IO_tools

A number of input / output have been written for the MFIX RES/SPX files. Most of the routines were written in response to specific issues encountered by NETL users. However, some could have utility to others, such as sample_RES_SPX_reader. All of the tools are located in the tool/IO_tools sub-folder of the main mfix directory. There are README files for each code with usage instructions located in the tools/IO_tools/doc folder.

sample_RES_SPX_reader

This is sample C++ code to read MFIX RES and SPX files. This could be used as a starting point for code that translates MFIX output into a different format.

spatial_subset

This code creates a spatial subset of a MFIX run. The resulting RES and SPX files can be used in MFIX post processing routines. For example, the code can carve out a 11x51x16 subset of an original data set of size 42x203x62 starting at cell (10, 100, 15). If you have a section of special interest in a very large grid, you can create a spatial subset of that section and improve post processing speed.

split_spx

This code will split a SPX file into two separate files based on a user supplied split time. This can be useful to remove the initial section of a file.

combine_spx

This code appends the data on one SPX file to another SPX file. This can be used to combine results when doing 'RESTART_2' runs. That is, one folder has the results from 0 to 5 seconds. Another folder has the results from a ‘restart_2' for the next 5 seconds. This program will append the second folder results to the file in the first folder to allow for analysis of the entire data set.

time_subset

This code creates a time subset of the SPX files. For example, if the data in the SPX files ranges from time = 0 to 10, you can create SPX files with a subset of those time. For example, a subset with data ranging from time = 7 to 8 can be created.

fix_RES_file

The purpose of this code is to fix the following problems:

- In some versions of the Intel compiler, the values for NMAX[0] and NMAX[1] are not written correctly in the RES file. They are very large numbers, causing many post-processing codes to hang. The first option in this code outputs the current value in the files and prompts for a new value.
- In some cases when interpolating a grid with post_mfix, the cell FLAGS all get set to zero. This causes some post processing routines that look for fluid cells to not work. The second option does a fix to this by setting all interior cells to fluid, and all exterior cells to wall. Obviously this will not be appropriate for all runs.
- Modify the date stored in the header record of the RES file. The m2e translator (MFIX to Ensight case file) will not run if there is a mismatch in the header times in the RES files and SPx files. The mismatch sometimes occurs when doing restart_2 runs.

modify_time

This code will let you modify a time stored in a SPX file. Occasionally during a restart, the times in a SPX file will be out of order. The MFIX to Ensight Case translator (m2e) would not process a file in which this occurred. For example:

- t = 1.000
- t = 0.999
- t = 1.1

This code will allow you to manual change the 0.999 to a time compatible with m2e (such as 1.00001).

FILE_POSITION in MfixData.h

In MfixData.h, there is a typedef for the type of the variable for position in the file (seekg() and tellg()). Because files can get very large, this should be a 64 bit integer. The code uses ‘long long’, which works for g++, but is not standard. If your compiler does not support long long, use a 64 bit integral variable type that is supported by your compiler.

```
        typedef long long FILE_POSITION;
```