Guidelines for Compiling VASP 5.4.4

Oliver Conquest

July 2020

1 VASP Versions

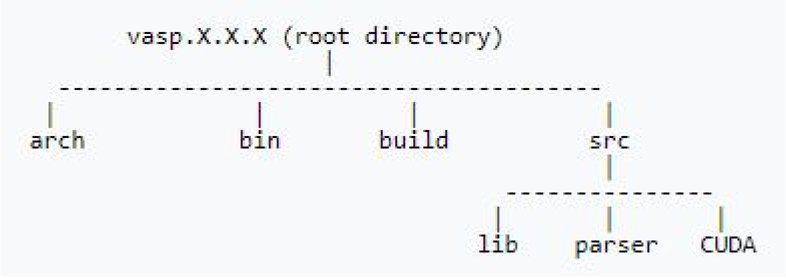
As of writing this guide version 5.4.4 is the recommended build for VASP. Version 6.1.1 is available but is not as stable and might be prone to crashing. There are also some differences between the root file structure of VASP 5.4.x and VASP 6.1.x.

2 VASP Installation Steps for Artemis

As of VASP 5.4.x installation has become more stream lined compared to older versions, with generalised `make.include` files provided in arch folder and the format of the `main.F` file has been changed to exclude the `SOURCE` section which is now located in the `.objects` file. The `.objects` file which may need to be edited when including certain VASP extensions.

The steps to compile VASP in Artemis are as follows:

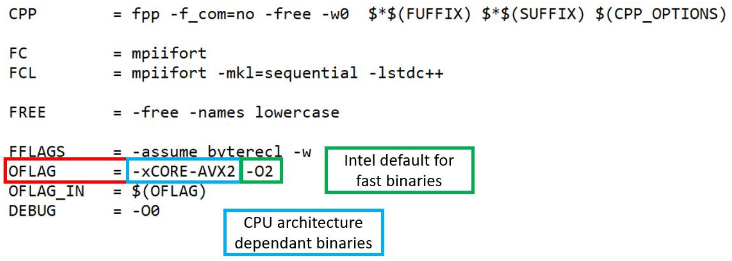
1. Make sure your VASP directory has the following structure (default for VASP 5.4.x):



```
        vasp.X.X.X (root directory)
                   |
  ---------------------------------------
 |            |          |            |
arch         bin       build         src
                                      |
                               ---------------
                              |       |       |
                             lib    parser   CUDA
```

2. Copy one of the `make.include.xxx_xxx` files from the `arch` folder to the root folder and rename to `make.include`, selecting the make file which best represents your system, in the case of Artemis the `makefile.include.linux_intel`. A quick bash command to copy the file to the root directory is `cp arch/makefile.include.linux_intel ./makefile.include` while in the VASP root directory.

3. This step is **optional** but may provide an improvement in the efficiency of VASP. In your root directory open the `makefile.include` file you copied and next to the `OFLAG` option add `-xCORE-AVX2` which is specific to Artemis's CPU architecture. The line in your `makefile.include` will be `OFLAG = -xCORE-AVX2 -O2`, as per the image below. To include the option from the command line use the following command (making sure you are located in your VASP root directory): `sed -i ’s/OFLAG\s*=/OFLAG = -xAVX/’ makefile.include`



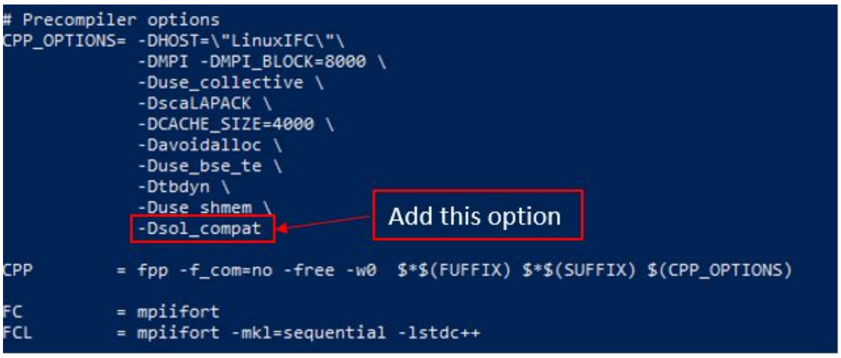
```
CPP        = fpp -f_com=no -free -w0  $*$(FUFFIX) $*$(SUFFIX) $(CPP_OPTIONS)

FC         = mpiifort
FCL        = mpiifort -mkl=sequential -lstdc++

FREE       = -free -names lowercase

FFLAGS     = -assume byterecl -w
OFLAG      = -xCORE-AVX2 -O2
OFLAG_IN   = $(OFLAG)
DEBUG      = -O0
```

Intel default for fast binaries
CPU architecture dependant binaries

For the **Physics Cluster** add the: `-xSSE4.2` option. For **Gadi** since it is using the latest CPU architecture (Cascade Lake) there does not seem to be binary flags for Cascade Lake available at the time of writing this guide, so leaving `OFLAG = -O2` will also work. For **Pawsey (Magnus)** add the same option as for Artemis: `-xCORE-AVX2` option.

If you would like to check which binary flag/s to use, look up the CPU architecture for your system, and find the corresponding binary flag which matches your CPU architecture from the link: https://software.intel.com/content/www/us/en/develop/articles/performance-tools-for-software-developers-intel-compi
html

4. If you want to install VASPSol with your VASP 5.4.4 you should include the `-Dsol_compat` flag in the `make.include` file (you copied to your VASP root directory) under the `CCP_OPTIONS` variable, see image below. This option ensures compatibility of VASPSol with the VASP's updated error function.



```
# Precompiler options
CPP_OPTIONS= -DHOST=\"LinuxIFC\"\
             -DMPI -DMPI_BLOCK=8000 \
             -Duse_collective \
             -DscaLAPACK \
             -DCACHE_SIZE=4000 \
             -Davoidalloc \
             -Duse_bse_te \
             -Dtbdyn \
             -Duse_shmem \
             -Dsol_compat

CPP        = fpp -f_com=no -free -w0  $*$(FUFFIX) $*$(SUFFIX) $(CPP_OPTIONS)

FC         = mpiifort
FCL        = mpiifort -mkl=sequential -lstdc++
```

It is recommend you use a text editor in linux such as nano to edit the pre-comiler file (`makefile.include`) as this file can be choosy about the use of spaces and tabs.

5. Copy the `solvation.F` file from the `/src/` folder in the VASPsol download folder to the `VASP5.4.4/src/` directory. You can download VASPSol from this link: https://github.com/henniggroup/VASPsol

6. Inorder to include the VTST extension which allows for NEB and other useful calculations in VASP download the source files from this link: http://theory.cm.utexas.edu/vtsttools/download.html and copy all the files with a `.F` format from the download folder to the `/src/` folder located in your VASP root directory.

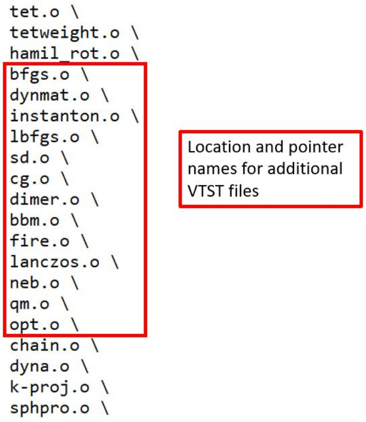
   We need to edit the `main.F` and `.objects` files in order to provide VASP with pointers to the additional files in the installation process. For the `main.F` file search for the following line:

   `CALL CHAIN_FORCE(T_INFO%NIONS,DYN%POSION,TOTEN,TIFOR, & LATT_CUR%A,LATT_CUR%B,IO%IU6)`

   and replace it with this line:

   `CALL CHAIN_FORCE(T_INFO%NIONS,DYN%POSION,TOTEN,TIFOR, & TSIF,LATT_CUR%A,LATT_CUR%B,IO%IU6)`

   Then for the `.objects` file we need to add the following file pointers before the `chain.o` pointer in the `SOURCE` variable (located at the start of the file), see image below. VTST tools have a guide on how to install their code in the `/src/` folder located in the VASP root directory, see the link: http://theory.cm.utexas.edu/vtsttools/installation.html **Note**, VTST have not updated their installation guide for VASP 5.4.4. File pointers are now located in the `.objects` file where as in previous versions $\leq$ 5.3.x the `SOURCE` variable was located in the `main.F` file.



```
tet.o \
tetweight.o \
hamil_rot.o \
bfgs.o \
dynmat.o \
instanton.o \
lbfgs.o \
sd.o \
cg.o \
dimer.o \
bbm.o \
fire.o \
lanczos.o \
neb.o \
qm.o \
opt.o \
chain.o \
dyna.o \
k-proj.o \
sphpro.o \
```

7. Once your confident you have changed and added the files you need to you can navigate back to the `vasp5.4.4/` root directory. Before we run the compile command we need to make sure we have the correct Intel compiler modules loaded (Intel compiler suites have a Fortran compiler which is able to compile VASP's Fortran code).

   For **Artemis** we need to load the following modules: `intel-mpi/19.1` and `libfabric/1.7.1`

   For the **Physics Cluster** it may only be the case of loading the `IntelCompilerSuite` module (but do check this, if a `libfabric` module is available then maybe load this as well to be safe.

   For **Gadi** try using the `intel-compiler/2020.1.217` and `libfabric/1.8.0a1` modules (maybe try earlier versions of the intel compiler suite if it doesn't work).

   Again with **Pawsey** try the latest version of there Intel Compiler Suite module and libfabric module (I'm currently not familiar with the module names on pawsey).

8. Lastly make sure you are located in your VASP root directory and type `make all` into the command line then the pre-compiler will run and VASP will compile. It usually takes around an 30-60 minutes to compile when using Intel Compilers (recommended), and maybe slower when using gfortran (gcc) compilers.