Gromacs - Task #1545

test Random123 on unsupported platforms

07/01/2014 02:58 PM - Szilárd Páll

Status: Closed
Priority: Normal
Assignee:
Category: core library
Target version: 5.0.1

Description

The Random123 library officially only supports x86/AMD64 and PPC and on all other platforms a compilation error prevents it use without modification.

While it does compile and seems to work e.g. on 32-bit ARM, we need to test it on the platforms of our interest:

- 32/64-bit ARM
- SPARC
- Power
- ?

I suggest that we give feedback to the Random123 developers (they expressed their interest in it) so that at least Threefry gains (official) support and can be compiled out of the box on the platforms we tests on. For this we would need a reliable test that can easily confirm that the library works as expected.

Related issues:

Related to Gromacs - Bug #995: Better parallel random number generator Closed 08/18/2012
Related to Gromacs - Bug #1542: two unit tests fail on 32-bit ARM Closed 06/30/2014

History

#1 - 07/01/2014 03:00 PM - Szilárd Páll
Is the recently uploaded change good enough to confirm that the RNG works correctly? Does Random123 provide a complete set of unit or regression tests that we could run?

#2 - 07/01/2014 03:00 PM - Szilárd Páll
- Related to Bug #995: Better parallel random number generator added

#3 - 07/01/2014 03:00 PM - Szilárd Páll
- Related to Bug #1542: two unit tests fail on 32-bit ARM added

#4 - 07/02/2014 01:22 AM - Roland Schulz
They validated the library using http://simul.iro.umontreal.ca/testu01/tu01.html and we could run that on ARM. It tests that the random numbers fulfill all statistical properties one expects from random numbers. But I don’t think we need to do that for ARM. If we test that ARM gives the same results as x86 and we know that x86 gives good numbers, that is sufficient. That is what their known-answer-tests (kat) does. That is the only test which comes with Random123 and tests the random numbers themselves (there are further tests for features we don’t use). Because we only use one generator (threefry2x64 with 20 rounds) we only need to test that one. If you take their kat test and reduce it to the one generator than you get the same thing as what my unit test does.

#5 - 07/02/2014 09:00 PM - Mark Abraham
I agree with Roland that agreement in generating number is all we need for GROMACS purposes, but if we’d like to influence the Random123 devs to declare more architectures as supported, we would probably need to show something more substantial was also correct.
Yes in that case we should run all the tests in the original distribution. Even than I don't think we need to run the crush tests.

I agree with Roland that agreement in generating number is all we need for GROMACS purposes, but if we'd like to influence the Random123 devs to declare more architectures as supported, we would probably need to show something more substantial was also correct.

Yeah, I was going to write that we may need to ask/suggest a more modular design which allows including only a certain algorithm (in our case Threefry) as well as limitations imposed separately on each algorithm. However, I ended up avoiding this suggestion because it sounds quite over-engineered and would probably not be worth the complication for the Random123 devs.

However, if there are non-SIMD fallback code-paths for all algorithms as well as appropriate kat (or other tests) that are ready to use, it wouldn't be a big effort to test the entire library on the few platforms of interest which we can anyway access right now.

threefry is being proposed to being added to boost: http://lists.boost.org/Archives/boost/2014/04/212865.php. If that goes through and we convert the users of our random from C to C++, we can use that in the future.

Some algorithms (ARS) are only available with hardware support. But I assume then they are automatic left out from the testing.

On an Tegra3, 32-bit ARMv7, the pi_cppapi test fails with with >-O1 using both gcc 4.6 (4.6.3-1ubuntu5 and 4.6.4-1ubuntu1~12.04 too) and 4.7 (4.7.2-2ubuntu1).

CC=gcc-4.6 CXX=g++-4.6 make runpi_cppapi CFLAGS="-std=c99" CXXFLAGS="-std=c++0x" CPPFLAGS="-O3 -Wall -Wstrict-aliasing=2" g++-4.6 -std=c++0x -O3 -Wall -Wstrict-aliasing=2 -l.../include pi_cppapi.cpp -o pi_cppapi
../pi_cppapi
Throwing 10000000 darts at a square board using Threefr4x64 7499459 out of 10000000 darts thrown at a square board hit the inscribed circle pi is approximately 2.9997836 (diff = -4.5 %) May not be OK, # of hits is more than three 'sigma'. Worth looking into. (chisquared = 7.5e+04) make: *** [runpi_cppapi] Error 1

As the "pi_capi" test passes, I assume that this may be some ARM-specific g++ problem. I'll try to install some newer gcc (or clang) later.

Additionally, the "time_thread" test also fails with an error, but the time_serial passes, so this may be an issue with the multi-threading in the tests themselves.

For curiosity I also checked the performance too and based on the "time_serial" throughput tests, a single Tegra 3 is 20-40x slower than x86 (lower limit based on AMD Thuban, upper based on Intel Haswell).
Hi,

I would reformulate the task slightly. We don't want to be in the situation of having to port Random123 to each new platform anybody wants to use Gromacs on ourselves, and we similarly don't want Random123 to be a new bottleneck that prevents people from running Gromacs on non-standard platforms - that portability is something we have invested two decades of work into.

Thus, rather than specifically testing it on one platform at a time, we need to assess whether we can make Random123 truly portable with some general defines (even if that means sub-optimal performance), and if that is not possible or very fragile, we need to start looking into alternative RNGs.

PS: Having said that, if we have a version that we think should be portable, I can help test it both on Fujitsu Sparc64 and two other special new platforms that we can't disclose officially yet.

Erik Lindahl wrote:

PS: Having said that, if we have a version that we think should be portable, I can help test it both on Fujitsu Sparc64 and two other special new platforms that we can't disclose officially yet.

I think Roland's done this in the patches that are in gerrit - the proposal is to hack out the compiler-hardware-OS-portability stuff provided by Random123, and use our own such gear, for the subset of Random123 that is of interest, and a unit test for the one function we use. I've inspected the threefry.h file, and indeed it uses integer arithmetic and basic bitwise operations, as expected. So if Erik can try https://gerrit.gromacs.org/#/c/3736/ on such platforms, then I think our work here is nearly done - correctness is assured by our unit test if the compiler and hardware work correctly, we've eliminated the defensive compile-time warnings, and nobody needs to do new work for future platforms.

Hopefully Szilard can show that ARM + gcc behaves better with Crush for more recent gcc because of some bug fix, but its a separate issue from "GROMACS should remain easy to port" (which I strongly support, of course!).

gcc 4.8 fixes the ARM issue. Adding the the Pi tests as regression tests too could be useful, especially as the ARM issue was revealed only by these.

Szilard Páll wrote:

gcc 4.8 fixes the ARM issue. Adding the the Pi tests as regression tests too could be useful, especially as the ARM issue was revealed only by these.
Having had a look at the tests' source, the ARM issue I come across is probably caused by either Threelfry4x64 or the use of ReinterpretCtr to get 32-bit values in contrast with the other test where Threelfry2x64 used with simple bit extraction.

Still, the simple pi_capi could be useful as a unit test.

#15 - 07/03/2014 06:27 PM - Roland Schulz

Our existing regressions tests (in particular complex/address) are sensitive to correct random numbers. I don't think pi_capi can help further.

#16 - 10/01/2014 05:34 PM - Mark Abraham

- Status changed from New to Closed
- Target version set to 5.0.1