Compiling Zacros

Zacros uses a build system called cmake. It has the distinct advantage of working on Linux, Mac, and Windows. Installing it should prove relatively easy and is not covered here.

Compilation on Unices

This section covers all Unix-like operating systems, including Linux and Mac OS/X. It has been tested with gfortran, ifort, pgfortran. Compiling should be done using a terminal. It simply comes down to the following:

```
cd path/to/source/of/Zacros
mkdir build # If it does not exist yet
cd build
cmake .. -DCMAKE_BUILD_TYPE=Release
make
```

1. The first line above moves to the directory where the source code can be found.
2. Then we create a build directory. All the build files will be located here, rather than polluting the source.
3. We move to the build directory
4. The compilation is first configured for the current platform.
5. Then the code is compiled.

At the end of this process, there should be an executable called `kmcce` in the build directory.

The code can be tested by running:

```
cd /path/to/build/directory/
make test
```

Some of the tests do take a while...

Finally, it is possible to disable threading when compiling. This is only interesting when running on a computer with a single core. Simply replace (or rerun) line 4 with:

```
cmake .. -DCMAKE_BUILD_TYPE=Release -Ddoopenmp=OFF
```

Compilation on Windows using MSYS

MSYS provides a collection of GNU utilities that make it possible to code just like on any unix platform, and without shelling out for a proprietary compiler.

Starting from the MSYS command-line:

```
cd path/to/source/of/Zacros
mkdir build # If it does not exist yet
cd build
cmake .. -DCMAKE_BUILD_TYPE=Release -G "MSYS Makefiles"
make
```

This operation is quite similar to the the one performed in the case of Unices. Please look there for details. However, line 4 has changed a bit: the `-G "MSYS Makefiles"` option tells cmake that we want to compile for MSYS. That's it!

Tests can be run by entering `make test`. Be warned that some of the tests are fairly long. Threading can be removed by adding the option `-Ddoopenmp=OFF` to line 4 above.

Compilation on Windows using Visual something

There should be a Visual something project file lying around that can be used for such a purpose. Otherwise, if one has access to ifort, it is possible to recreate the project file using the command-line/powershell commands:

```
cd path/to/source/
mkdir build
cd build
cmake ..
```

The above should create the project files. The project should contain several possible configurations. For speed, one should build the Release configuration.

It is also possible to build and test directly from the command-line:

```
cmake --build . --config Release
ctest . -C Release
```

Thus happily bypassing inefficient GUI systems.