C reference code for optimized SIMD kernels (nbnxnk4xN_SIMD_2xNN and nbnxnk4xN_SIMD_4xN)

03/03/2013 12:48 PM - Mikhail Plotnikov

Status: Closed
Priority: Normal
Assignee: Berk Hess
Category: mdrun
Target version: 4.6.4
Difficulty: uncategorized

Description
Currently SIMD kernels are written in intrinsics and don’t have C reference code. Plain C kernels (nbnxnk4x4_PlainC) cannot be considered as reference because different data format is used (xyzxyzxyz instead of xxxxyyyyyzzzz in SIMD kernels) and different size of UNROLLJ (in PlainC it is fixed to 4, while in SIMD kernels it varies depending on register width). Thus, neither functionally nor from performance point of view PlainC and SIMD kernels cannot be compared.

I wonder why this step is missing because intrinsics or assembly code should always have a good C reference. The purpose of such reference is the following:
1) Portability of the code. For any hardware, which doesn’t yet have optimized SIMD kernels, adequate C reference would be a good point to start.
2) Development purpose. When developing optimized SIMD kernels for a new hardware, C reference would be very convenient for debugging.
3) Testing of GROMACS on new coming architectures with new instructions sets, such as AVX2, AVX3 and beyond, which are not going to be optimized for on intrinsics in the nearest future.
4) Compiler testing. GROMACS is a part of different test suites, which are widely used for compiler abilities evaluation. This helped significantly improve compiler tools and even in some cases it started generate faster code than provided assembly kernels. This became possible only because old non-bonded group kernels had adequate C reference for assembly code. Such C reference code also will let us understand how good SIMD optimized kernels compared to what compiler may generate now.

Mostly appreciated are C references for 256bit/512bit nbnxnk4xN_SIMD_2xNN kernels.
Then go 256bit/512bit nbnxnk4xN_SIMD_4xN kernels.

Associated revisions
Revision eb153417 - 07/24/2013 03:30 PM - Berk Hess
implemented plain-C SIMD macros for reference

This is mainly code reorganization.
Adds reference plain-C, slow, arbitrary width SIMD for testing.
Adds FMA for gmx_calc_rsq_pr.
Adds generic SIMD acceleration (also AVX or double) for pme solve.
Moved SIMD vector operations to gmx_simd_vec.h
The math functions invsqrt, inv, pmecorrF and pmecorrV have been copied from the x86 specific single/double files to generic files using the SIMD macros from gmx_simd Macros.h.
Moved all architecture specific nbnxn_kernel_simd Utils code to separate files for each SIMD architecture and replaced all macros by inline functions.
The SIMD reference nbnxn 2xnn kernels now support 16-wide SIMD.
Adds FMA for in nbnxn kernels for calc_rsq and Coulomb forces.

Refs #1173
Change-Id: leda78cc3bcb499e8c17ef8ef539c49cbbc2d6d74d

Revision 5dee8e8a0 - 10/17/2013 03:50 PM - Roland Schulz
Fix SIMD C reference nbnxn kernels

Got broken by ace006a86 and 022581b388.
An additional fix for nbnxn 4x8 reference code, broken by c0cf8ce, is in a separate patch.
Also changed the AVX256 double precision nbfp_stride from 4 to 2.

Refs #1173
Created SIMD module

Moved header files to new module, renamed their #include guards appropriately.

Removed two unused headers (content mostly duplicated)

Refs #1173

---

History

#1 - 03/03/2013 04:29 PM - Szilárd Páll
- Assignee deleted (Szilárd Páll)
- Target version changed from 4.6.1 to 4.6.2

Not blocker for 4.6.1, bumped to 4.6.2 and removed myself as assignee because it's not my turf. :)

Otherwise, I think it is a valid request with some good points on why is such reference code needed. IMO it should be rather simple to write such kernels, but I'll let others decide how they feel about the gain/effort ratio.

Mikhail, are the plain C group scheme kernels adequate for your purposes?

#2 - 03/04/2013 06:57 PM - Mark Abraham

I agree that this would be a nice thing to help cater for future kernels. Once upon a time, the mismatch between reference and SIMD kernels was worse than it is now :-() Mikhail's point 4 is particularly interesting. We know GROMACS kernels make for challenging targets for auto-vectorization, but in principle future compilers should be able to do that for us.

The way the SIMD kernels use macros should make it fairly straightforward to implement plain C reference kernels using the various SIMD data formats. Berk's done quite a bit of work recently aimed at making future kernel ports easier to do. I'm not quite sure about a "drop in" replacement for gmx_mm_pr (see [https://github.com/gromacs/gromacs/blob/release-4-6/include/gmx_simd_macros.h](https://github.com/gromacs/gromacs/blob/release-4-6/include/gmx_simd_macros.h)). A real[4] could serve, but the C implementation of all the functions that use it would be forced to do lots of looping. That's OK, and a good vectorizing compiler might even do a better job with that than the current reference kernel!

#3 - 04/30/2013 05:04 PM - Mark Abraham

- Status changed from New to Accepted
- Assignee set to Berk Hess
- Target version deleted (4.6.2)

Work is in progress

#4 - 05/24/2013 07:48 PM - Mark Abraham

- Target version set to 4.6.3

Or 4.6.2 if someone else reviews it

#5 - 05/27/2013 06:07 AM - Mark Abraham

- Status changed from Accepted to Fix uploaded

[https://gerrit.gromacs.org/2326](https://gerrit.gromacs.org/2326)

#6 - 06/26/2013 12:36 AM - Mark Abraham

- Target version changed from 4.6.3 to 4.6.x

#7 - 08/14/2013 12:11 AM - Mark Abraham

- Target version changed from 4.6.x to 4.6.4

#8 - 10/16/2013 03:30 PM - Mark Abraham

- Status changed from Fix uploaded to Closed
Some bug fixes are still in train.