User Guide Part 2: Building ZEUS-MP

Author: John Hayes

NOTE:

Unlike Version 1.0, ZEUS-MP Version 2 **REQUIRES A FORTRAN 90 COMPILER.**

The Tar File Directory Tree

Upon untarring the tar file, the user will see a top-level directory named "zeusmp2". Under zeusmp2 are 4 subdirectories:

- src90: containing all ZEUS-MP source code files and Makefiles
- exe90: containing the zeusmp.x executable (once built), input file(s) required for execution, and any output files produced.
- test: containing input data decks for some example test problems
- pp: containing a minimal post-processing code for concatenating local output files written by each processor.

Customizing the Makefile

To compile ZEUS-MP, one needs to have a copy of "Makefile" in place in the src90 subdirectory. A sophisticated Makefile-building script is not currently in place, although one may be offered in a future release. Two currently valid example Makefiles are provided in the 2.0.0 tarball. **Makefile.sdsc.datastar** is valid on DataStar at the San Diego Supercomputing Center. **Makefile.nersc.seaborg** is valid on Seaborg at National Energy Research Scientific Computing Center. To create a working Makefile on your machine, use one of the provided Makefiles as a template and edit the following environment variables set in the Makefile:

- ZMP_EXE -- The relative path of the zeusmp.x executable.
- ZMP_CPP -- Contains CPP macros used to configure the code.
- ZMP_FC -- The name of the Fortran 90 compiler.
- ZMP_CC -- The name of the C compiler.
- ZMP_MOD -- The relative path to the directory containing the ".mod" F90 module objects generated during compilation of the ZEUS fortran routines.
- ZMP_OPTS -- Fortran compiler options.
- ZMP_LIB -- The absolute path to system libraries (e.g. HDF4).
- ZMP_LDR -- The command invoked to link the object files.

Including MGMPI

ZEUS-MP supports the use of the MGMPI elliptic linear system solver for the solution of Poisson's equation for problems that include self-gravity on 3D Cartesian meshes with non-periodic boundary conditions. The use of MGMPI in ZEUS-MP is enabled at compilation time by including the string, "-DUSE_MGMPI" in the ZMP_CPP definition line. In this case, the loader will look in the src90 subdirectory for a binary file called "libdmgmpi.a" This file is not included in the ZEUS-MP distribution, but rather must be built separately from the MGMPI source code, which may be obtained here.

If MGMPI is not used (USE_MGMPI is left undefined), then ZEUS-MP will default to the 3D conjugate gradient linear solver for 3D non-periodic meshes. Current (November 2005) research suggests that MGMPI will perform better than the CG solver once the mesh size exceeds 256^3.

The FFTw Gravity Solver Libraries

Compiling the code with the FFTw solver enabled requires that the Makefile be edited such that:

1. The string **-DFFT** be appended to the **ZMP_CPP** definition.
2. C routines **fftw_ps.c** and **fftw_plan.c** be included in the COBJ file list.

Makefile.sdsc.datastar provides an example of a Makefile properly configured to include the needed ZEUS-MP source files and the system library files on Datastar. **Makefile.nersc.seaborg** is written with all FFT references eliminated and thus provides a template for machines on which the FFTw libraries are absent.

An Operating System Issue: calling C routines

ZEUS-MP calls a handful of C functions, some of which are supplied in the source code and one of which ("etime") is a system library function. On some systems, trailing underscores are automatically appended to C routines during compilation, and on other systems (like IBM) they are not but must be for compilation to succeed. Files **intchk.F** and **clocks.F** employ a CPP macro, ARCH_IBM, which appends underscores when "-DARCH_IBM" is part of the ZMP_CPP definition in the Makefile. On non-IBM compilers, leaving this macro defined may break compilation, depending upon the needs of the compiler.

Compilation

With a proper Makefile, the command **make compile** is used to compile the code (**gmake** may also be used). During compilation, each fortran file with a ".F" suffix is processed by the C preprocessor, which generates a corresponding file with a ".f" suffix. These processed files are then used to create object files which are subsequently linked into the executable.

Desired changes to any subroutine must be applied to the ".F" copy of that routine.

Comments:

- The primary use of the C preprocessor is to expose/hide sections of the code to/from the compiler. The majority of macros defined in the ZMP_CPP variable pertain to MPI, and they are easily identified since "MPI" appears in the character string of each macro. Leaving "-DMPI_USED" out of the ZMP_CPP macro line causes all MPI code to be hidden from the compiler; in this way ZEUS-MP can be compiled as a serial code on systems which don't have MPI.
- The name of the subroutines which handle initialization specific to the problem and restarting a run for that particular problem are specified by the "PROBLEM" and "PROBRES" CPP macros,

respectively. For example, to run the sod shock test included with the code, your ZMP_CPP line must include "-DPROBLEM=sod" and would also include "-DPROBRES=sodres" in the unlikely event that you need to continue the Sod test from a restart file.

- The MGMPI multigrid Poisson solver is available as a separate package from the MGMPI Web page. It must be built as a separate library and included in the link path if it is to be used. When building ZEUS-MP, all references to MGMPI can be hidden from the compiler by leaving the "USE_MGMPI" CPP macro undefined, so it is not necessary to download MGMPI to use ZEUS-MP.