Gromacs - Bug #1428

rdtscp and illegal instructions

01/31/2014 05:40 AM - Teemu Murtola

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Mark Abraham</td>
</tr>
<tr>
<td>Category:</td>
<td>build system</td>
</tr>
<tr>
<td>Target version:</td>
<td>4.6.6</td>
</tr>
<tr>
<td>Affected version - extra info:</td>
<td>4.6.*</td>
</tr>
</tbody>
</table>

Description

There have been several reports on gmx-users where our use of rdtscp has caused confusion and complaints for users. The current behavior is:

- If the build host has a CPU that provides the instruction, it is silently used in mdrun. The only way to see that this is enabled is to run mdrun -version, but most users probably don't understand the output.
- If such a binary is used on a host that doesn't support the instruction, then mdrun aborts with an illegal instruction. Many people seem to have now understood that switching GMX_CPU ACCELERATION to a common denominator for the hosts is a way to solve this (and this is even what people usually suggest on the mailing list), but even setting GMX_CPU ACCELERATION=None does not solve this problem.
- It is possible to disable rdtscp by providing -DGMX_DISTRIBUTABLE_BINARY=ON to CMake, but this is not documented anywhere, nor can this be set in any CMake GUI, since it is not provided as a cache variable.

Possible solutions of making this more user-friendly while keeping rdtscp:

- Minimal solution: document GMX_DISTRIBUTABLE_BINARY and make it a proper cache variable, so that users actually have a chance of finding this option.
- If the feature is compiled in, make mdrun (or the gmx binary) check in the beginning whether the instruction is actually supported by the CPU, and provide a clear error message in such a case. Unlike for the SIMD instruction set, this instruction shouldn't get automatically inserted by the compiler, so it should be easy to do the check early enough.

So the question now is that is the benefit of using rdtscp worth this trouble? Does it influence the timings that much?

Related issues:

- Related to Gromacs - Bug #857: Sequencing rdtsc in cycle-counting funcs
  - Closed 01/04/2012

Associated revisions

Revision a997205e - 02/20/2014 06:38 PM - Mark Abraham

avoid mdrun crash when rdtscp is not supported

When using rdtscp, mdrun now detects at runtime whether the CPU supports this instruction and if this is not the case, it issues a fatal error and instructs the user to recompile mdrun for the compute host. Note that this will happen rarely, only when cross-compiling from a newer host for a rather old one.

Additionally, when the user manually picks AVX, we also turn on RDTSCP as all AVX-capable CPUs support it.

Also made CMake advanced cache option for GMX_USE_RDTSCP. This replaces the previously hidden GMX_DISTRIBUTABLE_BUILD option.

Fixes #1428

Change-Id: i8bc884ef9ea8ea4661626bb60490182ae2b302648

11/19/2015
portability aspects in install guide + minor tweaks

Added information on portability aspects related to CPU instruction sets, related to #1428.

Additionally, made several minor updates and tweaks related to compilers, platforms, cmake, etc.

Change-Id: i621262c939c119e5bdd5e7c91dda0ae3f60b7b

History

#1 - 01/31/2014 05:40 AM - Teemu Murtola
- Related to Bug #857: Sequencing rdtsc in cycle-counting funcs added

#2 - 01/31/2014 03:12 PM - Mark Abraham

Teemu Murtola wrote:

There have been several reports on gmx-users where our use of rdtscp has caused confusion and complaints for users. The current behavior is:

- If the build host has a CPU that provides the instruction, it is silently used in mdrun. The only way to see that this is enabled is to run mdrun -version, but most users probably don't understand the output.

This is poor, and I have fixed it.

- If such a binary is used on a host that doesn't support the instruction, then mdrun aborts with an illegal instruction. Many people seem to have now understood that switching GMX_CPU_ACCELERATION to a common denominator for the hosts is a way to solve this (and this is even what people usually suggest on the mailing list), but even setting GMX_CPU_ACCELERATION=None does not solve this problem.

I've been suggesting this because I didn't know GMX_DISTRIBUTABLE_BINARY even existed! :-)

- It is possible to disable rdtscp by providing -DGMX_DISTRIBUTABLE_BINARY=ON to CMake, but this is not documented anywhere, nor can this be set in any CMake GUI, since it is not provided as a cache variable.

Possible solutions of making this more user-friendly while keeping rdtscp:

- Minimal solution: document GMX_DISTRIBUTABLE_BINARY and make it a proper cache variable, so that users actually have a chance of finding this option.

Yes

- If the feature is compiled in, make mdrun (or the gmx binary) check in the beginning whether the instruction is actually supported by the CPU, and provide a clear error message in such a case. Unlike for the SIMD instruction set, this instruction shouldn't get automatically inserted by the compiler, so it should be easy to do the check early enough.

Indeed, gmx_detect_hardware provides this information, and is called early in runner.c before any timing code, so the fix is straightforward.

So the question now is that is the benefit of using rdtscp worth this trouble? Does it influence the timings that much?
I can't answer that. But my patch at least now informs the user that they might want to arrange use it if they could and are not.

#3 - 01/31/2014 03:27 PM - Mark Abraham
- Status changed from New to Fix uploaded
- Target version changed from 5.0 to 4.6.6

#4 - 01/31/2014 03:27 PM - Mark Abraham
https://gerrit.gromacs.org/#/c/3051/

#5 - 01/31/2014 04:56 PM - Szilárd Páll
Change looks good.

One thing that I'd like to mention is that with this change we implicitly assume that GMX_DISTRIBUTABLE_BINARY will be present in the GROMACS build system for a while and it will do what its name means. Hence, anything we add in the future that can make the binary non-portable, will have to be turned off by this option. Right now it's only RDTSC and if we want to keep it that way, it is perhaps more advantageous to name the option GMX_USE_TSC_COUNTER.

#6 - 01/31/2014 06:26 PM - Mark Abraham
For implementations, it makes sense to have specific names until there is a second case from which to start generalizing. For user interfaces, it is sometimes friendlier to describe the intent or effect, rather than the detail. But with Erik's suggestion on the patch, very few people will need to use it, and making the name specific makes good sense.

Is there an advantage to "GMX_USE_TSC_COUNTER" as a name? HAVE_RDTSCP is the name of the #define that gets set, and the assembly instructions are similar. I'd like to avoid referring to TSC_COUNTER in one place and not in others. (Kinda like avoiding "pinning" vs "setting thread affinity.")

#7 - 02/05/2014 01:31 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1428.
Uploader: Szilárd Páll (pall.szilard@gmail.com)
Change-Id: I8bc884ef9ea8ea4661626b60490182ae2b302648
Gerrit URL: https://gerrit.gromacs.org/3082

#8 - 02/10/2014 02:22 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1428.
Uploader: Szilárd Páll (pall.szilard@gmail.com)
Change-Id: l621262c939c119e5bddd5e7c91ddd0a3e3ff60b7b
Gerrit URL: https://gerrit.gromacs.org/3121

#9 - 02/20/2014 06:45 PM - Mark Abraham
- Status changed from Fix uploaded to Resolved
- % Done changed from 0 to 100

Applied in changeset a997205edbf8b550904d1c6e9a0b2af7727067.
- Status changed from Resolved to Closed