Um die MAC Adresse zu erhalten, führt man das Kommando

```
/sbin/ifconfig -a
```

aus. Man erhält information zu allen Netzwerkkarten. Die Ethernet Hardware Adresse ist die MAC Adresse.

6.19.3 Rechnerarchitektur

Das Linux Kommando "`arch`" oder "`uname -m`" lieferet die Rechnerarchitektur. z.B. x86_64.

Die wichtigsten Rechnerarchitekturen sind:

- IA-32 (Intel Architecture 32bit), i386, x86 ist die seit 87 von Intel verwendete Prozessorarchitektur bis zur Einführung der 64bit Rechner.
- x86-64, AMD64, EM64T, IA-32e, x64 Architektur der ersten 64-bit Prozessoren von AMD. EM64T steht für "Extended Memory 64 Technology".
- IA-64 ist die volkommen neue 64 Architektur von Intel. Sie kommt bei den Itanium Prozessoren zum Einsatz. IA-64 steht für "Intel Architecture 64".
- POWER ist die Architektur der RISC Prozessoren von IBM. Power steht hier für Performance optimized with enhanced RISC)

6.19.4 Linux version

Unter Suse erhält man die Versionsnummer durch `cat /etc/SuSE-release`. Die Kernel version erhält man durch `uname -r`

6.20 Hintergrund zum Configure Skript

Es mag von Interesse sein sein, zu verstehen, woher das configure script kommt, welches die Konfiguration vornimmt.

6.20.1 What does the configure script do

The configuration helps the user to compile the PAW-Code without having to find out about a lot of parameters himself. The configure script explores the availability of the compilers and libraries, and it sets the corresponding compiler flags and preprocessor variables. In the current version, however, we have to specify most parameters explicitely in a parameter file.

The configure script uses the parameter file ***parmfile*** in order to construct the necessary Makefiles from corresponding templates. The templates in use are

- `Makefile_targets.in` is converted in the `Makefile` located in the ***PAW-directory***. This is the primary makefile executed by the user.
- `Makefile_bare.in` is converted into `Makefile_bare` located in the ***PAW-directory***. However this make file is never executed itself, but it is itself only a template for the makefiles located in the directories

  paw-directory`/bin/`***arch***`/Objects/`***type***

  where ***type*** is one of `none`,`dbg`, `fast`, `prof`, `none_parallel`, `dbg_parallel`, `fast_parallel`, `prof_parallel`.

The configure script makes a copy of the templates for the Makefiles and replaces strings of the type `@VARIABLE@` in the copy of the template by the corresponding value. The variables are identifies by the two `@` at the beginning and the end.

In addition, the configure script constructs a directory Tree that will hold the specific makefiles, objects, binaries etc.

6.20.2 How the configure script is constructed

The configure script is constructed with the help of the GNU `autoconf` tool . The `autoconf` tool uses an input file `configure.ac`, which describes the configuration process in the `autoconf` macro language.

Especially important is the first part with the *user adaptable variables*. Here all the values can be set - the rest of the script just uses the variables. This is the place to make permanent changes to the installation scheme (e.g. change the default compiler flags).

By invoking

```
autoconf
```

in the PAW directory the `configure` file – which is a `/bin/sh` script – will be generated.

Appendix A

Output produced by the configure script

If the configure script completes successfully, the output looks like this.

```
checking for parms.g95_guam... yes
check for make
checking for gmake... gmake
checking for cpp... cpp
checking for g95... yes
checking for xlf90... no
checking for ifort... no
checking for ifc... no
checking for f90... yes
checking for fort... no
resolve parms.g95_guam
${MPI_HEADER}

============================
check MPI directory
checking for /home/tools/libs/mpich-1.2.6/... yes
check FFT directory
checking for /home/tools/libs/fftw-2.1.5_no-second-underscore/... yes
copy parms.g95_guam to parms.in_use
creating subdirectories and copying shell scripts
configure: creating ./config.status
config.status: creating /home/ptpb/Tree/PAW/main/bin/g95_guam/f90pp
config.status: creating Makefile
config.status: creating Makefile_bare
modify makefile_bare for lowercase module files
creating Makefile in none
creating Makefile in none_parallel
creating Makefile in fast
creating Makefile_parallel in fast
creating Makefile in dbg
creating Makefile_parallel in dbg
creating Makefile in prof
creating Makefile_parallel in prof
-------------------------------------------------------------
----------------------SUMMARY--------------------------------
-------------------------------------------------------------
directory of distribution      : /home/ptpb/Tree/PAW/main
directory with binaries        : /home/ptpb/Tree/PAW/main/bin/g95_guam
architecture                   : g95_guam
```

```
preprocessor variables         : -DCPPVAR_COMPILER_G95 -DCPPVAR_FFT_FFTW  -DCPPVAR_LAPAC
architecture name              : g95_guam
parallel environment           : T
compile command (scalar)       : g95 -fno-second-underscore
F90 file extension             : f90
compile flags (none)           : -c
compile flags (fast)           : -c -O3 -fshort-circuit -funroll-loops -fomit-frame-poir
compile flags (dbg)            : -c -g -std=f95 -Wall -ftrace=full  -fimplicit-none -fbo
compile flags (prof)           : -c -pg -O3 -fshort-circuit -funroll-loops  -msse2
link command w.flags (scalar)  : g95 -fno-second-underscore  -Wl,-dy -I${OBJDIR} -L${OBJ
external libraries (scalar)    : -Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c
uppercase module names?        : F
blas library                   :
lapack library                 : /home/tools/libs/acml3.0.0_64/gnu64/lib/
libs for Fourier transforms    : /home/tools/libs/fftw-2.1.5_no-second-underscore/
parallel envirnment considered : yes
compile command (parallel)     : g95 -fno-second-underscore
link command w. flags(parallel): g95 -fno-second-underscore  -Wl,-dy -I${OBJDIR} -L${OBJ
external libraries (parallel)  : -Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c -Wl,-dy -lf
mpi library                    : /home/tools/libs/mpich-1.2.6/
MAKE command                   : gmake
CPP command                    : cpp  -traditional
-----------------------------------------------------------------
----done!---configuration completed successfully!--------------
-----------------------------------------------------------------
```

Appendix B

Examples for Parameter Files

B.0.3 Example of a Parameter File for a Simple Installation

This is a simple example for a parameter file

```
ARCH="G95_mycomputer"
TPARALLEL="T"
TUPPERCASEMOD="F"
TPARALLEL="T"
SPECIAL="none"
BLASDIR=""
LAPACKDIR="/opt/acml4.0.1/libs/acml3.0.0_64/gnu64/lib/"
FFTDIR=""
MPIDIR="/opt/mpich-1.2.6/"
FFT_HEADER=" "
MPI_HEADER="${MPIDIR}/include/mpif.h"
COMPILER_SCALAR="g95 -fno-second-underscore "
COMPILER_PARALLEL="g95 -fno-second-underscore "
FCFLAGS_NONE="-c "
FCFLAGS_OPT="-c -O3 -fshort-circuit -funroll-loops -fomit-frame-pointer -msse2"
FCFLAGS_PROF="-c -pg -O3 -fshort-circuit -funroll-loops  -msse2"
FCFLAGS_DBG="-c -g -std=f95 -Wall -ftrace=full  -fimplicit-none -fbounds-check"
LDFLAGS_SCALAR="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} "
LDFLAGS_PARALLEL="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR}  -L${MPIDIR}/lib"
LIBS_SCALAR="-Wl,-dn -lacml -Wl,-dy -lg2c"
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c \
            -Wl,-dy -lfmpich -Wl,-dy -lmpich"
CPPFLAGS="-DCPPVAR_COMPILER_G95 -DCPPVAR_FFT_ACML  \
    -DCPPVAR_LAPACK_LAPACK "-DCPPVAR_BLAS_BLAS "
FEXT="f90"
```

B.1 Parameter File for G95

```
ARCH="g95_guam"
TUPPERCASEMOD="F"
TPARALLEL="T"
SPECIAL="none"
BLASDIR=""
LAPACKDIR="/home/tools/libs/acml3.0.0_64/gnu64/lib/"
FFTDIR="/home/tools/libs/fftw-2.1.5_no-second-underscore/"
MPIDIR="/home/tools/libs/mpich-1.2.6/"
FFT_HEADER="${FFTDIR}/fortran/fftw_f77.i"
```

```
MPI_HEADER="${MPIDIR}/include/mpif.h"
COMPILER_SCALAR="g95 -fno-second-underscore "
COMPILER_PARALLEL="g95 -fno-second-underscore "
FCFLAGS_NONE="-c "
FCFLAGS_OPT="-c -O3 -fshort-circuit -funroll-loops -fomit-frame-pointer -msse2"
FCFLAGS_PROF="-c -pg -O3 -fshort-circuit -funroll-loops  -msse2"
FCFLAGS_DBG="-c -g -std=f95 -Wall -ftrace=full  -fimplicit-none -fbounds-check"
LDFLAGS_SCALAR="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR}\
      -L${FFTDIR}/fftw/.libs/"
LDFLAGS_PARALLEL="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
        -L${FFTDIR}/fftw/.libs/ -L${MPIDIR}/lib"
LIBS_SCALAR="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c"
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lg2c \
         -Wl,-dy -lfmpich -Wl,-dy -lmpich"
CPPFLAGS="-DCPPVAR_COMPILER_G95 -DCPPVAR_FFT_FFTW  \
         -DCPPVAR_LAPACK_LAPACK "-DCPPVAR_BLAS_BLAS "
FEXT="f90"
```

B.2 Parameter File for IFORT

```
ARCH="ifc10_guam"
TUPPERCASEMOD="F"
TPARALLEL="T"
SPECIAL="none"
BLASDIR=""
LAPACKDIR="/opt/intel/mkl/9.1/lib/em64t/"
FFTDIR="/home/tools/libs/fftw-2.1.5_no-second-underscore/"
MPIDIR="/home/tools/libs/mpich-1.2.6_ifc10/"
FFT_HEADER="${FFTDIR}/fortran/fftw_f77.i"
MPI_HEADER="${MPIDIR}/include/mpif.h"
COMPILER_SCALAR="ifc10"
COMPILER_PARALLEL="${MPIDIR}/bin/mpif90 -choicemod "
FCFLAGS_NONE="-c "
FCFLAGS_OPT="-c -O2 -fast -finline-functions -finline-limit=50"
FCFLAGS_PROF="-c -pg -O3 "
FCFLAGS_DBG="-c -g -check bounds -check format -check pointers \
            -check uninit -debug full -debug-parameters all \
            -fp-stack-check -ftrapuv -stand f95 -traceback \
             -warn declarations"
LDFLAGS_SCALAR="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
               -L${FFTDIR}/fftw/.libs/"
LDFLAGS_PARALLEL="-Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
                 -L${FFTDIR}/fftw/.libs/"
LIBS_SCALAR="-Wl,-dn -lfftw -Wl,-dn -lmkl_lapack -Wl,-dn -lmkl_em64t \
             -Wl,-dn -lguide  -Wl,-dy -lpthread  -Wl,-dy -lg2c"
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dn -lmkl_lapack -Wl,-dn -lmkl_em64t \
               -Wl,-dn -lguide  -Wl,-dy -lpthread -Wl,-dy -lg2c "
CPPFLAGS="-DCPPVAR_COMPILER_IFC -DCPPVAR_FFT_FFTW \
          -DCPPVAR_LAPACK_LAPACK -DCPPVAR_BLASK_BLAS "
FEXT="f90"
```

B.3 Parameter File for PGI

```
ARCH="pgi_guam"
TUPPERCASEMOD="F"
TPARALLEL="T"
SPECIAL="none"
BLASDIR=""
LAPACKDIR="/opt/pgi/linux86-64/7.1/lib"
FFTDIR="/home/tools/libs/fftw-2.1.5_no-second-underscore/"
MPIDIR="/opt/pgi/linux86-64/7.1/mpi/mpich/"
FFT_HEADER="${FFTDIR}/fortran/fftw_f77.i"
MPI_HEADER="${MPIDIR}/include/mpif.h"
COMPILER_SCALAR="pgf90 -fpic"
COMPILER_PARALLEL="pgf90 -fpic"
FCFLAGS_NONE=" -c "
FCFLAGS_OPT="-c -fast -fastsse -Mipa=fast,inline"
FCFLAGS_PROF="-c -pg -fast -fastsse -Mipa=fast,inline"
FCFLAGS_DBG="-c -g -Mlist -m -C -Mbounds "
LDFLAGS_SCALAR=" -g77libs -Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
                -L${FFTDIR}/fftw/.libs/"
LDFLAGS_PARALLEL="-g77libs -Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
                 -L${FFTDIR}/fftw/.libs/ -L${MPIDIR}/lib"
LIBS_SCALAR="-Wl,-dn -lfftw -Wl,-dy -lacml  "
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dy -lacml  -Wl,-dy -lfmpich -Wl,-dy -lmpich"
CPPFLAGS="-DCPPVAR_COMPILER_PGI -DCPPVAR_FFT_FFTW \
          -DCPPVAR_LAPACK_LAPACK -DCPPVAR_BLAS_BLAS"
FEXT="f90"
```

B.4 Parameter File for PATHSCALE

```
ARCH="pathscale_guam"
TUPPERCASEMOD="T"
TPARALLEL="T"
SPECIAL="none"
BLASDIR=""
LAPACKDIR="/opt/acml4.0.1/pathscale64/lib/"
FFTDIR="/home/tools/libs/fftw-2.1.5_no-second-underscore/"
MPIDIR="/opt/mpich-1.2.7p1_pathscale_ssh"
FFT_HEADER="${FFTDIR}/fortran/fftw_f77.i"
MPI_HEADER="${MPIDIR}/include/mpif.h"
COMPILER_SCALAR="pathf95 -fno-second-underscore  "
COMPILER_PARALLEL="pathf95 -fno-second-underscore  "
FCFLAGS_NONE="-c "
FCFLAGS_OPT="-c -O3   -OPT:Ofast -fno-math-errno -ffast-math "
FCFLAGS_PROF="-c -pg -O3 -profile "
FCFLAGS_DBG="-c -C -g -Wall "
LDFLAGS_SCALAR="-ipa -Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
              -L${FFTDIR}/fftw/.libs/ -L/usr/lib64/gcc/x86_64-suse-linux/4.1.2/ "
LDFLAGS_PARALLEL="-ipa -Wl,-dy -I${OBJDIR} -L${OBJDIR} -L${LAPACKDIR} \
           -L${FFTDIR}/fftw/.libs/  -L/usr/lib64/gcc/x86_64-suse-linux/4.1.2/ \
            -L${MPIDIR}/lib "
LIBS_SCALAR="-Wl,-dn -lfftw -Wl,-dn -lacml -Wl,-dy -lgcc_s "
LIBS_PARALLEL="-Wl,-dn -lfftw -Wl,-dn -lacml  -Wl,-dy -lfmpich \
               -Wl,-dy -lmpich -lpmpich -Wl,-dy -lgcc_s"
CPPFLAGS="-DCPPVAR_COMPILER_PATHSCALE -DCPPVAR_FFT_ACML  \
          -DCPPVAR_LAPACK_LAPACK -DCPPVAR_BLAS_BLAS "
FEXT="f90"
```

Appendix C

Input Data Files

C.1 Control Input File

```
!CONTROL
 !GENERIC  TRACE=f DT=10.0 NSTEP=180 NWRITE=100 START=t !END
 !DFT    TYPE=10 !END
 !FOURIER EPWPSI=20 CDUAL=2 !END
 !PSIDYN  FRIC=0.005
    !AUTO FRIC(-)=0.3 FACT(-)=0.97
          FRIC(+)=0.3 FACT(+)=1. !END
 !END
!END
!EOB
```

C.2 Structure Input File

```
!STRUCTURE
  !GENERIC LUNIT=10.26 !END
  !KPOINTS DIV=1 1 1 !END
  !OCCUPATIONS EMPTY=5 NSPIN=1  !END
  !LATTICE T= 0.00000  0.50000  0.50000
             0.50000  0.00000  0.50000
             0.50000  0.50000  0.00000 !END
  !SPECIES NAME= 'Si' ZV=4. M=5.  NPRO= 2 2 1 lrhox=2
           FILE='si_.75_6.0.out'
  !END
  !ATOM   NAME= 'Si_1'  R= 0.00  0.00 0.00 !END
  !ATOM   NAME= 'Si_2'  R= 0.25  0.25 0.25 !END
!END
!EOB
```

The string `'si_.75_6.0.out'` needs to be adjusted. It specifies a setup file.

Index