

PWscf

---

Input and Output description

Stefano de Gironcoli

SISSA & DEMOCRITOS – Trieste

Where can I find some useful information about PWscf ?

Where can I find some useful information about PWscf ?

```
prompt > ls $0-sesame_ROOT/pwdocs/
```

BUGS

ChangeLog

democritos.png

INPUT\_CHDENS

INPUT\_D3

INPUT\_NC

INPUT\_PH

INPUT\_PP

INPUT\_PROJWFC

INPUT\_PW

INPUT\_PWCOND

manual.tex

pwscf.png

README

refman.tex

The input file for PWscf is structured in a number of **NAMELISTS** and **INPUT\_CARDS**.

```
&NAMELIST1 ... /
```

```
&NAMELIST2 ... /
```

```
&NAMELIST3 ... /
```

```
INPUT_CARD1
```

```
.....
```

```
.....
```

```
INPUT_CARD2
```

```
.....
```

```
.....
```

**NAMELISTS** are a standard input construct in fortran90.

The use of **NAMELISTS** allows to specify the value of an input variable **only when it is needed** and to define **default values** for most variables that then need not be specified. Variable can be inserted **in any order**.

```
&NAMELIST
```

```
needed_variable2=XX, needed_variable1=X,
```

```
/
```

**NAMELISTS** are read in a specific order

**NAMELISTS** that are not required are ignored

**INPUT\_CARDS** are specific of O-sesame codes and are used to provide input data that are **always needed** and would be boring to specify with the `variable_name=variable_value` syntax used by NAMELIST.

**INPUT\_CARDS** require data in specific order (which may depend on the situation and on the value of a **card\_format\_specifier** )

For instance:

```
INPUT_CARD      card_format_specifier
data(1,1) data(1,2) data(1,3) ...
data(2,1) data(2,2) data(2,3) ...
data(3,1) data(3,2) data(3,3) ...
... ..
```

Logically independent **INPUT\_CARDS** can be given in any order

There are **three mandatory NAMELISTS** in PWscf:

There are three mandatory NAMELISTS in PWscf:

&CONTROL    input variables that control the flux of the calculation and the amount of I/O on disk and on the screen.

There are **three mandatory NAMELISTS** in PWscf:

**&CONTROL**     input variables that control the flux of the calculation and the amount of I/O on disk and on the screen.

**&SYSTEM**     input variables that specify the system under study.

There are **three mandatory NAMELISTS** in PWscf:

**&CONTROL** input variables that control the flux of the calculation and the amount of I/O on disk and on the screen.

**&SYSTEM** input variables that specify the system under study.

**&ELECTRONS** input variables that control the algorithms used to reach the self-consistent solution of KS equations for the electrons.

There are **three additional NAMELISTS** in PWscf that **must** be specified under certain circumstances:

There are **three additional** NAMELISTS in PWscf that **must** be specified under certain circumstances:

**&IONS**        needed when ATOMS MOVE! IGNORED otherwise !  
input variables that control ionic motion in  
molecular dynamics run or structural relaxation

There are **three additional NAMELISTS** in PWscf that **must** be specified under certain circumstances:

**&IONS**        needed when ATOMS MOVE! IGNORED otherwise !  
input variables that control ionic motion in  
molecular dynamics run or structural relaxation

**&CELL**        needed when CELL MOVES! IGNORED otherwise !  
input variables that control the cell-shape  
evolution in a variable-cell-shape MD or  
structural relaxation

There are **three additional NAMELISTS** in PWscf that **must** be specified under certain circumstances:

- &IONS**        needed when ATOMS MOVE! IGNORED otherwise !  
input variables that control ionic motion in  
molecular dynamics run or structural relaxation
  
- &CELL**        needed when CELL MOVES! IGNORED otherwise !  
input variables that control the cell-shape  
evolution in a variable-cell-shape MD or  
structural relaxation
  
- &PHONON**     needed when preparing input for PHONON !  
IGNORED otherwise!  
will be described with the PHONON code.

There are **three mandatory** INPUT\_CARDS in PWscf

There are **three mandatory INPUT\_CARDS** in PWscf

**ATOMIC\_SPECIES**    name, mass and pseudopotential used for  
each atomic species present in the system

There are **three mandatory INPUT\_CARDS** in PWscf

**ATOMIC\_SPECIES** name, mass and pseudopotential used for each atomic species present in the system

**ATOMIC\_POSITIONS** type and coordinates of each atom in the unit cell

There are **three mandatory INPUT\_CARDS** in PWscf

ATOMIC_SPECIES	name, mass and pseudopotential used for each atomic species present in the system
ATOMIC_POSITIONS	type and coordinates of each atom in the unit cell
K_POINTS	coordinates and weights of the k-points used for BZ integration

There are **three additional INPUT\_CARDS** in PWscf that must be specified only under certain circumstances.

They are

There are **three additional INPUT\_CARDS** in PWscf that must be specified only under certain circumstances.

They are

**CELL\_PARAMETERS**

There are **three additional INPUT\_CARDS** in PWscf that must be specified only under certain circumstances.

They are

CELL\_PARAMETERS

OCCUPATIONS

There are **three additional** INPUT\_CARDS in PWscf that must be specified only under certain circumstances.

They are

CELL\_PARAMETERS

OCCUPATIONS

CLIMBING\_IMAGES will be described with NEB

## The **&CONTROL** namelist

**&CONTROL**     input variables that control the flux of the calculation and the amount of I/O on disk and on the screen.

**FLUX**            : calculation

**I/O**                : title, verbosity, iprint, outdir, prefix,  
                      pseudo\_dir, tprnfor, tstress, disk\_io

**RECOVER**        : restart\_mode, max\_seconds

**MISC**             : dt, nstep, etot\_conv\_thr, forc\_conv\_thr

**WRONGPLACE:**    tefield

## The `&CONTROL` namelist (FLUX)

`calculation CHARACTER (default = 'scf')`

a string describing the task to be performed:

`'scf', 'nscf', 'phonon', 'relax', 'md',`

`'vc-relax', 'vc-md', 'neb' (vc=variable-cell).`

## The &CONTROL namelist (I/O)

`title` CHARACTER ( default = ' ' )  
reprinted on output.

`verbosity` CHARACTER ( default = 'default' )  
'high' | 'default' | 'low' | 'minimal'

`iprint` INTEGER  
band energies are written every  
`iprint` iterations  
default: write only at convergence

... to be continued

## The &CONTROL namelist (I/O) continued

outdir CHARACTER ( default = current directory './' )  
input, temporary, output files are found

prefix CHARACTER ( default = 'pwscf' )  
prepended to input/output filenames:  
prefix.wfc, prefix.rho, etc.

pseudo\_dir CHARACTER ( default = '\$HOME/pw/pseudo/' )  
directory containing pseudopotential files  
Note that the maximum allowed length is 50

... to be continued

## The &CONTROL namelist (I/O) continued

tprnfor LOGICAL  
calculate and print forces. Default = .FALSE.;  
It is set .TRUE. if calculation='\*relax', '\*md'

tstress LOGICAL  
calculate and print stress. Default = .FALSE.;  
It is set .TRUE. if calculation='vc-\*

disk\_io CHARACTER ( default = 'default' )  
'high', 'default', 'low', 'minimal'

## The &CONTROL namelist (RECOVER)

restart\_mode CHARACTER

'from\_scratch' : from scratch ( default )

'restart' : from previous interrupted run

max\_seconds INTEGER (default = 1.0D+6 s)

job stops smoothly after max\_seconds CPU s

## The **&CONTROL** namelist (MISC/WRONGPLACE?)

**dt** REAL ( 1.0 Rydberg a.u.)  
time step for molecular dynamics  
Rydberg a.u. [ 1 hbar/ryd =  $4.84 \times 10^{-17}$  s]

**nstep** INTEGER  
number of ionic + electronic steps  
default: 1 if calculation = 'scf', 'nscf'  
0 if calculation = 'neb'  
50 for the other cases

... to be continued

## The **&CONTROL** namelist (MISC/WRONGPLACE?) continued

etot\_conv\_thr REAL ( default = 1.0D-4 )  
convergence threshold on total energy (Ryd)  
for ionic minimization.

forc\_conv\_thr REAL ( default = 1.0D-3 )  
convergence threshold on forces (Ryd/bohr)  
for ionic minimization.

Convergence is reached when both criteria are satisfied.

## The **&CONTROL** namelist (WRONGPLACE)

```
tefield    LOGICAL ( default = .FALSE. )  
           If .TRUE. a sawlike potential is added  
           to the bare ionic potential.
```

It should be moved to the **&SYSTEM** namelist

## The `&SYSTEM` namelist

`&SYSTEM` input variables that specify the system under study.

`STRUCTURE` : `ibrav, celldm(i),`  
              `a, b, c, cosab, cosac, cosbc,`  
              `nat, ntyp, nbnd, nelec`

`CUTOFF` : `ecutwfc, ecutrho, nr1,nr2,nr3, nr1s,nr2s,nr3s`

`METALS` : `occupations, degauss, smearing`

`MAGNETIC` : `nspin, starting_magnetization(i)`

... to be continued

The **&SYSTEM** namelist continued

SMOOTH CUTOFF: ecfixed, qcutz, q2sigma

LDA+U : lda\_plus\_u, Hubbard\_U(I), Hubbard\_alpha(I)  
starting\_ns\_eigenvalue(m,ispin,I)

TEFIELD : edir, emaxpos, eopreg, eamp

NOSYM : nosym

## The `&SYSTEM` namelist (STRUCTURE)

`ibrav` Bravais-lattice index (NO default, must be specified)

0	read unit cell information from CELL_PARAMETERS card
1	cubic P (sc)
2	cubic F (fcc)
3	cubic I (bcc)
4	Hexagonal and Trigonal P
5	Trigonal R
6	Tetragonal P (st)
7	Tetragonal I (bct)
8	Orthorhombic P
9	Orthorhombic base-centered(bco)
10	Orthorhombic face-centered
11	Orthorhombic body-centered
12	Monoclinic P
13	Monoclinic base-centered
14	Triclinic P

The **&SYSTEM** namelist (STRUCTURE) continued

Crystallographic constants: there are two options

- 1) `celldm(i)`,  $i=1,2,\dots,6$
- 2) `a`, `b`, `c`, `cosab`, `cosac`, `cosbc`,

```
celldm(1) = a / bohr_radius_angs = alat (internal unit of length)
celldm(2) = b / a
celldm(3) = c / a
celldm(4) = cosab
celldm(5) = cosac
celldm(6) = cosbc
```

BEWARE:

```
alat = celldm(1) is the lattice parameter "a" in BOHR
a,b,c are given in ANGSTROM
```

Specify either `a,b,c,...` , **OR** `celldm` **but not both**. Only crystallographic constants needed for chosen Bravais lattice must be specified; other parameters are **IGNORED**.

## The **&SYSTEM** namelist (STRUCTURE) continued

ibrav	celldm -->	a	b	c	cosab	cosac	cosbc
		1	2	3	4	5	6
1	cubic P (sc)	*					
2	cubic F (fcc)	*					
3	cubic I (bcc)	*					
4	Hexagonal and Trigonal P	*		*			
5	Trigonal R	*			*		
6	Tetragonal P (st)	*		*			
7	Tetragonal I (bct)	*		*			
8	Orthorhombic P	*	*	*			
9	Orthorhombic base-centered(bco)	*	*	*			
10	Orthorhombic face-centered	*	*	*			
11	Orthorhombic body-centered	*	*	*			
12	Monoclinic P	*	*	*	*		
13	Monoclinic base-centered	*	*	*	*		
14	Triclinic P	*	*	*	*	*	*

The **&SYSTEM** namelist (STRUCTURE) continued

If **ibrav = 0** BL fundamental vectors are read from an optional card **CELL\_PARAMETERS** to be inserted after all needed NAMELISTS.

```
&LAST_REQUIRED_NAMELIST
```

```
...
```

```
/
```

```
CELL_PARAMETERS    symmetry_class
```

```
  a(1,1) a(2,1) a(3,1)
```

```
  a(1,2) a(2,2) a(3,2)
```

```
  a(1,3) a(2,3) a(3,3)
```

Where **symmetry\_class** is **cubic** or **hexagonal** depending on the expected symmetry of the system.

If **celldm(1)≠0**, lattice vectors are given in **these units**

If **celldm(1)=0**, lattice vectors are given in **BOHR**, and the length of the first lattice vector defines **alat**.

## The `&SYSTEM` namelist (STRUCTURE) continued

`nat`      `INTEGER`    (NO default, must be specified)  
          number of atoms in the unit cell

`ntyp`     `INTEGER`    (NO default, must be specified)  
          number of types of atoms in the unit cell

`nelec`    `REAL`  
          number of electron in the unit cell  
          Default: the same as ionic charge (neutral cell)

`nbnd`     `INTEGER`  
          number of electronic bands to be calculated.  
          Default: in insulators,  $\text{nbnd} = \text{nelec}/2$   
                  in metals,        20% more (minimum 4 more)

## The `&SYSTEM` namelist (CUTOFF)

`ecutwfc`            REAL (NO default, must be specified)  
                    kinetic energy cutoff (Ryd) for wfc

`ecutrho`            REAL ( default = 4 \* `ecutwfc` )  
                    kinetic energy cutoff (Ryd) for charge  
                    density and potential.

Values larger than the default may be needed  
for ultrasoft PP or GGA XC-functionals

... to be continued

## The `&SYSTEM` namelist (CUTOFF) continued

`nr1, nr2, nr3` INTEGER (default: computed from `ecutrho`)  
3D FFT mesh for charge density and potential

`nr1s, nr2s, nr3s` INTEGER (default: computed from `4*ecutwfc`)  
3D FFT mesh for `wfc` and the smooth part of `rho` and `V` (smooth grid). Used in `h_psi`

Can be useful:

- for systems with **non-symmorphic space groups**, when calculated values of `nr1, ...` are **not compatible** with the fractional translation needed by some symmetry operation, which is then rejected.
- to allow **exact comparison** of calculations performed on **different platforms**.

## The `&SYSTEM` namelist (METALS)

```
occupations CHARACTER ( default = 'fixed' )  
    'smearing'      : requires a value for degauss  
    'tetrahedra'   : (see PRB49, 16223 (1994))  
                    requires uniform grid of k-points  
                    automatically generated.  
    'fixed'        : for insulators with a gap  
    'from_input'   : read from input file. 1 k-point!
```

The `&SYSTEM` namelist (METALS) continued  
If `occupation='smearing'` one needs also :

`degauss`      `REAL`  
                value of the broadening used in BZ integration

`smearing`     `CHARACTER`  
                '`gaussian`', '`gauss`':  
                        gaussian spreading (Default)

                '`methfessel-paxton`', '`m-p`', '`mp`':  
                        Methfessel-Paxton first-order broadening  
                        see PRB 40, 3616 (1989).

                '`marzari-vanderbilt`', '`cold`', '`m-v`', '`mv`':  
                        Marzari-Vanderbilt "cold smearing"  
                        see PRL 82, 3296 (1999).

                '`fermi-dirac`', '`f-d`', '`fd`':  
                        Fermi-Dirac thermal distribution

The **&SYSTEM** namelist (METALS) continued  
If **occupations='from\_input'** one needs to specify the **OCCUPATIONS**  
**INPUT\_CARD** with the input occupation values for all bands.

There must be only 1 k-point, LSDA allowed.

OCCUPATIONS

```
f_inp(1,1)  f_inp(2,1)  f_inp(3,1)  ...  f_inp(10,1)
f_inp(11,1) f_inp(12,1) ... f_inp(nbnd,1)
```

or

OCCUPATIONS

```
f_inp(1,1)  f_inp(2,1)  f_inp(3,1)  ...  f_inp(10,1)
f_inp(11,1) f_inp(12,1) ... f_inp(nbnd,1)
f_inp(1,2)  f_inp(2,2)  f_inp(3,2)  ...  f_inp(10,2)
f_inp(11,2) f_inp(12,2) ... f_inp(nbnd,2)
```

## The `&SYSTEM` namelist (MAGNETIC)

`nspin`     INTEGER

`nspin = 1` : non-polarized calculation (default)

`nspin = 2` : spin-polarized calculation

`starting_magnetization(I)`     REAL

starting spin polarization [  $-1 < \text{zeta} < +1$  ]

on atomic type 'I' in a spin-polarized calculation.

Breaks the symmetry and provides a starting point for self-consistency.

If `nspin=2`, at least one `starting_magnetization(I)` must be non zero; not used otherwise.

## The **&SYSTEM** namelist (LDA+U)

```
lda_plus_u      LOGICAL      ( default = .FALSE.)  
Hubbard_U(I),  Hubbard_alpha(I),  
                starting_ns_eigenvalue(m,ispin,I)  
Only needed if lda_plus_u=.TRUE.
```

## The **&SYSTEM** namelist (TEFIELD)

```
edir, emaxpos, eopreg, eamp  
Only needed if tefield=.TRUE. in &CONTROL namelist
```

## The **&SYSTEM** namelist (SMOOTH CUTOFF)

```
ecfixed, qcutz, q2sigma (default 0.0, 0.0, 0.0)  
parameters for modified functional to be used in  
variable-cell molecular dynamics or in stress calculation
```

## The **&SYSTEM** namelist (NOSYM)

```
nosym LOGICAL ( default = .FALSE. )  
      if (.TRUE.) symmetry is not used.
```

### **NB:**

- a k-point grid provided in input is used "as is";
- automatically generated k-point grids will contain points in the whole BZ (not only in the IBZ).

**Use with care** in low-symmetry large cells if you cannot afford a k-point grid with the correct symmetry.

## The `&ELECTRONS` namelist

`&ELECTRONS` input variables that control the algorithms used to reach the self-consistent solution of KS equations for the electrons.

`START` : `startingpot`, `startingwfc`  
`MIXING` : `conv_thr`, `mixing_mode`, `mixing_beta`, `mixing_ndim`,  
`mixing_fixed_ns`, `electron_maxstep`  
`DIAGO` : `diagonalization`, `diago_thr_init`,  
`diago_cg_maxiter`, `diago_david_ndim`,  
`diago_diis_ndim`, `diago_diis_ethr`

## The &ELECTRONS namelist (START)

startingpot CHARACTER

'atomic': starting potential from superposition of atomic charge densities.

Default for scf, \*relax, \*md, neb.

'file' : start from existing "prefix".pot file

Default and only possibility

for nscf and phonon calculations.

startingwfc CHARACTER

'atomic': (default) start from superposition of atomic orbitals. If not enough atomic orbitals are available, complete with random wfcs.

'random': start from random wfcs

'file' : start from a wavefunction file

## The &ELECTRONS namelist (MIXING)

conv\_thr        REAL    ( default = 1.D-6 Rydberg )  
                 Convergence threshold for selfconsistency:  
                 estimated energy error < conv\_thr

mixing\_mode    CHARACTER (default: 'plain')  
                 'plain' : charge density Broyden mixing  
                 'TF' : as above, with simple TF screening  
                 'local-TF': as above, with TF screening that  
                                 depends on the local density  
                 'potential':(OBSOLETE) potential mixing

mixing\_beta    REAL    ( default = 0.7D0 )  
                 mixing factor

mixing\_ndim    INTEGER ( default = 8)  
                 number of iterations used in mixing

... to be continued

## The `&ELECTRONS` namelist (MIXING) continued

`mixing_fixed_ns` INTEGER ( default = 0 )

sometime used by LDA+U : see INPUT\_PW

`electron_maxstep`

INTEGER ( default = 50 )

maximum number of iterations in a scf step

## The &ELECTRONS namelist (DIAGO)

```
diagonalization CHARACTER ( default = 'david' )  
    'david': Davidson diagonalization  
    'diis'  : DIIS-like diagonalization  
    'cg'   : band-by-band conjugate-gradient  
  
diago_thr_init REAL ( default = 1.D-2 )  
    Convergence threshold for first iterative  
    diagonalization.
```

## The &ELECTRONS namelist (DIAGO)

diagonalization CHARACTER ( default = 'david' )  
    'david': Davidson diagonalization  
    'diis' : DIIS-like diagonalization  
    'cg' : band-by-band conjugate-gradient

diago\_thr\_init REAL ( default = 1.D-2 )  
    Convergence threshold for first iterative  
    diagonalization.

Only used with DAVID: diago\_david\_ndim

Only used with DIIS: diago\_diis\_ndim, diago\_diis\_ethr

Only used with CG: diago\_cg\_maxiter

## There are three mandatory INPUT\_CARDS

- ATOMIC\_SPECIES      name, mass and pseudopotential used for  
                         each atomic species present in the system
- ATOMIC\_POSITIONS    type and coordinates of each atom in the  
                         unit cell
- K\_POINTS              coordinates and weights of the k-points  
                         used for BZ integration

## The `ATOMIC_SPECIES` card

For each atomic species (`ntyp` in `&SYSTEM` namelist) one must specify a label, the atomic mass and the name a PP file.

```
ATOMIC_SPECIES
```

```
X(1)      Mass_X(1)      PseudoPot_X(1)
```

```
...
```

```
X(ntyp)  Mass_X(ntyp)  PseudoPot_X(ntyp)
```

## The `ATOMIC_SPECIES` card

For each atomic species (`ntyp` in `&SYSTEM` namelist) one must specify a label, the atomic mass and the name a PP file.

```
ATOMIC_SPECIES
```

```
X(1)      Mass_X(1)      PseudoPot_X(1)
```

```
...
```

```
X(ntyp)  Mass_X(ntyp)  PseudoPot_X(ntyp)
```

example

```
ATOMIC_SPECIES
```

```
O  16.00  O.LDA.US.RRKJ3.UPF
```

```
C   12.00  CUS.RRKJ3.UPF
```

## The `ATOMIC_SPECIES` card

For each atomic species (`ntyp` in `&SYSTEM` namelist) one must specify a label, the atomic mass and the name a PP file.

```
ATOMIC_SPECIES
X(1)      Mass_X(1)      PseudoPot_X(1)
...
X(ntyp)  Mass_X(ntyp)   PseudoPot_X(ntyp)
```

example

```
ATOMIC_SPECIES
O  16.00  O.LDA.US.RRKJ3.UPF
C   12.00  CUS.RRKJ3.UPF
```

Masses are actually used only if atoms move.

## The `ATOMIC_POSITIONS` card

This card specifies the atomic species `label` and `positions` of each atom in the unit cell (`nat` in `&SYSTEM` namelist).

```
ATOMIC_POSITIONS  position_format  
X(1)    x(1)    y(1)    z(1)  
...  
X(nat)  x(nat)  y(nat)  z(nat)
```

where `position_format` is `alat`, `bohr`, `crystal` or `angstrom`

## The `ATOMIC_POSITIONS` card

This card specifies the atomic species `label` and `positions` of each atom in the unit cell (`nat` in `&SYSTEM` namelist).

```
ATOMIC_POSITIONS  position_format
X(1)  x(1)  y(1)  z(1)
...
X(nat) x(nat) y(nat) z(nat)
```

where `position_format` is `alat`, `bohr`, `crystal` or `angstrom`

It is also possible to specify that some coordinates should be kept fixed in relaxation or dynamics.

```
ATOMIC_POSITIONS bohr
C 2.256 0.0 0.0
O 0.0 0.0 0.0 0 0 0
```

The **ATOMIC\_POSITIONS** card

This card specifies the atomic species **label** and **positions** of each atom in the unit cell (nat in &SYSTEM namelist).

```
ATOMIC_POSITIONS  position_format
X(1)  x(1)  y(1)  z(1)
...
X(nat) x(nat) y(nat) z(nat)
```

where **position\_format** is **alat**, **bohr**, **crystal** or **angstrom**

It is also possible to specify that **some coordinates should be kept fixed** in relaxation or dynamics.

```
ATOMIC_POSITIONS bohr
C 2.256 0.0 0.0
O 0.0 0.0 0.0 0 0 0
```

If calculation is NEB first\_image and last\_image configurations are given

The **K\_POINTS** card

This card specifies set of k-points used to sample the BZ.

```
K_POINTS kpoint_format
```

```
.....
```

The **K\_POINTS** card

This card specifies set of k-points used to sample the BZ.

```
K_POINTS kpoint_format
```

```
.....
```

there are four possible values for **kpoint\_format**

## The `K_POINTS` card

This card specifies set of k-points used to sample the BZ.

```
K_POINTS kpoint_format
```

```
.....
```

there are four possible values for `kpoint_format`

```
K_POINTS { tpiba | crystal }
```

```
nks
```

```
  kx(1)   ky(1)   kz(1)   weight(1)
```

```
  ...
```

```
  kx(nks) ky(nks) kz(nks) weight(nks)
```

## The `K_POINTS` card

This card specifies set of k-points used to sample the BZ.

```
K_POINTS kpoint_format
```

```
.....
```

there are four possible values for `kpoint_format`

```
K_POINTS { tpiba | crystal }
```

```
nks
```

```
  kx(1)   ky(1)   kz(1)   weight(1)
```

```
  ...
```

```
  kx(nks) ky(nks) kz(nks) weight(nks)
```

```
K_POINTS automatic
```

```
  nk1, nk2, nk3, k1, k2, k3
```

## The `K_POINTS` card

This card specifies set of k-points used to sample the BZ.

```
K_POINTS kpoint_format
```

```
.....
```

there are four possible values for `kpoint_format`

```
K_POINTS { tpiba | crystal }
```

```
nks
```

```
  kx(1)   ky(1)   kz(1)   weight(1)
```

```
  ...
```

```
  kx(nks) ky(nks) kz(nks) weight(nks)
```

```
K_POINTS automatic
```

```
  nk1, nk2, nk3, k1, k2, k3
```

```
K_POINTS gamma
```

There are **three additional NAMELISTS** in PWscf that **must** be specified under certain circumstances:

- &IONS**        needed when ATOMS MOVE! IGNORED otherwise !  
input variables that control ionic motion in  
molecular dynamics run or structural relaxation
- &CELL**        needed when CELL MOVES! IGNORED otherwise !  
input variables that control the cell-shape  
evolution in a variable-cell-shape MD or  
structural relaxation
- &PHONON**     needed when preparing input for PHONON !  
IGNORED otherwise!  
will be described with the PHONON code.

## The IONS namelist

only needed if calculation = 'relax', 'md', 'vc-relax', 'vc-md', 'neb'

ion\_dynamics CHARACTER specify the type of ionic dynamics.

For different type of calculation different possibilities are allowed and different default apply:

```
CASE ( calculation = 'relax' )
    'bfgs' (default), 'old-bfgs',
    'constrained-bfgs', 'damp'
CASE ( calculation = 'md' )
    'verlet' (default), 'constrained-verlet',
    'beeman'
CASE ( calculation = 'vc-md' )
    'beeman' (default)
CASE ( calculation = 'vc-relax' )
    'damp' (default)
```

Many more variables can be specified (check INPUT\_PW)

## The **CELL** namelist

only needed if calculation = 'vc-relax', 'vc-md'

cell\_dynamics CHARACTER specify the type of cell dynamics.  
For different type of calculation different possibilities are allowed and different default apply:

```
CASE ( calculation = 'vc-md' )
```

```
  'none' (default), 'pr', 'w'
```

```
CASE ( calculation = 'vc-relax' )
```

```
  'none' (default), 'damp-pr', 'damp-w'
```

press, wmass, cell\_factor

for these variables check INPUT\_PW

## Input structure for a SCF run

```
&CONTROL ... /
&SYSTEM ... /
&ELECTRONS ... /
ATOMIC_SPECIES
ATOMIC_POSITIONS
K_POINTS

&CONTROL ... /
&SYSTEM ibrav=0 ... /
&ELECTRONS ... /
CELL_PARAMETERS
ATOMIC_SPECIES
ATOMIC_POSITIONS
K_POINTS
```

```
&CONTROL ... /
&SYSTEM ... /
&ELECTRONS occupations=fixed ... /
OCCUPATIONS
ATOMIC_SPECIES
ATOMIC_POSITIONS
K_POINTS
```

## Input structure for a RELAX / MD run

```
&CONTROL calculation='relax' ... /
```

```
&SYSTEM ... /
```

```
&ELECTRONS ... /
```

```
&IONS ... /
```

```
ATOMIC_SPECIES
```

```
ATOMIC_POSITIONS
```

```
K_POINTS
```

```
&CONTROL calculation='vc-relax' ... /
```

```
&SYSTEM ... /
```

```
&ELECTRONS ... /
```

```
&IONS ... /
```

```
&CELL ... /
```

```
ATOMIC_SPECIES
```

```
ATOMIC_POSITIONS
```

```
K_POINTS
```

## An example

```
&control
  pseudo_dir = './',
  outdir='/scratch/stefano/Be0001/',
  prefix='be0001'
  tprnfor = .true.
/
&system
  ibrav=4, celldm(1)=4.247, celldm(3)=16.0, nat=12, ntyp=1, nbnd=20,
  occupations='smearing', smearing='methfessel-paxton', degauss=0.05
  ecutwfc=22.0, nr1=16, nr2=16,
/
&electrons
/
ATOMIC_SPECIES
Be 1.0 Be.vbc2
ATOMIC_POSITIONS alat
Be 0.000000000 -0.288675135 4.359667099
Be 0.000000000 0.288675135 3.548485449
Be 0.000000000 -0.288675135 2.754655986
.. .....
```

```

..      .....      .....      .....
Be      0.000000000    0.288675135   -2.754655986
Be      0.000000000   -0.288675135   -3.548485449
Be      0.000000000    0.288675135   -4.359667099

```

K\_POINTS tpiba

30

```

0.000000000    0.000000000    0.000000000    1.00
0.062500000    0.036084392    0.000000000    6.00
0.125000000    0.072168784    0.000000000    6.00
0.187500000    0.108253175    0.000000000    6.00
.....
.....
0.250000000    0.433012702    0.000000000    6.00
0.312500000    0.469097094    0.000000000   12.00
0.375000000    0.505181486    0.000000000    6.00
0.312500000    0.541265877    0.000000000    6.00

```

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

On parallel machines the workload and the required RAM memory can be distributed

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

On parallel machines the workload and the required RAM memory can be distributed **in two ways** (at least):

- 1) K-points can be distributed:  
different processors operate on a subset of the K-points.

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

On parallel machines the workload and the required RAM memory can be distributed in two ways (at least):

- 1) K-points can be distributed:  
different processors operate on a subset of the K-points.
- 2) PW's and FFT grids can be distributed:  
different processors keep in memory and operate on a subset of the total number of PW's and FFT grid points.

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

On parallel machines the workload and the required RAM memory can be distributed **in two ways** (at least):

- 1) K-points can be distributed:  
different processors operate on a subset of the K-points.
- 2) PW's and FFT grids can be distributed:  
different processors keep in memory and operate on a subset of the total number of PW's and FFT grid points.

PWscf implements **both ways**: processors are divided in pools  
(ex: 16 proc = 8 pool x 2 proc/pool)

- 1) **K-points** are distributed **among the pools** and
- 2) **PW's and FFT grids** are distributed **within each pool**.

```
prompt> $0-sesame_ROOT/bin/pw.x < pw.in > pw.out
```

On parallel machines this become something like

```
prompt> $0-sesame_ROOT/bin/pw.x -n 16 -npool 8 < pw.in > pw.out
```

or

```
prompt> poe $0-sesame_ROOT/bin/pw.x -n 16 -npool 8 < pw.in > pw.out
```

or

```
prompt> mpirun -np 16 $0-sesame_ROOT/bin/pw.x -npool 8 < pw.in > pw.out
```

The way **number of processors** is set depends on the particular parallel environment.

The **number of pools** is instead specific of PWscf code and its input format does not change on different parallel environment.

## The output

Program PWSCF v.2.0 starts ...

Today is 16Feb2004 at 16: 6:28

Ultrasoft (Vanderbilt) Pseudopotentials

Current dimensions of program pwscf are:

ntypx =10 npk =40000 lmax = 3

nchix = 6 ndim = 2000 nbrx = 8 nqfm = 8

bravais-lattice index = 4

lattice parameter (a\_0) = 4.2470 a.u.

unit-cell volume = 1061.4448 (a.u.)<sup>3</sup>

number of atoms/cell = 12

number of atomic types = 1

kinetic-energy cutoff = 22.0000 Ry

charge density cutoff = 88.0000 Ry

convergence threshold = 1.0E-06

beta = 0.7000

number of iterations used = 8 plain mixing

Exchange-correlation = PZ (1100)

iswitch = 0

celldm(1)= 4.247000 celldm(2)= 0.000000 celldm(3)= 16.000000  
celldm(4)= 0.000000 celldm(5)= 0.000000 celldm(6)= 0.000000

crystal axes: (cart. coord. in units of a\_0)  
a(1) = ( 1.000000 0.000000 0.000000 )  
a(2) = ( -0.500000 0.866025 0.000000 )  
a(3) = ( 0.000000 0.000000 16.000000 )

reciprocal axes: (cart. coord. in units 2 pi/a\_0)  
b(1) = ( 1.000000 0.577350 0.000000 )  
b(2) = ( 0.000000 1.154701 0.000000 )  
b(3) = ( 0.000000 0.000000 0.062500 )

PSEUDO 1 is Be (vbc)      zval = 2.0      lmax= 1      lloc= 1  
          i=            1            2            3

core

alpha =            0.99964            0.0000

a(i) =            1.0000            0.0000

l = 0

alpha =            1.7068            0.0000            0.0000

a(i) =            5.4710            0.0000            0.0000

a(i+3)=           -1.6312            0.0000            0.0000

l = 1

alpha =            0.78031            0.0000            0.0000

a(i) =            -1.6972            0.0000            0.0000

a(i+3)=            0.48457            0.0000            0.0000

nonlinear core correction:  $\rho(r) = (a + b r^2) \exp(-\alpha r^2)$

a =            0.95153E-01

b =            0.24127

alpha=            2.7594

atomic species	valence	mass	pseudopotential
Be	2.00	1.00000	Be( 1.00)

12 Sym.Ops. (with inversion)

Cartesian axes

site n.	atom	positions (a_0 units)
1	Be tau( 1) = (	0.0000000 -0.2886751 4.3596671 )
2	Be tau( 2) = (	0.0000000 0.2886751 3.5484854 )
3	Be tau( 3) = (	0.0000000 -0.2886751 2.7546560 )
4	Be tau( 4) = (	0.0000000 0.2886751 1.9655547 )
5	Be tau( 5) = (	0.0000000 -0.2886751 1.1789015 )
6	Be tau( 6) = (	0.0000000 0.2886751 0.3929197 )
7	Be tau( 7) = (	0.0000000 -0.2886751 -0.3929197 )
8	Be tau( 8) = (	0.0000000 0.2886751 -1.1789015 )
9	Be tau( 9) = (	0.0000000 -0.2886751 -1.9655547 )
10	Be tau( 10) = (	0.0000000 0.2886751 -2.7546560 )
11	Be tau( 11) = (	0.0000000 -0.2886751 -3.5484854 )
12	Be tau( 12) = (	0.0000000 0.2886751 -4.3596671 )

number of k points= 30 gaussian broad. (ryd)= 0.0500 ngauss = 1

cart. coord. in units  $2\pi/a_0$

k( 1)	=	(	0.0000000	0.0000000	0.0000000)	, wk =	0.0078125
k( 2)	=	(	0.0625000	0.0360844	0.0000000)	, wk =	0.0468750
k( 3)	=	(	0.1250000	0.0721688	0.0000000)	, wk =	0.0468750
k( 4)	=	(	0.1875000	0.1082532	0.0000000)	, wk =	0.0468750
..	..	.	.	.....	.....	.....	.. .
..	..	.	.	.....	.....	.....	.. .
k( 27)	=	(	0.2500000	0.4330127	0.0000000)	, wk =	0.0468750
k( 28)	=	(	0.3125000	0.4690971	0.0000000)	, wk =	0.0937500
k( 29)	=	(	0.3750000	0.5051815	0.0000000)	, wk =	0.0468750
k( 30)	=	(	0.3125000	0.5412659	0.0000000)	, wk =	0.0468750

G cutoff = 40.2057 ( 14795 G-vectors) FFT grid: ( 16, 16,216)

nbndx	=	80	nbnd	=	20	natomwfc	=	12	npwx	=	1887
nelec	=	24.00	nkb	=	12	ngl	=	943			

warning: negative or imaginary core charge      -0.000003      0.000000

Initial potential from superposition of free atoms  
Starting wfc are atomic +    8 random wfc

total cpu time spent up to now is      31.14 secs

iteration # 1      ecut=      22.00 ryd      beta=0.70  
Davidson diagonalization (with overlap)  
ethr = 1.00E-02, avg # of iterations = 8.0  
total energy      =      -29.14912003 ryd  
estimated scf accuracy      <      0.47112901 ryd

total cpu time spent up to now is      220.47 secs

iteration # 2      ecut=      22.00 ryd      beta=0.70  
Davidson diagonalization (with overlap)  
ethr = 1.96E-03, avg # of iterations = 9.2  
total energy      =      935.30786090 ryd  
estimated scf accuracy      <      979.55647128 ryd

.....

total cpu time spent up to now is 1635.03 secs

iteration # 15 ecut= 22.00 ryd beta=0.70

Davidson diagonalization (with overlap)

ethr = 2.32E-08, avg # of iterations = 3.6

k = 0.0000 0.0000 0.0000 ( 1883 PWs) bands (ev):  
-8.6594 -8.3732 -8.0061 -7.5629 -7.0379 -6.4263 -5.7252 -4.9375  
-4.0779 -3.1891 -2.3981 -0.2625 -0.2366 4.3785 5.4202 6.5117  
7.1274 7.7717 7.8515 9.1438

.....

k = 0.3125 0.5413 0.0000 ( 1887 PWs) bands (ev):  
-0.4220 -0.1336 0.2091 0.5662 0.9013 1.1965 1.4595 1.6954  
1.7326 1.8983 1.9058 1.9856 2.1627 2.4288 2.6712 2.9470  
3.2659 3.6570 4.2349 5.0255

the Fermi energy is 2.3995 ev

! total energy = -29.53349448 ryd  
estimated scf accuracy < 0.00000041 ryd

band energy sum = -3.75904228 ryd  
one-electron contribution = -847.59119742 ryd  
hartree contribution = 431.32053221 ryd  
xc contribution = -16.79591716 ryd  
ewald contribution = 403.53337095 ryd  
correction for metals = -0.00028305 ryd

convergence has been achieved

Forces acting on atoms (Ry/au):

atom	1	type	1	force =	0.00000000	0.00000000	-0.00004555
atom	2	type	1	force =	0.00000000	0.00000000	0.00003219
atom	3	type	1	force =	0.00000000	0.00000000	-0.00011340
atom	4	type	1	force =	0.00000000	0.00000000	0.00007865
atom	5	type	1	force =	0.00000000	0.00000000	0.00005442
atom	6	type	1	force =	0.00000000	0.00000000	-0.00001113
atom	7	type	1	force =	0.00000000	0.00000000	0.00001113
atom	8	type	1	force =	0.00000000	0.00000000	-0.00005442
atom	9	type	1	force =	0.00000000	0.00000000	-0.00007865
atom	10	type	1	force =	0.00000000	0.00000000	0.00011340
atom	11	type	1	force =	0.00000000	0.00000000	-0.00003219
atom	12	type	1	force =	0.00000000	0.00000000	0.00004555

Total force = 0.000225      Total SCF correction = 0.001059

Writing file be0001.pun      for program phonon

PWSCF : 28m48.18s CPU time

init\_run : 31.12s CPU

electrons : 1696.01s CPU

forces : 0.95s CPU

electrons : 1696.01s CPU

c\_bands : 1496.42s CPU ( 15 calls, 99.761 s avg)

sum\_band : 195.59s CPU ( 15 calls, 13.039 s avg)

v\_of\_rho : 2.04s CPU ( 31 calls, 0.066 s avg)

mix\_rho : 1.91s CPU ( 15 calls, 0.127 s avg)

c\_bands : 1496.42s CPU ( 15 calls, 99.761 s avg)

init\_us\_2 : 1.40s CPU ( 960 calls, 0.001 s avg)

cegterg : 1493.49s CPU ( 450 calls, 3.319 s avg)

sum\_band : 195.59s CPU ( 15 calls, 13.039 s avg)

wfcrot	:	30.20s	CPU (	30	calls,	1.007	s avg)
cegterg	:	1493.49s	CPU (	450	calls,	3.319	s avg)
h_psi	:	1398.80s	CPU (	2473	calls,	0.566	s avg)
g_psi	:	3.22s	CPU (	1993	calls,	0.002	s avg)
overlap	:	46.05s	CPU (	1993	calls,	0.023	s avg)
cdiaghg	:	14.71s	CPU (	2023	calls,	0.007	s avg)
update	:	26.08s	CPU (	1993	calls,	0.013	s avg)
last	:	18.87s	CPU (	586	calls,	0.032	s avg)

h_psi	:	1398.80s	CPU (	2473	calls,	0.566	s avg)
init	:	2.32s	CPU (	2473	calls,	0.001	s avg)
firstfft	:	669.67s	CPU (	32391	calls,	0.021	s avg)
secondfft	:	653.82s	CPU (	32391	calls,	0.020	s avg)
add_vuspsi	:	6.94s	CPU (	2473	calls,	0.003	s avg)

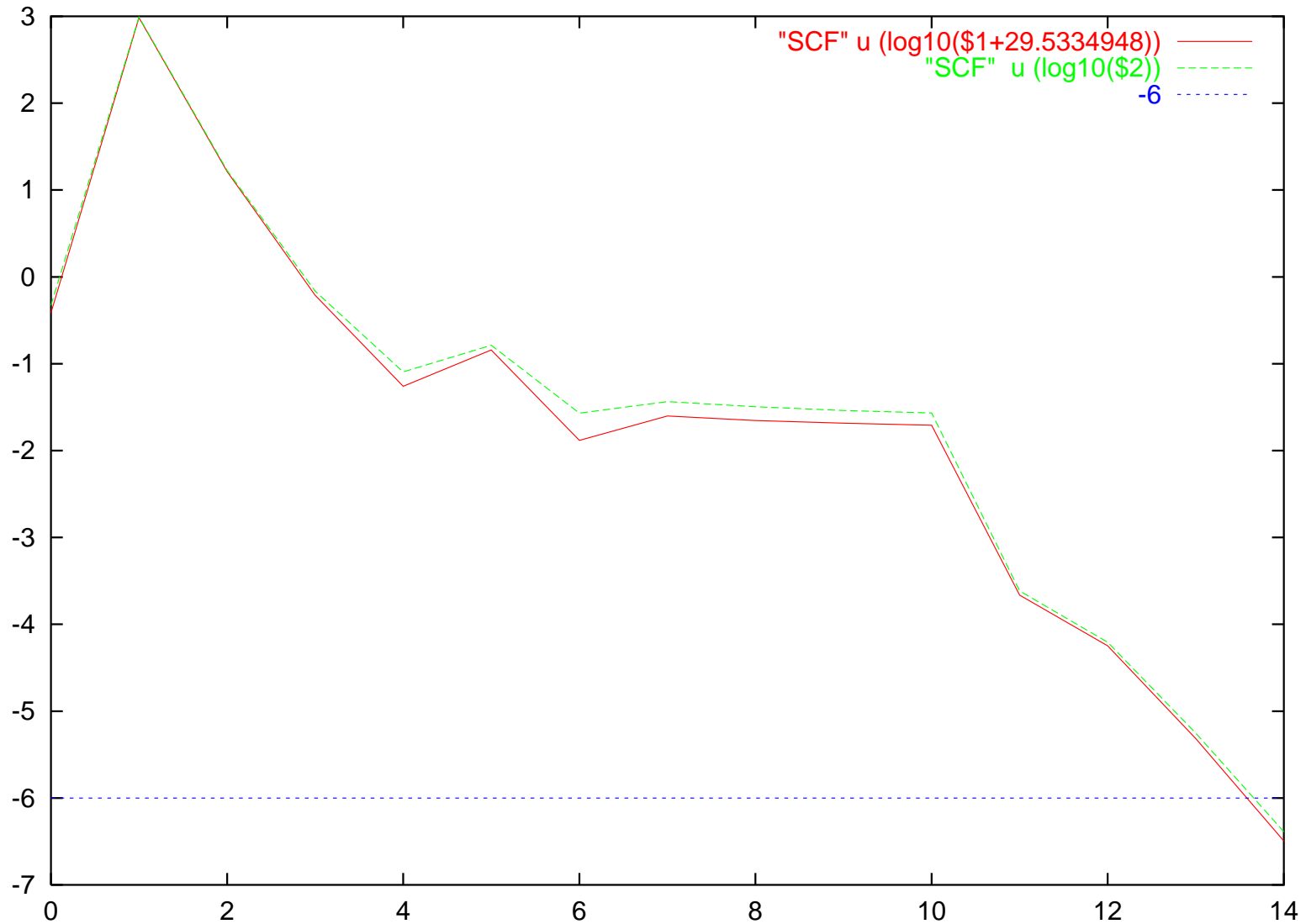
### General routines

ccalbec	:	8.00s	CPU (	2503	calls,	0.003	s avg)
cft3	:	2.74s	CPU (	126	calls,	0.022	s avg)
cft3s	:	1432.65s	CPU (	73782	calls,	0.019	s avg)
davcio	:	2.67s	CPU (	1410	calls,	0.002	s avg)

```
prompt> grep -e 'l e' -e ' scf ' pw.out | \  
awk '/l e/{e=$(NF-1)}/ scf /{print e, $(NF-1)}'
```

```
-29.14912003 0.47112901  
935.30786090 979.55647128  
-13.21357118 16.83007169  
-28.92055625 0.68658733  
-29.47844813 0.08060113  
-29.38920267 0.16317196  
-29.52040351 0.02695532  
-29.50844717 0.03671080  
-29.51131284 0.03206405  
-29.51277583 0.02891886  
-29.51395038 0.02707970  
-29.53327797 0.00024233  
-29.53343821 0.00006216  
-29.53348998 0.00000557  
-29.53349448 0.00000041
```

```
prompt> grep -e 'l e' -e ' scf ' pw.out | \  
awk '/l e/{e=$(NF-1)}/ scf /{print e, $(NF-1)}' > SCF
```



Where can I find some useful information about PWscf ?

```
prompt > ls $0-sesame_ROOT/pwdocs/
```

In particular **INPUT\_PW** contains a rather complete description of the input of PWscf.

Similarly **INPUT\_PP**, **INPUT\_PH**,... contain descriptions of post processing, phonon...

```
prompt > ls $0-sesame_ROOT/pw_examples/
```

This directory contains a number of example scripts that illustrate (some) of the features implemented in PWscf and related codes.

The code stops with an error that I don't understand

- read the error message and look at your input file
- read the manual
- check the pw\_forum mailing-list
- write a message to the pw\_forum
- have a look to the source

THE END