Continuing the Debian architecture tests, "make check" fails on at least seven architectures: arm64, armhf†, mipsel, alpha, hppa, mips64el, and sparc64, with one or both of the following failures:

```
5/20 Test #5: HardwareUnitTests ................***Failed 0.08 sec
[----------] Running 5 tests from 2 test cases.
[----------] 1 test from CpuInfoTest
 [ RUN ] CpuInfoTest.SupportLevel
/«PKGBUILDDIR»/src/gromacs/hardware/tests/cpuinfo.cpp:72: Failure
Expected: (c.supportLevel()) >= (gmx::CpuInfo::SupportLevel::Features), actual: 4-byte object <01-00 00-00> vs 4-byte object <02-00 00-00>
No CPU features could be detected.
GROMACS might still work, but it will likely hurt your performance.
Please mail gmx-developers@gromacs.org so we can try to fix it.
[ F A I L E D ] CpuInfoTest.SupportLevel (1 ms)
[----------] 1 test from CpuInfoTest (1 ms total)
```

```
[----------] 4 tests from HardwareTopologyTest
 [ RUN ] HardwareTopologyTest.Execute
 [ OK ] HardwareTopologyTest.Execute (15 ms)
 [ RUN ] HardwareTopologyTest.HwlocExecute
/«PKGBUILDDIR»/src/gromacs/hardware/tests/hardwaretopology.cpp:86: Failure
Expected: (hwTop.supportLevel()) >= (gmx::HardwareTopology::SupportLevel::Full), actual: 4-byte object <02-00 00-00> vs 4-byte object <03-00 00-00>
Cannot determine full hardware topology from hwloc. GROMACS will still work, but it might affect your performance for large nodes.
Please mail gmx-developers@gromacs.org so we can try to fix it.
[ F A I L E D ] HardwareTopologyTest.HwlocExecute (14 ms)
 [ RUN ] HardwareTopologyTest.ProcessorSelfconsistency
 [ OK ] HardwareTopologyTest.ProcessorSelfconsistency (13 ms)
 [ RUN ] HardwareTopologyTest.NumaCacheSelfconsistency
 [ OK ] HardwareTopologyTest.NumaCacheSelfconsistency (13 ms)
[----------] 4 tests from HardwareTopologyTest (55 ms total)
```

Should these be informative messages rather than full failures? I can acquire more information about the build environment on each architecture, but maybe it's best to simply disable this test on them. There are probably not a lot of people running GROMACS on an Alpha workstation.

† armhf also encounters illegal instructions in the mdrun tests. The other architectures pass all remaining tests.
Associated revisions

Revision 0a7428cf - 07/08/2016 01:47 AM - Erik Lindahl

Reduce hwloc & cpuid test requirements

On some non-x86 linux platforms hwloc does not report caches, which means it will fail our strict test requirements of full topology support. There is no problem whatsoever with this, so we reduce the test to only require basic support from hwloc - this is still better than anything we can get ourselves. Similarly for CPUID, it is not an error for an architecture to not provide any of the specific flags we have defined, so avoid marking it as such.

Fixes #1987.

Change-Id: I0a065296bc647b7f7f5d3cb178e88df80fac81a7

History

#1 - 06/09/2016 10:11 PM - Mark Abraham

ARM64 we care about, and the others, probably we don't care as far as I know. I would guess from https://www.open-mpi.org/community/lists/hwloc-announce/2014/03/0068.php that the above behaviour might be expected from versions of hwloc before 1.9. What versions were in use?

#2 - 06/09/2016 10:25 PM - Nicholas Breen

hwloc 1.11.3 for all builds.

Between this and the other failures, beta2 is only successfully building on 3 of the 10 Debian release architectures, so I would like to see at least arm* + mipsel (+ mips, most likely, still in the build queue) fixed such that it can be included in the next Debian release.

#3 - 06/10/2016 01:50 AM - Szilárd Páll

Even if we don't care about running on somewhat exotic arch, I think building and running tests on some if not all (or e.g. allowing a subset of tests to run) would provide useful feedback and allow us to harden the portability of the code.

In particular, not being able to determine hardware topology should not trigger a test failure, I'd think. Not sure what's the best workaround, but could we not emit warnings instead of test failure -- the lack of hardware topology support would not prevent mdrun from functioning (not that anybody would want to use mdrun on these arch).

#4 - 06/10/2016 08:51 AM - Mark Abraham

Szilárd Páll wrote:

```
Even if we don't care about running on somewhat exotic arch, I think building and running tests on some if not all (or e.g. allowing a subset of tests to run) would provide useful feedback and allow us to harden the portability of the code.

Sure

In particular, not being able to determine hardware topology should not trigger a test failure, I'd think. Not sure what's the best workaround, but could we not emit warnings instead of test failure -- the lack of hardware topology support would not prevent mdrun from functioning (not that anybody would want to use mdrun on these arch).
```

Sure, but the purpose of those tests is to be noisy. If it's just a warning in mdrun, or in the test binary, nobody is likely to read it, which makes us feel more effective than we are. Unfortunately there's no effective way of diagnosing the actual problem without logging into a machine and seeing what hwloc returned and what Erik's new code does with it. For example, this could be down to our use of the hwloc-1.5 API not being rich enough for the platforms newer to hwloc.

One strategy would be to compile this code at configure time, and emit the warning there, and consequently disabling the test. It's more likely to be read by a human, and less likely to cause build-farm issues.
Mark Abraham wrote:

In particular, not being able to determine hardware topology should not trigger a test failure, I'd think. Not sure what's the best workaround, but could we not emit warnings instead of test failure -- the lack of hardware topology support would not prevent mdrun from functioning (not that anybody would want to use mdrun on these arch).

Sure, but the purpose of those tests is to be noisy. If it's just a warning in mdrun, or in the test binary, nobody is likely to read it, which makes us feel more effective than we are.

To the best of my knowledge unit test frameworks often support emitting a "WARNING" or similar status, a harmless message that provides useful information but does not lead to a failed "make check". In this case, it's safe to say that topology is not very relevant on some hardware that is exotic and Erik's comment hints exactly that. However I tend to disagree with the reasoning in that comment ("we might as well flag it so we add it to our detection code") because we don't want to "flag" everything that's not explicitly supported in the code, do we?

One strategy would be to compile this code at configure time, and emit the warning there, and consequently disabling the test. It's more likely to be read by a human, and less likely to cause build-farm issues.

Would it be enough to just compile without hwloc on platforms where one does not care about detailed CPU and topology info? It still seems to me that it's entirely legitimate to disable (or accept the lack of) such low-level detection on some architecture and not just exotic ones, but also e.g. during porting to a new platform. It would certainly make the dev's life who's porting e.g. for Power 9 easier if failed hardware topology detection does not keep spamming the test results.

Mark Abraham wrote:

Sure, but the purpose of those tests is to be noisy. If it's just a warning in mdrun, or in the test binary, nobody is likely to read it, which makes us feel more effective than we are.

To the best of my knowledge unit test frameworks often support emitting a "WARNING" or similar status, a harmless message that provides useful information but does not lead to a failed "make check". In this case, it's safe to say that topology is not very relevant on some hardware that is exotic and Erik's comment hints exactly that. However I tend to disagree with the reasoning in that comment ("we might as well flag it so we add it to our detection code") because we don't want to "flag" everything that's not explicitly supported in the code, do we?

I'd say we'll probably know of things we care about before someone runs such a test on it, so we could downgrade these to not be a failure.

One strategy would be to compile this code at configure time, and emit the warning there, and consequently disabling the test. It's more likely to be read by a human, and less likely to cause build-farm issues.

Would it be enough to just compile without hwloc on platforms where one does not care about detailed CPU and topology info? It still seems to me that it's entirely legitimate to disable (or accept the lack of) such low-level detection on some architecture and not just exotic ones, but also e.g. during porting to a new platform. It would certainly make the dev's life who's porting e.g. for Power 9 easier if failed hardware topology detection does not keep spamming the test results.

Maybe. But first we should find out if hwloc is working on the ARM system that we can test.

Mark Abraham wrote:

Sure, but the purpose of those tests is to be noisy. If it's just a warning in mdrun, or in the test binary, nobody is likely to read it, which makes us feel more effective than we are.

To the best of my knowledge unit test frameworks often support emitting a "WARNING" or similar status, a harmless message that provides useful information but does not lead to a failed "make check". In this case, it's safe to say that topology is not very relevant on some hardware that is exotic and Erik's comment hints exactly that. However I tend to disagree with the reasoning in that comment ("we might as well flag it so we add it to our detection code") because we don't want to "flag" everything that's not explicitly supported in the code, do we?

I'd say we'll probably know of things we care about before someone runs such a test on it, so we could downgrade these to not be a failure.

One strategy would be to compile this code at configure time, and emit the warning there, and consequently disabling the test. It's more likely to be read by a human, and less likely to cause build-farm issues.

Would it be enough to just compile without hwloc on platforms where one does not care about detailed CPU and topology info? It still seems to me that it's entirely legitimate to disable (or accept the lack of) such low-level detection on some architecture and not just exotic ones, but also e.g. during porting to a new platform. It would certainly make the dev's life who's porting e.g. for Power 9 easier if failed hardware topology detection does not keep spamming the test results.

Maybe. But first we should find out if hwloc is working on the ARM system that we can test.

Mark Abraham wrote:

Sure, but the purpose of those tests is to be noisy. If it's just a warning in mdrun, or in the test binary, nobody is likely to read it, which makes us feel more effective than we are.

To the best of my knowledge unit test frameworks often support emitting a "WARNING" or similar status, a harmless message that provides useful information but does not lead to a failed "make check". In this case, it's safe to say that topology is not very relevant on some hardware that is exotic and Erik's comment hints exactly that. However I tend to disagree with the reasoning in that comment ("we might as well flag it so we add it to our detection code") because we don't want to "flag" everything that's not explicitly supported in the code, do we?

I'd say we'll probably know of things we care about before someone runs such a test on it, so we could downgrade these to not be a failure.

One strategy would be to compile this code at configure time, and emit the warning there, and consequently disabling the test. It's more likely to be read by a human, and less likely to cause build-farm issues.

Would it be enough to just compile without hwloc on platforms where one does not care about detailed CPU and topology info? It still seems to me that it's entirely legitimate to disable (or accept the lack of) such low-level detection on some architecture and not just exotic ones, but also e.g. during porting to a new platform. It would certainly make the dev's life who's porting e.g. for Power 9 easier if failed hardware topology detection does not keep spamming the test results.

Maybe. But first we should find out if hwloc is working on the ARM system that we can test.

We have some ARM+CUDA systems available now, so hopefully we can address the targets we think are important for the long term.

Erik Lindahl wrote:

All these failures appear to be benign. Hwloc reports the basic topology just fine, but since there is not full info about caches, etc., it does not reach the level we require for x86. Similarly, while the ARM architectures we originally tested on had Neon, it is not an error if we run on architectures where we don't detect any specific CPU features.

So, for now I'm taking the test requirements down one notch - it won't affect any runs, and it hasn't exposed any errors, just that hwloc doesn't have
full bells & whistles everywhere, which is a-ok.

#9 - 07/05/2016 12:59 AM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1987.
Uploader: Erik Lindahl (erik.lindahl@gmail.com)
Change-id: I0a065296bc647b71f5d3cb178e68df80iac681a7
Gerrit URL: https://gerrit.gromacs.org/6007

#10 - 07/05/2016 12:59 AM - Erik Lindahl
- Status changed from Accepted to Fix uploaded

#11 - 07/08/2016 01:48 AM - Erik Lindahl
- Status changed from Fix uploaded to Resolved

#12 - 07/08/2016 08:24 AM - Erik Lindahl
Applied in changeset 0a7428cfec22280685ad1206a7aaeae8163ed98c.

#13 - 07/08/2016 01:38 PM - Erik Lindahl
- Status changed from Resolved to Closed

#14 - 07/15/2016 07:55 PM - Nicholas Breen
At least as of 2016-rc1, CpuInfoTest.SupportLevel is still a hard failure.

Example on mips: https://buildd.debian.org/status/fetch.php?pkg=gromacs&arch=mips&ver=2016%7Erc1-1&stamp=1468307509