Simulating the Shallow Water Equations on Accelerator Architectures

Scott Rostrup
Advisor: Hans De Sterck

Applied Mathematics Department
University of Waterloo
In Collaboration with SHARCNET

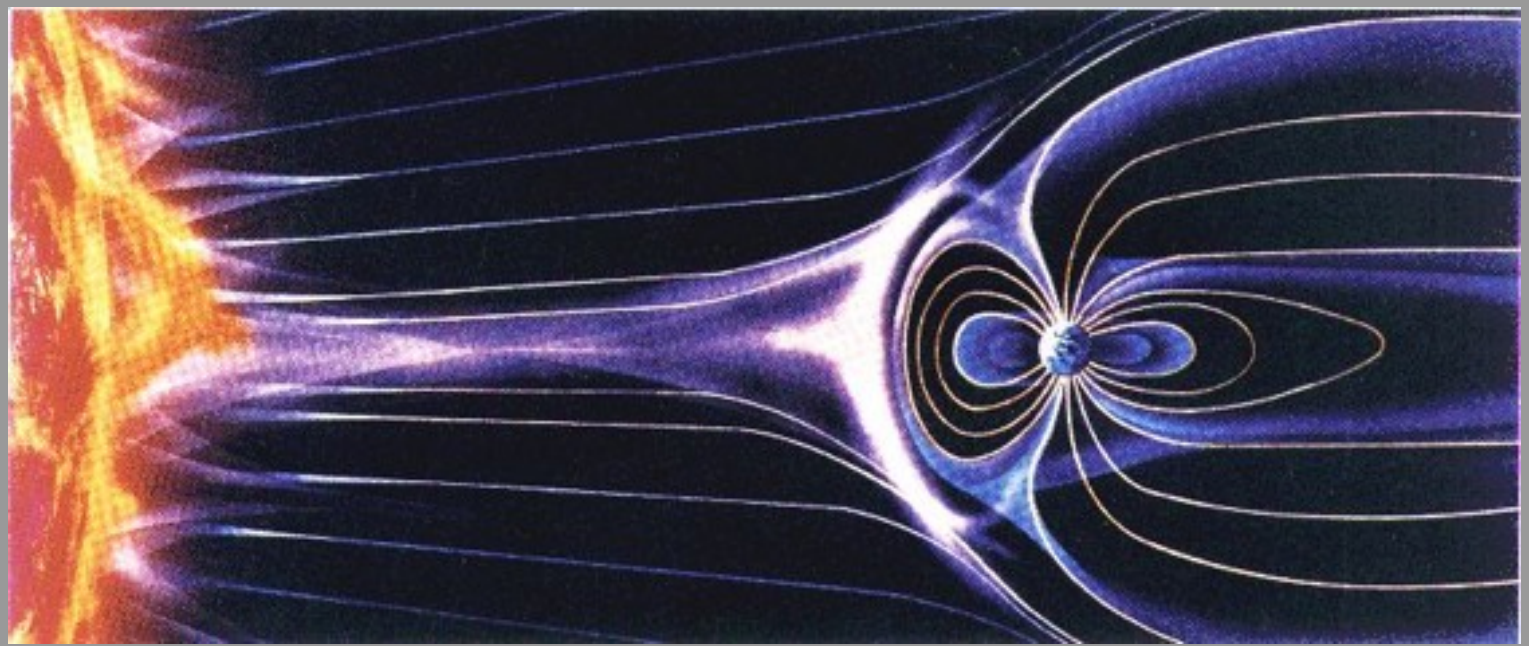
May 20, 2009

Research Aims

Space Weather

- 3D Simulation of the solar wind using the MHD Equations

Solar Wind



Research Aims

Motivation

- 3D Simulation of the solar wind using the MHD Equations
- 3D Hyperbolic System:

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} + \frac{\partial G(U)}{\partial y} + \frac{\partial H(U)}{\partial z} = S(U, x, y, z)$$

- We want to speed things up with parallel architectures

Research Aims

Motivation

- 3D Simulation of the solar wind using the MHD Equations
- 3D Hyperbolic System:

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} + \frac{\partial G(U)}{\partial y} + \frac{\partial H(U)}{\partial z} = S(U, x, y, z)$$

- We want to speed things up with parallel architectures
- As a simpler prototype we use a 2D model, the Shallow Water equations

Research Aims

Investigations

- How much improvement in performance can be expected?

Research Aims

Investigations

- How much improvement in performance can be expected?
- How do we obtain it?

Research Aims

Investigations

- How much improvement in performance can be expected?
- How do we obtain it?
- Build a prototype code!

Shallow Water Equations

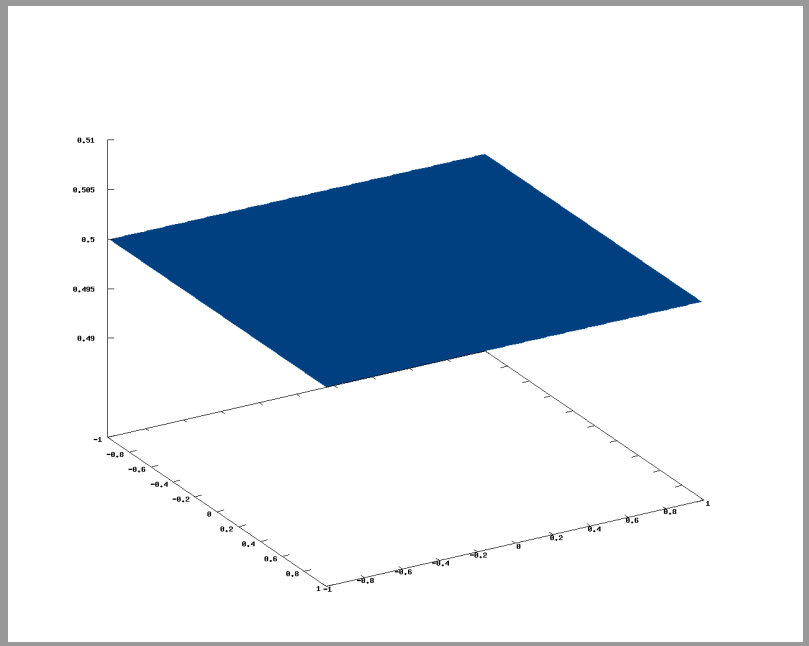
What are they?

- Model surface waves in a shallow layer of fluid
- Height scale small compared to length scale

Waves on a Sandy Shore

Images: Courtesy University of Waterloo Fluid Mechanics Group

Water Drops

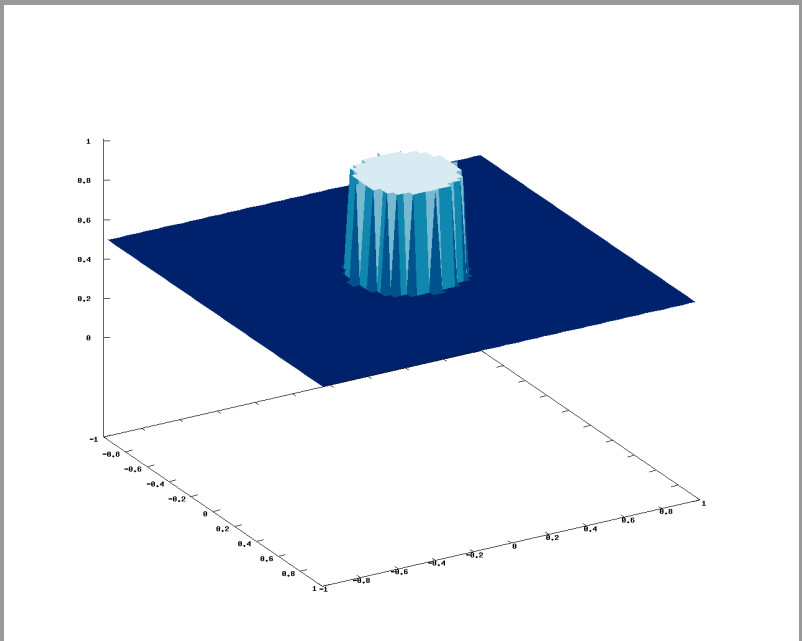


Shallow Water Equations

What are they?

- Model surface waves in a shallow layer of fluid
- Height scale small compared to length scale

Dam Break



Tsunami

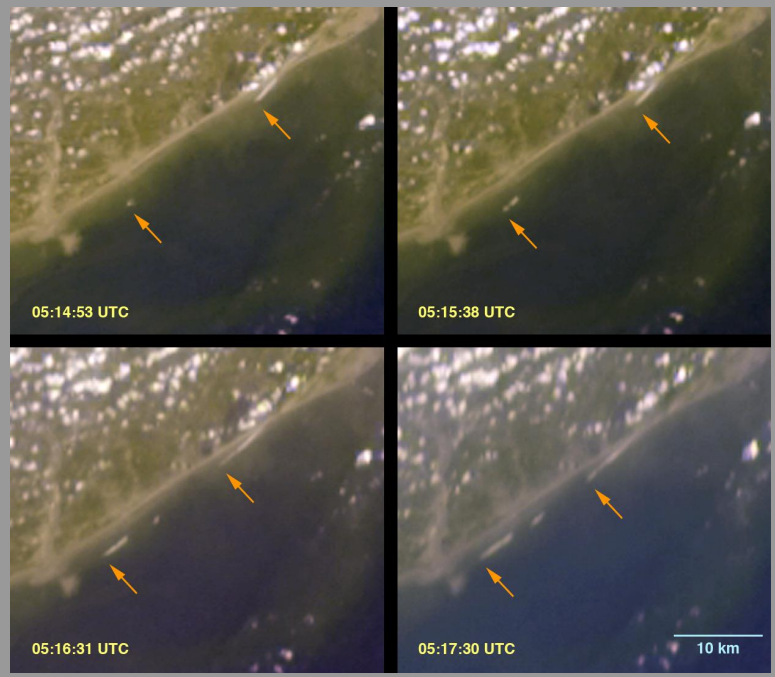
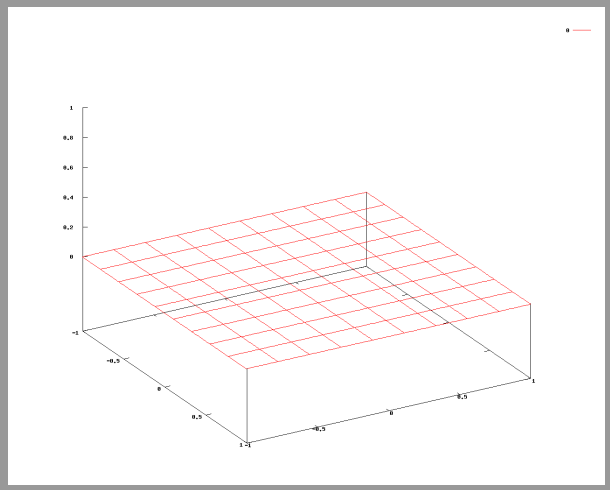
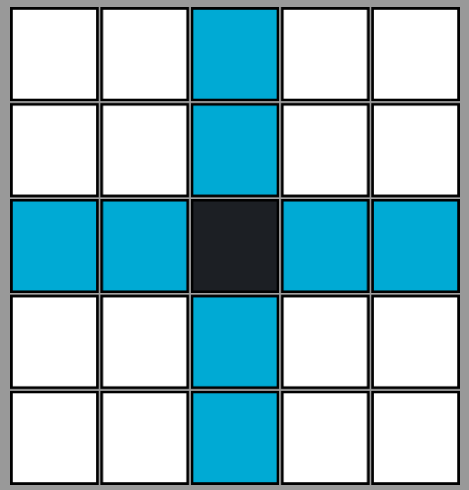


Image from NASA satellite imagery

Grid Based Solver

Simplified Problem Domain: a rectangular grid with walls





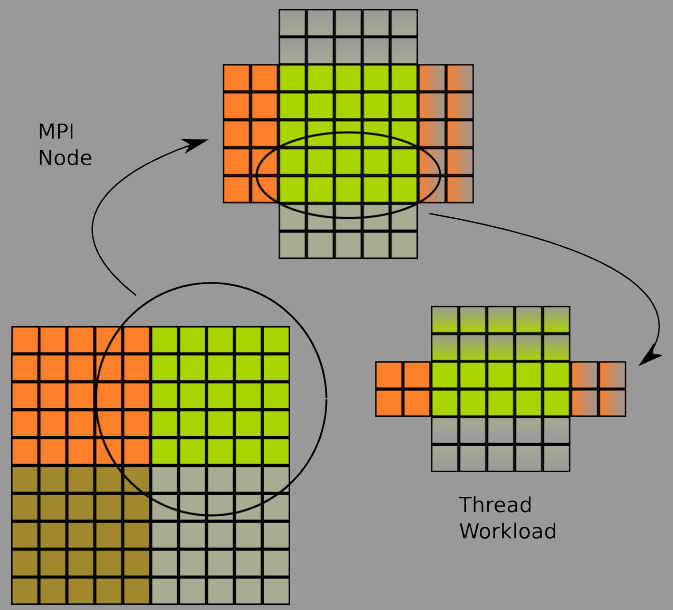
Stencil Calculation

$$U_{i,j}^{n+1} = U_{i,j}^{n} - \frac{\Delta t}{\Delta x}(F_{i+\frac{1}{2},j}^{n} - F_{i-\frac{1}{2},j}^{n}) - \frac{\Delta t}{\Delta y}(G_{i,j+\frac{1}{2}}^{n} - G_{i,j-\frac{1}{2}}^{n}),$$

Coarse Level Parallelism

1. Start with serial CPU code
2. Add Threading (pthreads or OpenMP)
3. Add MPI support for distributed memory systems

Node and Thread Parallelism

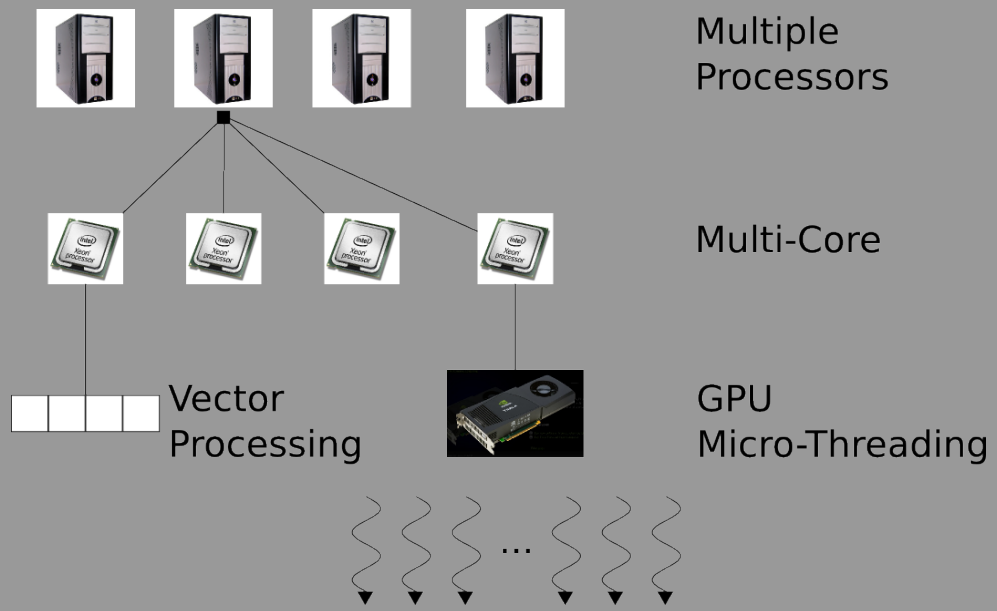


Fine Level Parallelism

Depends on the hardware

- Cell: Vector Intructions (SIMD)
- GPU: Micro-Threading (SIMT)

Vector and Microthread Parallelism

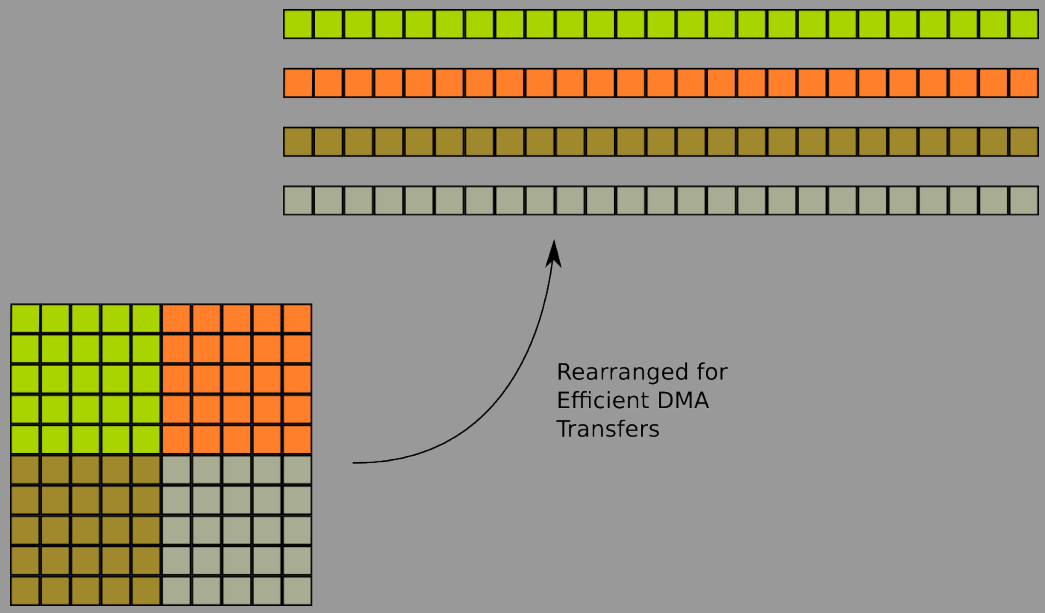


Memory Latency

Need to Hide Memory Latency (Cell)

- Double buffer data going in and out of the spe's local store

Block Memory Layout

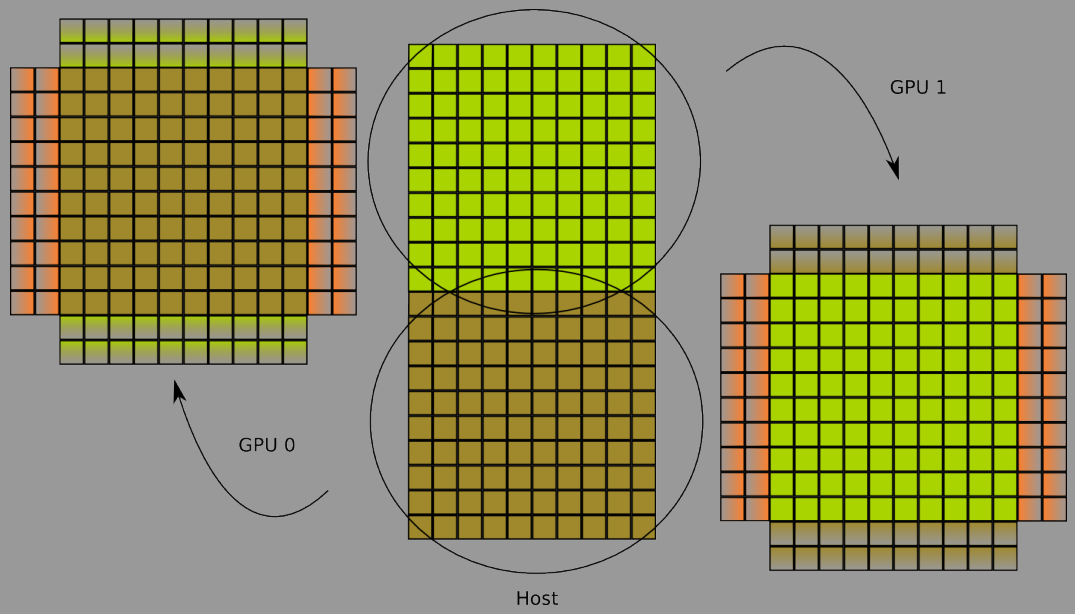


Memory Latency

Need to Hide Memory Latency (GPU)

- Minimize transfers between Host and Device memory
- Coalesce Loads and Stores
- Hide the remainder through context switching

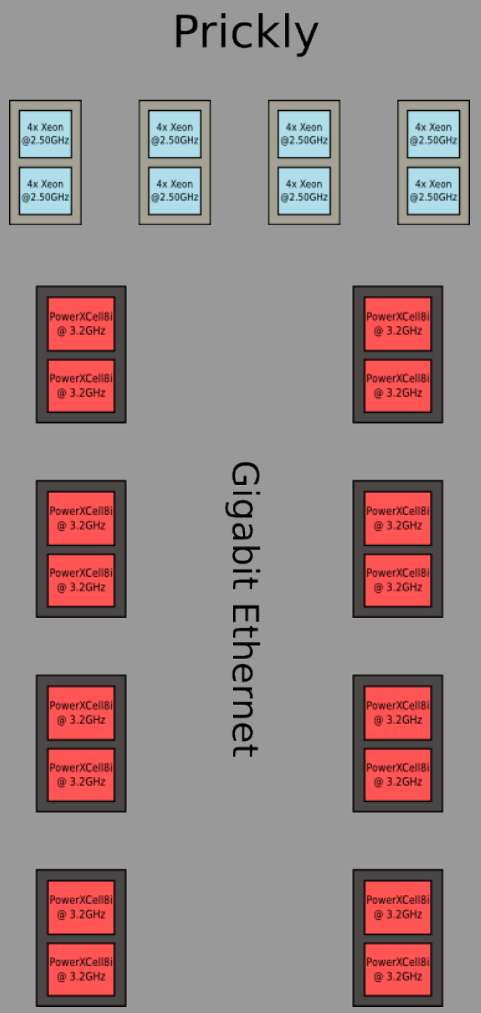
Tiled Memory Layout



SHARCNET Resources Used

QS22 Cluster (prickly)

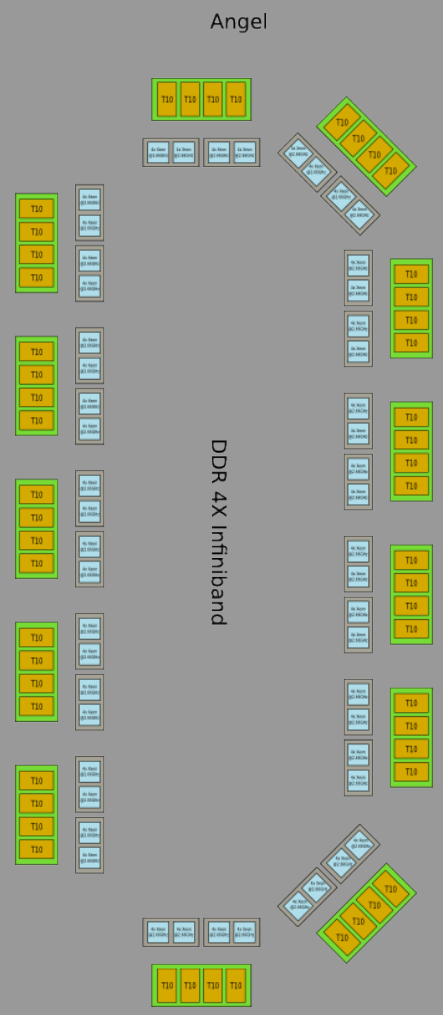
- 8x QS22 blade servers
- Dual-socket PowerXCell 8i @ 3.2GHz (16Gb)
- GigE Interconnect



SHARCNET Resources Used

Tesla S1070 Cluster (angel)

- 22x Xeon Nodes, dual-socket quad-core Xeon @ 2.66GHz (8 GB)
- 11x Tesla S1070 (4 GB)
- 1 CPU per T10 GPU
- Infiniband Interconnect



Cuda Results

Xeon vs Tesla T10

	Xeon 2x4	T10 x1	T10 x2
SP time (s)	4096	395 (10x)	197 (20x)
DP time (s)	4609	Pending	Pending

Cuda Results

Xeon vs Tesla T10

	Xeon 2x4	T10 x1	T10 x2
SP time (s)	4096	395 (10x)	197 (20x)
DP time (s)	4609	Pending	Pending

Cluster Results (SP)

N	Xeon 2x4	T10 x2
1	4096	197
2	1991	103
4	990	53
8	520	29
16	301	17

Cell Results

Xeon vs QS22

	Xeon 2x4	QS22 (16)	Speed-Up
SP time (s)	4096	321	12.7x
DP time (s)	4609	1156	4.0x

Cell Results

Xeon vs QS22

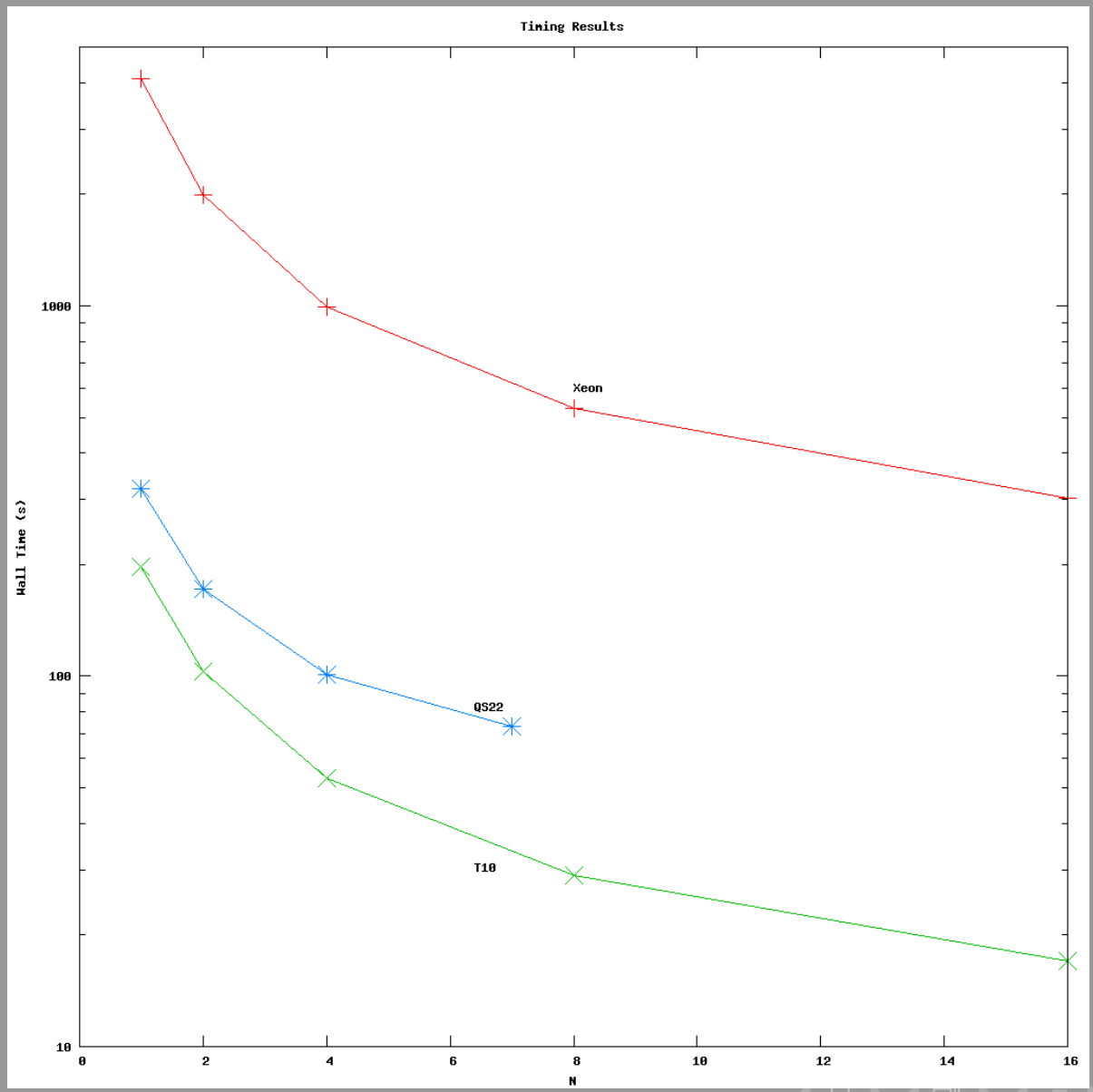
	Xeon 2x4	QS22 (16)	Speed-Up
SP time (s)	4096	321	12.7x
DP time (s)	4609	1156	4.0x

Cluster Results (DP)

N	Xeon 2x4	QS22 (16)
1	4609	1156
4	1143	329
8 (7)	576	(199)

One QS22 was down when these tests were run

Single Precision Results



Conclusions

- Cell: 4x DP and 13x SP speed up over 2x quad core Xeon
- GPU: 20x SP speed up over 2x quad core Xeon

We've been able to obtain some acceleration especially in single precision but more should be possible

Conclusions

- Cell: 4x DP and 13x SP speed up over 2x quad core Xeon
- GPU: 20x SP speed up over 2x quad core Xeon

We've been able to obtain some acceleration especially in single precision but more should be possible

Further Optimizations

- GPU: Texture memory, better use of shared memory, double precision
- Cell: Reducing instruction overhead, unrolling, better dual issues
- x86: SSE4 instructions
- Suggestions?

Presentation Over

- Thanks for coming.
- Questions?