Efficient Calculation of Pairwise Nonbonded Forces

Matt Chiu Md. Ashfaquzzaman Khan Martin C. Herbordt
Department of Electrical and Computer Engineering
Boston University, Boston, MA

***Abstract*—A major bottleneck in molecular dynamics (MD) simulations is the calculation of the pairwise nonbonded interactions. Previous work on FPGAs has shown that these calculations can be implemented with a number of force computation pipelines operating in parallel (4 and 8 for the Stratix-III and Stratix-V, respectively). Optimization has received some attention previously in CPU, GPU, FPGA, and ASIC implementations, with direct computation of the equations of interaction being replaced with table lookup with interpolation, and the order and granularity of those interpolations being optimized. FPGAs lend themselves to a particularly rich design space both of opportunities and constraints. We explore and evaluate this space with respect to both resource requirements and simulation quality. We find that FPGAs' BRAM architecture makes them well suited to support unusually fine-grained intervals. This leads to a reduction in other logic and a proportional increase in performance. We demonstrate these designs with prototype implementations supporting full electrostatics and integrated into NAMD-lite. Throughput is improved by 50% over the previous best FPGA implementation while simulation quality is maintained.**

***Keywords*-High Performance Reconfigurable Computing; Molecular Dynamics**

I. Introduction

We have shown previously [1] that FPGA-based MD acceleration can be highly competitive even with GPU-based methods. Recently we described progress towards extending this work into a production FPGA-accelerated MD system [2] through integration into NAMD-lite [3] and mapping onto a Gidel PROCStar III board. To achieve the necessary compatibility and simulation quality, the force pipelines, which compute the pairwise nonbonded forces, were extended in two ways: (i) to support the short-range part of the Particle Mesh Ewald method of computing the electrostatic potential (in addition to the Multigrid method previously implemented [1]) and (ii) with the addition of a switching function to the van der Waals calculation. These extensions resulted in a reduction in the number of force pipelines from 8 to 4 (on an Altera Stratix-III SE260).

The problem addressed here is optimizing the computation of the pairwise nonbonded force in light of this added complexity. The equations for the pairwise nonbonded forces, as with any non-trivial equations, lend themselves to being computed in a number of ways. These can be grouped into two categories: direct and by table lookup with (or without) interpolation. Within each method are many further variations; here we focus on table lookup and the critical parameters of table size and the interpolation order. Besides performance, design decisions also affect simulation quality. In general, more accuracy requires more hardware. Optimization of performance versus quality, however, is non-trivial. Accuracy only affects simulation quality indirectly and highest quality simulations may not be needed.

In this study we explore and evaluate this space of possible solutions with respect to both resource requirements and simulation quality. We find that FPGAs' BRAM architecture makes them well suited to support unusually fine-grained intervals. This leads to a reduction in other logic and a proportional increase in performance. We demonstrate these designs with prototype implementations supporting full electrostatics and integrated into NAMD-lite. Throughput is improved by 50% over the previous best FPGA implementation while simulation quality is maintained.

II. Preliminaries

A. Molecular Dynamics Review

MD is an iterative application of Newtonian mechanics to ensembles of atoms and molecules (see, e.g., [1], [4] for details). MD simulations generally proceed in iterations each of which consists of two phases, force computation and motion integration, of which the force computation dominates. Within the force computation, a bottleneck is calculating the pairwise nonbonded forces:

$$\frac{\mathbf{F}_{ji}^{short}}{\mathbf{r_{ji}}} = A_{ab}r_{ji}^{-14} + B_{ab}r_{ji}^{-8} + QQ_{ab}(r_{ji}^{-3} + \frac{g_a'(r)}{r}) \quad (1)$$

where the first two terms compute the van der Waals force and the third the Coulombic force. A_{ab}, B_{ab}, and QQ_{ab} are distance independent coefficient look-up tables indexed with atom types a and b, and the g term is a correction for integration with the long-range force.

We now describe issues in their actual implementation. While the van der Waals term shown in Equation 1 converges quickly, a switching function must still be implemented to effect smoothness at the cutoff distance (see Equations 2-4 and also [2]) and so ensure energy conservation.

$$\begin{gathered} s = (cutoff^2 - r^2)^2 * \\ (cutoff^2 + 2 * r^2 - 3 * switch\_dist^2) * denom \end{gathered} \quad (2)$$

$$ds_r = 12 * (cutoff^2 - r^2)^2 * (switch\_dist^2 - r^2) * denom \quad (3)$$

$$denom = 1/(cutoff^2 - switch\_dist^2)^3 \quad (4)$$

The van der Waals force and energy can be computed directly as shown here:

This work was supported in part by the NIH through award #R01-RR023168-01A1.

IF $(r^2 \le switch\_dist^2)$ $\quad U_{vdW} = U,\ F_{vdW} = F$
IF $(r^2 > switch\_dist^2\ \&\&\ r^2 < cutoff^2)$

$$U_{vdW} = U * s,\ F_{vdW} = F * s + U_{vdw} * ds_r$$

IF $(r^2 \ge cutoff^2)$ $\quad U_{vdW} = 0,\ F_{vdW} = 0$

We now examine the Coulomb term. The most flexible method in NAMD-lite of calculating the electrostatic force/energy is Particle Mesh Ewald (PME), where the pairwise component is as follows:

$$E_s = \sum_{i=1}^{N-1} \sum_{j>1}^{N} \frac{q_i q_j erfc(\beta r_{ij})}{r_{ij}} \tag{5}$$

where $erfc(x)$ is the complementary error function and β is the Ewald parameter.

B. Table Look-Up with Interpolation

A major consideration is whether to compute directly or to use table look-up with interpolation. Most MD codes use multiple tables; Equation 1 can be rewritten as

$$\frac{\mathbf{F}^{short}_{ji}(|r_{ji}|^2(a,b))}{\mathbf{r_{ji}}} = \tag{6}$$

$$A_{ab}R_{14}(|r_{ji}|^2) + B_{ab}R_8(|r_{ji}|^2) + QQ_{ab}R_3(|r_{ji}|^2)$$

where R_{14}, R_8, and R_3 are lookup tables indexed with $|r_{ji}|^2$ (rather than $|r_{ji}|$ to avoid the square-root operation).

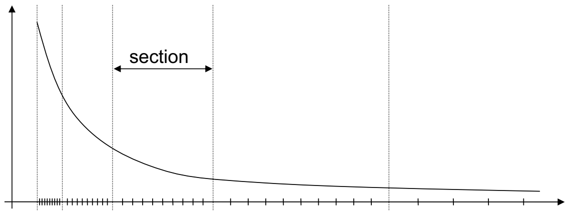

Figure 1. Table look-up varies in precision across r^{-k}. Each section has a fixed number of *intervals*.

The intervals in the tables are shown in Figure 1. Curves are divided into several sections such that the length of each section is twice that of the previous. Each section, however, is cut into the same number of intervals N. To improve the accuracy, higher order terms can be used. When the interpolation is order M, each interval needs $M+1$ coefficients, and each section needs $N*(M+1)$ coefficients. Equation $F(x) = a_0 + a_1x + a_2x^2 + a_3x^3$ shows third order with coefficients a_i. Accuracy increases with both the number of intervals per section and the interpolation order.

III. Design Space

A. Sample Implementations

We now present a sample of the methods of force computation used in widely used MD packages and systems.

NAMD (CPU) – Ref: Source code of NAMD2.7
Order = 2 bins/segment = 64 Index: r^2
Segments: 12 – size increases exponentially, from $0.0625\AA$

NAMD (GPU) – Ref: Source code of NAMD2.7 and [5]
Order = 1 bins/segment = 64 Index: $1/\sqrt{r^2}$
Segments: 12 – segment size increases exponentially, starting from $0.0625\AA$

CHARMM – Ref: [6]
Order = 2 bins/segment = 10-25 Index: r^2
Segments: Uniform segment size of $1\AA^2$ is used which results in relatively more precise values near cut-off

ANTON – Ref: [7] — Force Table Order = Says 3 but that may be for energy only. Value for force may be smaller.
# of bins = 256 Index: r^2
Segments: Segments are of different widths, but values not available, nor whether the number of bins is the total or per segment.

GROMACS – Ref: GROMACS Manual 4.5.3, page 148
Order = 2 bins = 500 (2000) per nm for single (double) precision
Segments: 1 Index: r^2
Comment: Allows user-defined tables.

Clearly there are a wide variety of parameter settings that have been chosen with regard to cache size (CPU), routing and chip area (Anton), and the availability of special features (GPU texture memory). The parameters also have an effect on simulation quality.

B. Force Pipeline Designs

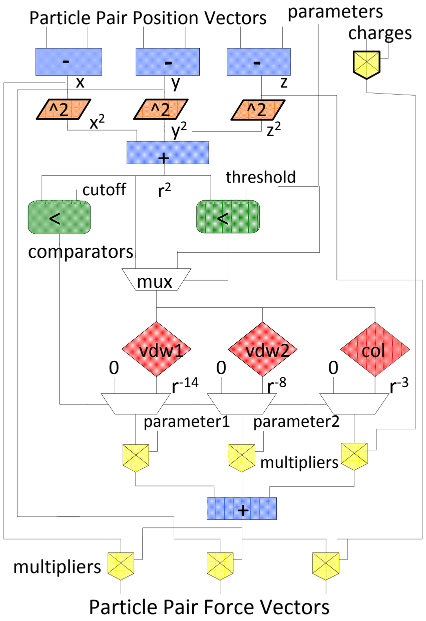

Figure 2. Force pipeline template.

Figure 2 illustrates the major functional units of the force pipelines. The force function evaluators are the diamonds marked in red; these are the components which can be implemented with the various schemes. The other units remain mostly unchanged. The three function evaluators are for the R_{14}, R_8, and R_3 components of Equation 6, respectively. In particular, **Vdw Function 1** and **Vdw Function 2** are the R_{14} and R_8 terms but also include the cutoff shown in Equations 2-4. **Coulomb Function** is the R_3 term but also includes the correction shown in Equation 5.

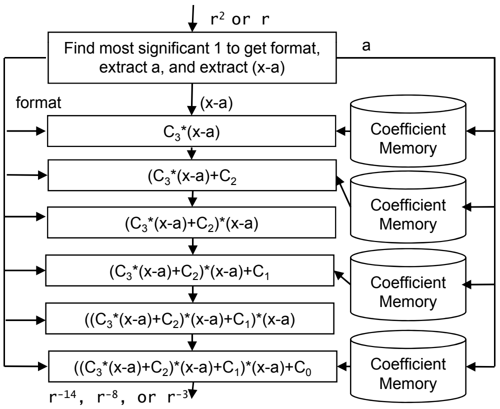

Figure 3. Arithmetic flow of a function evaluated with table lookup and 3rd order interpolation.

Figure 3 shows the basic flow to compute the force function with 3rd order interpolation. This consumes three multipliers and three adders for each function, as well as four coefficients per bin (interval).

IV. Results: Quality and Performance

A. Target System Overview

Our accelerated MD system is currently running on one FPGA of a Gidel PROCStar III board. Each processing unit contains an Altera Stratix III SE260 FPGA and three memory banks, each of which has a 128-bit interface. The host PC is a 64-bit 3GHz Xeon quad processor (Harpertown X5412) with 8GB of memory. The accelerator has been embedded into NAMD-lite (see [2] for details).

B. Simulation Quality

Since MD is chaotic, simulation quality must be validated. For example, systematic error can be introduced, e.g., as the motion integration algorithm generally assumes the force is continuous and differentiable [8]. Quality measures can be classified as follows (see, e.g., [6], [9], [10]).

1. Arithmetic error in the approximation is the deviation from the ideal (direct) computation done at high precision (double precision). A frequently used measure is the relative RMS force error, which is defined as follows [11]:

$$\Delta F = \sqrt{\left(\frac{\sum_i \sum_{\alpha \in x,y,z} [F_{i,\alpha} - F^*_{i,\alpha}]^2}{\sum_i \sum_{\alpha \in x,y,z} [F^*_{i,\alpha}]^2}\right)} \qquad (7)$$

While this can be computed precisely, it may hide effects of discontinuities in piecewise approximations [8].

2. Physical invariants should remain so in simulation. Energy can be monitored through fluctuation (e.g., in the relative RMS value) and drift. We use the following expression (suggested by Shan et al. [11]):

$$\Delta E = \frac{1}{N_t} \sum_{i=1}^{N_t} |\frac{E_0 - E_i}{E_0}| \qquad (8)$$

where E_0 is the initial value, N_i is the total number of time steps in time t, and E_i is the total energy at step i. Acceptable numerical accuracy is achieved when $\Delta E \leq 0.003$.

The results presented here are for the NAMD benchmark NAMD2.6 on ApoA1. It has 92,224 particles, a bounding box of $108\mathring{A} \times 108\mathring{A} \times 78\mathring{A}$, and a cut-off radius of $12\mathring{A}$.

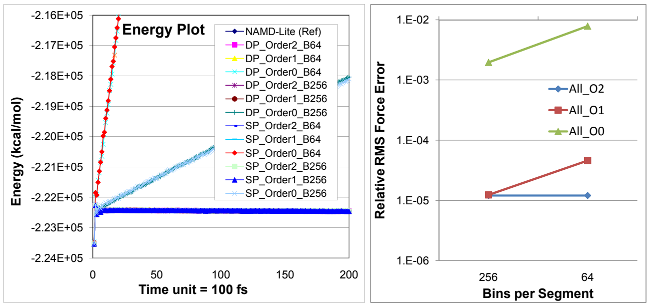

Figure 4. Right graph shows Relative RMS Force Error versus bin density for interpolation orders 0, 1, and 2. Left graph shows energy for various designs run for 20,000 timesteps. Except for 0-order, plots are indistinguishable from the reference code.

To determine the force error, NAMD-lite was modified to support the various functions. The simulation was first run for 1000 timesteps using direct computation. Then in the next timestep, both direct computation and table interpolation were used to find the relative RMS force error for table interpolation. Only the range limited forces (switched vdw and short-range portion of PME) were considered. All computations were done in double precision; Equation 7 was used to compute the relative RMS. Results are shown in Figure 4. We note that 1st and 2nd order interpolation have two orders of magnitude less error than 0th order. We also note that with 256 bins per segment (and 12 segments) 1st and 2nd order are virtually identical.

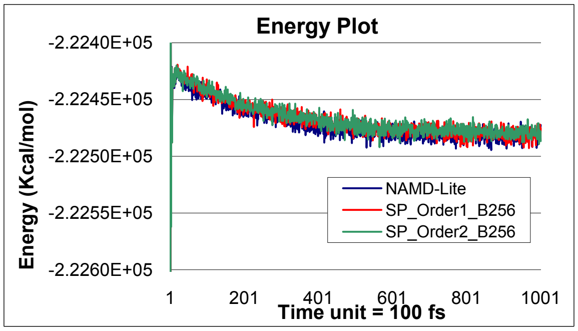

Figure 5. Graphs of energy for selected designs run for 100,000 timesteps.

Preliminary results with respect to energy fluctuation and drift are shown in Figure 4. A number of design alternatives were examined, including the original code and all combinations of the following parameters: bin density (64 and 256 per segment), interpolation order (0th, 1st, and 2nd), and single and double precision floating point. We note that all of the 0th order simulations are unacceptable, but that the others are all indistinguishable (in both energy fluctuation and drift) from the serial reference code running direct computation in double precision floating point.

Three implementations were chosen for longer simulation (shown in Figure 5). Using Equation 8 to compute ΔE we find that the value for the reference code is 1.1E-4 and for both of the FPGA-accelerated codes is 1.3E-4; all are much smaller than 0.003. After 70,000 timesteps, the values for ΔE are all less than 1.5E-07.

C. Performance

Table 1. Resource utilization and performance of various pipeline configurations on Stratix III EP3SE260 (bins/segment = 256)

	LUP0	LUP1	LUP2	DC
Multipliers	67%	63%	66%	68%
Logic	87%	88%	85%	94%
BRAMs (M9K)	89%	86%	89%	62%
BRAMs (M144K)	87.5%	75%	62.5%	50%
Number of Pipeline	7	6	5	4
Timing (ms) @ 200 MHz	NA	45	56	67

Performance is directly related to resources consumed (see Table 1). All of these designs have been implemented and run on the Gidel board. Time is per iteration. We note that the number of pipelines increases from 4 to 5 to 6 to 7 with interpolation order 2, 1, and 0, respectively. According to the quality results, the six pipeline design with 1st order interpolation is likely to be preferred. This design increases performance by almost 50% over direct computation.

The resource utilization results indicate that the limiting factor is the logic. This is used mostly for registers. An interesting observation is that number of bins is not a major concern and could be doubled if needed to achieve better simulation quality.

Table 2. Resource utilization and performance of various pipeline configurations on the Stratix IV EP4SE530 (bins/segment = 256)

	LUP0	LUP1	LUP2	DC
Multipliers	76%	87%	98%	100%
Logic	69%	75%	78%	86%
BRAM (M9K)	98%	98%	95%	67%
BRAM (M144K)	100%	100%	94%	75%
Number of Pipeline	12	11	10	8

We have also synthesized the designs with respect to the Stratix IV EP4SE530 (post place-and-route) with the results shown in Table 2. After optimization we anticipate achieving an operating frequency similar to that for the Stratix III. We anticipate a nearly proportional increase in performance resulting in a time per iteration of about 25ms.

V. Discussion

We have described a range of force pipelines appropriate for FPGA implementation of MD. These have undergone evaluation for quality, been integrated into NAMD-lite, and implemented and tested on a real system.

The most surprising result is how robust the low order interpolation is, with virtually no change in simulation quality by using 1st, rather than 2nd, order interpolation. This is largely a consequence of the number of interpolation intervals that we were able to use: over 3,000 for each of the three functions. Our ability to do this is a direct result of the BRAM availability on high-end FPGAs. By comparison, the 72KB storage needed for these tables would swamp the L1 data cache of a modern processor core and would likely reduce performance substantially.

Overall, these results are highly promising for MD on FPGAs, even in the face of competition from GPUs. The Stratix III is now over two generations old, and even the Stratix IV is dated. Moving to the Stratix V would again nearly double performance.

References

[1] M. Chiu and M. Herbordt, “Molecular dynamics simulations on high performance reconfigurable computing systems,” *ACM Transactions on Reconfigurable Technology and Systems*, vol. 3, no. 4, 2010.

[2] ——, “Towards production FPGA-accelerated molecular dynamics: Progress and challenges,” in *Proc. High Performance Reconfigurable Technology and Applications*, 2010.

[3] D. Hardy, “NAMD-lite,” http//www.ks.uiuc.edu/ Development/MDTools/namdlite/, University of Illinois at Urbana-Champaign, 2007.

[4] D. Rapaport, *The Art of Molecular Dynamics Simulation*. Cambridge University Press, 2004.

[5] J. Stone, J. Phillips, P. Freddolino, D. Hardy, L. Trabuco, and K. Schulten, “Accelerating molecular modeling applications with graphics processors,” *J. Computational Chemistry*, vol. 28, pp. 2618–2640, 2007.

[6] L. Nilsson, “Efficient table lookup without inverse square roots for calculation of pair wise atomic interactions in classical simulations,” *J. Computational Chemistry*, vol. 30, pp. 1490–1498, 2009.

[7] R. Larson, J. Salmon, M. Deneroff, C. Young, J. Grossman, Y. Shan, J. Klepseis, and D. Shaw, “High-througput pairwise point interactions in Anton, a specialized machine for molecular dynamics simulation,” in *Proc. High Performance Computer Architecture*, 2008, pp. 331–342.

[8] T. Andrea, W. Swope, and H. Anderson, “The performance implications of thread management alternatives for shared-memory multiprocessors,” *J. of Chemical Physics*, vol. 79, no. 9, pp. 4576–4584, 1983.

[9] Shaw, D.E., et al., “Anton, a special-purpose machine for molecular dynamics simulation,” in *Proc. International Symp. on Computer Architecture*, 2007, pp. 1–12.

[10] R. Engle, R. Skeel, and M. Drees, “Monitoring Energy Drift with Shadow Hamiltonians,” *J. Computational Physics*, vol. 206, pp. 432–452, 2005.

[11] Y. Shan, J. Klepeis, M. Eastwood, R. Dror, and D. Shaw, “Gaussian split Ewald: A fast Ewald mesh method for molecular simulation,” *J. Chemical Physics*, vol. 122, no. 4, 2005.