Preliminary Report: FPGA Acceleration of Molecular Dynamics Computations*

Yongfeng Gu Tom Van Court Douglas DiSabello Martin C. Herbordt

Department of Electrical and Computer Engineering
Boston University; Boston, MA 02215
EMail: {maplegu|tvancour|douglasd|herbordt}@bu.edu

Abstract: Molecular Dynamics (MD) is of central importance to computational chemistry and its myriad applications. Here we show that, at even a preliminary stage of development, MD can be implemented efficiently on a COTS FPGA board, and that a 57x speed-up over a PC implementation can be obtained. We sketch our FPGA implementation and describe how performance tuning and precision management (currently in progress) could double this factor.

1 Introduction

Molecular Dynamics (MD) is a fundamental part of computational chemistry. In the last few years MD has become, if anything, even more critical as it has been applied to modeling molecular interactions in drug design (see e.g. [4]), and to predicting molecule structure with applications to homeland security.

MD is an iterative technique that runs in phases: the forces on each atom/molecule are computed, then applied using equations of motion. Although modern force computations have become highly sophisticated, with terms describing ten or more phenomena, the complexity generally resides in computing the van der Waals (Lennard-Jones or LJ) and Coulombic terms. These are $O(N^2)$ in the number of particles N, while the motion updates are $O(N)$, and the other forces–which only look at bonds–are also $O(N)$. In this extended abstract, we describe preliminary work in accelerating MD using FPGAs. We restrict our attention to the motion updates and the $O(N^2)$ force terms.

MD is an obvious candidate for acceleration with special purpose hardware. Examples are the well-known GRAPE-derived boards (see e.g. [5]) and the FPGA implementation presented at the last FCCM [2]. In the latter study, 2001-era FPGA technology was used to obtain performance similar to that of a PC; this was extrapolated to a 20x speed-up by assuming modest hardware updates.

Our work differs from previous approaches in that we combine the following: on the hardware side, that we use a COTS board; on the implementation side, that we model the Coulombic as well as the LJ term, and that we support the simultaneous modeling of multiple types of molecules. There are also numerous implementation details which we mention only briefly.

Our primary result is that FPGA-based MD acceleration is likely to be many times more effective than previously indicated. We have already obtained a 57x speed-up while using a more detailed computational model; straightforward tuning and precision management could increase that speed-up substantially.

The primary significance is that a speed-up of two orders of magnitude is the often cited minimum threshold for initial acceptance of non-standard computing technology. Also significant is that this can be achieved using a flexible COTS board; that it is FPGA-based means that the hardware can ride the technology curve for commodity chips and that the configured algorithms can be updated as new discoveries are made.

2 Methods

The LJ force for particle i can be expressed as:

$$\mathbf{F}_i^{LJ} = \sum_{j \neq i} \frac{\epsilon_{ab}}{\sigma_{ab}^2} \left\{ 12 \left(\frac{\sigma_{ab}}{|r_{ji}|} \right)^{14} - 6 \left(\frac{\sigma_{ab}}{|r_{ji}|} \right)^{8} \right\} \mathbf{r}_{ji}$$

where the ϵ_{ab} and σ_{ab} are parameters related to the types of particles, i.e. particle i is type a and particle j is type b. The coulombic force can be expressed as:

$$\mathbf{F}_i^{C} = q_i \sum_{j \neq i} \left(\frac{q_j}{|\mathbf{r}_{ji}|^3} \right) \mathbf{r}_{ji}$$

We implement both coulombic and LJ forces; we also implement multiple atom types. We use the Verlet

*This work was supported in part by the NIH through award #RR020209-01 and facilitated by donations from Xilinx Corporation. Web: http://www.bu.edu/caadlab.

Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'05)
0-7695-2445-1/05 $20.00 © 2005 IEEE

method for motion updates. For serial reference code (as in [2]) we began with that described in [3]. We also created our own serial reference code that tracks the hardware implementation, e.g. in varying precision. The standard reference code ran at about 9.5s per MD time-step on a PC with a 2.4GHz Xeon CPU. This is similar to the 10.8s for a 2.4GHz P4 described in [2].

3 Implementation and Results

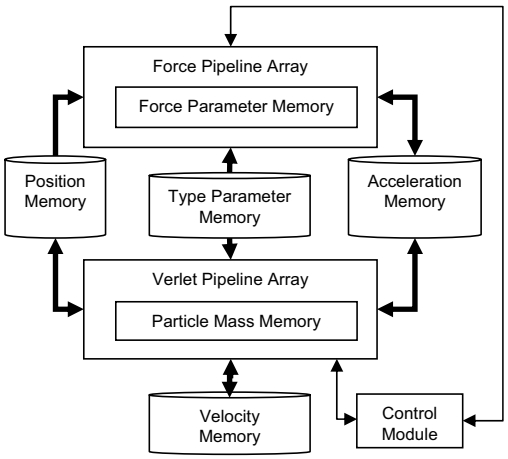

Figure 1: Shown is block diagram of system.

The high-level design is shown in Figure 1. As is common in MD hardware implementations, fixed point is used. Scaled units bound the datapath size while maintaining near maximum precision throughout the computation. The precisions for the various data are as in [1]. Also, the force computations use table look-ups with interpolation.

The design was implemented on the WildstarII-Pro board from Annapolis Micro Systems, Inc. with two Xilinx Virtex-II-Pro XC2VP70 -5 FPGAs. However, only one of the FPGAs is currently used. The critical resource for both speed and area is the hard multipliers. In order to get the multipliers off the timing critical path, we created customized 40-bit multipliers. Instead of using 3 hard multipliers with 25ns latency, we use 9 hard multipliers with 9ns pipelined latency. The implementation has four pipelines in both force and verlet computations. As mentioned, the critical resource preventing more pipelines from being implemented is the shortage of hard multipliers: 87% of these are used.

The implementation was validated against both reference codes. Against our own, the match was exact. Against the double precision floating point code (see [3]) there was a very close match as to be expected from the analysis in [1].

The clock delay is currently 9.98ns which, for a model with 8192 particles, yields a latency of .167s per time-step for a speed-up of 57x.

4 Optimizations and Extensions

We have only begun optimizing the design. It appears that a relatively straightforward change in the multipliers, i.e. using a mix of hard and soft components, will reduce the hard multiplier count by 30% without affecting the operating frequency. This would likely enable the implementation of two more pairs of pipelines. Another optimization is with the adders that currently comprise the critical timing path; a simple change using available resources should get them off and reduce the clock latency to 9ns. Further optimizations will almost certainly emerge from guiding placement, etc.

We are currently examining the effect of reducing the precision. This is a complex issue as it is not the variance in motion that matters, but rather the fidelity of observable macro phenomena. Still, a reduction of even a few bits will have a drastic effect on our design and our initial simulations are favorable.

The most interesting investigations involve comparing trade-offs in algorithm (particularly as a result of the choice of boundary conditions), FPGA resource utilization (and therefore performance), and simulation accuracy. The implementation described here works well for fixed and stochastic cut-offs. However, many implementations use periodic boundary conditions. These are susceptible to other kinds of error than the cut-offs, but perhaps more significantly, allow use of Ewald Sums and other $O(N \log N)$ methods.

Finally, this work is part of a larger project involving the acceleration of applications in computational biochemistry (e.g. [6]) and will be integrated into that.

References

[1] Amisaki, T., Fujiwara, T., Kusumi, A., Miyagawa, H., and Kitamura, K. Error evaluation in the design of a special-purpose processor that calculates nonbonded forces in molecular dynamics simulations. *J. Comp. Chem. 16*, 9 (1995), 1120–1130.

[2] Azizi, N., Kuon, I., Egier, A., Darabiha, A., and Chow, P. Reconfigurable molecular dynamics simulator. In *Proc. FCCM* (2004), pp. 197–206.

[3] Bargiel, M., Dzwinel, W., Kitowski, J., and Moscinski, J. C-language molecular dynamics program for the simulation of Lennard-Jones particles. *Computer Physics Communication 64* (1991), 193–205.

[4] Taufer, M., Crowley, M., Price, D., Chien, A., and Brooks III, C. Study of a highly accurate and fast protein-ligand docking algorithm based on molecular dynamics. In *Proc. High Perf. Comp. Bio.* (2004).

[5] Toyoda, S., Mihagawa, H., Kitamura, K., Amisake, T., Hashimoto, E., Ikeda, H., Kusumi, A., and Miyakawa, N. Development of MD engine: High-speed accelerator with parallel processor design for molecular dynamics simulations. *J. Comp. Chem. 20*, 2 (1999), 185–199.

[6] Van Court, T., Gu, Y., and Herbordt, M. C. FPGA acceleration of rigid molecule interactions. In *Proc. of Field Programmable Logic and Applications* (2004).

Proceedings of the 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'05)
0-7695-2445-1/05 $20.00 © 2005 **IEEE**