Accelerated Particle Tracking using GPUs and GPULib

Vahid Ranjbar, Ilya Pogorelov, Keegan Amyx, Peter Messmer
{vahid, ilya, amyx, messmer}@txcorp.com

Tech-X Corporation
5621 Arapahoe Ave., Boulder, CO 80303
http://www.txcorp.com

Tech-X UK Ltd
Daresbury Innovation Centre
Keckwick Lane
Daresbury
Cheshire WA4 4FS, UK
http://www.txcorp.co.uk

Tech-X GmbH
Claridenstrasse 25
CH-8027 Zurich
Switzerland
http://www.txcorp.ch

Work supported by DOE/BES SBIR Grant # DE-SC0004585 and DOE/NP Grant # DE-SC0004432, NASA SBIR Grant NNG06CA13C, Tech-X Corp and NERSC.

TECH-X CORPORATION

Particle Tracking

- Machines defined as a 'lattice' of elements 'steering' and accelerating the beam.
- Particles tracked through this lattice in terms of their 6D phase space coordinates:

$$\vec{r} = x, p_x, y, p_y, z, p_z$$

- Modeled by applying a transport map to propagate particles from one element to the next:

$$\vec{r}^{\,n+1} = M^n(\vec{r}^{\,n})$$

=> Pushing lots of 6D particles through a sequence of maps (matrices or higher order elements)

=> Calculating space charge effects

Spin Tracking

- We are interested in Track spin ½ particles through the lattice. So in addition to the orbit (r vector) we need to track the evolution of the spin vector S whose evolution obeys the Thomas-BMT equation (ignoring Electric fields):

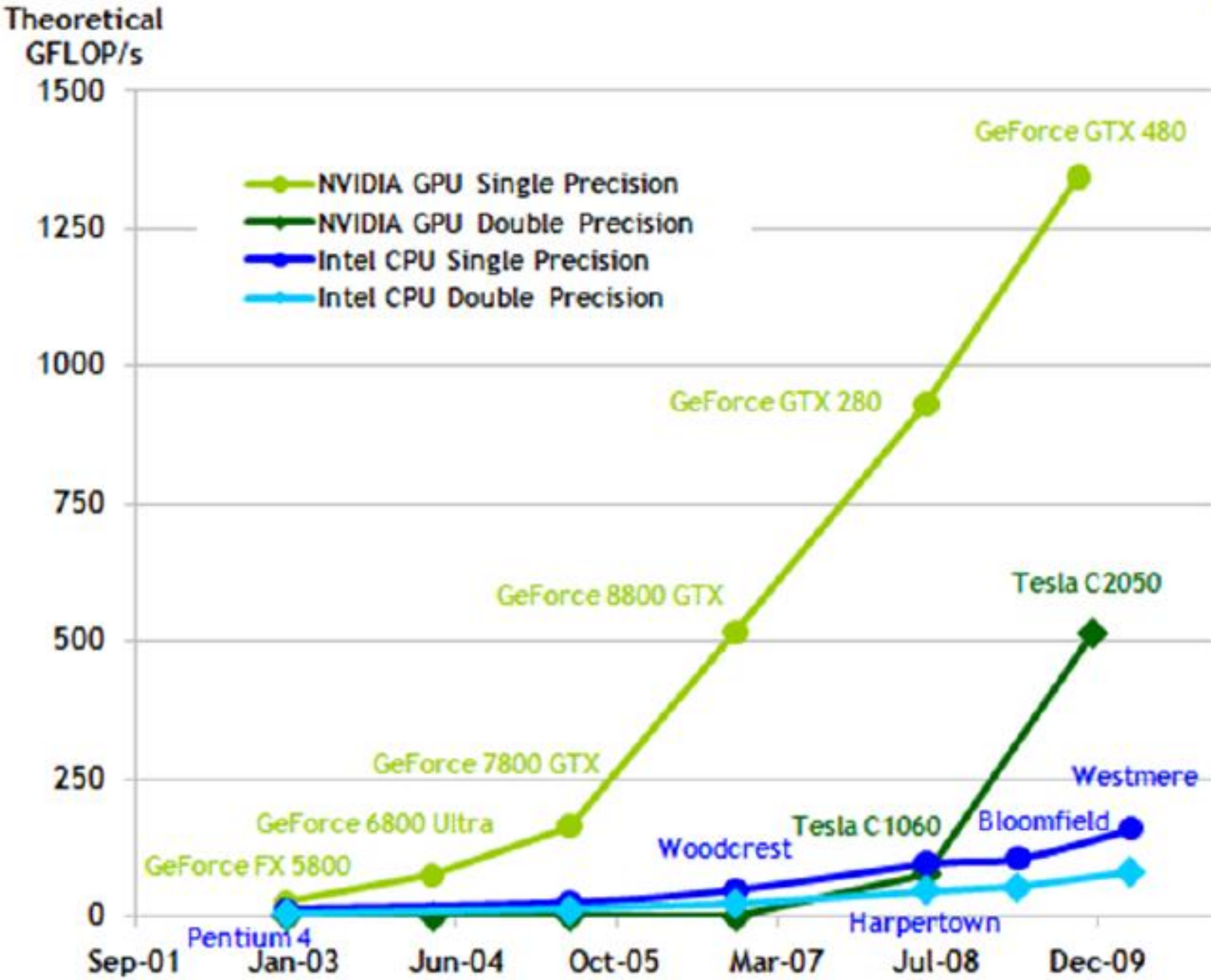
$$\frac{d\vec{S}}{dt} = \vec{S} \times \vec{\Omega}$$

- Spin transport expressed as 3x3 map with phase-space dependent matrix elements:

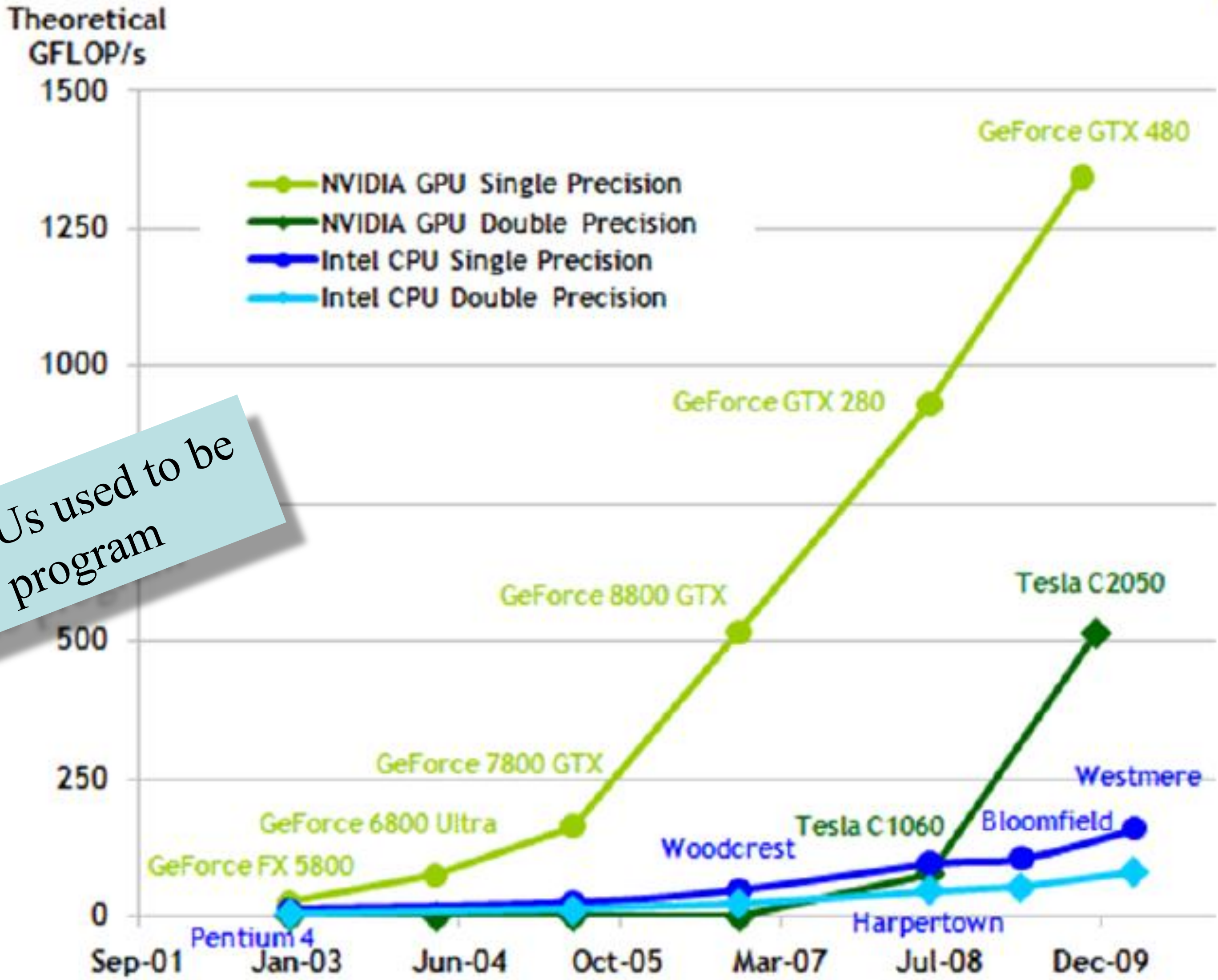
$$\begin{pmatrix} 1-(B^2+C^2)c & ABc+Cs & ACc-Bs \\ ABc-Cs & 1-(A^2+C^2)c & BCc+As \\ ACc+Bs & BCc-As & 1-(A^2+B^2)c \end{pmatrix}$$

=> Problem: Pushing a lot of independent particles through maps with phase-space dependent elements

Why scientific computing on GPUs?

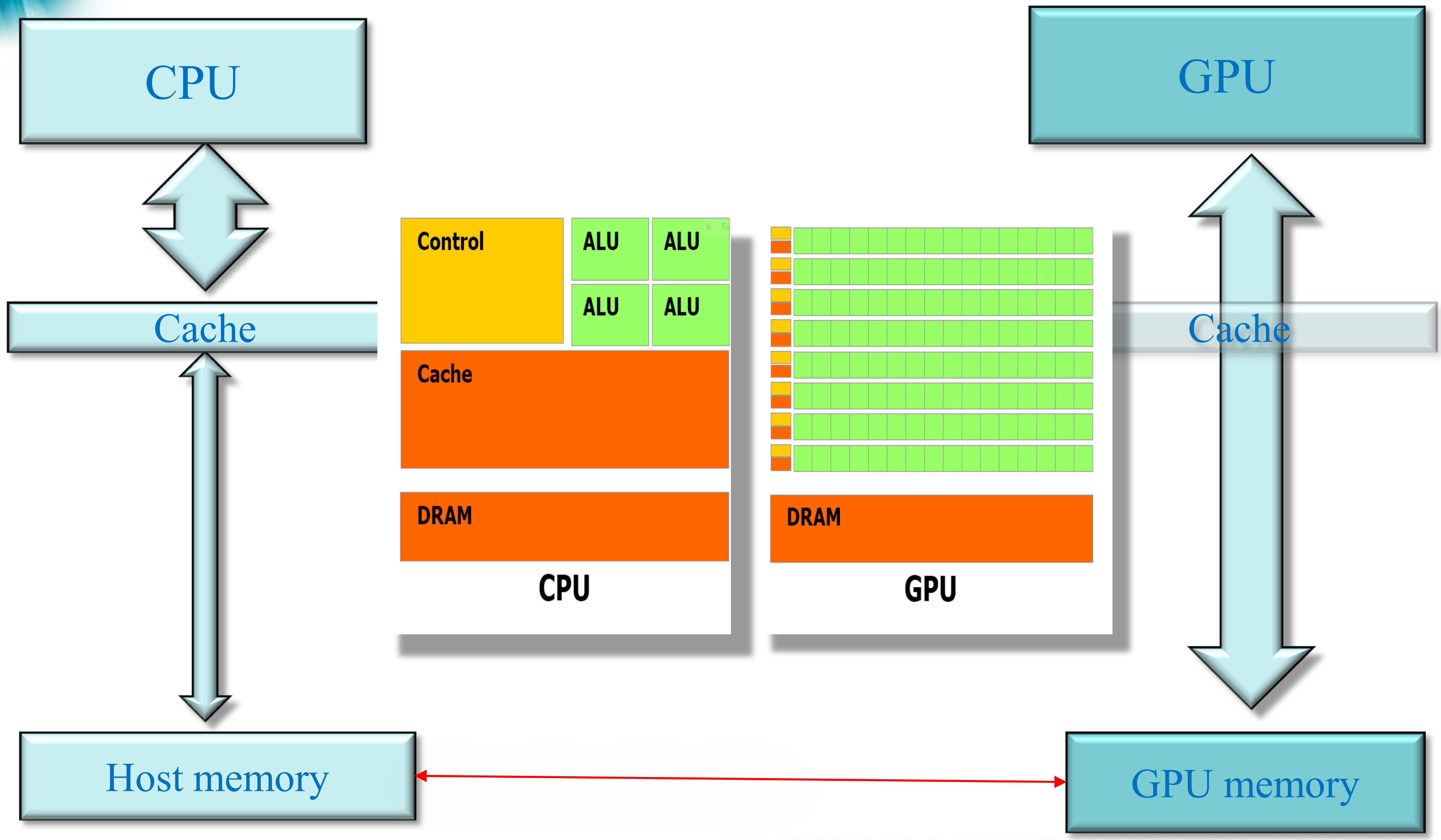


Why scientific computing on GPUs?



Problem: GPUs used to be hard to program

Massive Parallelism and Large Memory Bandwidth lead to High Performance of GPUs



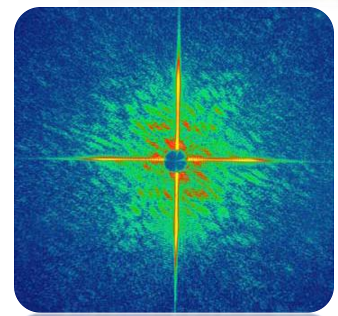
How to program these devices?

GPU Programming Overview

- OpenCL (Open Compute Lanugage)
  - Open standard, targeting GPUs, CELL, CPUs,..
  - Supported by AMD/ATI, NVIDIA, IBM, Intel,..
- CUDA (Compute Unified Device Architecture)
  - NVIDIA Proprietary
  - Wide-spread use
  - Strong influence on OpenCL
- Libraries
  - cuFFT, cuBLAS
  - cuLAPACK, ..

GPULib:
High-Productivity GPU Computing

- IDL (ITT Vis), MATLAB (Mathworks)
  C, Fortran
- Rich set of data parallel kernels
- Extensible with proprietary kernels
- Seamless integration into host language
- Explicit or implicit management of address spaces
- Interface to Tech-X' FastDL for multi-GPU/DMPP computing



FastDL

http://gpulib.txcorp.com

(free for non-commercial use)

Messmer, Mullowney, Granger, *"GPULib: GPU computing in High-Level Languages",* Computers in Science and Engineering, 10(5), 80, 2008.

Particle and Spin Tracking on GPUs

- Tracking Algorithm
  - Phase Space loaded onto GPU
  - Push particles through transport matrix
  - If Spin:
    - Compute average phase space positions
    - Compute Spin transport matrix
    - Push spin
- Particles never leave GPU

- Prototyping in GPULib
- Implemented as separate kernel

- Benchmark Tracking:
  - 20 lattice elements, quads and drifts
  - 100k particles

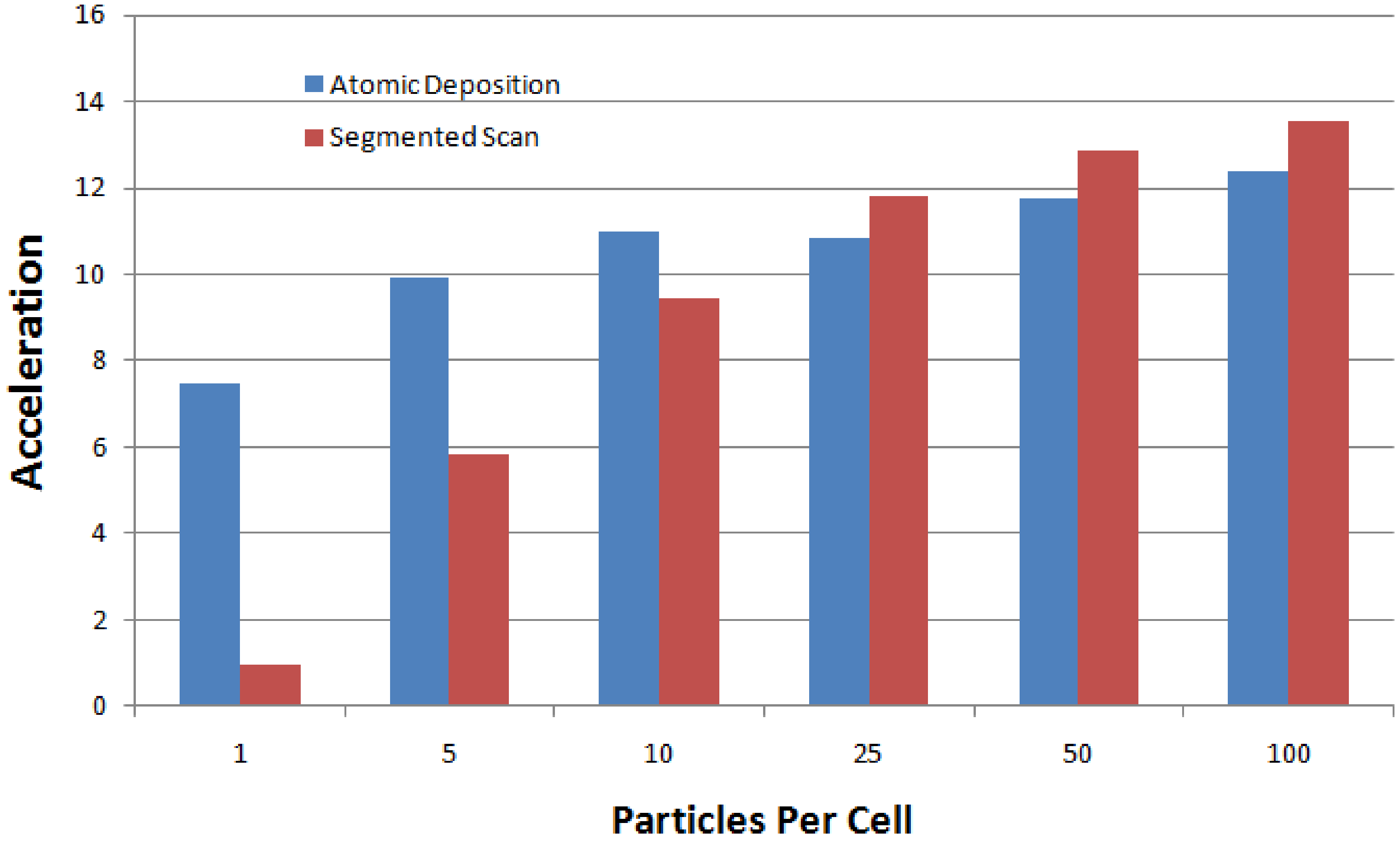
$\Rightarrow$ 40x speedup single (NVIDIA Tesla C2050 vs 2.8 GHz Westmere)

$\Rightarrow$ ~20x speedup in double

Collective Effects: Charge Deposition Challenge

- **Charge binning potential risk for memory conflict**
  - Multiple threads concurrently update same memory address
- **Data parallel approach**
  - Sort particles based on cell/bin
  - Sum contributions via prefix sum/segmented scan
  - Only global barrier needed
  - Complex implementation
- **Hardware assisted approach**
  - Atomic memory updates: prevent thread interference
  - Requires special hardware
  - “Useless” on pre-Fermi devices
  - Results in simple code

For low contention, atomic updates perform as well as data parallel deposition



CPU: 2.7GHz Westmere
GPU: NVIDIA C2050

Conclusion and future work

- **Particle/Spin Tracking computationally demanding**
  - Ideally suited for GPUs
- **Rapid prototyping on GPUs via GPULib**
  - Ultimately limited by bandwidth
- **Charge deposition conflicts**
  - Resolved via segmented scan, atomic updates
  - Atomics viable alternative
- **Ongoing/Future work**
  - Incorporation of GPU accelerated tracking into ELEGANT
  - Quad, LSCDRIFT (1D Space-Charge) first, other elements later
  - Spin Orbit Tracking classes for Unified Accelerator Library (UAL)