Gromacs - Bug #857
Sequencing rdtsc in cycle-counting funcs
01/04/2012 03:46 PM - Janne Blomqvist

Status: Closed
Priority: Low
Assignee:
Category:
Target version:
Affected version -
extra info:

Description
The x86 rdtsc instruction is not defined as a serializing instruction, thus the CPU is free to move rdtsc past other instructions in the pipeline. If using the TSC to, say, measure the time spent executing some piece of code, this is probably not what one wants, and the Intel system programming manual advises one to use "lfence;rdtsc" in that case. The AMD manual has similar wording, but instead advises "mfence;rdtsc". Also, an alternative to using a fence is "rdtscp" (which differs from rdtsc in that it is a serializing instr), if the CPU supports it.

This seems to affect at least

include/gmx_cyclecounter.h
include/thread_mpi/atomic/cycles.h

(checked on 4.5.5)

FWIW, on current Linux with high-resolution clocks (2.6.24+ or thereabouts IIRC) clock_gettime(CLOCK_MONOTONIC, ...) takes only about a factor of 3 as much time as a "{m,f}fence;rdtsc", so it's not like this particular ASM trickery buys one particularly much, but I digress.


Related issues:
Related to Gromacs - Bug #1428: rdtscp and illegal instructions

Associated revisions
Revision 5ba7125c - 06/16/2012 10:02 PM - Erik Lindahl
New CPU detection & AVX/SSE code, removed raw assembly files.

Removed all raw assembly files and deprecated altivec support.
Removed support for NASM and other assemblers, and replaced previous SSE detection code with a new module using CPUID instead.
Added detection for SSE2, SSE4.1, AVX 128-bit with FMA, and AVX 256-bit.
Added Cmake detection of build platform based on CPUID, and output this to the log file. The executables now compare the compile-time platform and selected acceleration with the run-time platform and most suitable acceleration and warns the user if they do not match. The compiler detection code has also been reordered slightly to produce more readable warnings when OpenMP is not available, and correctly disable pragma warnings.

Added intrinsics code and math functions for SSE2, SSE4.1, AVX128/256 both in single and double precision. All math functions and permutation code have been tested & verified. Single precision math functions are correct apart from the least significant bit, and double precision has roughly twice the accuracy.

This has forced me to temporarily disable the SSE & Fortran acceleration. SSE will be added back soon based on new intrinsics-only kernels currently in testing, and we will test if Fortran still makes sense then.

Finally, the patch includes a modification to gmx_rmsdist where
Revision 5ba7125c - 06/16/2012 10:02 PM - Erik Lindahl

New CPU detection & AVX/SSE code, removed raw assembly files.

Removed all raw assembly files and deprecated altivec support. Removed support for NASM and other assemblers, and replaced previous SSE detection code with a new module using CPUID instead. Added detection for SSE2, SSE4.1, AVX 128-bit with FMA, and AVX 256-bit. Added Cmake detection of build platform based on CPUID, and output this to the log file. The executables now compare the compile-time platform and selected acceleration with the run-time platform and most suitable acceleration and warns the user if they do not match. The compiler detection code has also been reordered slightly to produce more readable warnings when OpenMP is not available, and correctly disable pragma warnings.

Added intrinsics code and math functions for SSE2, SSE4.1, AVX128/256 both in single and double precision. All math functions and permutation code have been tested & verified. Single precision math functions are correct apart from the least significant bit, and double precision has roughly twice the accuracy.

This has forced me to temporarily disable the SSE & Fortran acceleration. SSE will be added back soon based on new intrinsics-only kernels currently in testing, and we will test if Fortran still makes sense then.

Finally, the patch includes a modification to gmx_rmsdist where a regression issue was introduced recently by using sqrtf() for the norm function. This caused the intel compiler to produce slightly different results at high optimization levels, which got evident here.

Closes #926 - Raw assembly code has been removed.
Refs #923 - Old kernels removed, new will be added shortly.
Fixes #914 - Cmake now does architecture-specific optimization.
Fixes #912, #913
Fixes #857 - We detect rdtscp support with CPUID and use it if possible.
Fixes #750
Closes #537, #574 - Altivec is now deprecated.

Change-Id: lcfca5a940762f8d82ae67b59c65b2d2ac683256d
Sander/Erik, could you comment on this?

Hi,

The only issues I've seen with RDTSC in practice is when executing it twice within a handful of instructions, and in that case it has usually been resolved through the volatile addition to the inline assembly (so it was probably the compiler moving it then).

The Linux high-resolution timers sound interesting, but we aren't yet ready to restrict support to 2.6.24+, and it would have to be tested to work on every single hardware platform, not only x86.

... but, before spending time on creating 3-4 separate code paths for Intel/AMD/old vs new Linux/rdtscp support, etc: Has there ever been a single example of a Gromacs run not executing as expected (incorrect load balance or whatever) due to this?

- Status changed from New to Closed

Fixed by 5ba7125c.

- Related to Bug #1428: rdtscp and illegal instructions added