Experiences with Seismic Algorithms on GPUs

Scott Morton

Geoscience Technology

Hess Corporation

Hess Commodity PC Cluster 2005

Hess Commodity PC Cluster 2005

John Hess: “What's next?”


DOOM3
DOOM3.COM
© 2003 id Software, Inc. All rights reserved.
ACTIVISION
id

HESS

Outline

- History
  - Other co-processors
  - Peakstream & ATI/AMD
  - Nvidia
- Pre-stack depth migration algorithms
  - Kirchhoff
  - Reverse-time
  - Wave-equation
- Status

Co-Processors

- Find co-processors "better" than PC CPUs
  - Better price-to-performance ratio
    - Want 10X improvement
    - Commodity volumes
    - Significant parallelism
      - Streaming
      - Multi-threaded
    - Must be "easy" to program
      - Similar programming tools
      - Able to handle similar computational algorithms

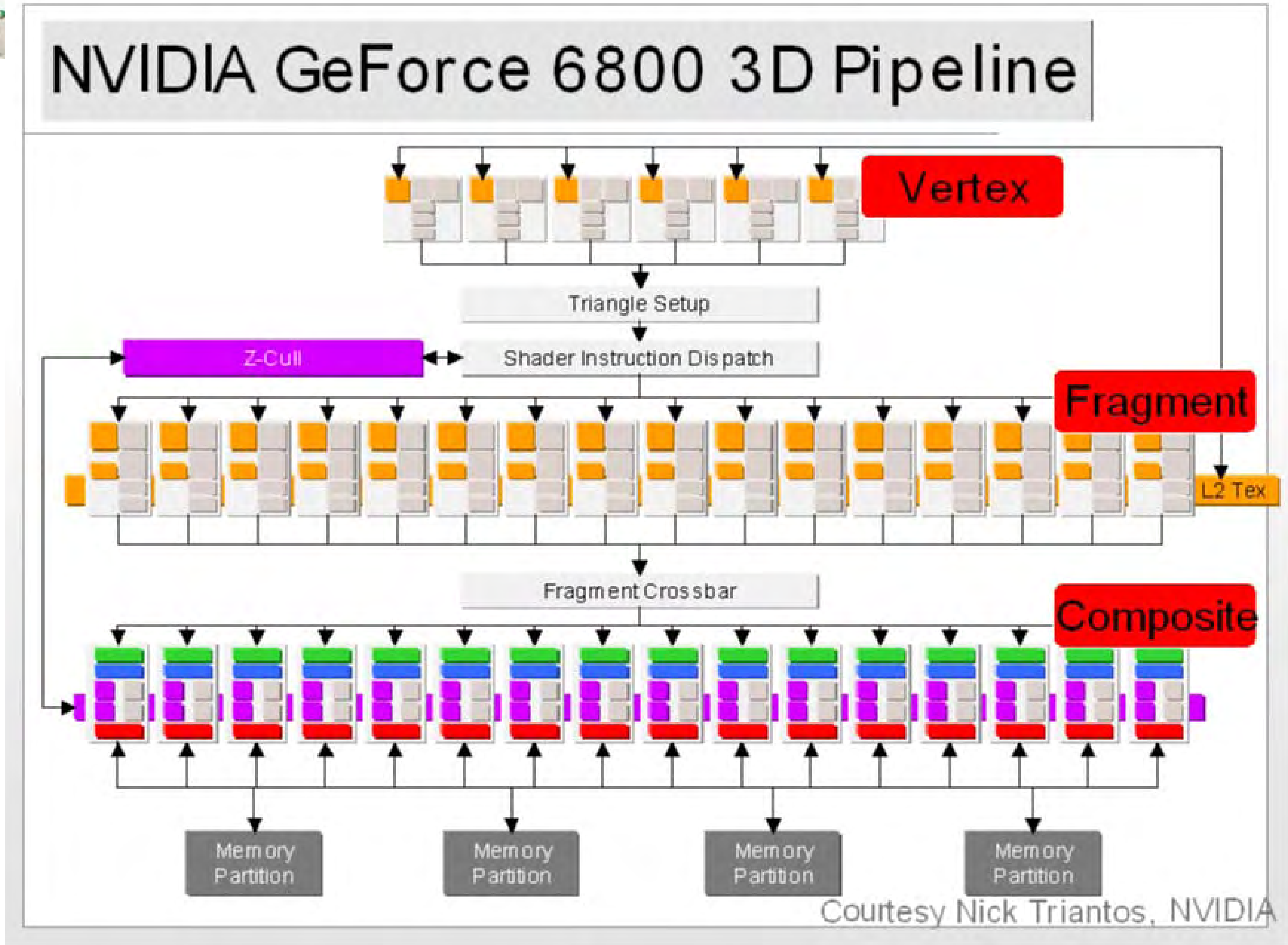
Co-Processors

- Effort expended varies from
  - Monitoring capabilities
  - Testing kernels
  - Buying hardware
- Candidates we've considered
  - DSPs
  - FPGAs
  - Cell processors
  - GPUs

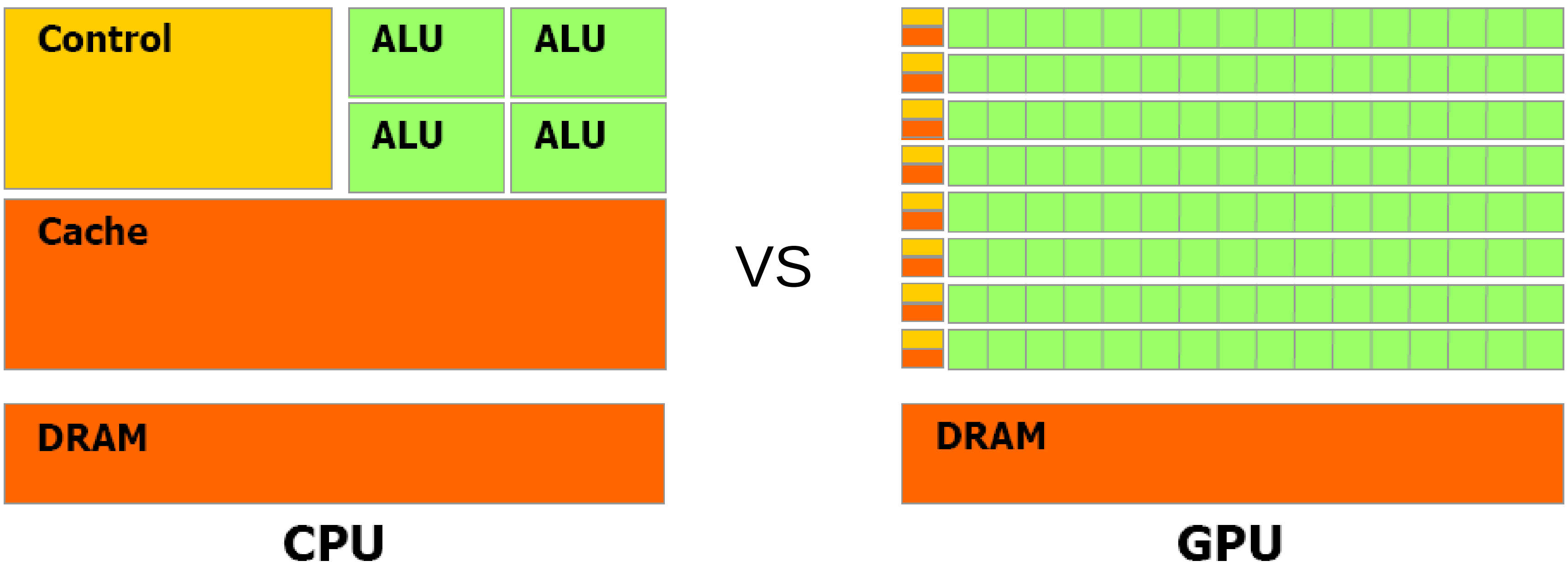
FPGAs

- SRC Computers, Inc
  - Co-processing board
  - Limited on-board memory
  - Tested wave-equation algorithm (2003)
    - ~10 X speed-up
    - ~10 X cost increase
    - Difficult to program
      - Libraries & compilers but …
      - Ended up using graphic scheduling of algorithm

Why GPUs?

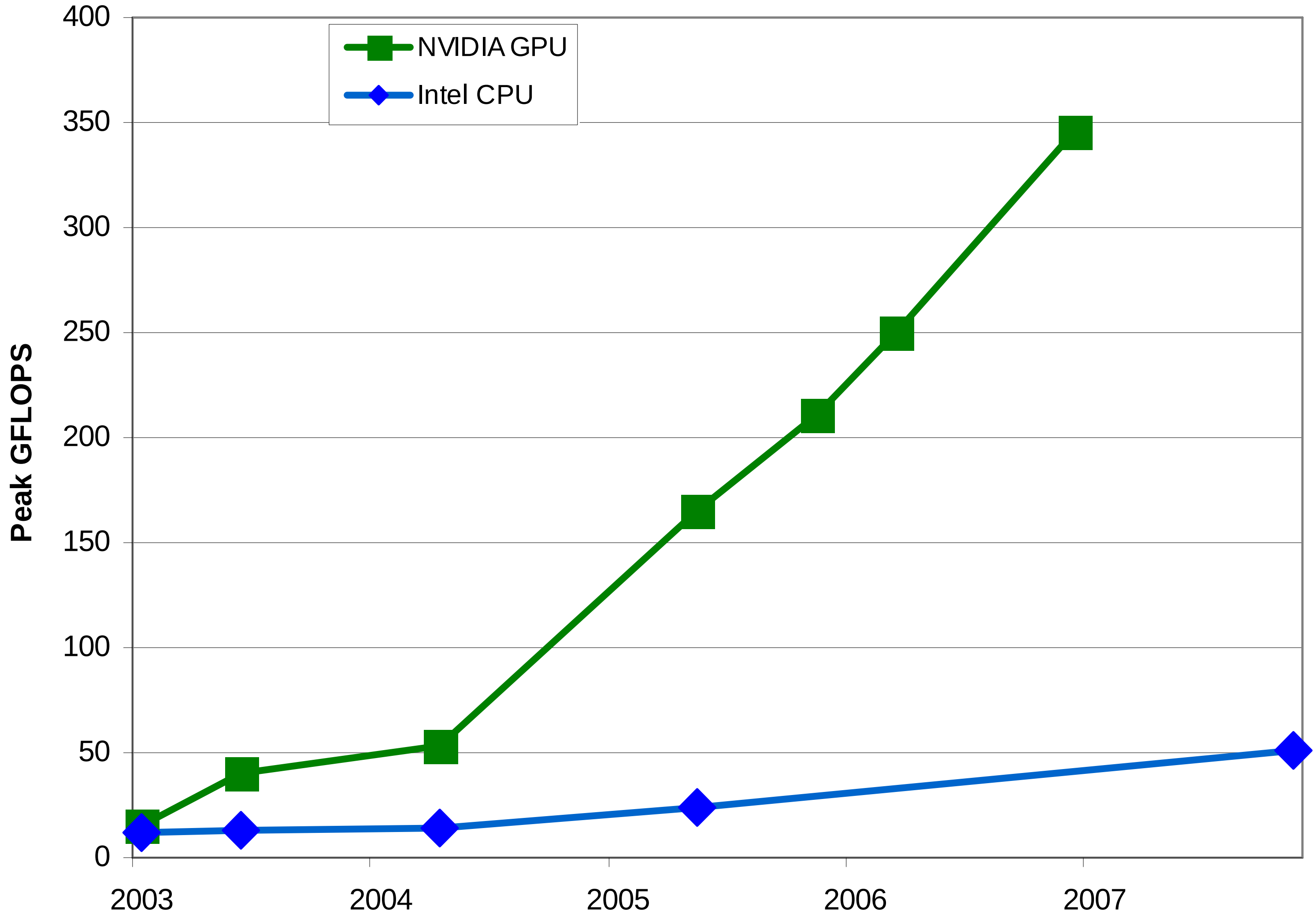

NVIDIA GeForce 6800 3D Pipeline
Vertex
Triangle Setup
Z-Cull
Shader Instruction Dispatch
Fragment
L2 Tex
Fragment Crossbar
Composite
Memory Partition
Memory Partition
Memory Partition
Memory Partition
Courtesy Nick Triantos, NVIDIA

Nvidia's G80 Architecture

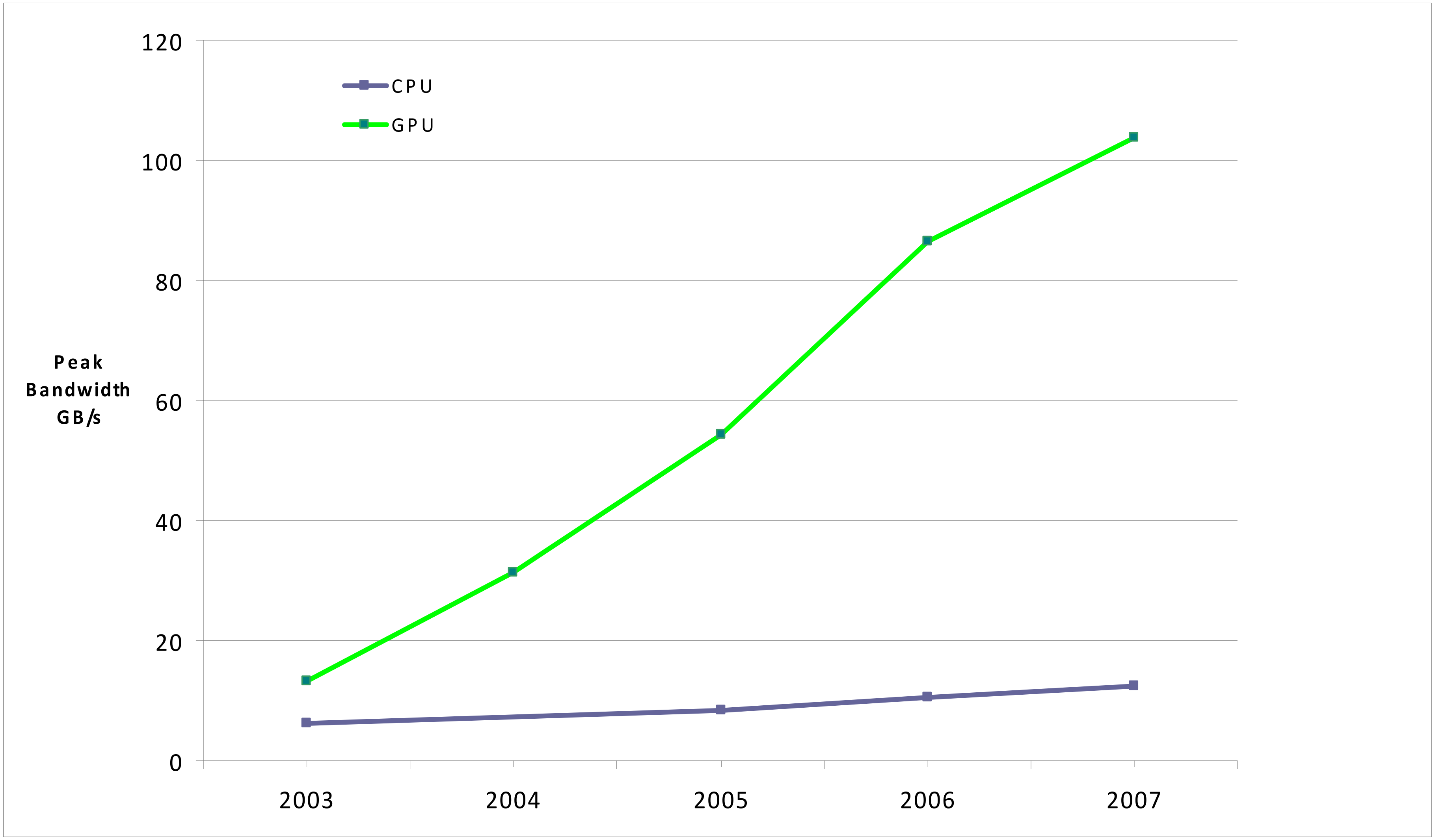


- Nvidia Corporation, CUDA Compute Unified Device Architecture (version 1.0), Jun. 2007, pg 2

Computational Performance



Bandwidth Performance



Accessing the GPU's Power

- Graphics people speak a different language
- APIs and languages
  - OpenGL API
  - Sh library
  - Nvidia's Cg
  - Stanford's Brook (for GPUs)

usability

Peakstream

- Software platform
  - Write in standard C++
  - Virtual machine & JIT compiler
    - Creates stream kernels
    - Run kernels on GPU or CPU or …
  - Partnered early with ATI (AMD)

Peakstream

- Tested several algorithms
  – Kirchhoff prototype
  – Finite-difference modeling
    - 2-D demo at SEG 2006
- Performance
  – 5 – 10 X speed-up in 2-D
  – Only 2 - 3 X in 3-D
- Promising but …

Peakstream

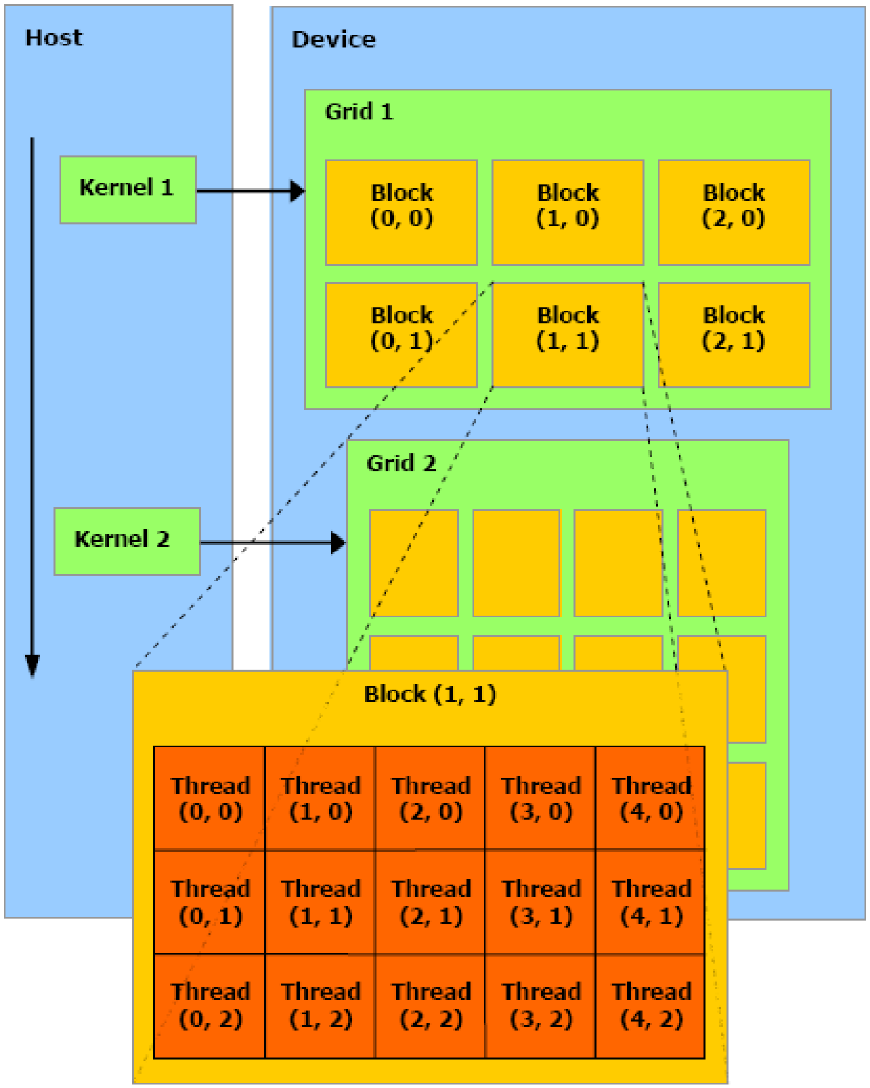
Gobbled by Google

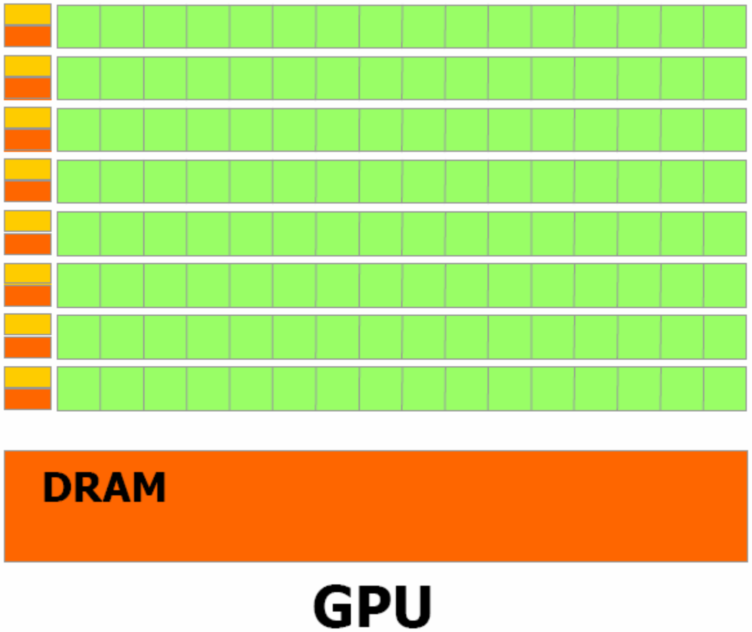
- Tested several algorithms
  - Kirchhoff prototype
  - Finite-differen[illegible] modeling
    - 2-D demo at S[illegible]
- Performance
  - 5 – 10 X speed-up in 2-D
  - Only 2 - 3 X in 3-D
- Promi[illegible] out[illegible]

Nvidia's CUDA

- C with parallel language extensions
  - SIMD programming language
  - Specify grids of blocks of threads
  - Complex memory model
- Programming involves
  - Copying data between CPU and GPU
  - multi-threaded kernels for GPU

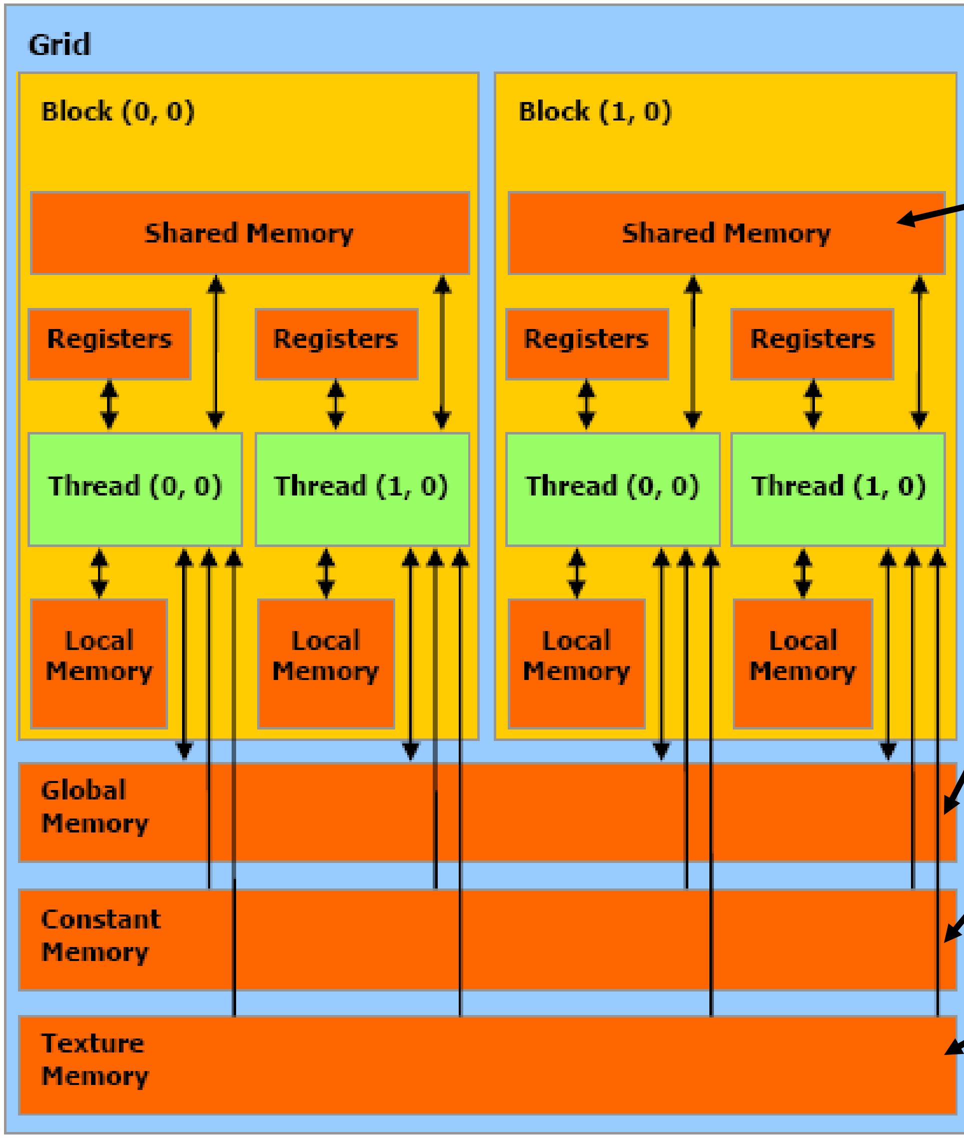
CUDA Programming Model





- Nvidia Corporation, CUDA Compute Unified Device Architecture (version 1.0), Jun. 2007, pg 9

CUDA Memory Model



Fast
Limited in size (16kB)
Has access requirements

Relatively slow
Large in size (1.5GB)
Has access requirements

Small but cached
Can not be changed during runtime

2-D cached
Can not be changed during runtime

- Nvidia Corporation, CUDA Compute Unified Device Architecture (version 1.0), Jun. 2007, pg 11

Nvidia's CUDA

- Relatively easy programming
  - Took 1.5 days to learn CUDA and write 2-D modeling program
  - But steeper learning curve for optimization

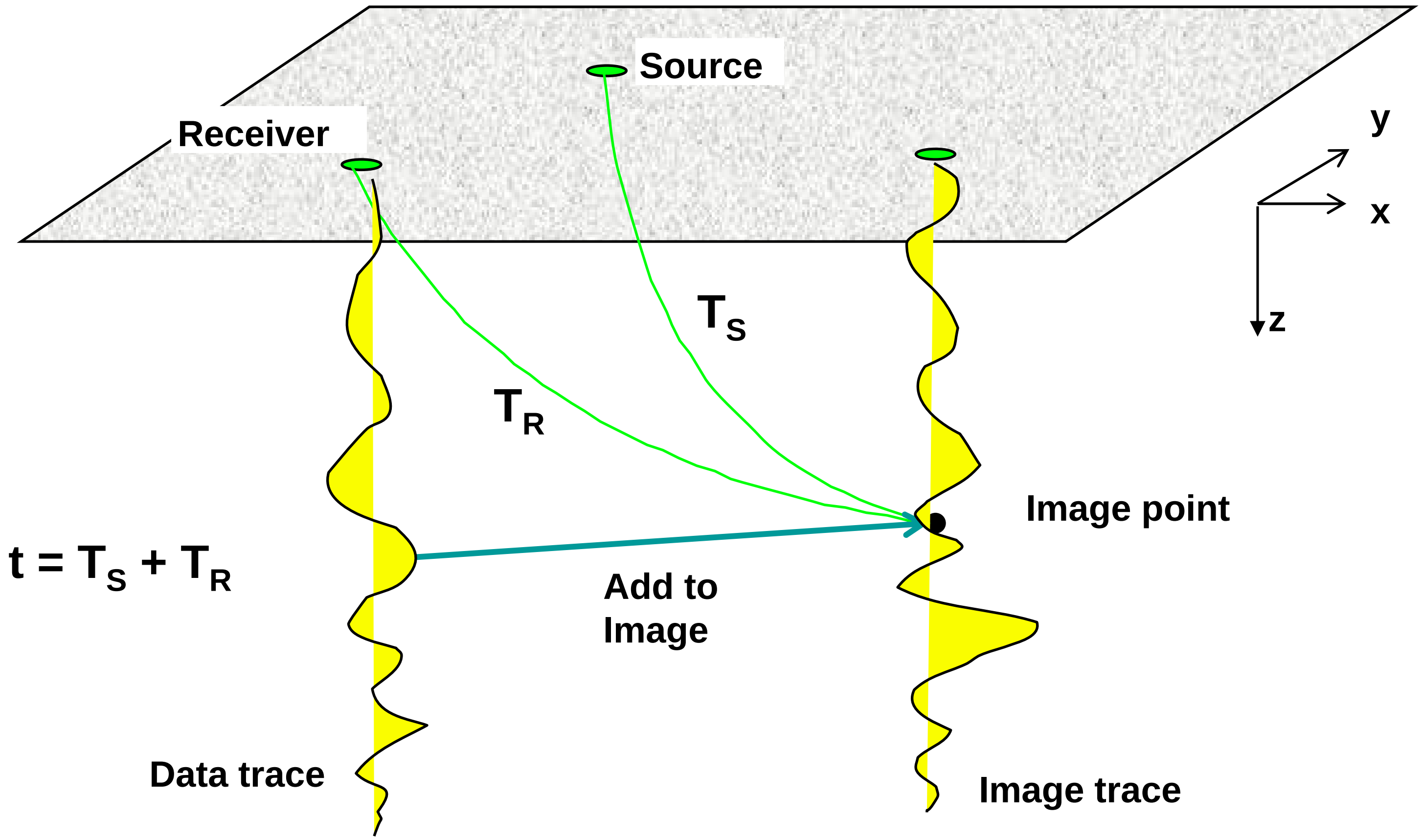
Prestack Depth Migration

- Algorithmic Efforts
  - Test with simplified kernels
  - If promising, port production code
- Candidates
  - Kirchhoff
  - Reverse-time
  - Wave-equation

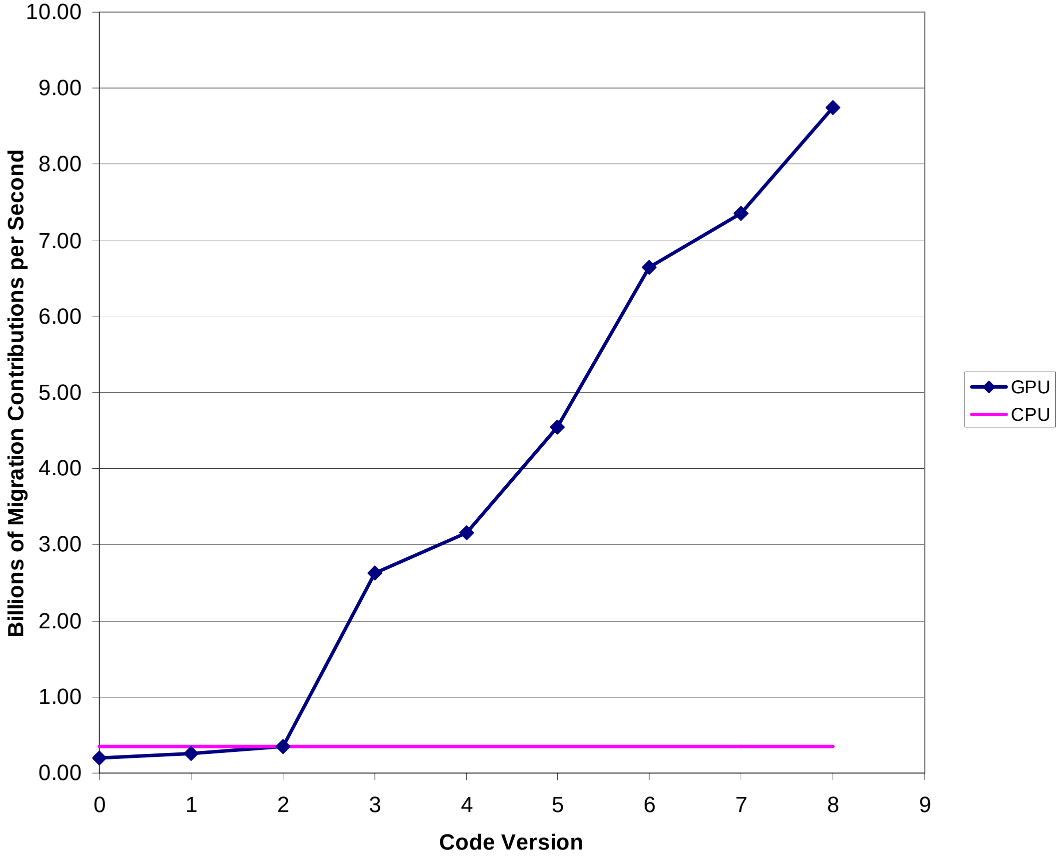
Prestack Depth Migration

- Algorithmic Efforts
  - Test with simplified kernels
  - If promising, port production code
- Candidates
  - Kirchhoff
  - Reverse-time
  - Wave-equation
- Disclaimer: *your/our* mileage *will* vary

Kirchhoff Kernel

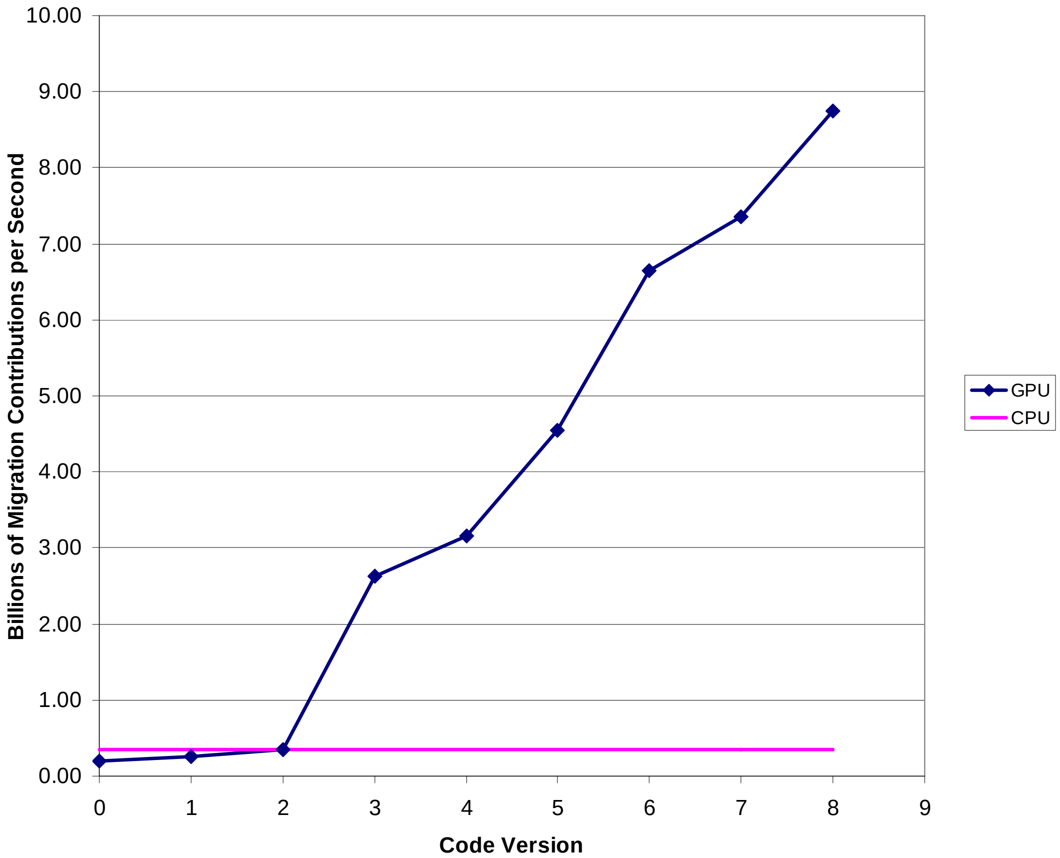


Kirchhoff Learning Curve



HESS

Kirchhoff Learning Curve



0 – Initial Kernel

1 – Used Texture Memory

2 – Shared Memory Image Cell

3 – Global Memory Coalescing

4 – Decreased Data Trace Shared Memory Use

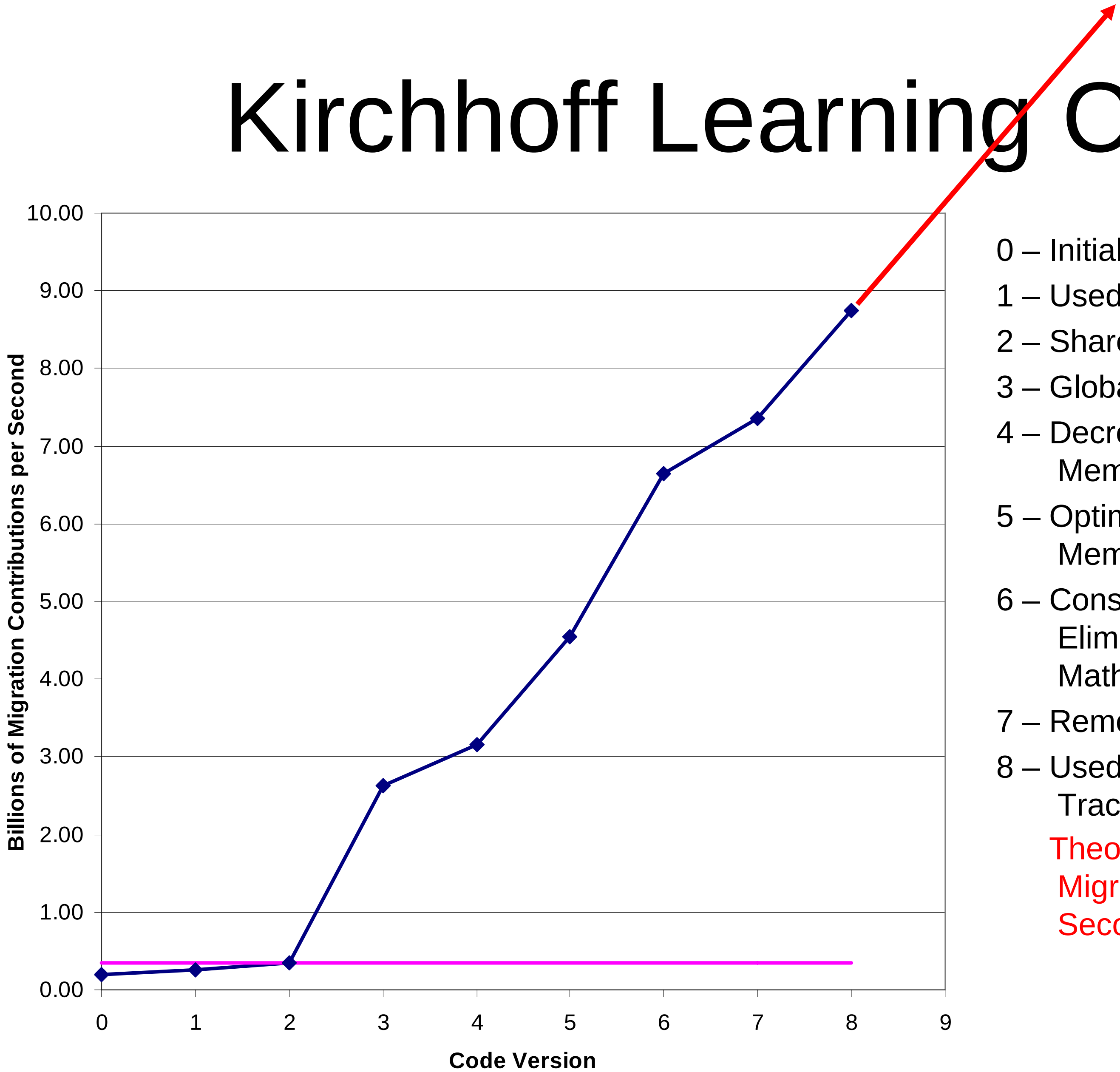
5 – Optimized Use of Shared Memory

6 – Consolidated “if” Statements, Eliminated or Substituted Some Math Operations

7 – Removed an “if” and “for”

8 – Used Texture Memory for Data-Trace Fetch

Kirchhoff Learning Curve



0 – Initial Kernel

1 – Used Texture Memory

2 – Shared Memory Image Cell

3 – Global Memory Coalescing

4 – Decreased Data Trace Shared Memory Use

5 – Optimized Use of Shared Memory

6 – Consolidated "if" Statements, Eliminated or Substituted Some Math Operations

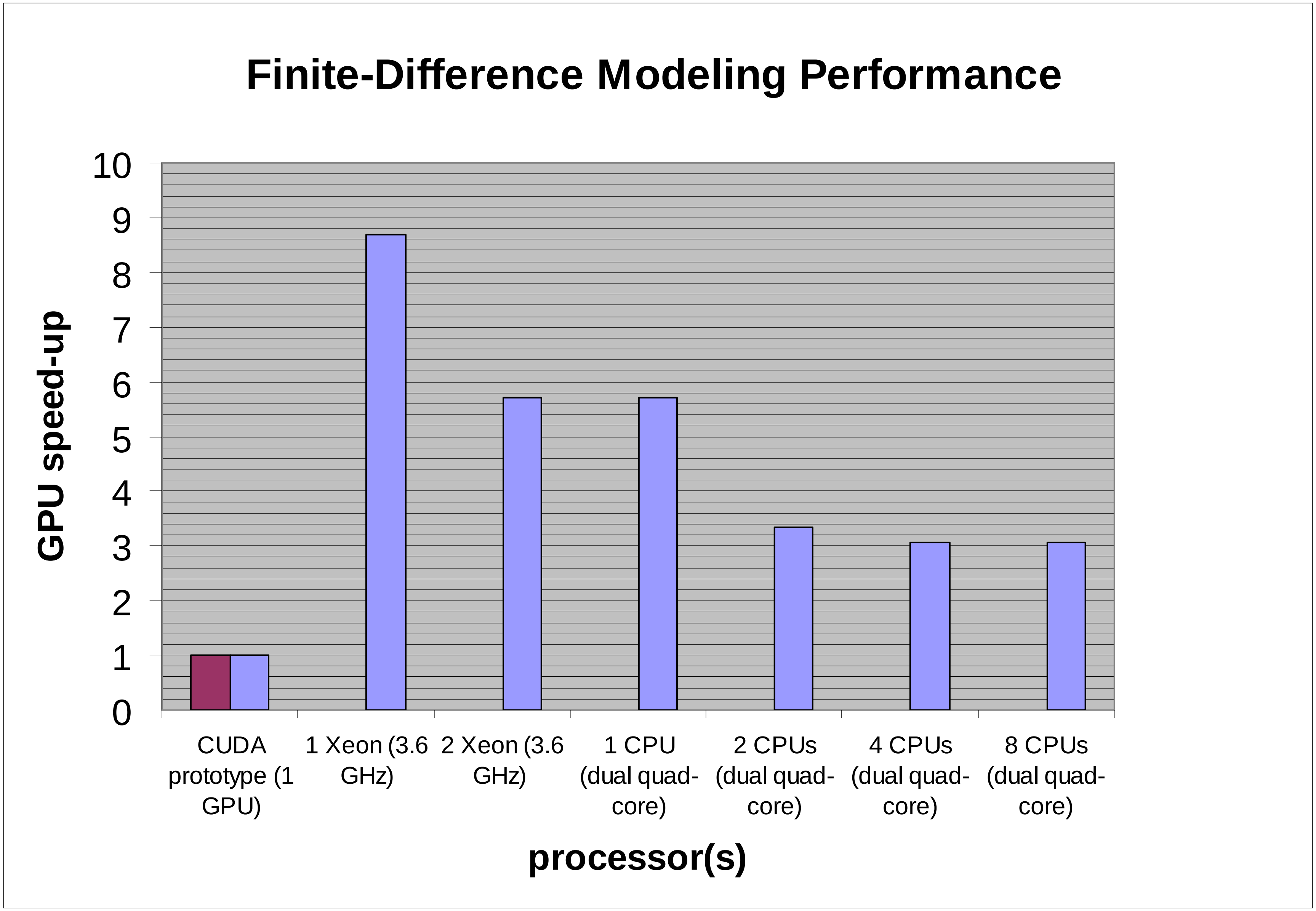
7 – Removed an "if" and "for"

8 – Used Texture Memory for Data-Trace Fetch

Theoretical Max of ~ 70 Billion Migration-Contributions per Second

Reverse-time & Modeling

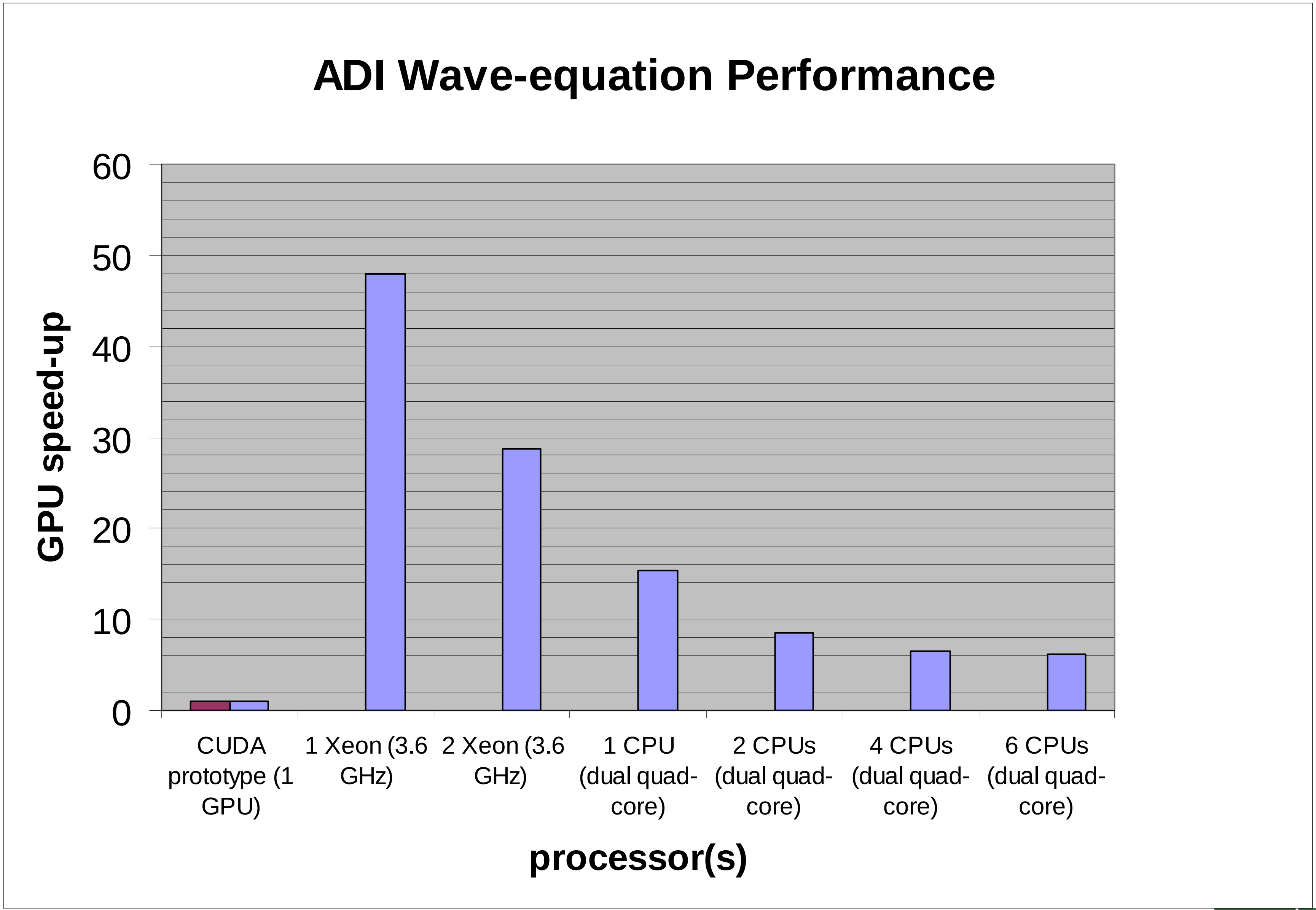
- Pseudo-spectral
  - Dominated by 3D FFT
  - 5 X speed-up over a single CPU
  - 20% speed-up over dual quad-core node
- Finite difference
  - 2^{nd} order time, 8^{th} order space


Finite-Difference Modeling Performance
GPU speed-up
0
1
2
3
4
5
6
7
8
9
10
CUDA prototype (1 GPU)
1 Xeon (3.6 GHz)
2 Xeon (3.6 GHz)
1 CPU (dual quad-core)
2 CPUs (dual quad-core)
4 CPUs (dual quad-core)
8 CPUs (dual quad-core)
processor(s)

HESS

Wave-equation Migration

- Implicit finite difference (ADI)
  - Dominated by complex tri-diagonal systems
  - Wrote prototype CUDA code
    - Benchmarked on 1 GPU
    - Compared against production kernel
  - Currently debugging full production code


ADI Wave-equation Performance
GPU speed-up
60
50
40
30
20
10
0
CUDA prototype (1 GPU)
1 Xeon (3.6 GHz)
2 Xeon (3.6 GHz)
1 CPU (dual quad-core)
2 CPUs (dual quad-core)
4 CPUs (dual quad-core)
6 CPUs (dual quad-core)
processor(s)

Status of Hess GPU Systems

- Got test system in spring ‘07
  - Used for prototyping Kirchhoff code
  - Benchmarked simple kernels

Status of Hess GPU Systems

- Ordered 32-node system in Jan ’08
  - Each node contains
    - Dual quad-core 1U server with 8 GB memory
    - Connected to 1U 4-GPU server with 1.5 GB each
  - Running test codes by Fri Feb 29th

Status of Hess GPU Systems

- Ordered 32-node system in Jan ’08
  - Each node contains
    - Dual quad-core 1U server with 8 GB memory
    - Connected to 1U 4-GPU server with 1.5 GB each
  - Running test codes by Fri Feb 29th
  - Running production code by …?
  - Out-performing 4000-CPU cluster by …?
    - 128 * 29 * 2 = 7424

Acknowledgements

- Hess management for their support and permission to publish.
- Thomas Cullison, Jeff Davis, Mac McCalla @ Hess
- David Caliga @ SRC Computers
- Matthew Papakipos & Philip Nenon @ Peakstream
- Paulius Micikevicius, Paul Holzhauer & Philip Nenon @ Nvidia