1 cell.in

```
fem bcc !name str
1.000000000000 0.000000000000 0.000000000000
0.000000000000 1.000000000000 0.000000000000
0.000000000000 0.000000000000 1.000000000000
len 7.030035560000 4.058792920000 2.485492910000
latpar 2.870000000000
```

The first line is a string to name the file. ????

The next three sets of numbers are the primitive lattice vectors, which are dimensionless.

len sets the length of the primitive lattice vectors in the order XYZ, this must be in units of Å.

latpar sets the length of the lattice parameter, again in angstrom, although it is not generally used (best to set this to 1.0 as it is not required and this will ensure it doesn't interfere).

```
nd 1014
Fe 1.00 1.00 0.00 0.00 2.8837815927610 !symb crds(3r) de mg
Fe 1.50 1.50 0.00 0.00 2.8837815927610 !symb crds(3r) de mg
...
```

nd defines the number of dynamic atoms that will be moving during the simulation

Each atom is defined by its species, its coordinates (which are then scaled by the **len** vectors so should be given in fractions of these vectors), the energy shift in eV (although this will be calculated to self consistency anyway) and the magnetic moment in Bohr magnetons (which will again be calculated to self consistency, however do not set it to 0 for Iron as this is a metastable state and so will halt the self-consistency cycle). As in Fortran anything following ! is commented out, used to define each number here.

```
ninert 336
Fe 0.00 0.00 0.00 0.00 2.8837815927610 !symb crds(3r) de mg
Fe 0.50 0.50 0.00 0.00 2.8837815927610 !symb crds(3r) de mg
...
```

ninert defines the number of inert atoms, which will not move or participate in self consistency although the moments must still be set, these atoms are defined in the same way as previously.

```
unrld
Fe 1.00 1.00 0.00 0.00 0.00 !symb crds(3r) de mg
Fe 1.50 1.50 0.00 0.00 0.00 !symb crds(3r) de mg
...
```

Atoms following **unrld** are the unrelaxed atoms, this defines the perfect lattice and is used to generate differential displacement plots.

```
nullheight 0.000000000000
```

nullheight is used in generating the gamma-surface, this will define the Z height at which the cut is made in order to generate it. This surface must always be along a Z plane.

```
dislocation_conf
0 ! gfbcon
0.0 ! radiusi
0.0 100.0 0.0 100.0 !
```

This is the dislocation configuration, **gfbcon** defines the Green's Function boundary conditions, gives dependence on all environment, set it to zero for it not to be used. This has not been fully implemented.

2 conf.in

This is the configuration file, defining the potential parameters and the number of recursions etc.

```
fefe
2 16 F F !pot_flg mvflag quiet paral
0 0 0 !scf writescale elplusrel
0 0 1 !rlim(3i)
1 0 1 0 0 0 t !fs env vpair fs_only emb pot_scl forces
3 12.0 !scf_cut rprune
4.02 4.02 !rcut(2r)
0 0 !idebug restart
```

fefe is just a string to define the atome names.
pot_flg is the potential flag deciding which is used, where a value of 2 means B.O.P. is used and a value of 4 means T.B., but with no overlap, just shifting of onsite energies until self consistency is reached; this is essentially B.O.P. with infinite moments.
mvflag defines the action performed within this simulation:

-5 D.O.S. plotting.
-2 Force check; Prints out a table with energies and forces in forcecheck.out for each displacement of the specified atom along the specified coordinate. The table may be useful for checking whether the energies and forces are consistent. There is a python script available to plot it all.
-1 A convergence test.
0 Evaluate the energy and forces of a configuration.
1 Relax the atoms.
2-5 Carry out molecular dynamics.
6 Evaluate the elastic constants. (Use 12 instead)
7 Perform a diffusion barrier calculation; push a chosen atom (D_ATOM) in a certain direction (DISP_VEC) in a series of steps (N_DISP). At each point on the diffusion path the diffusing atom is constrained to lie in the plane perpendicular to the direction of motion. All other atoms are relaxed unconstrained using a conjugate gradients algorithm.

9 Calculate force constant matrix.
10 Calculate the gamma surface.
11 Calculate the unit cell dimensions.
12 Calculates the elastic constants for the unit cell, output may be used in fitting procedure.
13 Calculate the energy surface vs. unit cell dimensions A and C.
14 Calculate gamma surface using Alex Girshick's code.
15 Calculate grain boundary structure.
16 Calculate dislocation structure.
17 Calculate the dislocation structure under applied stress.
18 Build the round blocks needed to calculate the dislocation self-energy.
19 Calculate the logariphmic plot for the dislocation self-energy.
20 Calculate the Rose universal equation of state curve.
21 Calculate the shortest distance between the atoms in the relaxed configuration.
22-23 Screened bond integral scaling.
24 Determine lattice Green's function boundary conditions.
25 Perform Monte Carlo structure optimisation.
27 Calculate energy volume curves.
29 Print bond scalings.
30 Use N.E.B.

quiet determines the verbosity of the simulation, if set to true will print very little information during operation.
paral determines whether the simulation is run in parallel or not, set to true if it is being run with M.P.I.
scf is the scaling factor.
writescale ???????????????
elplusrel ??????????????
rlim(3i) defines the periodicity in the XYZ axes, where 0 0 1 implies it is periodic in Z but not X or Y.
fs determines whether or not to include all terms that are not typical B.O.P. terms (classical terms like environment or pairwise), ie the Finnis-Sinclair model.
env determines whether or not to include the environment term.
vpair determines whether to include the pair potential term, ie the repulsive Coulomb interaction between atoms.
fs_only will cause the simulation to only calculate the classical terms and none of the B.O.P. terms.
emb is used to calculate the proper Finnis-Sinclair with the embedding potential.
pot_scl is used for the scaling of the pair potential.
forces determines whether or not to calculate the forces.
rprune is used to set the boundary condition for the furthest atom that can be reached with the number of moments being used, this produces a cell that contains all of the influencing atoms.

rcut(2r) defines two cutoffs; first is the cut-off in the neighbour finding (should be slightly further than the largest hopping integral) and second is for the pair potential, which should be just above the largest cut-off set for the pairs.
idebug prints the debugging reports.
restart determines how it will restart, generally obselete so leave at zero.

```
bop_conf
1 6 6 1 0 2 600 0.03 1.0 ! momflg nrec nbase term v1flag chi_meth mfac kt etail
```

This category sets the configuration for B.O.P.
momflg determines the form of the moments:
1 = Averaged moments.
2 = Mixed basis, locally oriented orbitals.
3 = Non-averaged basis, separated orbitals.
nrec sets the number of recursions, the number of moments produces will be twice this add one.
nbase sets the number of recursions it does fully, shoulds be equal to or less than nrec
term defines the terminator used, generally for metals it will be the square root terminator; 1 - Square root terminator, 2 - Finite terminator and 3 - No terminator.
v1flag dictates how the coefficients and Fermi energy are dealt with:
0 = Read coefficients from disk, find the Fermi energy.
1 = Evaluate the coefficients and write them to disk, Find the Fermi energy.
2 = Evaluate the coefficients but don't write them to disk, don't find the Fermi energy.
chi_meth sets the method for finding the response functions, if equal to 1 it will carry out analytic integration, if 2 it will be numeric integration.
mfac sets the m factor for calculating the energy points.
kt is the Boltzmann constant multiplied by the temperature, essentially sets the broadening for the energy integration, generally want it to be low for accuracy but if it's below 0.001 the calculation will be numerically unstable.
etail ???????????????

```
sc_conf
0.01 1000 1e-8 0.01 f 0.01 1000 ! qerr mxit eftol mag merr mgit mixt
```

Sets configuration for self-consistency. The mixing scheme is set so that before each single iteration of magnetic mixing the charge is mixed until self-consistency.
qerr sets the tolerance for the charge in self-consistency cycle.
mxit sets the maximum number of iterations.
eftol sets the tolerance for the Fermi energy in the cycle, can go as low as 1e-12.
mag defines whether magnetics are included or not, can slow process as each loop for the charge will have to include a loop for the magnetic moments (note: if they are included do not set the magnetic tolerance to be more lax than charge tolerance or it will not converge).
merr sets the magnetic error tolerance.

mgit sets the magnetic self-consistency cycle's maximum number of iterations.
mixt ????????????????????

```
ham_conf
2 4 ! natypes nbtypes
Al 13 0.100000 0.100000 0.030000 0.200000 !symb z ravs col(3r)
3.0 1 1 2 0 -2.543 3.6677 6.9037 0.0 0.5 0.0 0.0 0.0 !cc norbs orblist(3i) es ep ed
st poly_coeffs(4r)
2.3000 2.0000 0.0000 !lam0 m r_core
```

Sets the Hamiltonian configuration.
natypes sets the number of atom types.
nbtypes sets the number of bond types (ie Fe-Fe, H-Fe, Fe-H and H-H for a system with only iron and hydrogen).
symb is the symbol to call the atom in question.
z is its atomic number.
ravs and **col(3r)** are the average radius and colour of that atom respectively, only used for visualisation software such as xbs.
cc is the core charge, as it is neglected from calculations here it is included in screening the nuclear charge.
norbs is tyhe number of orbital types (s, p and d)
orblist(3i) sets orbitals in order of decreasing l, so d only would be 2 0 0, while sd would be 0 2 0.
es, **ep** and **ed** are the onsite energy levels for the orbitals relative to each other, so this may be left out unless you are including 2 or more.
st ????????????????????
poly_coeffs(4r) is not used any more, but was used for the coefficients of the polynomials augmenting the tails.
lam0 and **m** set the lambda and m terms in the environmental equations.
r_core sets the core radius ??????????????????????
This block must be done for each atom species being included, starting with their symbol.

```
AlAl
!     Vscr     DTau    Oscr   r0     rc     n      nc     tt   r1     rcut   n     m
sss   -0.8753   0.0     0.0    2.89   0.0    1.0    0.0    1    3.40   3.55   2.0   2.0
sps.... etc.
...

!          A          C        nu       rnu
env   150000.0     0.45      2.1      2.89

et1 1 3.10 3.75 2.00 2.00 ! tt r1 rcut n m
et2 1 3.10 3.75 2.00 2.00 ! tt r1 rcut n m

pwp 1 5 0.002506717245 2.832 5.0 2.0 0.20
ptl 1 3.10 3.75 2.0 2.0 ! tt r1 rcut n m
```

Each bond must then have its parameters defined in this way, where for a system of Aluminium and Titanium both AlTi and TiAl must be defined.

The orbitals are listed in order of angular momentum and the second orbital in the interaction is equal to or bigger than the first, so sps represents a s-p sigma interaction, the reverse need not be listed as it will just be the negative.
Vscr is the value of the potential at nearest neighbour spacing or at r0.
DTau ????????????????????
Oscr???????????????????
Currently B.O.P. is set up so that it can only use G.S.P. functions to describe the orbital potentials, where **r0**, **rc**, **n** and **nc** are parameters in the G.S.P., however if **n=0** it implies a simple exponential scaling and if **nc=0** it implies power law scaling.
tt is the tail type:

0 = No tail.
1 = Augmented polynomial
2 = Multiplicative type I.
3 = Multiplicative type II (see mod_tail).

r1 is the radius at which the tail function connects to the hopping integral.
rcut is the radius at which the tail function must go smoothly and differentiably to zero.
n and **m** are the number of parameters you want to zero (smoothing the function) where n is the number at the point to connects to the function and m is on the remote side.
env sets the environmental terms. ???????
et1 and **et2** set the two tails, one for each atom considered in the production of the environment terms, where the parameters defining them have the same meaning as in the hopping integral tails.
pwp sets the pairwise potential, where the first number sets its type, the second the number of parameters to follow and the numbers after set the parameters, which will be different depending on the type.

0 = Null function.
1 = GSP function.
2 = Exponential.
3 = Plain polynomial potential (not yet implemented).
4,5 = Spline.

ptl sets the parameters for the pp tail, where **tt** is tail type, **r1** where the tail begins, **rcut** where it goes to zero and **n** and **m** the number of derivatives to set to zero on either side.

tt:
0 = Null tail.
1 = Polynomial tail.
2 = Binomial tail.
3 = **dstptail**.

```
elastic_const_conf
1.0   0.0   0.0
0.0   1.0   0.0
0.0   0.0   1.0
5   !polynomial order
-0.0025   0.0025   5
```

This sets the configuration of the elastic constants, the first three sets of numbers are the numbers by which the primitive lattice is multiplied in order to introduce shear or strain.

Energy is then calculated at various points and a polynomial is fitted to the values, the order of which is set in **polynomial order**.

The first two numbers at the bottom defines the set range that the polynomial is to cover and the last is the number of points at which the energy is calculated.

```
relaxation_conf
1 2 25000 0.01 0.01 ! reltype rlxflg mxiter ftol step
1 0 0 0.03 ! writeper datform autosave ok
0 ! cnst_n !XEC1 YEC1
1 1.0 0.0 0.0 ! cnst_a cnst_v(3r)
```

This category specifies the relaxation configuration.

reltype defines whether the relaxation is carried out at constant volume or not (2 for constant volume, 1 to relax the volume)

rlxflg defines the method of relaxation; 1 is Variable metric relaxation, 2 is Steepest descent relaxation, 3 is Conjugate gradient relaxation, 4 is the Fast Internal Relaxation Engine scheme and 5 is Steepest descent relaxation.

mxiter defines the maximum number of iterations in the relaxation cycle.

ftol defines what value the maximum force on an atom must have for the relaxation to complete.

step defines the tolerance in change in a step for the relaxation to be considered complete.

writeper specifies how many steps between each time the process writes information.

datform specifies the form of the written data.

autosave defines whether to save automatically or not.

cnst_n is the number of constraints, after this number list the constraints.

cnst_a cnst_v(3r) ???????????????

3 Dislocations

The code can be used to find the core and relax the structure of a dislocation in mvflag=16 or plot dislocation dynamics under strain with mvflag=17, in order to add dislocations a dislocation control file must be produced and saved as part of the run sequences as fort.21, possibly by including the command:

```
ln -s screw.in fort.21
```

where screw.in is the aforementioned control file.

This file must have a very specific structure in order to be processed by the system, with no additional vertical spaces etc.

```
4
C11, C22,    C12, C13,    C33 C66, C44, C55, C45
1.951 1.951 0.757 0.612 2.096 0.597 0.452 0.452 0.0
ORDINARY SCREW DISLOCATION
1/2[1,1, 1]
        A*D
        (4.209144E-01,2.034065E-01)
        (4.209146E-01,-2.034064E-01)
        (-8.418291E-01,-1.074751E-07)
        (-7.290445E-01,1.174381E-01)
        (7.290446E-01,1.174381E-01)
        (-1.529464E-08,-2.348728E-01)
        (-8.285043E-01,1.246910E-06)
        (-8.285043E-01,-1.346049E-06)
        (-8.284914E-01,-3.301518E-08)
        P(N)
        (3.831265E-01,6.409506E-01)
        (-3.831265E-01,6.409506E-01)
        (2.980232E-08,1.905314)
        CORE
        8.144553395      8.732768748      0      1/2(111)
     BX            BY            BZ
     0.0           0.0         -2.4681724
ORDINARY SCREW DISLOCATION
1/2[1,1, 1]
        A*D
        (-4.209144E-01,-2.034065E-01)
        (-4.209146E-01,2.034064E-01)
        (8.418291E-01,1.074751E-07)
        (7.290445E-01,-1.174381E-01)
        (-7.290446E-01,-1.174381E-01)
        (1.529464E-08,2.348728E-01)
        (8.285043E-01,-1.246910E-06)
        (8.285043E-01,1.346049E-06)
        (8.284914E-01,3.301518E-08)
        P(N)
        (3.831265E-01,6.409506E-01)
        (-3.831265E-01,6.409506E-01)
        (2.980232E-08,1.905314)
        CORE
        25.59716781      9.40452019      0      1/2(111)
     BX            BY            BZ
     0.0           0.0          2.4681724
......
```

This displacement file is to be applied to a cell with [111] as the z-axis, $[1\bar{1}0]$ as the y-axis and $[\bar{1}\bar{1}2]$ as the x-axis. The first number given is the number of dislocations, and so in this instance there would be four dislocations, although the specifications are only shown for the first two. After that each of the elastic constants must be given for the lattice. The displacements are produced by solving the linear elasticity equations for anisotropic media (as in Theory of dislocations, Price and Lothe (1982)) and so the parameters of the elasticity equations are required, where P(N) are the roots of the sixth order equation involving the elastic constants and A*D are involved in the equation:

$Re\,[\sum \pm A_{k(n)} D_{(n)}] = b_k$

The three numbers under CORE then specify the dislocation core position and those under BX,BY and BZ specify the components of the Burger's vector, note that these mustbe given in Angstrom as they will not be scaled by the len vectors, however note that in order to change the Burger's vector the A*D values must be changed as well.

The other two dislocations must be specified in the same way to those shown here.

Included below is a configuration file for a $\frac{1}{2}\langle 111\rangle\,\{1\bar{1}0\}$edge dislocation, to be applied to a cell with [111] as the x-axis, $[1\bar{1}0]$ as the y-axis and $[\bar{1}\bar{1}2]$ as the z-axis.

```
1
C11, C22,    C12, C13,    C33 C66, C44, C55, C45
2.096 1.95 0.611 0.612 1.951 0.452 0.597 0.452 0.0
ORDINARY EDGE DISLOCATION
1/2[1,1, 1]
        A*D
        (-1.768494E-02,4.976752E-01)
        (-1.768496E-02,-4.976759E-01)
        (2.520870,2.027857E-07)
        (-1.095423,-6.361356E-01)
        (1.095423,-6.361365E-01)
        (4.802389E-08,5.468352E-01)
        (-6.248119E-01,-1.279852)
        (-6.248116E-01,1.279853)
        (1.249624,5.963046E-08)
        P(N)
        (1.977849E-01,6.478031E-01)
        (-1.977850E-01,6.478031E-01)
        (-6.332994E-08,2.139978)
        CORE
        39.49075841        36.94635144        0.        1/2(111)
      BX              BY              BZ
      2.4855           0.0            0.0
```

4 NEB

BOP has the Nudged Elastic Band method built in, which can be used to calculate the minimum energy path between two system configurations when mvflag is set to 30. The initial and final configurations are specified in coordinate files labelled cell.in and cell_neb_fin.in respectively, which are both laid out exactly as in a normal cell file, the parameters for NEB may then be defined in the conf.in file:

```
neb_conf
11 0 10 ! nimg climb k
```

nimg defines the number of images in the chain that is interpolated between the two end points.

climb should set whether climbing image NEB is used or not, however this has not been implemented as of yet.

k defines the spring constant for the springs between images, however this is also not parsed from the conf file and so must be set in the source code. Within the code variable springs may also be set so that they become stiffer the closer they are to the saddle-point to ensure a tight grouping of images in that area, this is done using the parameters kmax and kmin.

Note that FIRE is the only relaxation mechanism thus far implemented for NEB and so the rlxflg must be set to 4. Also note that when moving atoms NEB will compute the distance they are to move and if it is greater than half the length of the cell and there is a possibility of taking a shorter route through a repeating boundary then the NEB will automatically move the atom in that direction.