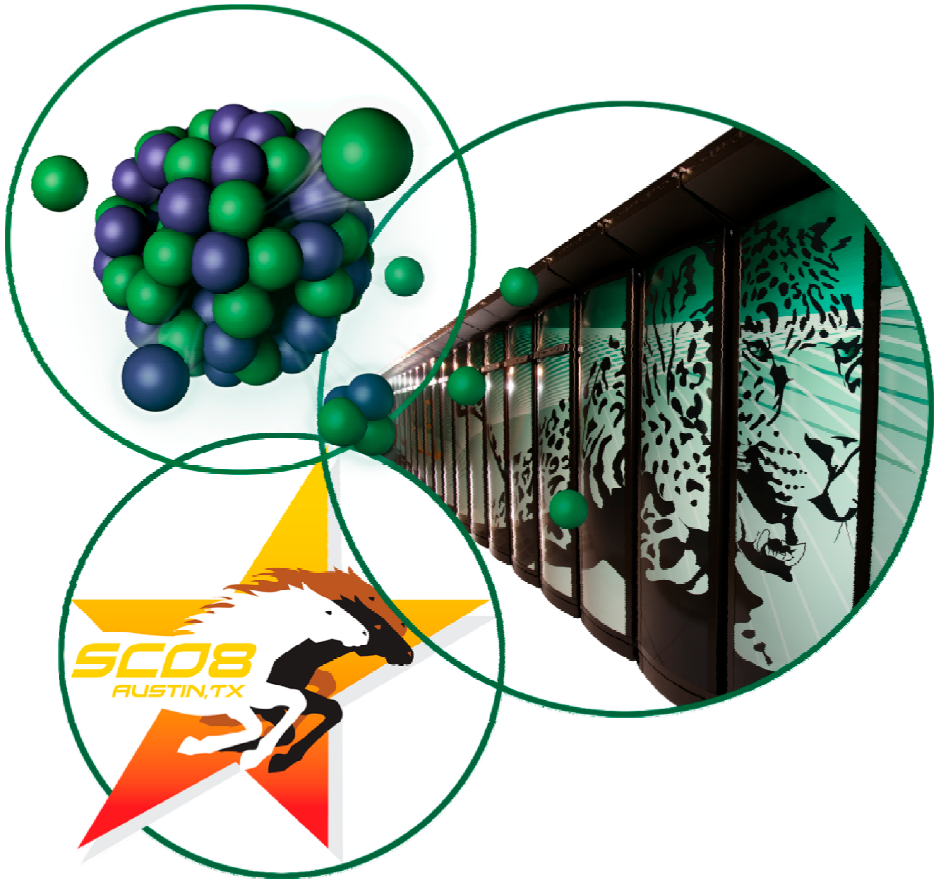
Accelerating Biomolecular Simulations on Reconfigurable Computing Hardware



Presented by

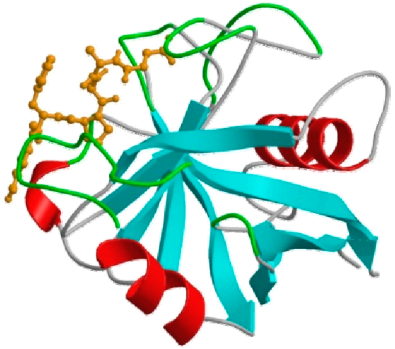
Pratul K. Agarwal
Sadaf R. Alam
Oak Ridge National Laboratory

Melissa Smith
Clemson University

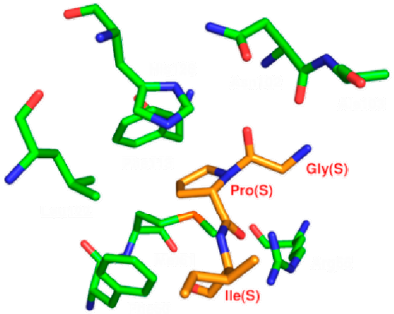
Managed by UT-Battelle
for the Department of Energy

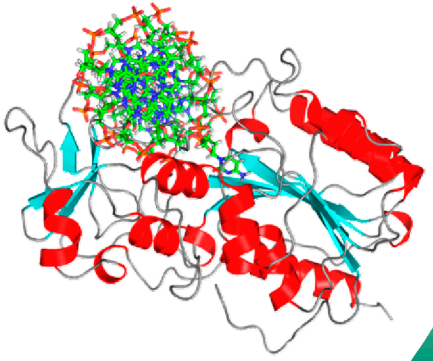
Biomolecular modeling

Biomolecules including proteins, enzymes, and DNA are highly efficient molecular machines working in all cells



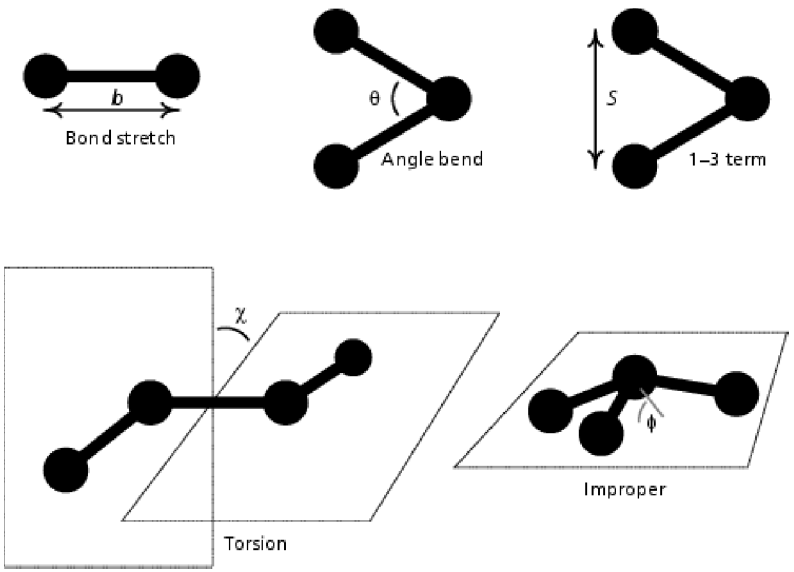
- **Atomistic-level biomolecular modeling and simulations**
  - **Provides detailed biophysical characterization of molecular machines, protein-structure prediction, and docking**
  - **Power of computational method: novel insights that are beyond the reach of experimental techniques**
  - **Enables multiscale approach: investigate a wide range of timescales and lengths**





Molecular dynamics (MD) simulations

- **Atomistic modeling**
- ***N* atoms in a system**
- **Not homogenous**
- **Computationally expensive**
- **Random memory access**
- **Dynamic array indices**
- **O(N^2) nonbond interactions**



$$E(potential) = \sum_{bonds} f(dist) + \sum_{angles} f(angle) + \sum_{dihedral} f(d-angles)$$

$$+ \sum_{j=1}^{N} \sum_{i=1}^{N} \left(-\frac{A_{ij}}{r_{ij}^{12}} + \frac{B_{ij}}{r_{ij}^{6}} \right) + \sum_{j=1}^{N} \sum_{i=1}^{N} \left(\frac{q_i q_j}{r_{ij}} \right)$$

Nonbonded calculations (Particle Mesh Ewald method)

Supercomputers offer some relief

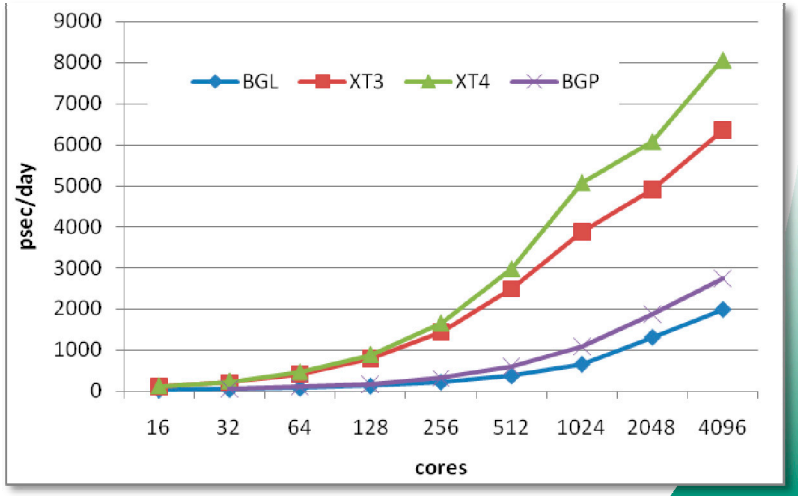
- **Desktop computers can only simulate a fraction of scales**

Multiscale modeling—structure, dynamics, and function
Desired/current capability ratio: 10^4–10^6

- **MD software does not scale well on the current MPP machines, particularly for small proteins**
- **Legacy problem: the underlying code was designed decades ago**

Jaguar: Cray XT4 263 TeraFLOPs

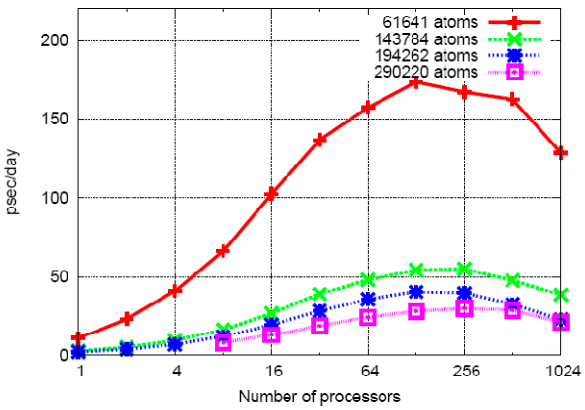
LAMMPS (RAQ with 290220 atoms)



Oak Ridge National Laboratory

Computational bottleneck: PME method

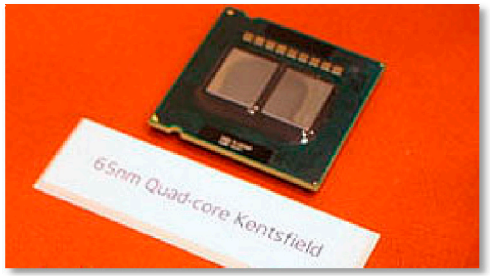
- **Particle Mesh Ewald (PME) method is used for electrostatic forces**
- **Underlying physics limits the performance of MPP architecture**
  - **Fixed communication volume per processor depends on problem size (number of atoms in the system)**
  - **Computation-to-communication ratios do not change with problem size**
  - **Direct PME scales, but reciprocal calculations with FFT operations do not**
  - **FFT communication volume increases as log2(MPI tasks)**
- **The cost of communications and computations is not well balanced**
- **Communication latency and bandwidth putting upper limits on science**



Weak-scaling experiments on the Blue Gene/L processor; similar behavior is observed on the XT3 system

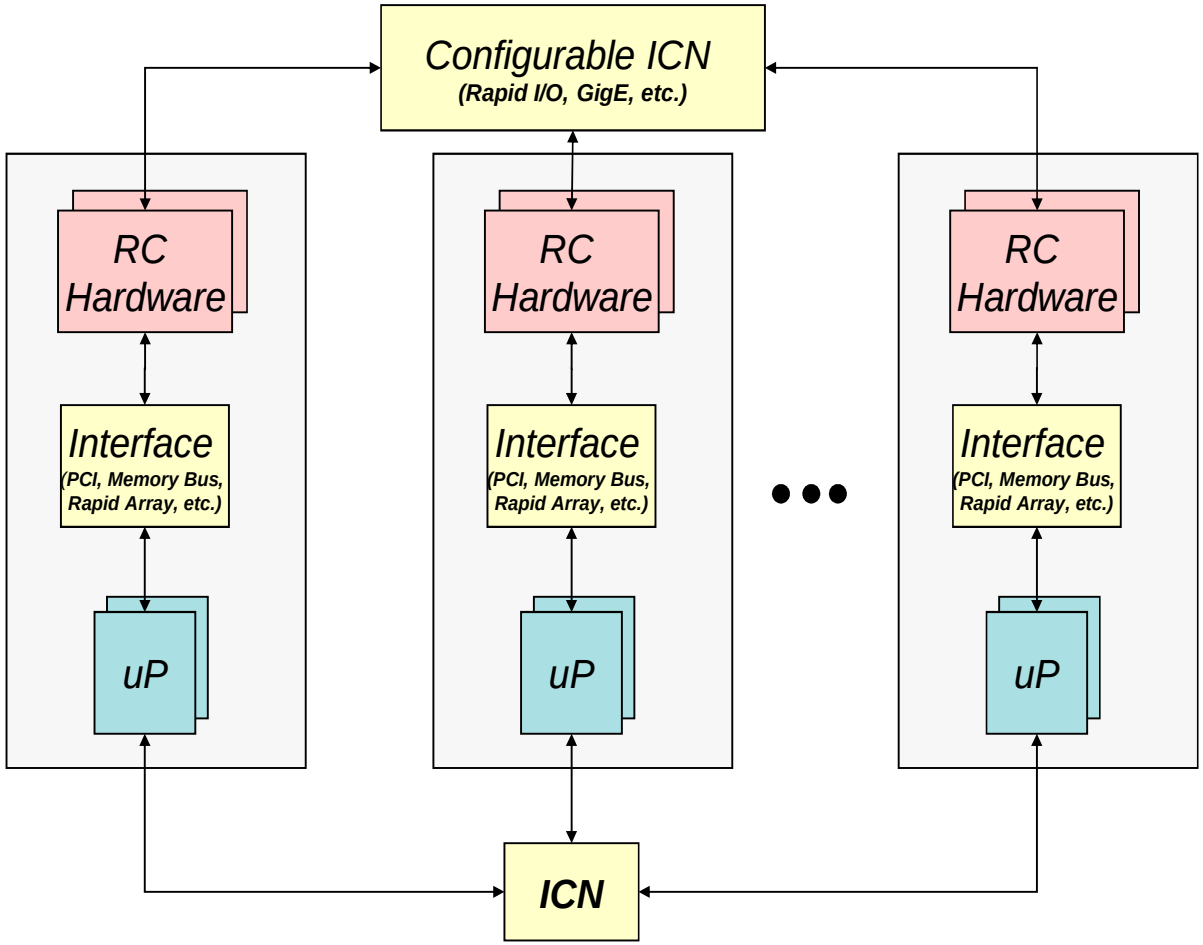
Promise of reconfigurable computing hardware: multicore, FPGAs, GPGPUs, and beyond

- Optimization of the modeling software is required
- Legacy code designed >20 years ago
- Collaborating with Future Technologies Group
- Performance characterization and optimization on multicore, FPGAs, GPGPUs, and other specialized hardware



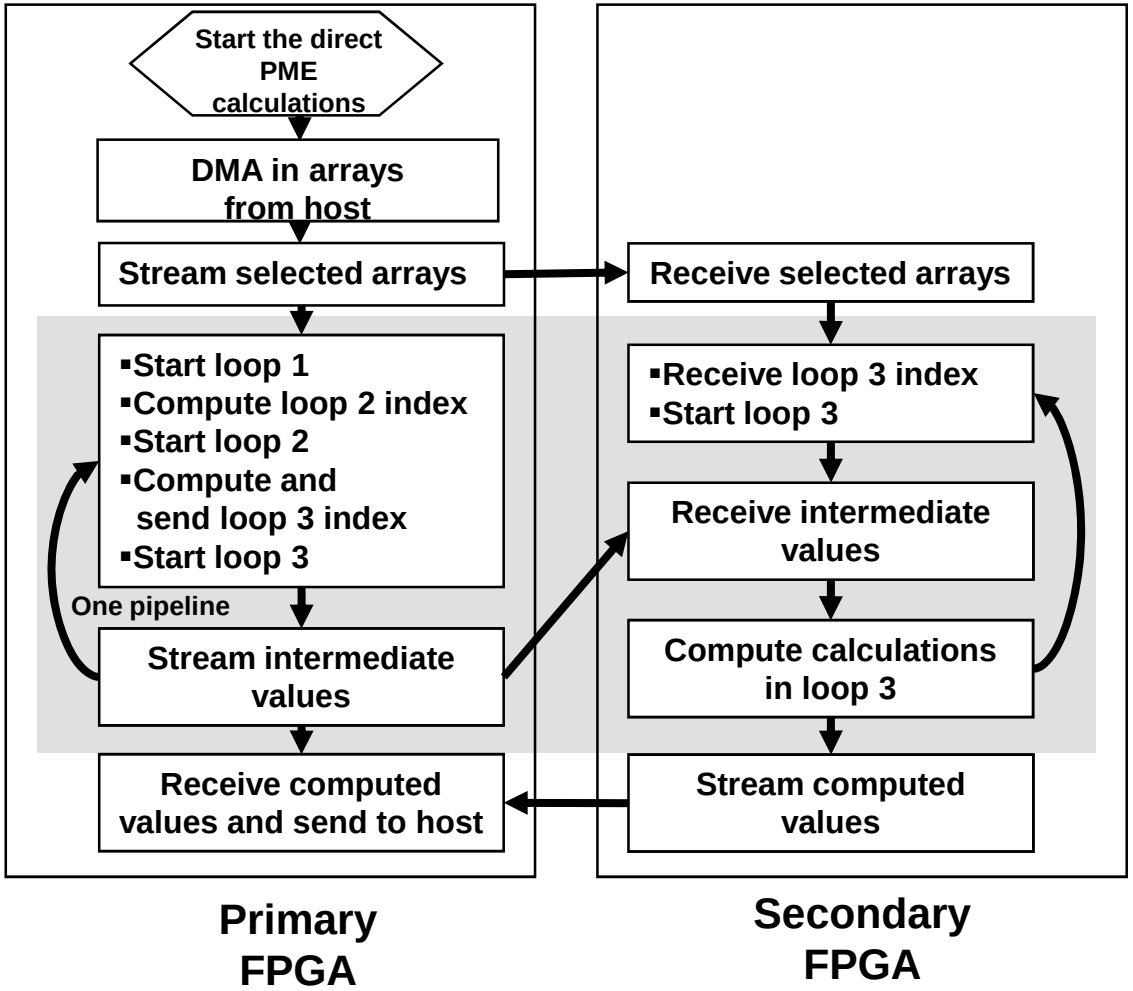
MD on FPGA: 15x speed-up

Field–programmable gate array (FPGA) accelerators



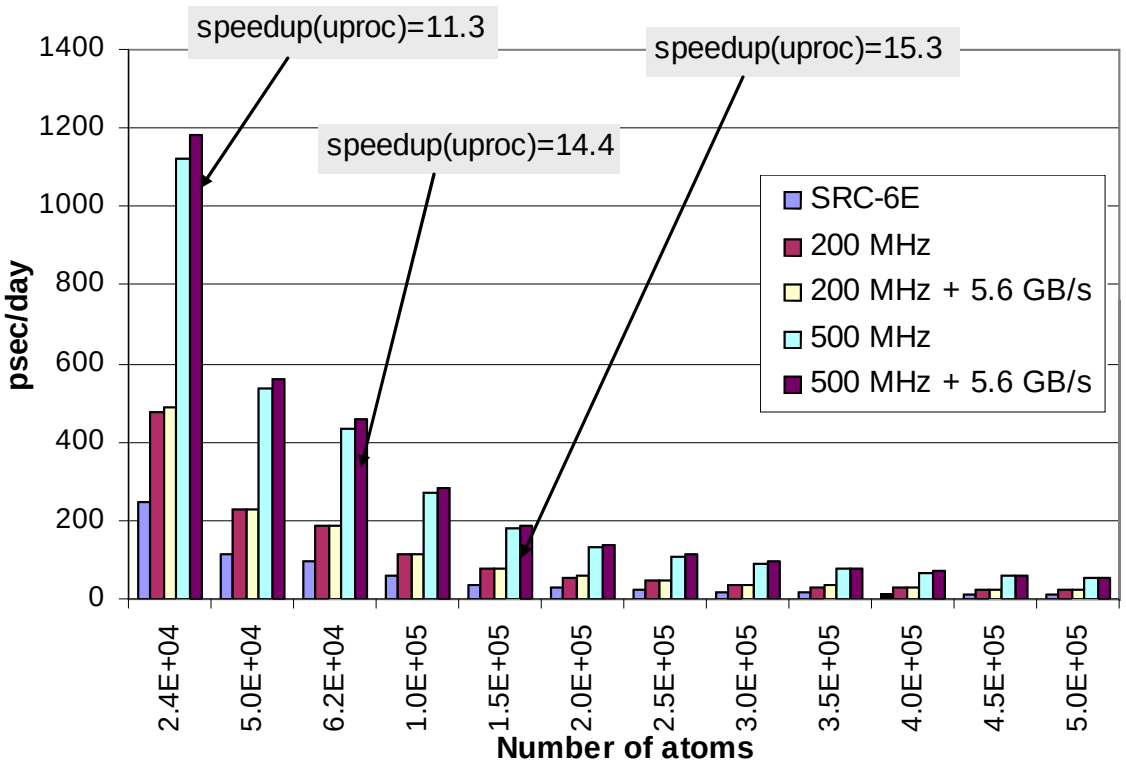
MD on FPGA: 15x speed-up (cont.)

Mapping onto FPGA devices

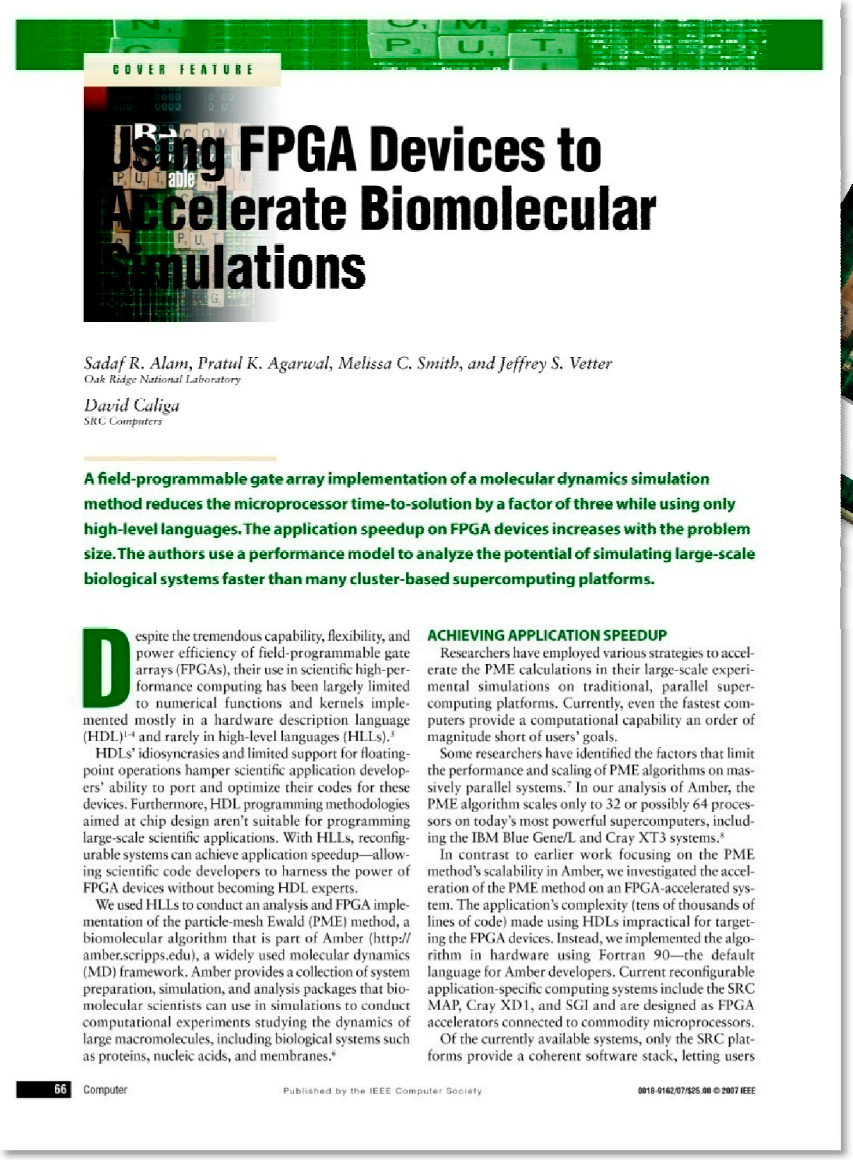


MD on FPGA: 15x speed-up (cont.)

Application acceleration: measurements and projections



Cover feature: IEEE's *Computer*



COVER FEATURE

Using FPGA Devices to Accelerate Biomolecular Simulations

Sadaf R. Alam, Pratul K. Agarwal, Melissa C. Smith, and Jeffrey S. Vetter
Oak Ridge National Laboratory

David Caliga
SRC Computers

A field-programmable gate array implementation of a molecular dynamics simulation method reduces the microprocessor time-to-solution by a factor of three while using only high-level languages. The application speedup on FPGA devices increases with the problem size. The authors use a performance model to analyze the potential of simulating large-scale biological systems faster than many cluster-based supercomputing platforms.

Despite the tremendous capability, flexibility, and power efficiency of field-programmable gate arrays (FPGAs), their use in scientific high-performance computing has been largely limited to numerical functions and kernels implemented mostly in a hardware description language (HDL)[1-4] and rarely in high-level languages (HLLs).[5]

HDLs' idiosyncrasies and limited support for floating-point operations hamper scientific application developers' ability to port and optimize their codes for these devices. Furthermore, HDL programming methodologies aimed at chip design aren't suitable for programming large-scale scientific applications. With HLLs, reconfigurable systems can achieve application speedup—allowing scientific code developers to harness the power of FPGA devices without becoming HDL experts.

We used HLLs to conduct an analysis and FPGA implementation of the particle-mesh Ewald (PME) method, a biomolecular algorithm that is part of Amber (http://amber.scripps.edu), a widely used molecular dynamics (MD) framework. Amber provides a collection of system preparation, simulation, and analysis packages that biomolecular scientists can use in simulations to conduct computational experiments studying the dynamics of large macromolecules, including biological systems such as proteins, nucleic acids, and membranes.[6]

ACHIEVING APPLICATION SPEEDUP

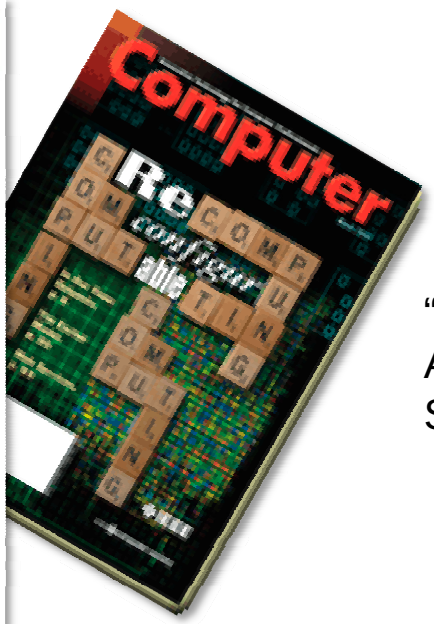
Researchers have employed various strategies to accelerate the PME calculations in their large-scale experimental simulations on traditional, parallel supercomputing platforms. Currently, even the fastest computers provide a computational capability an order of magnitude short of users' goals.

Some researchers have identified the factors that limit the performance and scaling of PME algorithms on massively parallel systems.[7] In our analysis of Amber, the PME algorithm scales only to 32 or possibly 64 processors on today's most powerful supercomputers, including the IBM Blue Gene/L and Cray XT3 systems.[8]

In contrast to earlier work focusing on the PME method's scalability in Amber, we investigated the acceleration of the PME method on an FPGA-accelerated system. The application's complexity (tens of thousands of lines of code) made using HDLs impractical for targeting the FPGA devices. Instead, we implemented the algorithm in hardware using Fortran 90—the default language for Amber developers. Current reconfigurable application-specific computing systems include the SRC MAP, Cray XD1, and SGI and are designed as FPGA accelerators connected to commodity microprocessors.

Of the currently available systems, only the SRC platforms provide a coherent software stack, letting users

Published by the IEEE Computer Society
0018-9162/07/$25.00 © 2007 IEEE



"Using FPGA Devices to Accelerate Biomolecular Simulations," March 2007

- **Speeding up of biomolecular simulations**
  - **Collaborating with computer scientists**
  - **Having broad impact**
  - **Benefits to other scientists**

OAK RIDGE National Laboratory

LAMMPS on XtremeData

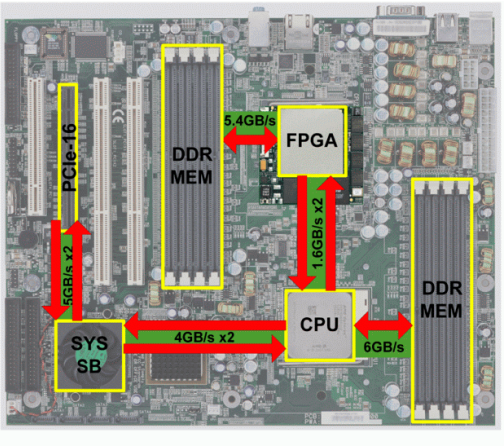
- **Examined for "hot spots" using Xprofiler on a 1.3 GHz IBM Power 4 (Rhodopsin benchmark)**
- **68% of execution time spent in *pair_lj_charmm_coul_long:compute***
  - **Computes the electrostatic and van der Waals forces between "pairs" of atoms**
  - **The function has 96 double–precision multiplications, 7 divisions, 81 additions/subtractions**

$$E(potential) = \sum_{bonds} f(bond) + \sum_{angles} f(angle) + \sum_{torsions} f(torsion) + \sum_{i=1}^{N}\sum_{j<i}^{N}\left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}\right) + \sum_{i=1}^{N}\sum_{j<i}^{N}\left(\frac{q_i q_j}{\varepsilon r_{ij}}\right)$$

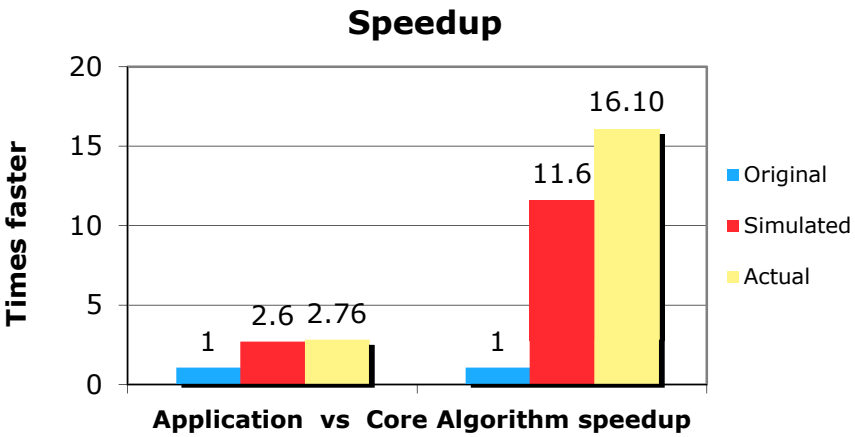
- **Estimated hardware runtime using StageMaster Explorer:**
  - **Ported LAMMPS function requires 364 clock cycles to compute forces on one atom (FPGA clock rate 100 Mhz limited by FP libraries)**
  - **114 ms to compute 1 timestep 32 K atoms → 11.5× speedup for core algorithm → 2.7× speedup for entire application**

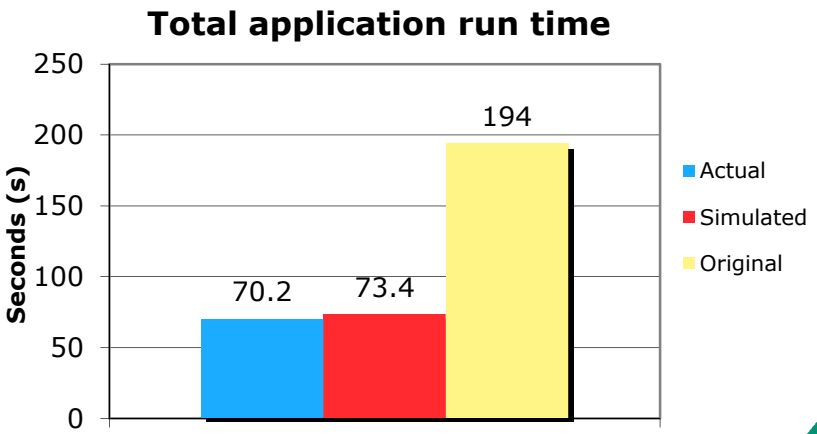
Results: XtremeData XD1000

- **XtremeData XD1000 development system**
  - Single AMD Opteron
  - Altera Stratix II FPGA (Opteron socket)
  - HyperTransport bus provides 1.6 GB/s between FPGA and system hardware



- **ImpulseC CoDeveloper Suite**
  - Provides C-to-VHDL or Verilog capability
  - Performance estimation of hardware implementation
  - ImpulseC CoDeveloper Suite
- **Single- and double-precision floating-point support**
- **16× increase for algorithm or 2.77× application speedup***





* Based on current implementation

NVIDIA's Tesla: promising avenue

- TeraFLOP computing power at desktop level
  - Powerful hardware at affordable price
  - User community has easy access
  - High language programming support
- CUDA: NVIDIA's software
  - C for GPGPUs
  - Support for FFT and BLAS libraries
- Porting popular MD code
  - LAMMPS and PMEMD (AMBER)
  - Ported code would run on other hardware supporting CUDA
- Hybrid computing systems
  - Future platforms expected to have some form of reconfigurable computing devices

Summary and future directions

- Promises and challenges of emerging acceleration devices
  - GPGPU 50x+ potential speedup with a restricted programming model
  - FPGA 10X+ potential speedup with programming challenges
  - Ease of programming vs. portability
  - Speedup vs. cost
- Acceleration of a production-level application
  - Benefits end user
  - Must be done on a widely available technology and programming paradigm (e.g., GPGPU)

Contact

Pratul K. Agarwal

(865)-574-7184
agarwalpk@ornl.gov
www.vigyaancd.org/pratul

Acknowledgements

Jeff Vetter
ORNL

David Calliga
SRC Computers, Inc.

Phillip Martin
Clemson University