## GROMACS - Task #1982

**work around compilation error with nvcc + glibc 2.23**

05/31/2016 11:20 PM - Szilárd Páll

<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>build system</td>
</tr>
<tr>
<td>Category</td>
<td>build system</td>
</tr>
<tr>
<td>Target version</td>
<td>2016</td>
</tr>
<tr>
<td>Difficulty</td>
<td>uncategorized</td>
</tr>
</tbody>
</table>

### Description

With some versions of nvcc and glibc compilation errors occur; reported on gmx-users [first](#) and [second](#) as well as by multiple other projects.

The issue seems to be caused by a string.h change in glibc and I've reproduced it with CUDA 6.5, 7.0 and 7.5, but not with 8.0.

### Related issues:
- Related to GROMACS - Bug #2022: Compilation fails with CUDA and gcc 5 Closed

### Associated revisions

**Revision e2cd2e2e - 06/29/2016 12:12 PM - Mark Abraham**

Work around glibc 2.23 with CUDA

Fixes #1982

Change-Id: I24671fcbdfdf1fb8bcc178edaeb801e849959266

### History

**#1 - 06/01/2016 07:51 AM - Mark Abraham**

Further, the issue is not specific to the compiler version (whether supported by CUDA or not), but rather to the version of glibc. Our source code being compiled in those reports is only including stdio.h, but including string.h is something that might happen (transitively) by whatever transformation nvcc does. And probably the issue is not unique to that single file in our source code (since other projects observe the same effect).

The workaround being suggested in various places is to add -D_FORCE_INLINES to the CUDA compiler command line, which disables the new code path in string.h. It is not yet clear what other effects that has in other glibc headers, but some quick grepping suggests it is specific to string-handling headers. In that case we don't care, and can just deploy the hack.

We could compile a test CUDA program (with `execute_process()`? I'm not sure how we arrange to call nvcc now) and add the flag if that fails, and then give up if there's a further problem. That infrastructure could also help address #1616

**#2 - 06/27/2016 07:21 PM - Gerrit Code Review Bot**

Gerrit received a related patchset ‘1’ for Issue #1982

Uploader: Mark Abraham (mark.j.abraham@gmail.com)

Change-Id: I24671fcbdfdf1fb8bcc178edaeb801e849959266

Gerrit URL: https://gerrit.gromacs.org/5991

**#3 - 06/27/2016 07:25 PM - Mark Abraham**

- Status changed from New to Fix uploaded
- Target version set to 2016

**#4 - 06/27/2016 08:03 PM - Szilárd Páll**

Thanks for the fix!

Mark Abraham wrote:

> We could compile a test CUDA program (with `execute_process()`? I'm not sure how we arrange to call nvcc now) and add the flag if that fails, and then give up if there's a further problem. That infrastructure could also help address #1616

I thought about hacking up a try_compile_nvcc() in the past, but some technical issues seemed to require more complex solution that what was worth the effort. Otherwise, it should be as simple as calling
nvcc -c test.cu -ccbin /path/to/compiler.binary
and checking its retval.

#5 - 06/30/2016 09:35 PM - Mark Abraham
- Status changed from Fix uploaded to Resolved

Applied in changeset e2cd2e2a10c7259ed58ddff26f0d5b0a38015c1.

#6 - 07/05/2016 12:40 AM - Mark Abraham
- Status changed from Resolved to Closed

#7 - 08/05/2016 12:33 AM - Mark Abraham
- Related to Bug #2022: Compilation fails with CUDA and gcc 5 added