Memory-Driven Algorithm Mapping of Molecular Dynamics for High-Performance Reconfigurable Computers

CHREC
NSF Center for High-Performance Reconfigurable Computing

MRSC 2008

UF UNIVERSITY of FLORIDA
Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
BYU
BRIGHAM YOUNG UNIVERSITY
THE GEORGE WASHINGTON UNIVERSITY

Brian Holland & Alan D. George

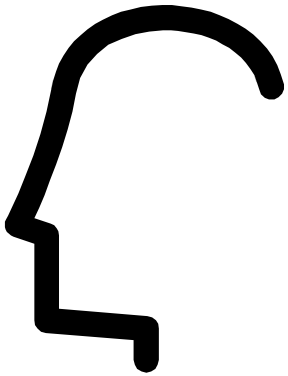
NSF CHREC Center

ECE Department
University of Florida

1-3 April 2008

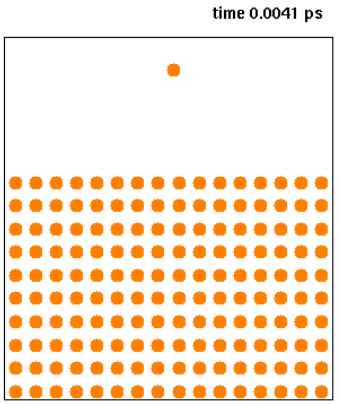
Outline

- Molecular Dynamics Introduction
- Strategic Design (Formulation)
- Performance Prediction Overview
  - RC Amenability Test (RAT)
- Impulse C & XtremeData XD1000
  - System configuration and linguistic factors
  - Performance prediction with MD
- Carte & SRC MAP-B
  - System configuration and linguistic factors
  - Performance prediction with MD
- Results, Speedup, & Prediction Accuracy
- Taking Knowledge Forward
  - Design Patterns
- Conclusions

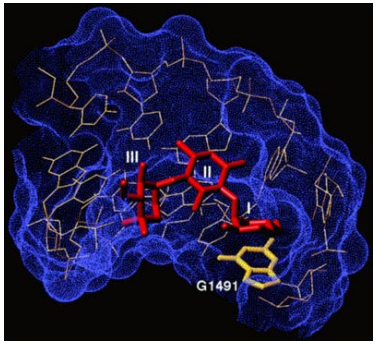


Molecular Dynamics

- Numerical simulation of physical interactions of atoms and molecules over given time interval



- Previous Work
  - Widely studied in HPC & HPRC
    - [Alam 06], [Azizi 04], [Cordov 05], [Gu 06], [Kindratenko06], etc.
  - Software baseline code for molecular dynamics provided to CHREC by Oak Ridge National Lab
    - Explored as HPC case study to uncover lessoned learned



- Primary Stages of MD Computation
  1. Reading molecular pair's position from memory
  2. Computing distance between two molecules
  3. Computing atomic interaction (if distance < threshold)
  4. Adjusting net acceleration for adjacent molecules

Images from Wikipedia Commons and folding.stanford.edu. For academic purposes only.

Molecular Dynamics

```
void ComputeAccel() {
  double dr[3],f,fcVal,rrCut,rr,ri2,ri6,r1;
  int j1,j2,n,k;

  rrCut = RCUT*RCUT;
  for(n=0;n<nAtom;n++) for(k=0;k<3;k++) ra[n][k] = 0.0;
  potEnergy = 0.0;

  for (j1=0; j1<nAtom-1; j1++) {
    for (j2=j1+1; j2<nAtom; j2++) {
      for (rr=0.0, k=0; k<3; k++) {
1       dr[k] = r[j1][k] - r[j2][k];
        dr[k] = dr[k]-SignR(RegionH[k],dr[k]-RegionH[k])
2                     - SignR(RegionH[k],dr[k]+RegionH[k]);
        rr = rr + dr[k]*dr[k];
      }
      if (rr < rrCut) {
        ri2 = 1.0/rr; ri6 = ri2*ri2*ri2; r1 = sqrt(rr);
3       fcVal = 48.0*ri2*ri6*(ri6-0.5) + Duc/r1;
        for (k=0; k<3; k++) {
          f = fcVal*dr[k];
4         ra[j1][k] = ra[j1][k] + f;
          ra[j2][k] = ra[j2][k] - f;
        }
        potEnergy+=4.0*ri6*(ri6-1.0)- Uc - Duc*(r1-RCUT);
      }
    }
  }
}
```

1. Reading Position
   - Potential contention for shared resource
2. Computing Distance
   - Simple hardware mapping if Step 1 contentions resolved
3. Computing Interaction
   - Conditional computations must be speculatively executed for pipelines
4. Adjusting Net Acceleration
   - Similar contention to Step 1 for shared resource

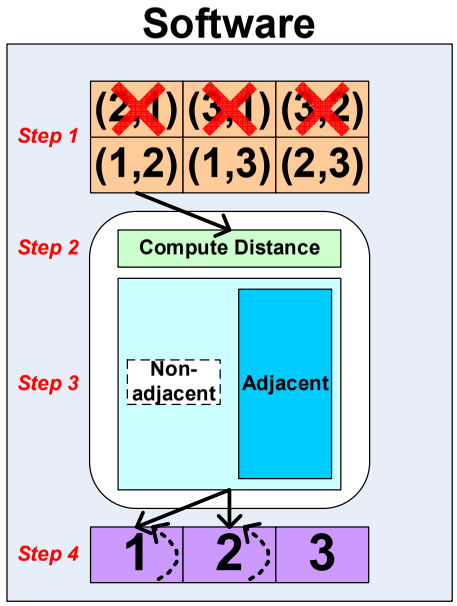
Molecular Dynamics

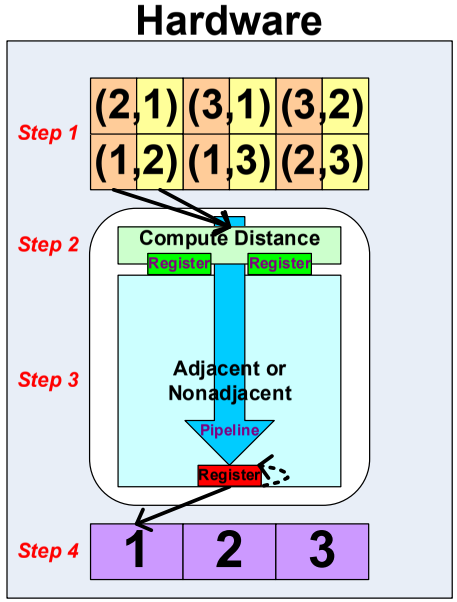
Step 1

```
for (rr=0.0, k=0; k<3; k++) {
  dr[k] = r[j1][k] - r[j2][k];
  …
}
```

Step 4

```
for (k=0; k<3; k++) {
  f = fcVal*dr[k];
  ra[j1][k] = ra[j1][k] + f;
  ra[j2][k] = ra[j2][k] - f;
}
```





- Memory-driven algorithm mapping
  - Memory contention resolution in Step 1
    - Replicate input data across multiple banks
      - Alternately, use multi-ported memory
  - Memory contention resolution in Step 4
    - Use multi-ported memory
      - Not always available, especially for external SRAM
    - Compute pairs twice only saving one value each time
      - Both updates are appends (read, add, write)
        - Potentially many stall cycles are necessary
      - One molecule per pair requires only one register append with a final memory write (after *N* iterations)

Strategic Design Methodology

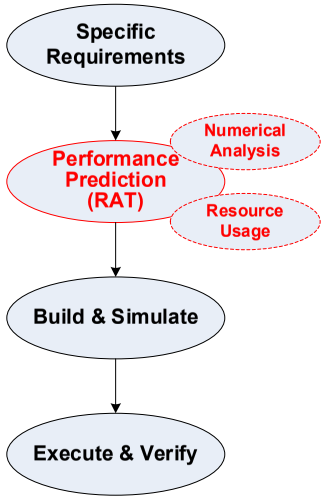
- What is next step towards an FPGA design?
  - Jump right in and start coding (ill-advised!)
  - Examine prior work and leverage success (better!!)
    - But what if you're in uncharted territory with algorithm?
  - Formulation/Strategic Design (best!!!)
    - What are specifications and requirements?
    - ***Can I quantitatively show my design will meet expectations?***
      - Develop algorithm then use RAT to gauge likely performance

- Molecular Dynamics
  - Requirements
    - Create highest performance design possible for 16,384 molecules
    - Minimize memory footprint to enable larger future designs
  - Quantitative Analysis
    - Will memory structure and bandwidth support the design?
    - Are enough computations operating in parallel to achieve speedup?

Performance Prediction

- RC Amenability Test (RAT)
  - Determines execution time of *specific* algorithm on *specific* hardware platform
  - Amenability is gauged based on performance and speedup requirements as determined by user
- RAT Analytic Model

FPGA Application Development Spectrum



Communication Time

$$t_{comm} = t_{read} + t_{write}$$

Computation Time

$$t_{comp} = \frac{N_{elements} \cdot N_{ops/element}}{f_{clock} \cdot throughput_{proc}}$$

Speedup

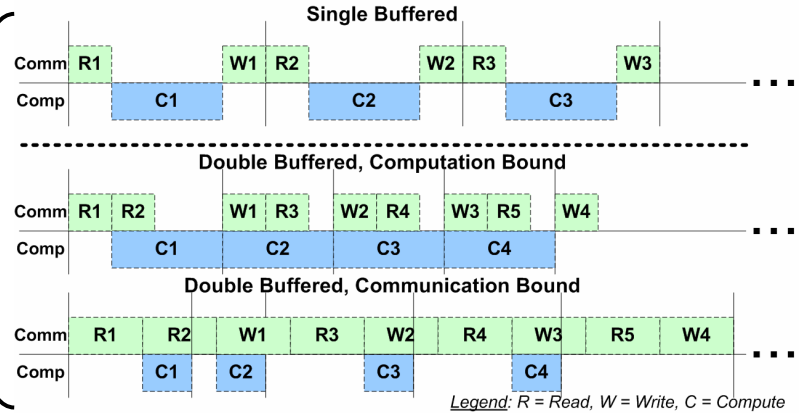
$$speedup = \frac{t_{soft}}{t_{RC}}$$

RC Execution Time

$$t_{rc_{SB}} = N_{iter} \cdot (t_{comm} + t_{comp})$$

$$t_{rc_{DB}} \approx N_{iter} \cdot Max(t_{comm}, t_{comp})$$

Comm. and Comp. Overlap for Single or Double Buffering



Performance Prediction

- Interconnect Parameters
  - Throughput, α_{input},α_{output}
    - Models CPU/FPGA interconnect
- Communication Parameters
  - $\#_{elements}$,input/output
    - Quantity of input and output data for algorithm
  - $N_{Bytes/element}$
    - Element is an algorithm's basic unit of data
    - Conversion factor for determining total data size
- Computation Parameters
  - $\#_{elements}$,comp
    - Computation is not always related to data transfer
  - $N_{ops/element}$
    - Amount of computational work needed to complete one element
  - $Throughput_{proc}$
    - Average number of operations per cycle
  - Clock Frequency
- Software Parameters
  - T_{soft}
    - Software execution time
  - $N_{iterations}$
    - Number of input, compute, output cycles

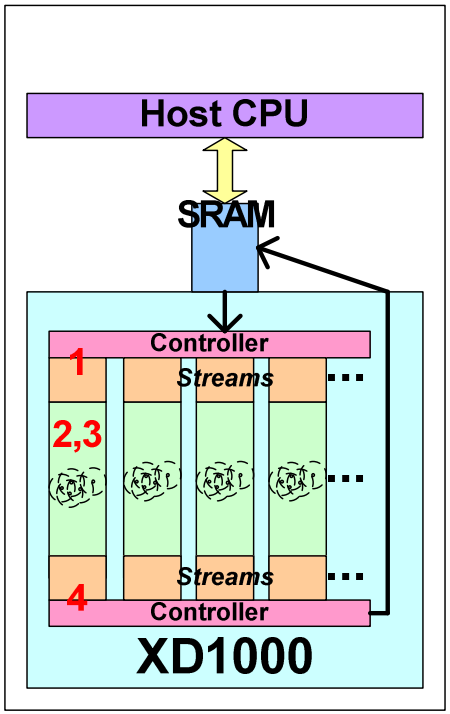
Constants/User-Defined Parameters		
Interconnect Parameters		
throughput(ideal)	(MB/s)	
α(input)	0<α<1	
α(output)	0<α<1	
Communication Parameters		
# of Input Elements	(elements)	
# of Output Elements	(elements)	
Bytes per element	(B)	
Computation Parameters		
# of Comp Elements	(elements)	
Ops per element	(ops/elem)	
throughput(proc)	(ops/cycle)	
f(clock)	(MHz)	
Software Parameters		
t(soft)	(sec)	
N	(iterations)	

Calculated Sub-metrics		
	Single Buffered	
t(soft)	(sec)	0.000E+00
t(comm)	(sec)	0.000E+00
t(comp)	(sec)	0.000E+00
t(RC)	(sec)	0.000E+00
speedup(kernel)	n/a	**0.00**

XD1000 System & Impulse C Language

- Xtreme Data XD1000 System
  - Single XD1000 FPGA module
    - One Altera Stratix II EP2S180
    - Connected to Opteron server via HyperTransport interconnect
    - Single 4MB SRAM bank
      - ~500MB/s throughput
- Linguistic Factors
  - Internal FPGA Network
    - Stream-oriented communication with serialized input and output
      - Links 14 MD computational kernels
  - Computation
    - Challenging to explore and evaluate optimal pipelining

impulse accelerated technologies



Images from xtremedatainc.com and impulsec.com. For academic purposes only.

MD Performance Prediction

Constants/User-Defined Parameters		
Interconnect Parameters		
throughput(ideal)	(MB/s)	500
α(input)	0<α<1	0.9
α(output)	0<α<1	0.9
Communication Parameters		
# of Input Elements	(elements)	16384
# of Output Elements	(elements)	16384
Bytes per element	(B)	12
Computation Parameters		
# of Comp Elements	(elements)	16384
Ops per element	(ops/elem)	163840
throughput(proc)	(ops/cycle)	40
f(clock)	(MHz)	100
Software Parameters		
t(soft)	(sec)	6.71
N	(iterations)	1

Calculated Sub-metrics		
Single Buffered		
t(soft)	(sec)	6.71E+00
t(comm)	(sec)	8.74E-04
t(comp)	(sec)	6.71E-01
t(RC)	(sec)	6.72E-01
speedup(kernel)	n/a	9.99

Impulse C & XD1000

- Interconnect Parameters
  - Throughput, α_{input},α_{output}
    - Models HyperTransport interconnect
- Communication Parameters
  - $\#_{elements}$,input/output
    - 16384 molecules (elements) are used in MD simulation
  - $N_{bytes/element}$
    - 4 (32 bits) × 3 (x,y,z dimensions)
- Computation Parameters
  - $\#_{elements}$,comp
    - 16384 (elements, iterations, etc.)
  - $N_{ops/element}$
    - 16384 other elements × 100 (ops each)
    - *Difficult to measure for this algorithm*
  - $Throughput_{proc}$
    - 40, number required to achieve 10× speedup
    - *Difficult to measure for this algorithm*
  - Clock Frequency
    - 100 MHz, default frequency of XD1000 system
- Software Parameters
  - T_{soft}
    - Execution time on 3.2GHz Xeon
  - $N_{iterations}$
    - Only 1 iteration necessary

SRC System & Carte Language

- SRC-6 System
  - 4 MAP-B units
    - Two Xilinx Virtex XC2V6000 each
      - Only one FPGA per unit used
    - Connected to host server via SNAP (memory) interconnect
    - Six 4MB 64-bit SRAM banks each
- Linguistic Issues
  - Mapping SRAM resources to computational needs
    - One MD kernel per MAP unit consuming all SRAM resources
      - Four computation kernels total
  - Computation
    - Mapping unrolled MD loops and removing pipeline stalls

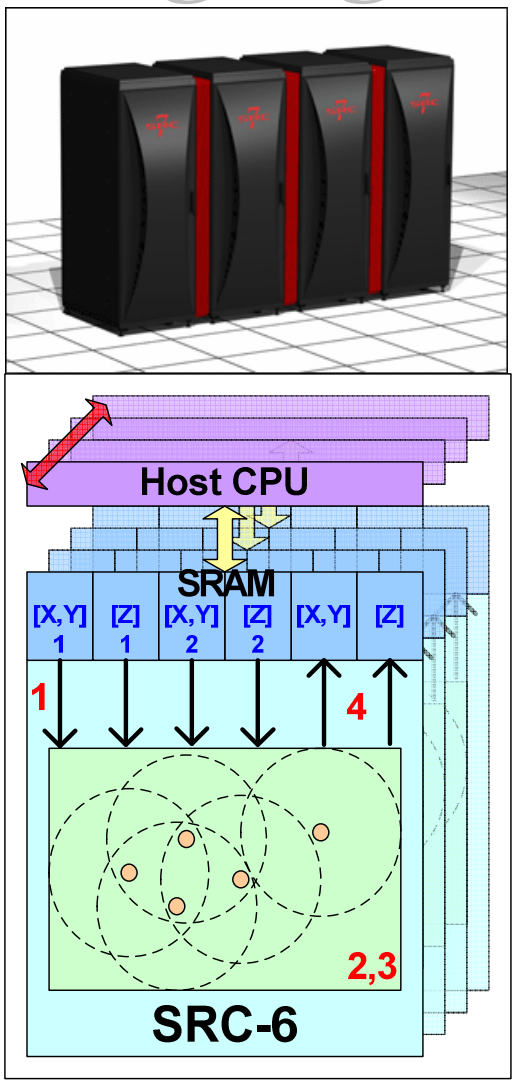


Image from www.srccomp.com. For academic purposes only.

MD Performance Prediction

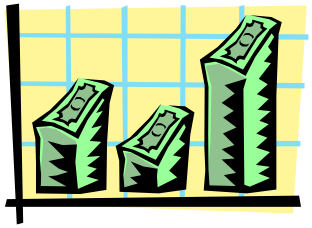
Constants/User-Defined Parameters		
Interconnect Parameters		
throughput(ideal)	(MB/s)	800
α(input)	0<α<1	0.95
α(output)	0<α<1	0.95
Communication Parameters		
# of Input Elements	(elements)	65536
# of Output Elements	(elements)	32768
Bytes per element	(B)	8
Computation Parameters		
# of Comp Elements	(elements)	16384
Ops per element	(ops/elem)	16383
throughput(proc)	(ops/cycle)	4
f(clock)	(MHz)	100
Software Parameters		
t(soft)	(sec)	6.71
N	(iterations)	1

Calculated Sub-metrics		
	Single Buffered	
t(soft)	(sec)	6.71E+00
t(comm)	(sec)	1.03E-03
t(comp)	(sec)	6.71E-01
t(RC)	(sec)	6.72E-01
speedup(kernel)	n/a	**9.98**

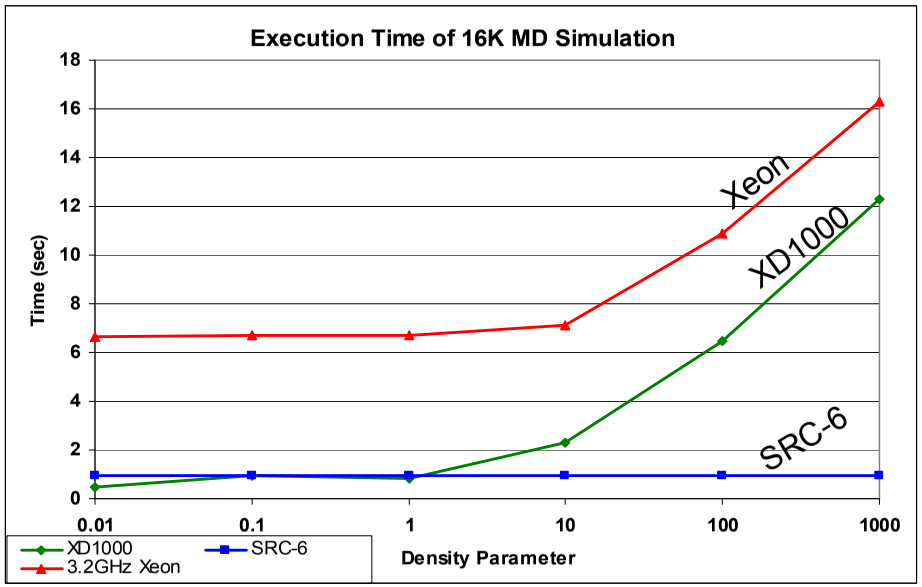
Carte C & SRC MAP-B

- Interconnect Parameters
  - Throughput, α_{input},α_{output}
    - Models SNAP interconnect
- Communication Parameters
  - $\#_{elements}$,input
    - 16384 molecules × 2 blocks (x/y,z) × 2 copies (i,j)
  - $\#_{elements}$,output
    - 16384 molecules × 2 blocks (x/y,z)
  - $N_{bytes/element}$
    - 8 bytes = 64-bit wide SRAM
- Computation Parameters
  - $\#_{elements}$,comp
    - 16384 (elements, iterations, etc.)
  - $N_{ops/element}$
    - 16383 operations (i.e. comparisons) again other atoms
    - Individual steps of comparison (i.e. pipeline depth) is difficult to quantify before design but not necessary
  - $Throughput_{proc}$
    - 4 parallel kernels, processing 4 comparisons per cycle
    - Assumes that pipelines will not stall
  - Clock Frequency
    - 100 MHz, default frequency
- Software Parameters
  - T_{soft}
    - Execution time on 3.2GHz Xeon
  - $N_{iterations}$
    - Only 1 iteration necessary

Results

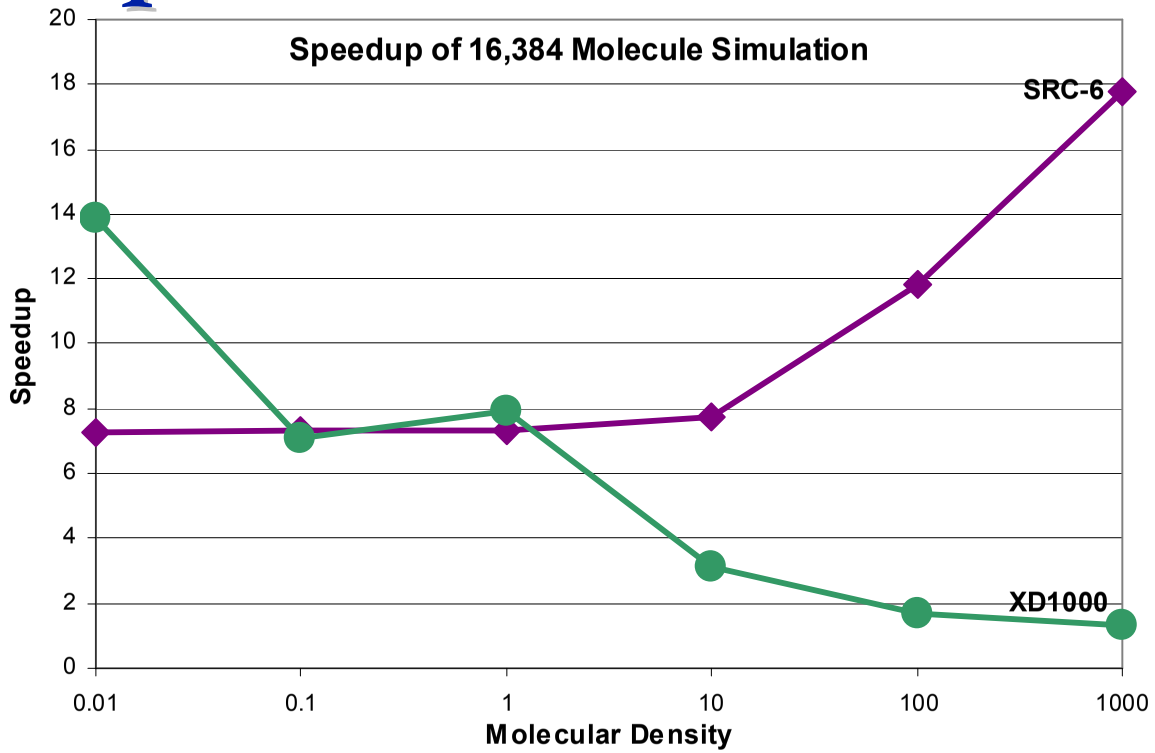


density	Execution Time (sec) XD1000	SRC-6	3.2GHz Xeon
0.01	0.48	0.92	6.67
0.1	0.94	0.92	6.70
1	0.85	0.92	6.71
10	2.28	0.92	7.12
100	6.49	0.92	10.87
1000	12.30	0.92	16.29



- Molecular Density
  - Pairs of molecules above threshold do not require extra computation
    - Sparse sets will have many pairs above distance threshold
    - Dense sets will have many pairs below distance threshold
- Impulse
  - Diminishing performance for denser sets
    - Pipelining inefficiencies
    - Load balancing issues
- SRC
  - Consistent pipelining performance
    - Best for dense sets of molecules
    - However, not as resource-efficient

Speedup & Prediction Accuracy

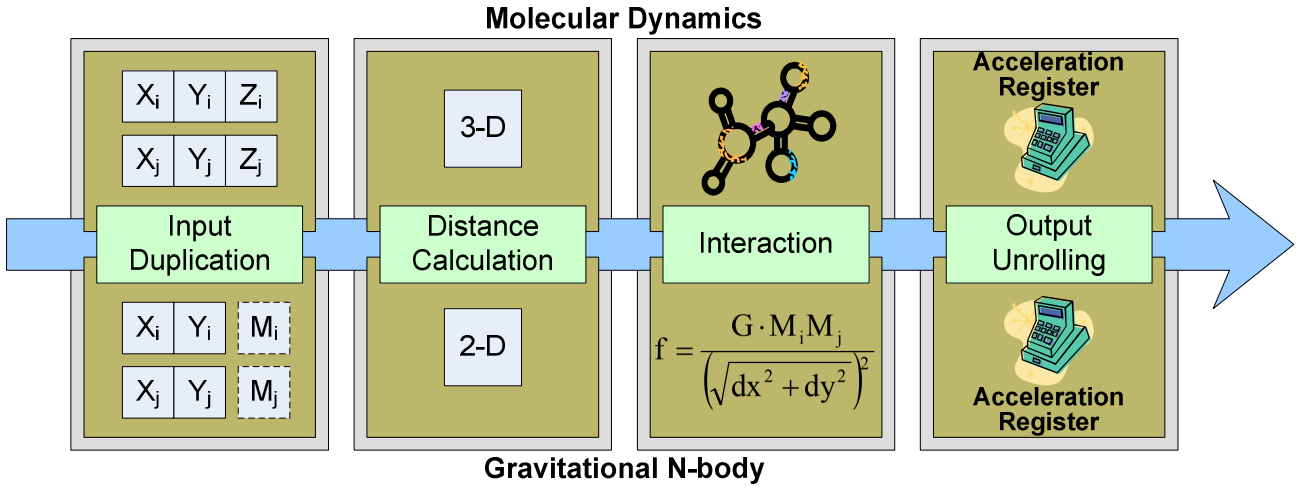


		Impulse C & XD1000		*Carte C & SRC-6*	
		Predicted	**Actual**	**Predicted**	**Actual**
tcomm	(sec)	8.74E-04	1.40E-03	1.04E-03	2.81E-03
tcomp	(sec)	6.71E-01	8.47E-01	6.71E-01	6.71E-01
toverhead	(sec)	0	N/A	0	2.46E-01
tRC	(sec)	6.71E-01	8.48E-01	6.71E-01	9.20E-01
Speedup	n/a	9.99	7.93	9.98	7.31

- XD1000 has CPU-initiated data transfers
  - All measurements are from wall-clock time
- SRC has FPGA-initiated data transfers
  - Total execution time is wall-clock time
  - Individual values take from FPGA cycle count
    - Discrepancy caused by system overhead

Taking Knowledge Forward

- Design Patterns
  - "A design pattern names, abstracts, and identifies the key aspects of a common design structure that make it useful for creating a reusable object-oriented design" [1]
  - "Design patterns offer us organizing and structuring principles that help us understand how to put building blocks (e.g., adders, multipliers, FIRs) together." [2]
- Gravitational N-Body



1. Gamma, Eric, et al., Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Boston, 1995.
2. DeHon, Andre, et al., "Design Patterns for Reconfigurable Computing", Proceedings of 12th IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04), April 20-23, 2004, Napa, California.

Conclusions

- Molecular Dynamics
  - Case study for evaluating algorithm mapping and optimization
    - Illustrates need for strategic design planning (formulation)
  - Memory hierarchy
    - Significant impact on overall algorithm performance
      - Without proper data locality & distribution, computation stalls unnecessarily
    - Balance requirements of overall application with features and strengths of target FPGA platform and design language
      - Efficient kernels had highest average speedup for MD
- Strategic Design
  - Performance prediction: RC Amenability Test
    - Knowledge of algorithm, FPGA system, & design language is critical
    - Accuracy will improve with larger problem sizes
      - Must balance accuracy of prediction with time required to make predictions
  - Knowledge is reusable for betterment of future algorithm design

Acknowledgements

- NSF I/UCRC Program (Grant EEC-0642422)
- CHREC members

- Altera Corporation (tools, devices)
- George Washington University (SRC-6 access)
- Impulse Accelerated Technologies (tools)
- SRC Computers (tools)
- XtremeData Inc. (tools, platform)

Questions?

References

1. S. R. Alam, J. S. Vetter, P. K. Agarwal, and A. Geist. Performance characterization of molecular dynamics techniques for biomolecular simulations. In 11th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, New York, NY, Mar 29-31 2006.
2. V. Kindratenko and D. Pointer. A case study in porting a production scientific supercomputing application to a reconfigurable computer. In Proc IEEE 14th Symp. Field-Programmable Custom ComputingMachines (FCCM), pages 13–22, Napa, CA, Apr 24-26 2006.
3. Azizi, N., Kuon, I., Egier, A., Darabiha, A., and Chow, P. 2004. Recongurable molecular dynamics simulator. In Proc IEEE 12th Symp. Field-Programmable Custom Computing Machines (FCCM). Napa, CA, 197-206.
4. Cordova, L. and Buell, D. 2005. An approach to scalable molecular dynamics simulation using supercomputing adaptive processing elements. In Proc. IEEE Int. Conf. Field Programmable Logic and Applications (FPL). 711-712.
5. Gu, Y., VanCourt, T., and Herbordt, M. 2006. Accelerating molecular dynamics simulations with congurable circuits. In Proc. IEE Computers and Digital Techniques. Vol. 153. 189{195.