MRI-Q optimization
(Download ZipFiles)

1) Unzip the mp template into your sdk project directory

2) Edit any part of the code that you like, so long as you don't change the user interface or the timing structure. Your goal is to make the GPU kernel execution (not counting the memory allocation or copying times) as fast as you can with the following restriction.

The restults must be deterministic and match the result of the unoptimized code perfectly. This means you may not use the fast math versions of sin and cos, and the order of accumulation operations must be the same. While some optimizations can trade off accuracy for speed, we're asking you to maintain current semantics exactly.

3) The given interface for the application is as follows.

You must specify using the -i option the input file. The data/sutton... file is the sample input we provide and encourage you to use.

You may specify the option -S to get more accurate timings (inserts synchronization after non-blocking events). This is how we will measure your final speed.

You may specify an output file using the -o option. You can then analyse the output file however you like, including comparing it to other output files using "diff" to check for a match.

You may specify as the last command line parameter an integer number to limit the number of input samples used. This can be useful in testing or verifying your code in a shorter amount of time. For reference, we also provide correct output files for using 512 or 10000 samples. Keep in mind that your optimizations should not put restrictions on the number of samples you may be provided with as input, although you could potentially pad or otherwise handle it internally.

4) Your report should simply contain a journal of all optimizations you tried, including those that ultimately were abandonded or worsened performance. Your report should have an entry for every optimization tried, and each entry should note:

1) the changes you made for the optimization
2) any difficulties with completing the optimization correctly
(debugging effort, etc.)
3) the amount of time spent on it (even if it was abandoned)

Grading:

Your submission will be graded on the following parameters.

Demo/knowledge: 25%
- 10% Produces correct result output file for our test inputs.
- 25% = (Your runtime / TA solution runtime) * 25
(Yes, you can get over 100 on this assignment. Maybe.)

Functionality: 40%
- Major optimizations enabled
Register promotion, Memory space usage (constant? shared?)

Report: 35%
- Complete and accurate report. We will at least check for discrepencies,
optimizations that you did but didn't report, etc.