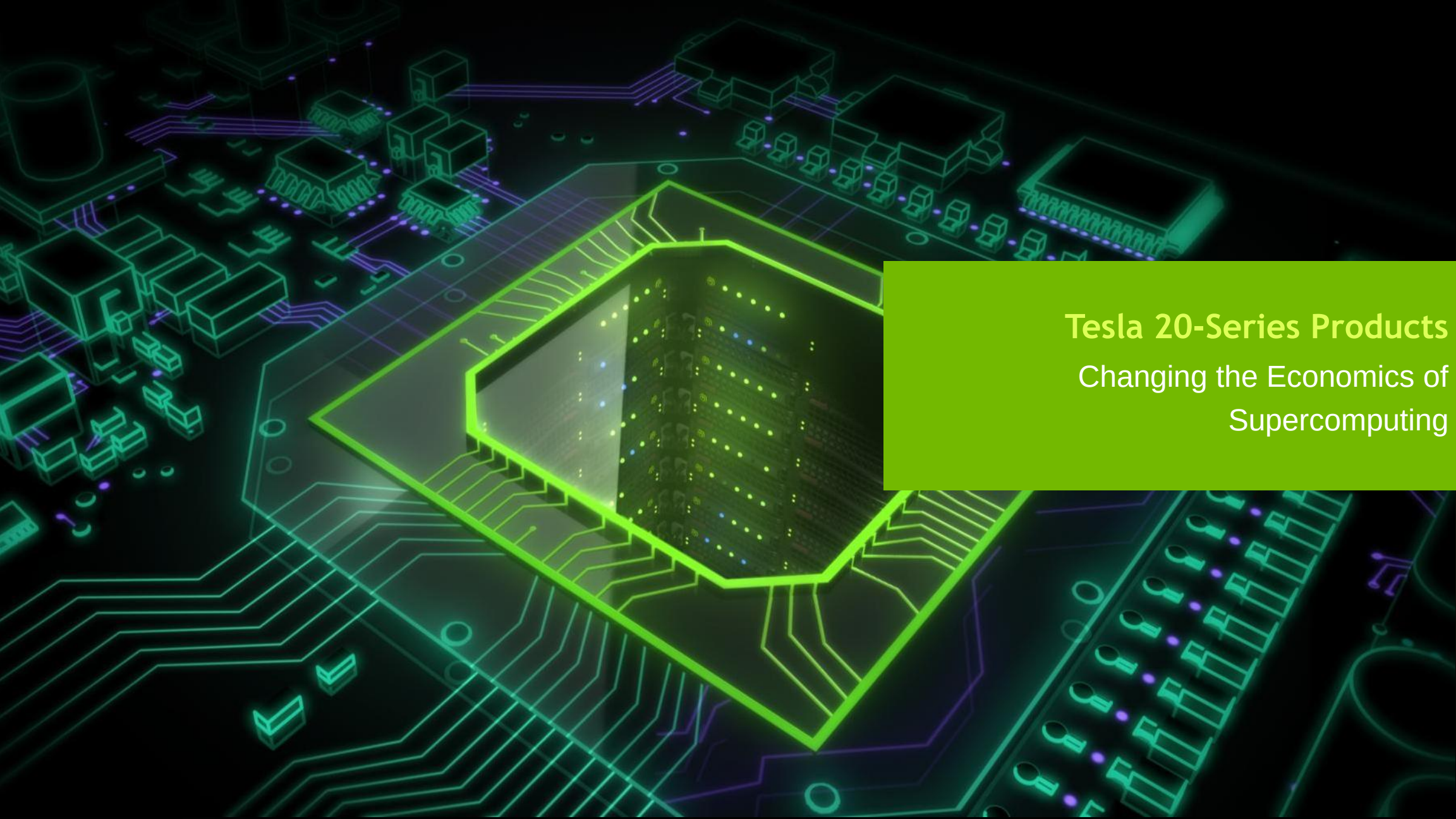
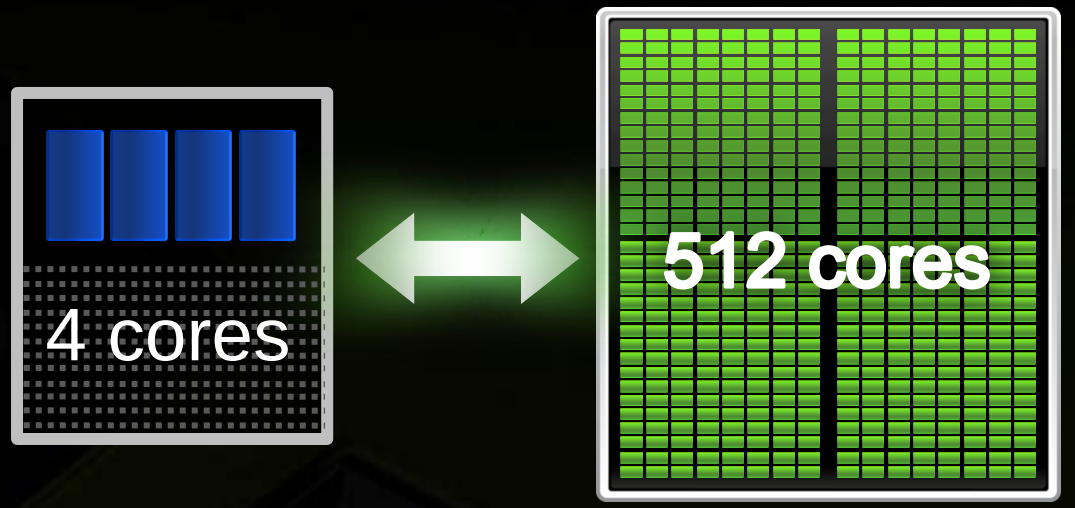


Tesla 20-Series Products

Changing the Economics of Supercomputing

GPU Computing



CPU + GPU Co-Processing

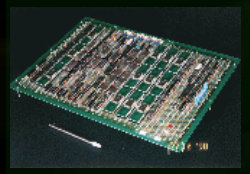
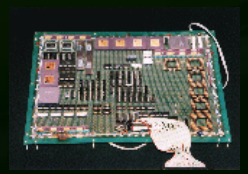
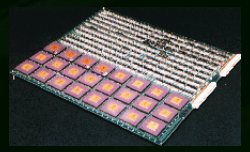
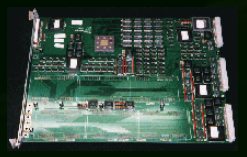
GPU Impact on AstroPhysics

GRAPE-4
1 Teraflop
Gordon Bell '96

GRAPE-6
48 Teraflops
Gordon Bell
2000, 01, 03

NVIDIA G80
91 Teraflops
Gordon Bell 2009

1989	1990	1991	1995	1998	1998	2007	2008
GRAPE-1	GRAPE-2	GRAPE-3	GRAPE-4	GRAPE-5	GRAPE-6	Tesla 8-series	Tesla 10-series
0.24 Gflops Single	0.04 GFlops Double	15 Gflops Single	0.64 GFlops Double	21.6 Gflops Single	30.8 Gflops Double	500 Gflops Single	1 TF Single 80 GF Double



NVIDIA GPUs at Supercomputing 09

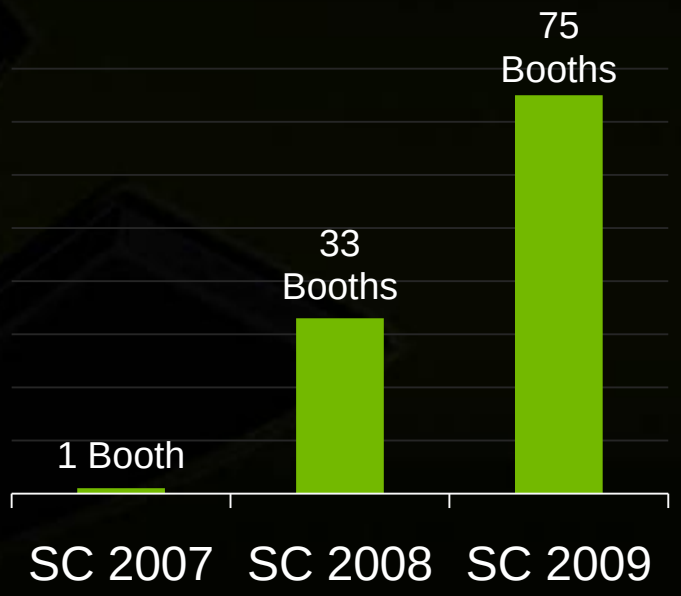
All Within 2 Years of launching CUDA GPUs

12% of Papers and nearly 50% of Posters used NVIDIA GPUs

Stellar speakers at NVIDIA Booth:

- *Jack Dongarra (UTenn)*
- *Klaus Schulten (UIUC)*
- *Ross Walker (SDSC)*
- *Jeff Vetter (ORNL)*
- *and many others*

1 in 4 Booths at SC 09 had NVIDIA GPUs



Prof Hamada won the Gordon Bell award for his CUDA-based work

Fermi: The Computational GPU

Performance

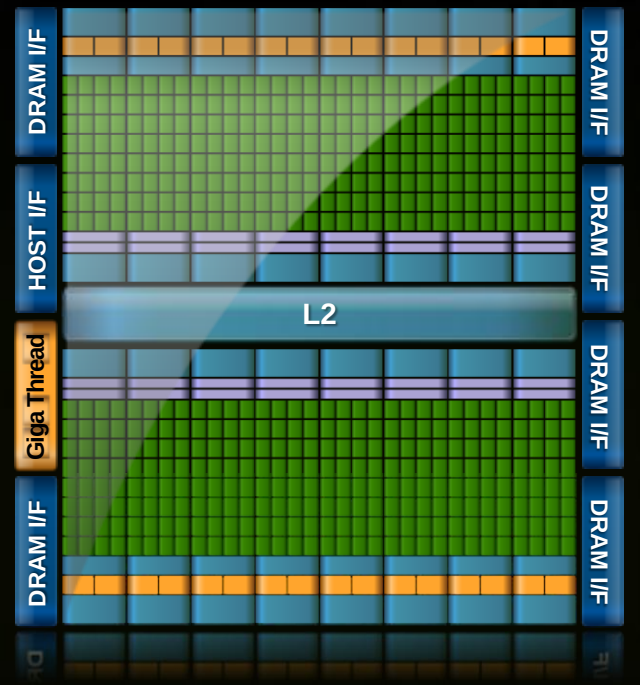
- 13x Double Precision of CPUs
- IEEE 754-2008 SP & DP Floating Point

Flexibility

- Increased Shared Memory from 16 KB to 64 KB
- Added L1 and L2 Caches
- ECC on all Internal and External Memories
- Enable up to 1 TeraByte of GPU Memories
- High Speed GDDR5 Memory Interface

Usability

- Multiple Simultaneous Tasks on GPU
- 10x Faster Atomic Operations
- C++ Support
- System Calls, printf support



http://www.nvidia.com/fermi

Tesla Personal Supercomputer

Tesla C2070
520-630 Gigaflop DP
6 GB Memory
ECC

Tesla C2050
520-630 Gigaflop DP
3 GB Memory
ECC

Tesla C1060
933 Gigaflop SP
78 Gigaflop DP
4 GB Memory

Performance

Large Datasets

8x Peak DP Performance

Mid-Range Performance

Q4	Q1	Q2	Q3	Q4
2009	**2010**			

Disclaimer: performance specification may change

Tesla Datacenter Systems

Performance

Tesla S1070-500
4.14 Teraflop SP
345 Gigaflop DP
4 GB Memory / GPU

Tesla S2050
2.1-2.5 Teraflop DP
3 GB Memory / GPU
ECC

Tesla S2070
2.1-2.5 Teraflop DP
6 GB Memory / GPU
ECC

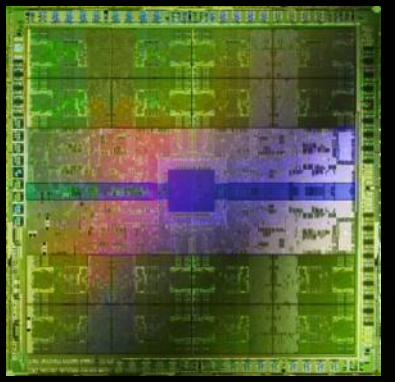
Large Datasets

8x Peak DP Performance

Mid-Range Performance

Q4	Q1	Q2	Q3	Q4
2009	**2010**			

Disclaimer: performance specification may change

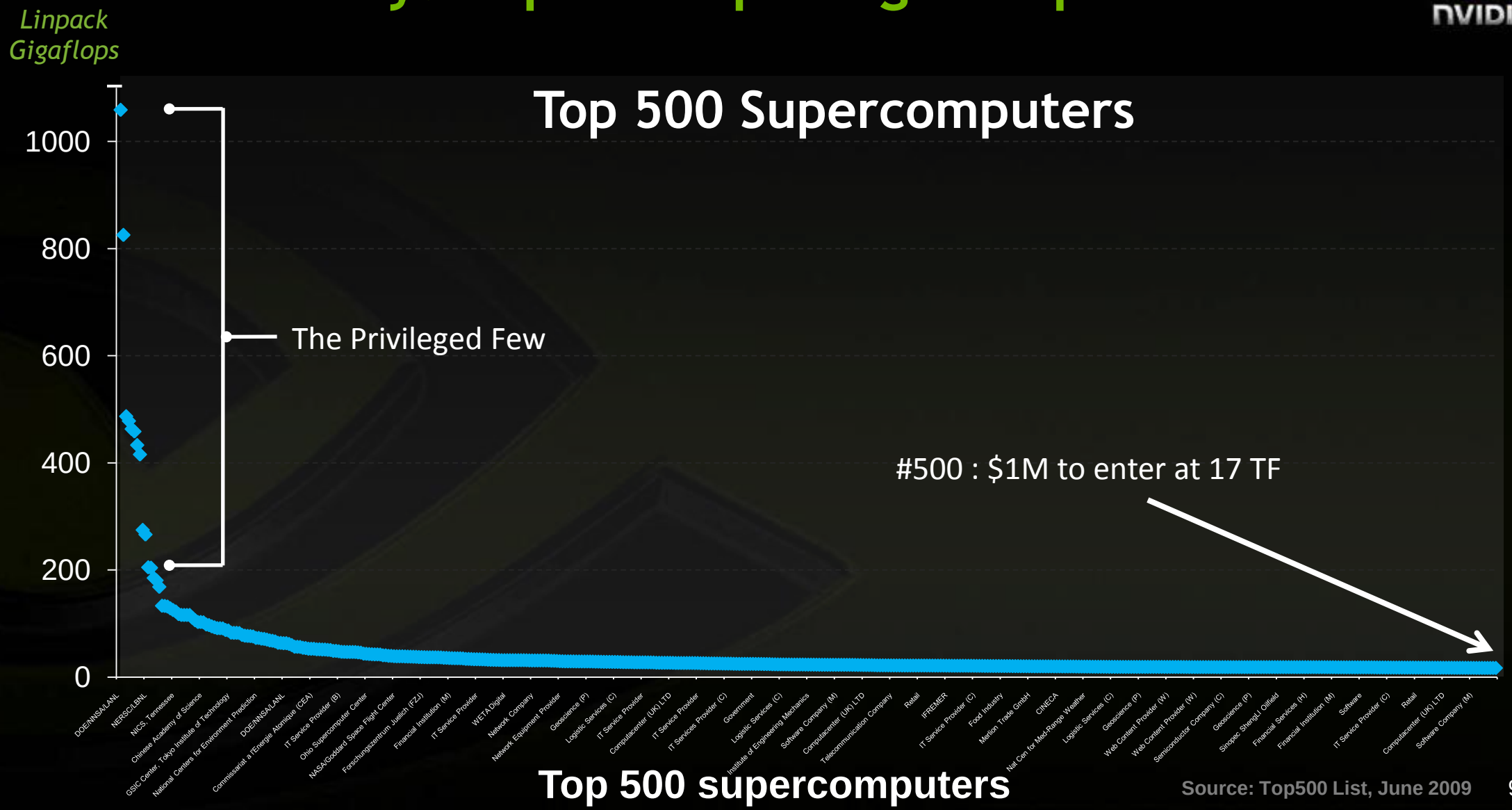


Fermi Transforms High Performance Computing

“I believe history will record Fermi as a significant milestone.”

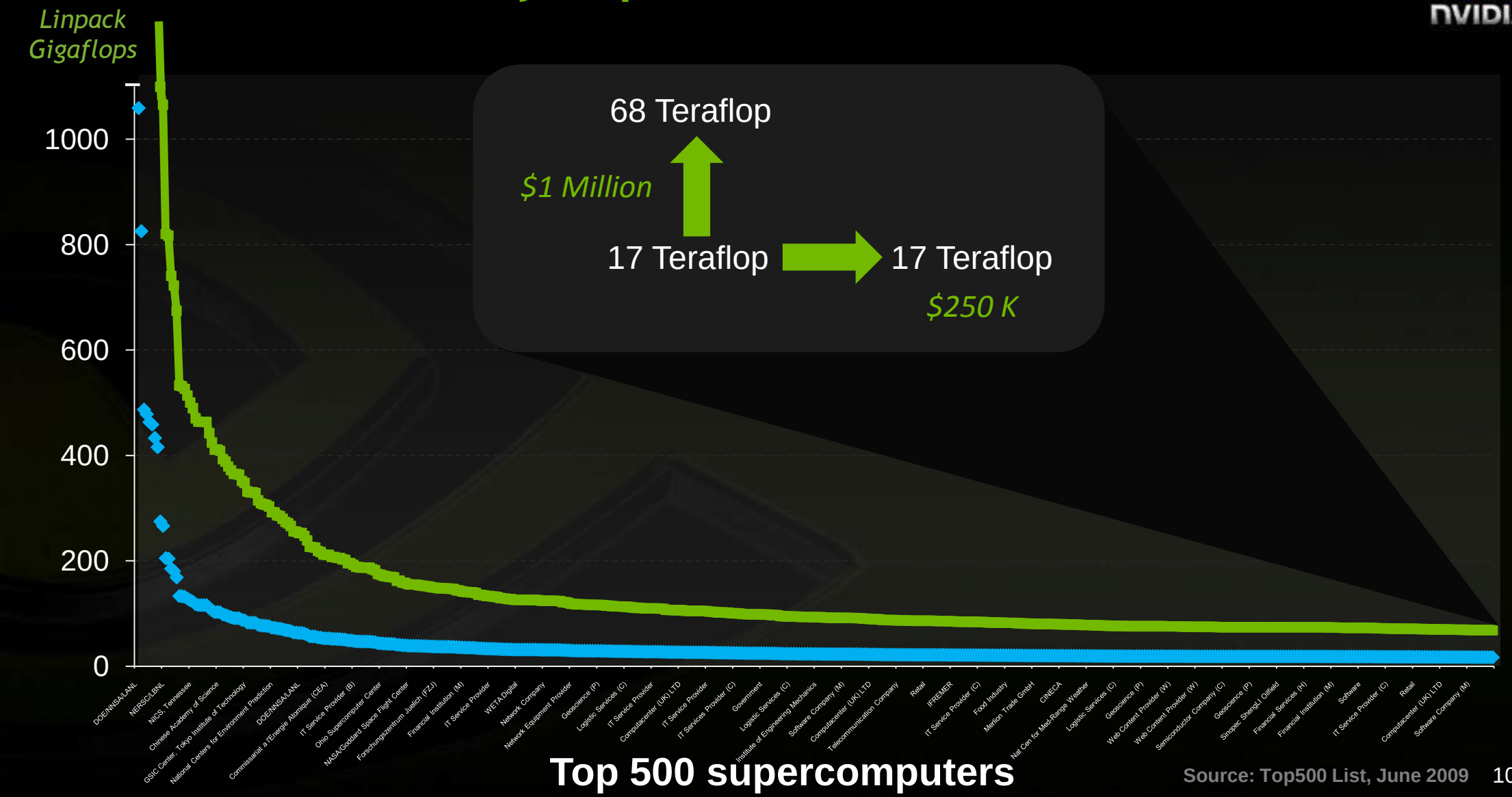
Dave Patterson
Director Parallel Computing Research Laboratory, U.C. Berkeley
Co-Author of Computer Architecture: A Quantitative Approach

Today: Supercomputing is Expensive



Source: Top500 List, June 2009

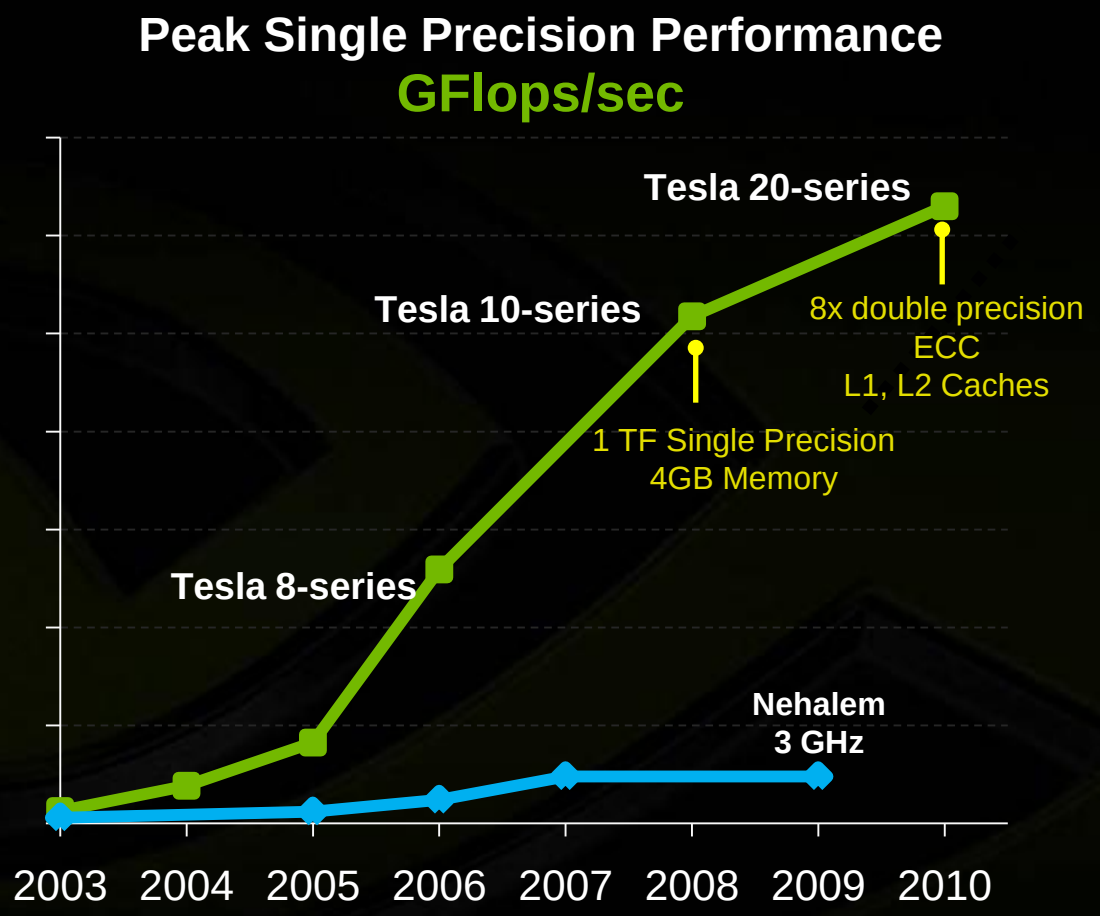
What if Every Top 500 Cluster Used Tesla GPUs

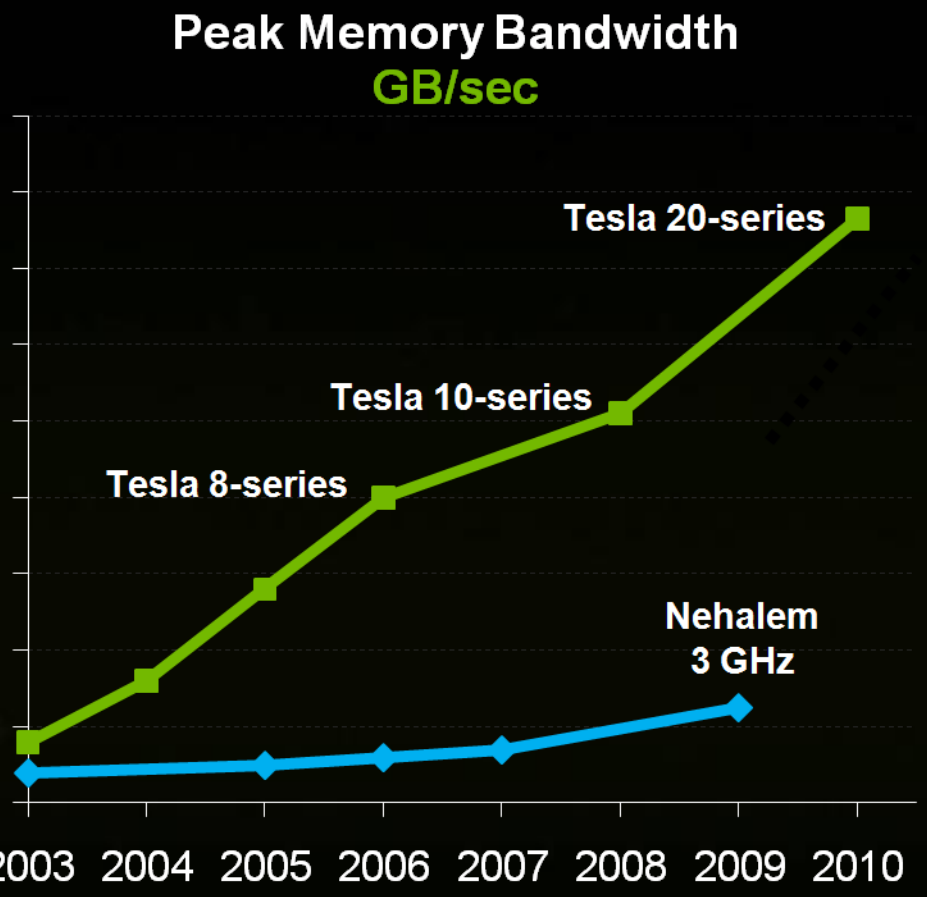


Top 500 supercomputers

Source: Top500 List, June 2009

The Performance Gap Widens Further



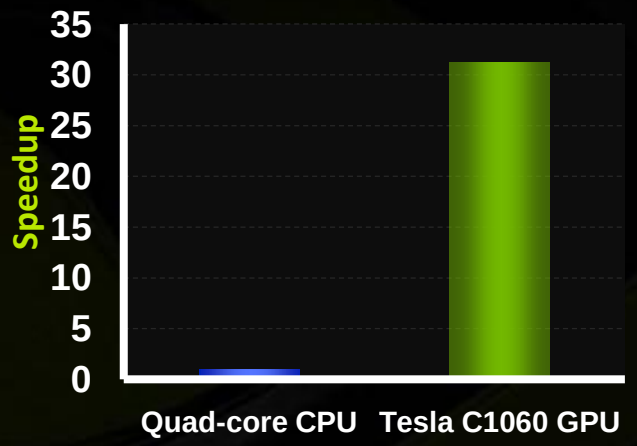


NVIDIA GPU
X86 CPU

Real Speedups in Real Applications

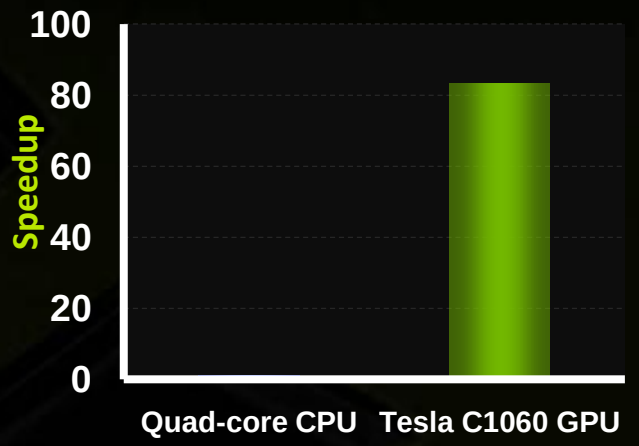
31 X

Seismic Processing
Wave Propagation



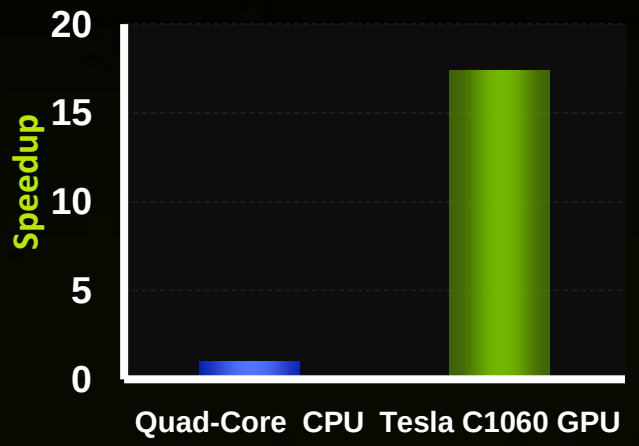
83 X

Financial Computing
Pricing



17 X

Molecular Dynamics
AMBER



Bloomberg: Bond Pricing

48 GPUs	**42x Lower Space**	2000 CPUs
$144K	**28x Lower Cost**	$4 Million
$31K / year	**38x Lower Power Cost**	$1.2 Million / year

GPU Parallel Computing Developer Eco-System

Numerical Packages

MATLAB
Mathematica
NI LabView
pyCUDA

Debuggers & Profilers

cuda-gdb
NV Visual Profiler
“Nexus” VS 2008
Allinea
TotalView

GPU Compilers

C
C++
Fortran
OpenCL
DirectCompute
Java
Python

Parallelizing Compilers

PGI Accelerator
CAPS HMPP
mCUDA
OpenMP

Libraries

BLAS
FFT
LAPACK
NPP
Video
Imaging

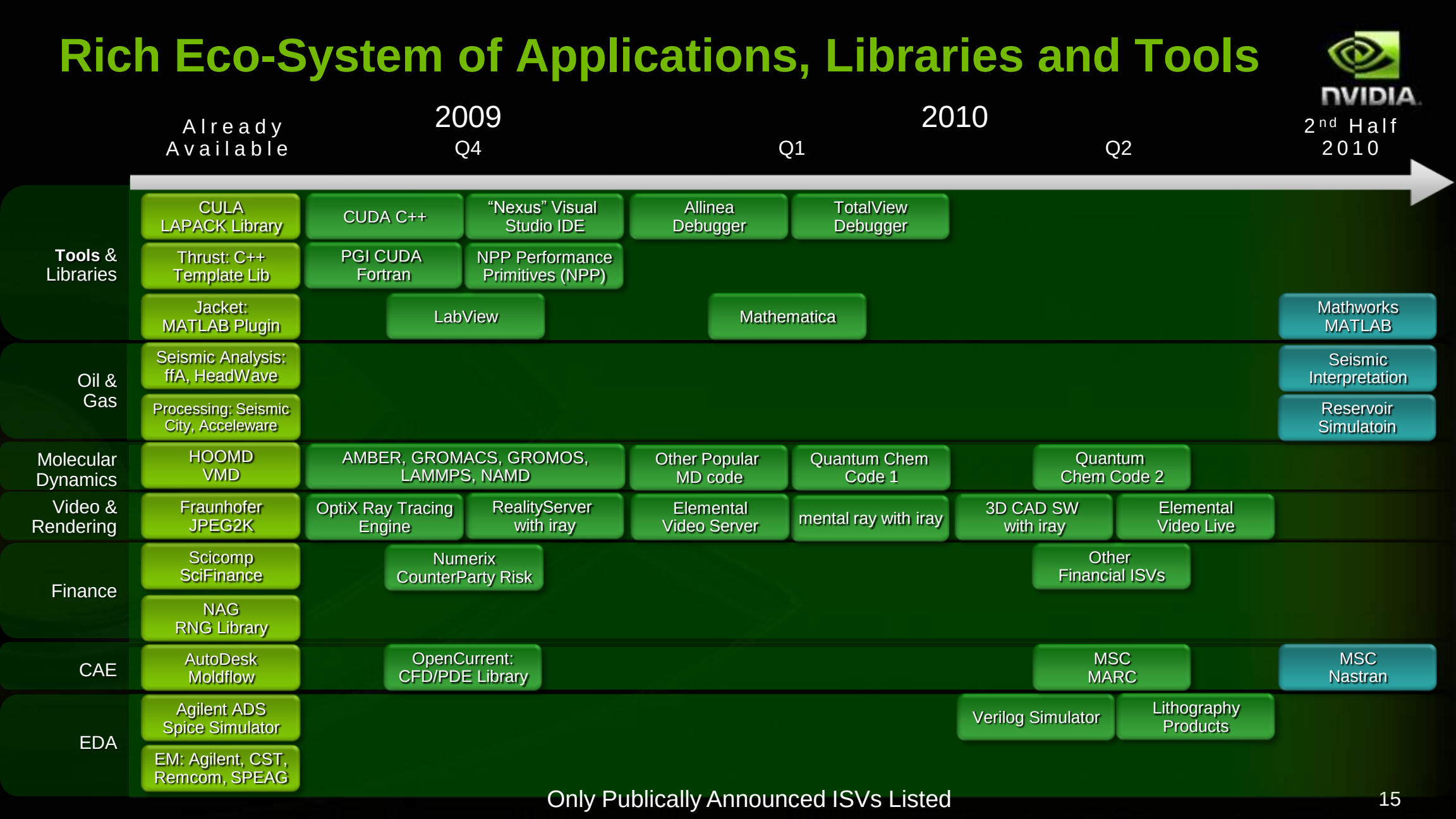
CUDA Consultants & Training

acceleware
GASS
STONE RIDGE TECHNOLOGY
ANEO
GPU Tech
TECH-X
CAPS
HPC
EM Photonics
SCALABLE GRAPHICS
WIPRO

Solution Providers

DELL
hp
Sun microsystems
CRAY
ASUS
SUPERMICRO
sgi
FUJITSU
BULL
APPRO
lenovo 联想
NEC

Rich Eco-System of Applications, Libraries and Tools

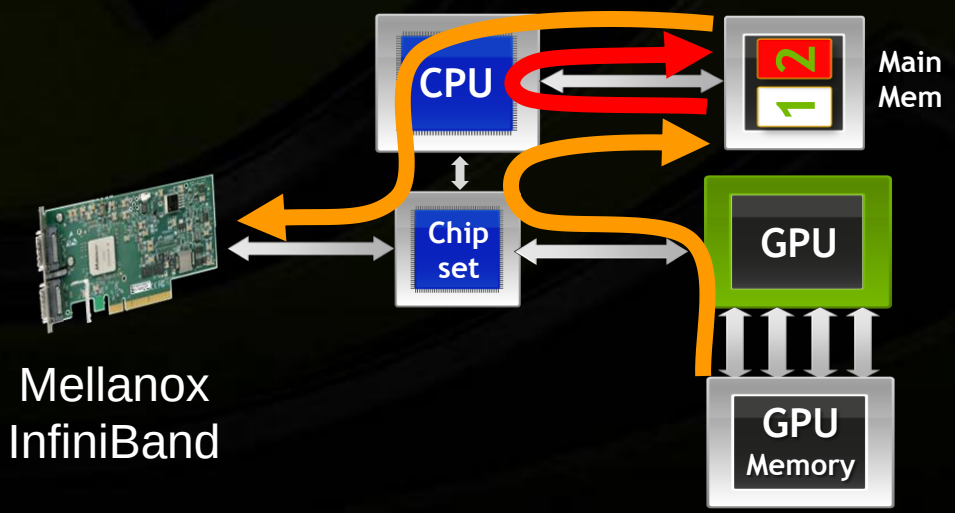


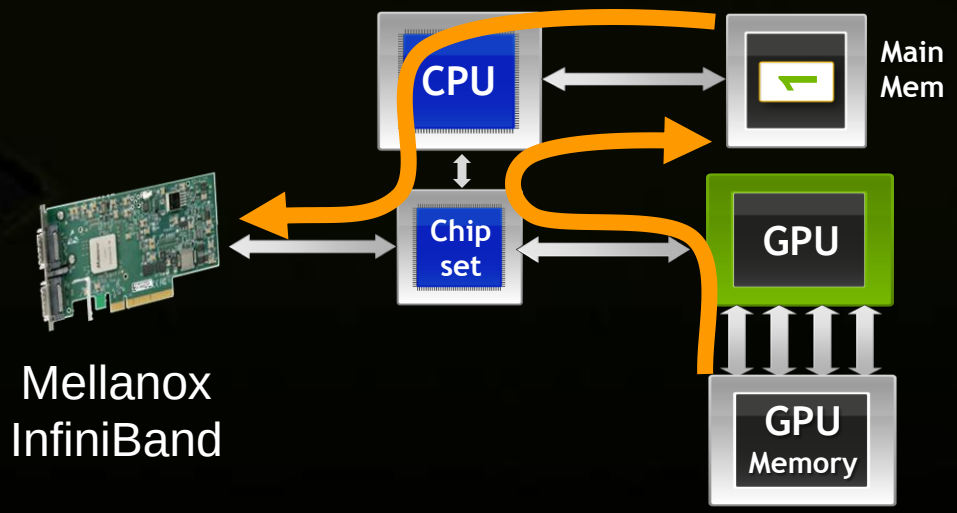
Only Publically Announced ISVs Listed

New GPU – InfiniBand Faster Communication

- Saves 30% in communication time
- Software available as CUDA C calls in Q2 2010
- Works with existing Tesla S1070, M1060 products

1. GPU Writes to Pinned Main Mem 1
2. ~~CPU Copies MMem 1 to MMem 2~~
3. INF Reads MMem 2





“We are heavy users of GPU for computing.

The conference confirmed to a large extent that we have made the right choice.

Fermi looks very good, and with Nexus the last reasons for not adopting GPUs over our complete portfolio are for the most part gone.

Excellent delivery.”

Olav Lindtjorn, Schlumberger

Mad Science Promotion:

Buy Tesla 10-series now, Be First in Line for Fermi

- **Special Promotional Pricing on Tesla 20-series Products**
  - **Available only till Jan 31, 2010**
- **Buy Tesla 10-series products today, and pre-order Fermi at promotional pricing**
- **Find out more at:**

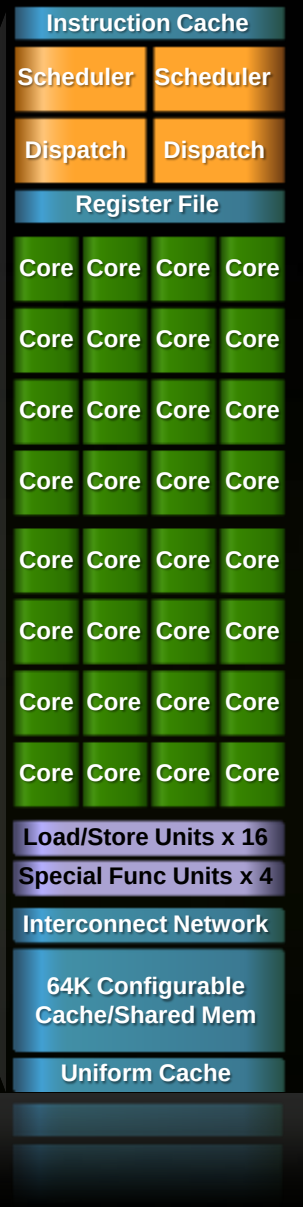
  http://www.nvidia.com/object/mad_science_promo.html

Next Gen CUDA GPU Architecture, Code Named “Fermi”

http://www.nvidia.com/fermi

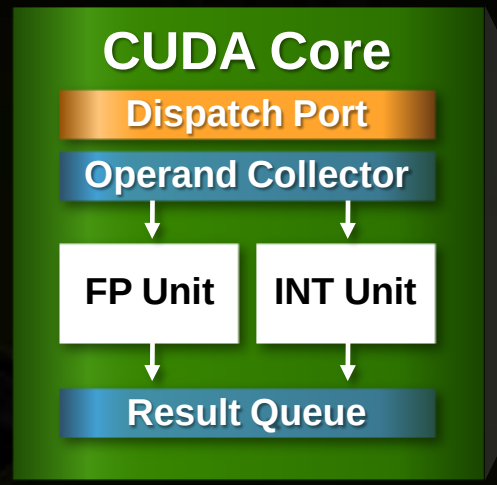
SM Architecture

- 32 CUDA cores per SM (512 total)
- 8x peak double precision floating point performance
  - 50% of peak single precision
- Dual Thread Scheduler
- 64 KB of RAM for shared memory and L1 cache (configurable)



CUDA Core Architecture

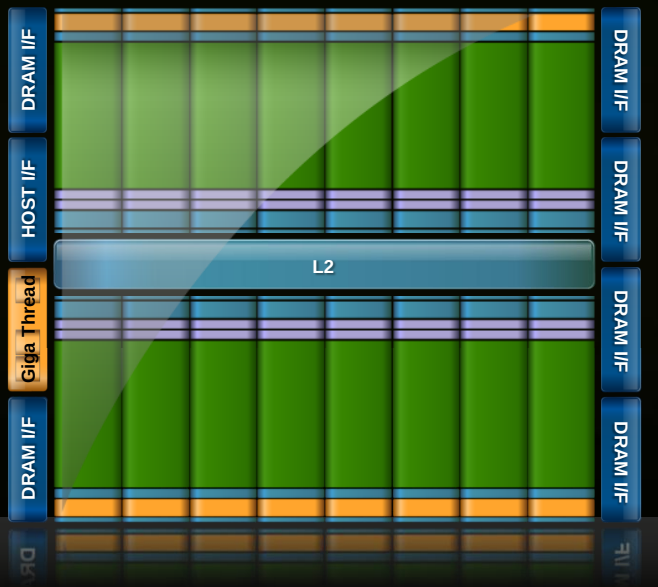
- New IEEE 754-2008 floating-point standard, surpassing even the most advanced CPUs
- Fused multiply-add (FMA) instruction for both single and double precision
- Newly designed integer ALU optimized for 64-bit and extended precision operations



Cached Memory Hierarchy

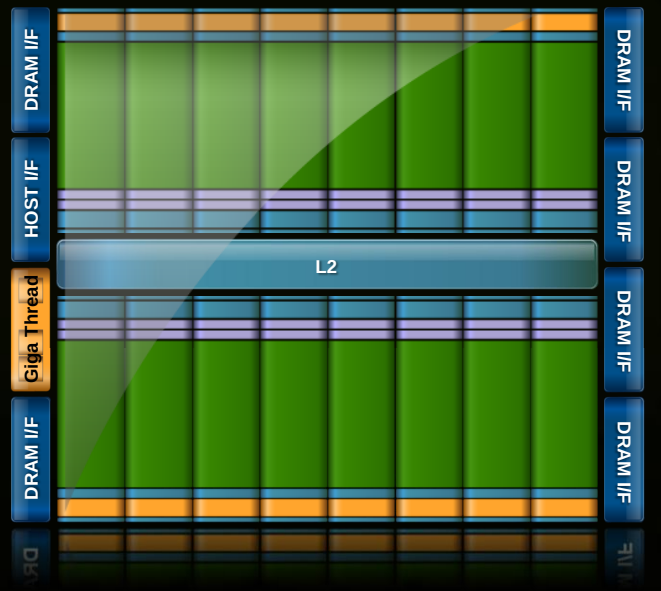
- First GPU architecture to support a true cache hierarchy in combination with on-chip shared memory
- L1 Cache per SM (32 cores)
  - Improves bandwidth and reduces latency
- Unified L2 Cache (768 KB)
  - Fast, coherent data sharing across all cores in the GPU

Parallel DataCache™ Memory Hierarchy



Larger, Faster Memory Interface

- GDDR5 memory interface
  - 2x speed of GDDR3
- Up to 1 Terabyte of memory attached to GPU
  - Operate on large data sets

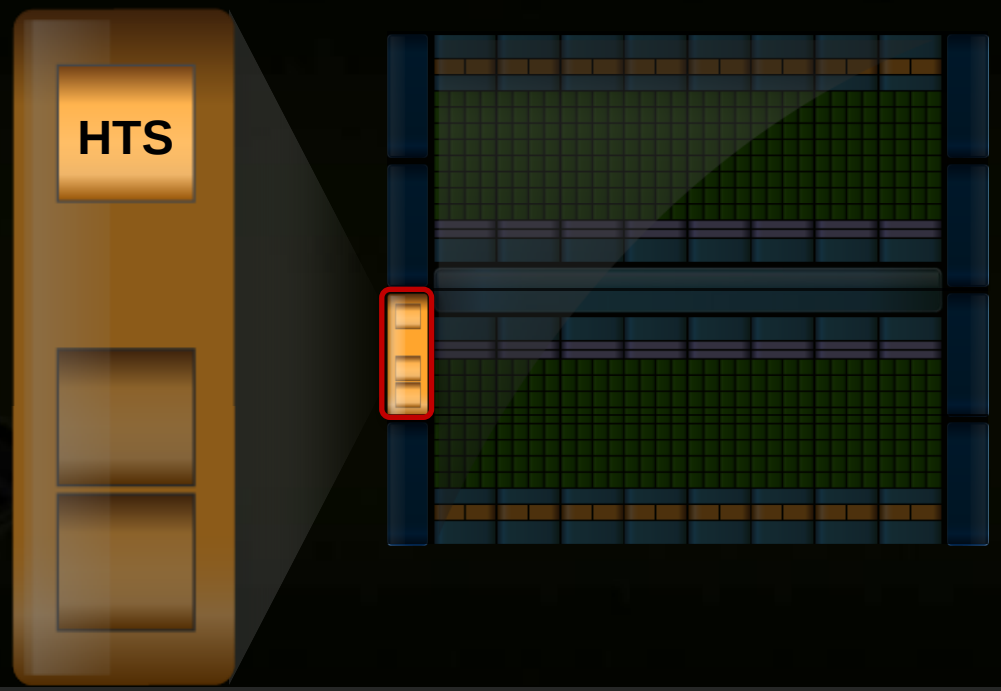


ECC

- **ECC protection for**
  - **DRAM**
    - ECC supported for GDDR5 memory
  - **All major internal memories are ECC protected**
    - Register file, L1 cache, L2 cache

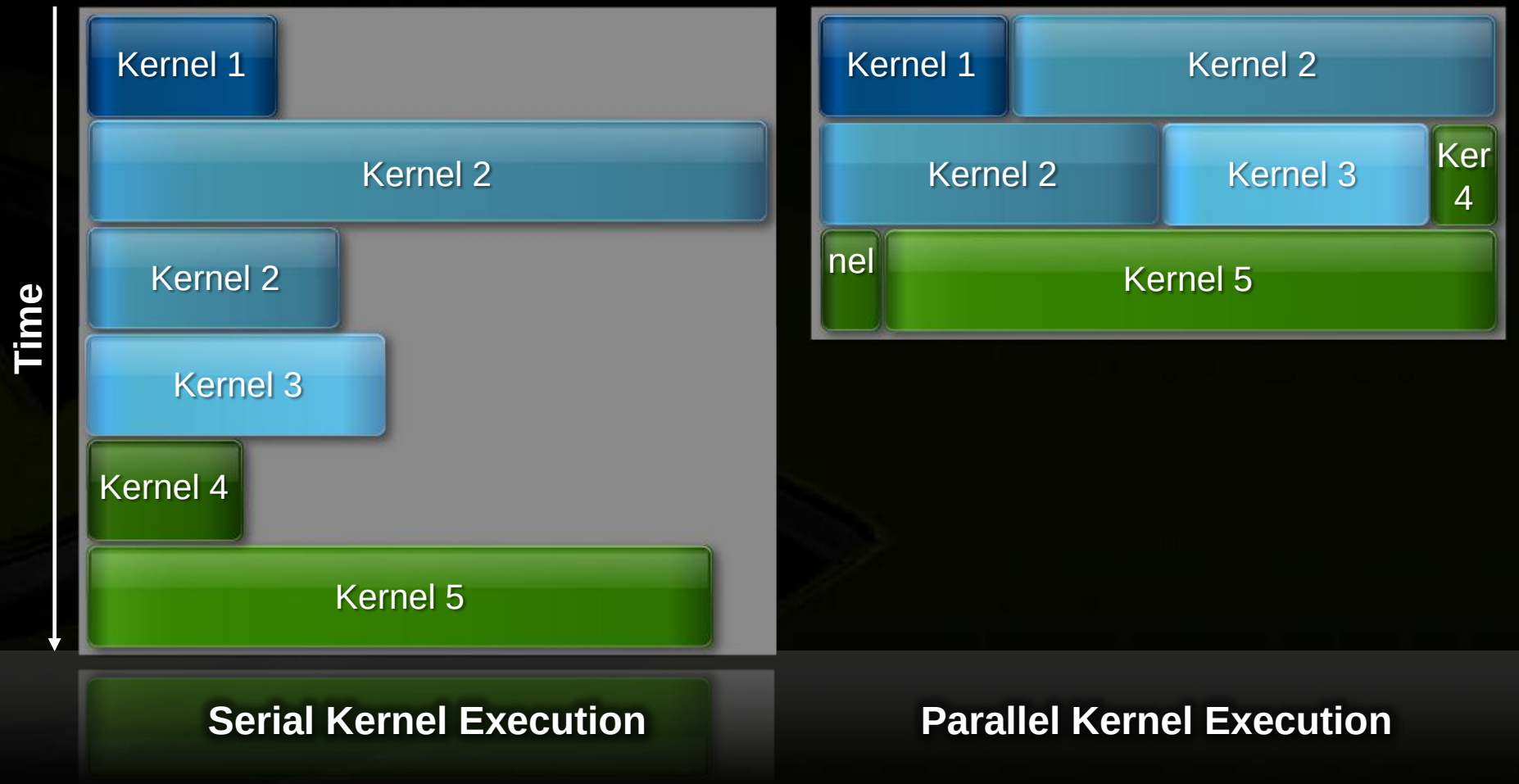
GigaThread™ Hardware Thread Scheduler (HTS)

- Hierarchically manages thousands of simultaneously active threads
- 10x faster application context switching
- Concurrent kernel execution



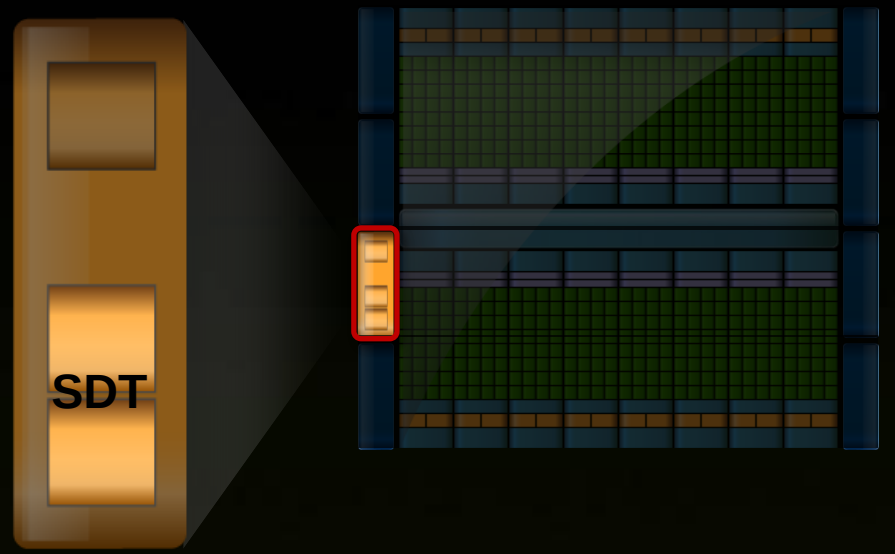
GigaThread Hardware Thread Scheduler

Concurrent Kernel Execution + Faster Context Switch

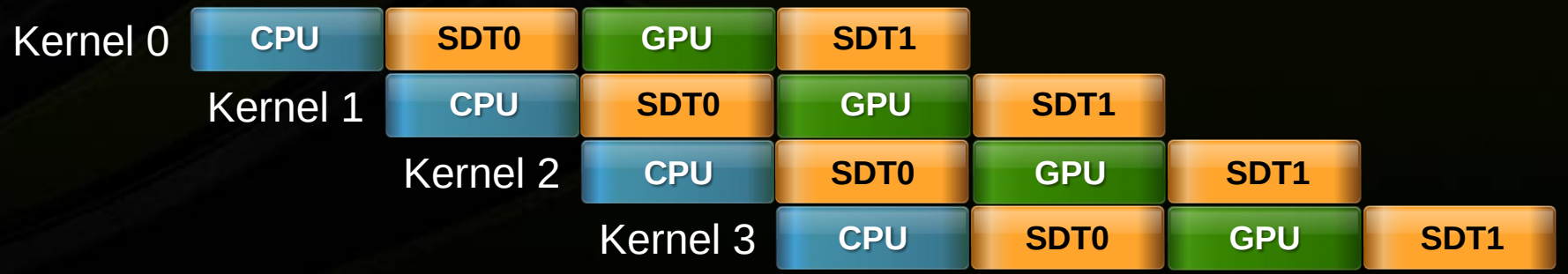


GigaThread Streaming Data Transfer Engine

- **Dual DMA engines**
  - Simultaneous CPU→GPU and GPU→CPU data transfer
  - Fully overlapped with CPU and GPU processing time



- **Activity Snapshot:**



Enhanced Software Support

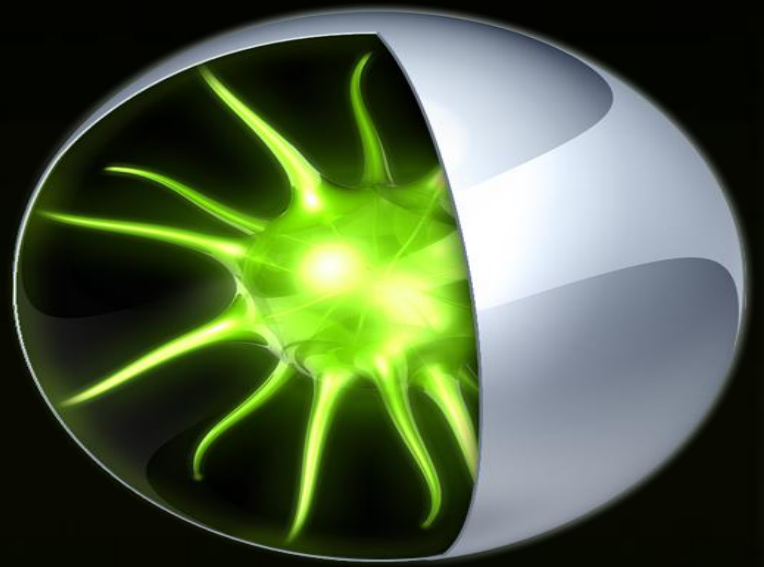
- Full C++ Support
  - Virtual functions
  - Try/Catch hardware support
- System call support
  - Support for pipes, semaphores, printf, etc
- Unified 64-bit memory addressing

NVIDIA Nexus IDE

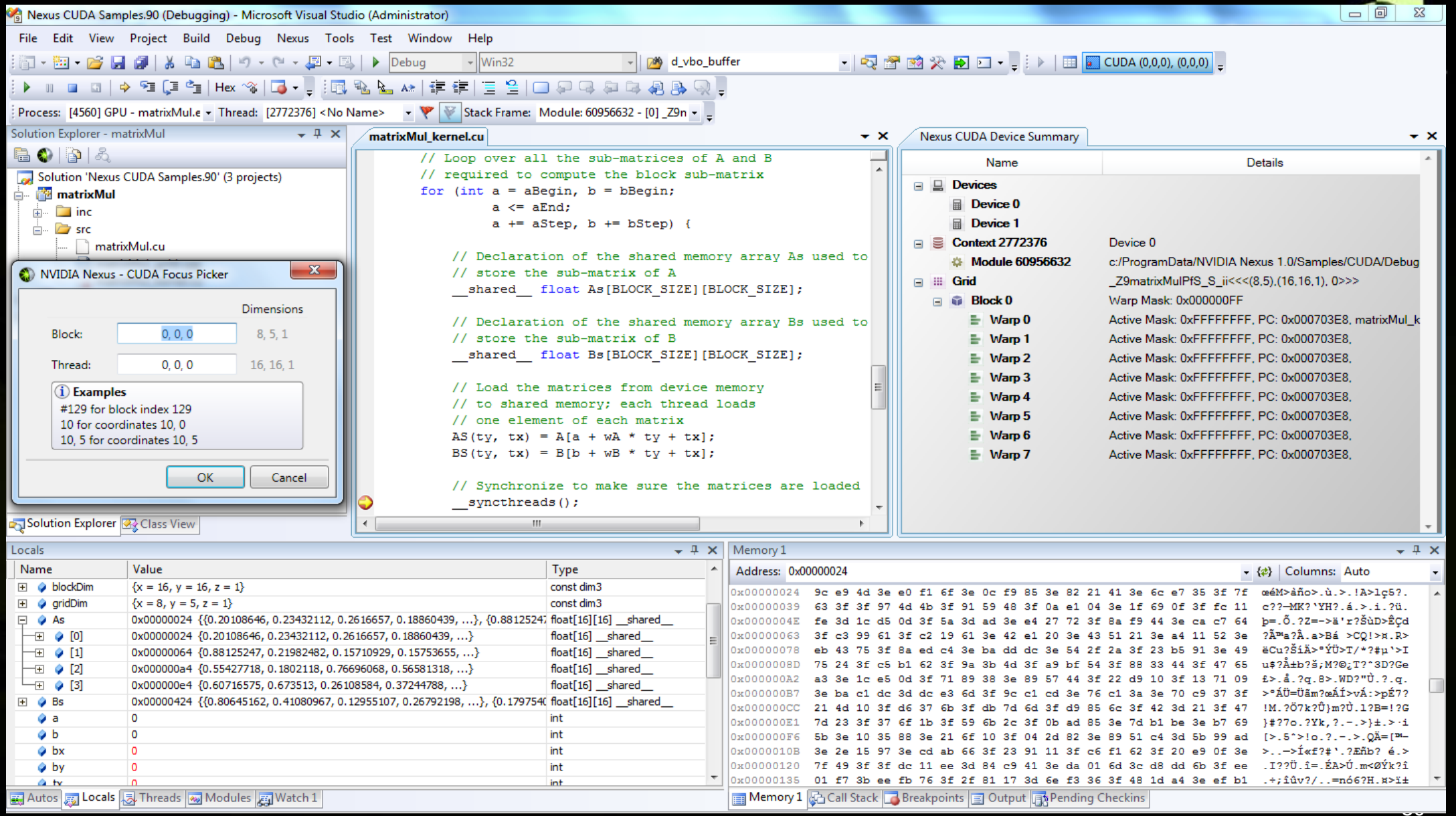
The industry's 1st IDE (Integrated Development Environment) for massively parallel applications

Accelerates co-processing (CPU + GPU) application development

Complete Visual Studio-integrated development environment



Microsoft Visual Studio PARTNER


Nexus CUDA Samples.90 (Debugging) - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Nexus Tools Test Window Help
Debug
Win32
d_vbo_buffer
CUDA (0,0,0), (0,0,0)
Process: [4560] GPU - matrixMul.e
Thread: [2772376] <No Name>
Stack Frame: Module: 60956632 - [0] _Z9n
Solution Explorer - matrixMul
Solution 'Nexus CUDA Samples.90' (3 projects)
matrixMul
inc
src
matrixMul.cu
NVIDIA Nexus - CUDA Focus Picker
Dimensions
Block:
0, 0, 0
8, 5, 1
Thread:
0, 0, 0
16, 16, 1
Examples
#129 for block index 129
10 for coordinates 10, 0
10, 5 for coordinates 10, 5
OK
Cancel
Solution Explorer
Class View
matrixMul_kernel.cu
// Loop over all the sub-matrices of A and B
// required to compute the block sub-matrix
for (int a = aBegin, b = bBegin;
a <= aEnd;
a += aStep, b += bStep) {
// Declaration of the shared memory array As used to
// store the sub-matrix of A
__shared__ float As[BLOCK_SIZE][BLOCK_SIZE];
// Declaration of the shared memory array Bs used to
// store the sub-matrix of B
__shared__ float Bs[BLOCK_SIZE][BLOCK_SIZE];
// Load the matrices from device memory
// to shared memory; each thread loads
// one element of each matrix
AS(ty, tx) = A[a + wA * ty + tx];
BS(ty, tx) = B[b + wB * ty + tx];
// Synchronize to make sure the matrices are loaded
__syncthreads();
Nexus CUDA Device Summary
Name
Details
Devices
Device 0
Device 1
Context 2772376
Device 0
Module 60956632
c:/ProgramData/NVIDIA Nexus 1.0/Samples/CUDA/Debug
Grid
_Z9matrixMulPfS_S_ii<<<(8,5),(16,16,1), 0>>>
Block 0
Warp Mask: 0x000000FF
Warp 0
Active Mask: 0xFFFFFFFF, PC: 0x000703E8, matrixMul_k
Warp 1
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 2
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 3
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 4
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 5
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 6
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Warp 7
Active Mask: 0xFFFFFFFF, PC: 0x000703E8,
Locals
Name
Value
Type
blockDim
{x = 16, y = 16, z = 1}
const dim3
gridDim
{x = 8, y = 5, z = 1}
const dim3
As
0x00000024 {{0.20108646, 0.23432112, 0.2616657, 0.18860439, ...}, {0.8812524
float[16][16] __shared__
[0]
0x00000024 {0.20108646, 0.23432112, 0.2616657, 0.18860439, ...}
float[16] __shared__
[1]
0x00000064 {0.88125247, 0.21982482, 0.15710929, 0.15753655, ...}
float[16] __shared__
[2]
0x000000a4 {0.55427718, 0.1802118, 0.76696068, 0.56581318, ...}
float[16] __shared__
[3]
0x000000e4 {0.60716575, 0.673513, 0.26108584, 0.37244788, ...}
float[16] __shared__
Bs
0x00000424 {{0.80645162, 0.41080967, 0.12955107, 0.26792198, ...}, {0.1797540
float[16][16] __shared__
a
0
int
b
0
int
bx
0
int
by
0
int
Autos
Locals
Threads
Modules
Watch 1
Memory 1
Address: 0x00000024
Columns: Auto
0x00000024 9c e9 4d 3e e0 f1 6f 3e 0c f9 85 3e 82 21 41 3e 6c e7 35 3f 7f
0x00000039 63 3f 3f 97 4d 4b 3f 91 59 48 3f 0a e1 04 3e 1f 69 0f 3f fc 11
0x0000004E fe 3d 1c d5 0d 3f 5a 3d ad 3e e4 27 72 3f 8a f9 44 3e ca c7 64
0x00000063 3f c3 99 61 3f c2 19 61 3e 42 e1 20 3e 43 51 21 3e a4 11 52 3e
0x00000078 eb 43 75 3f 8a ed c4 3e ba dd dc 3e 54 2f 2a 3f 23 b5 91 3e 49
0x0000008D 75 24 3f c5 b1 62 3f 9a 3b 4d 3f a9 bf 54 3f 88 33 44 3f 47 65
0x000000A2 a3 3e 1c e5 0d 3f 71 89 38 3e 89 57 44 3f 22 d9 10 3f 13 71 09
0x000000B7 3e ba c1 dc 3d dc e3 6d 3f 9c c1 cd 3e 76 c1 3a 3e 70 c9 37 3f
0x000000CC 21 4d 10 3f d6 37 6b 3f db 7d 6d 3f d9 85 6c 3f 42 3d 21 3f 47
0x000000E1 7d 23 3f 37 6f 1b 3f 59 6b 2c 3f 0b ad 85 3e 7d b1 be 3e b7 69
0x000000F6 5b 3e 10 35 88 3e 21 6f 10 3f 04 2d 82 3e 89 51 c4 3d 5b 99 ad
0x0000010B 3e 2e 15 97 3e cd ab 66 3f 23 91 11 3f c6 f1 62 3f 20 e9 0f 3e
0x00000120 7f 49 3f 3f dc 11 ee 3d 84 c9 41 3e da 01 6d 3c d8 dd 6b 3f ee
0x00000135 01 f7 3b ee fb 76 3f 2f 81 17 3d 6e f3 36 3f 48 1d a4 3e ef b1
Memory 1
Call Stack
Breakpoints
Output
Pending Checkins

Tesla 20-Series Products

Tesla C-Series Workstation GPUs

	C2050	C2070
Processors	1 Tesla 20-series GPU	
Double precision performance	520-630 Gigaflops DP	
GPU Memory	3 GB	6 GB
Memory Interface	GDDR5	
System I/O	PCIe x16 Gen2	
Power	190 W (typical) 225 W (max)	

Disclaimer: specification subject to change
Available memory may be lower with ECC

Tesla S-Series 1U GPU Systems

	S2050	S2070
Processors	4 Tesla 20-series GPUs	
Double precision performance	2.1 – 2.5 Teraflops DP	
GPU Memory	3 GB / GPU	6 GB / GPU
Memory Interface	GDDR5	
System I/O	2x PCIe x16 Gen2 (optional x8 available)	
Power	900 W (typical) 1200 W (max)	

Disclaimer: specification subject to change
Available memory may be lower with ECC

Tesla Data Center Software Tools

- System management tools
  - nv-smi Thermal, Power, and System monitoring software
  - Integration of nv-smi with Ganglia system monitoring software
  - IPMI support for Tesla M-based hybrid CPU-GPU servers

- Clustering software
  - Platform Computing : *Cluster Manager*
  - Penguin Computing : *Scyld*
  - Cluster Corp : *Rocks Roll*
  - Altair*: PBS Works*

Tesla Compute Cluster (TCC) Driver
Enables Windows HPC on Tesla

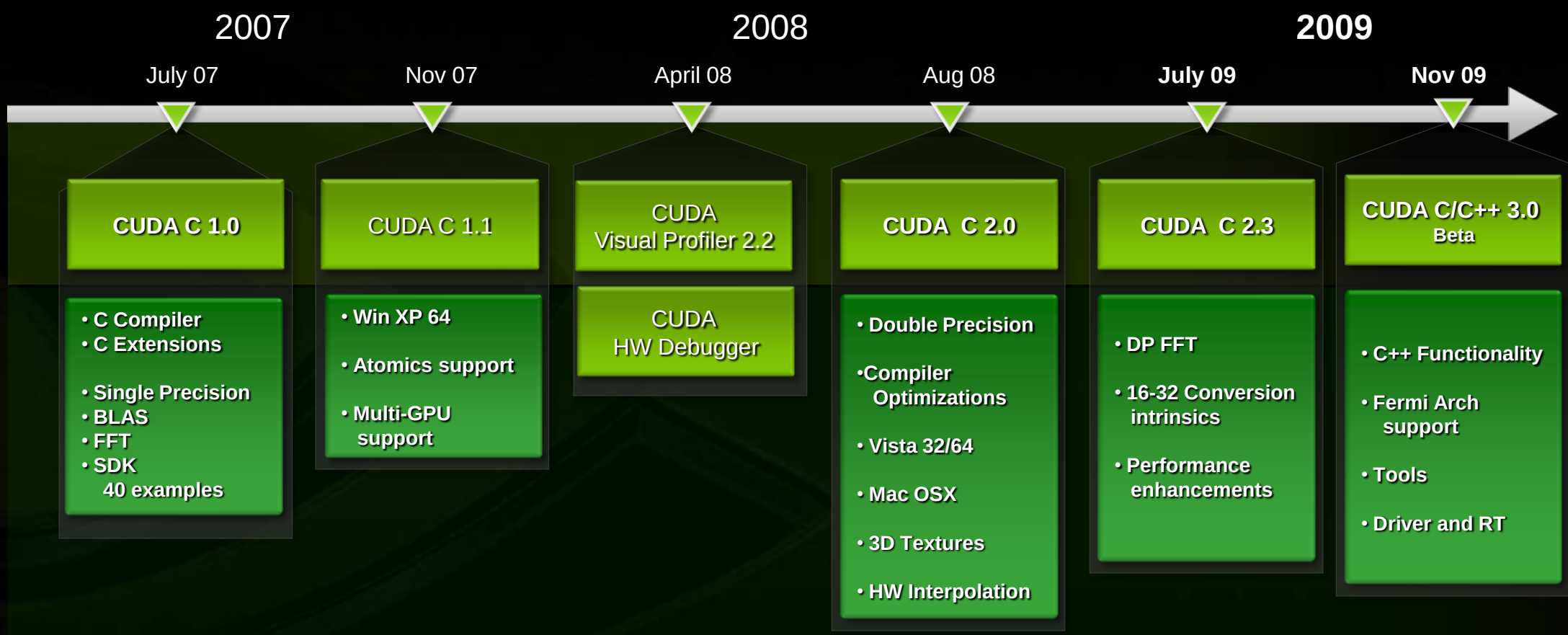
- Enables Tesla without a NVIDIA graphics card with
  - Windows 7, Server 2008 R2, Windows Vista, Server 2008
  - Only Tesla 8-series, 10-series and 20-series supported
  - Only works with CUDA
    - Does not support OpenGL and DirectX
  - Available in beta now, release in Jan 2010

- Enables the following features under Windows with CUDA
  - RDP (Remote Desktop)
  - Launch CUDA applications via Windows Services
  - No Windows Timeout issues
  - No penalty on launch overhead
  - KVM-over-IP enabled (CPU Server on-board graphics chipset enabled)

CUDA C/C++ Update

December ‘09

Evolution of C / C++ Programming Environment



CUDA C/C++ Leadership

CUDA C++ Language Functionality

- C++ Class Inheritance
- C++ Template Inheritance

(Productivity)

Tools

- Debugger support for CUDA Driver API
- ELF support (cubin format deprecated)
- CUDA Memory Checker (cuda-gdb feature)
- Fermi: Fermi HW Debugger support in cuda-gdb
- Fermi: Fermi HW Profiler support in CUDA Visual Profiler

Fermi Architecture Support

- Native 64-bit GPU support
- Generic Address space / Memheap rewr
- Multiple Copy Engine support
- ECC reporting
- Concurrent Kernel Execution

Architecture supported in SW now

CUDA Driver & CUDA C Runtime

- CUDA Driver / Runtime Buffer interoperability (Stateless CUDART)
- Separate emulation runtime
- Unified interoperability API for Direct3D and OpenGL
- OpenGL texture interoperability
- Fermi: Direct3D 11 interoperability

2009

Nov 09

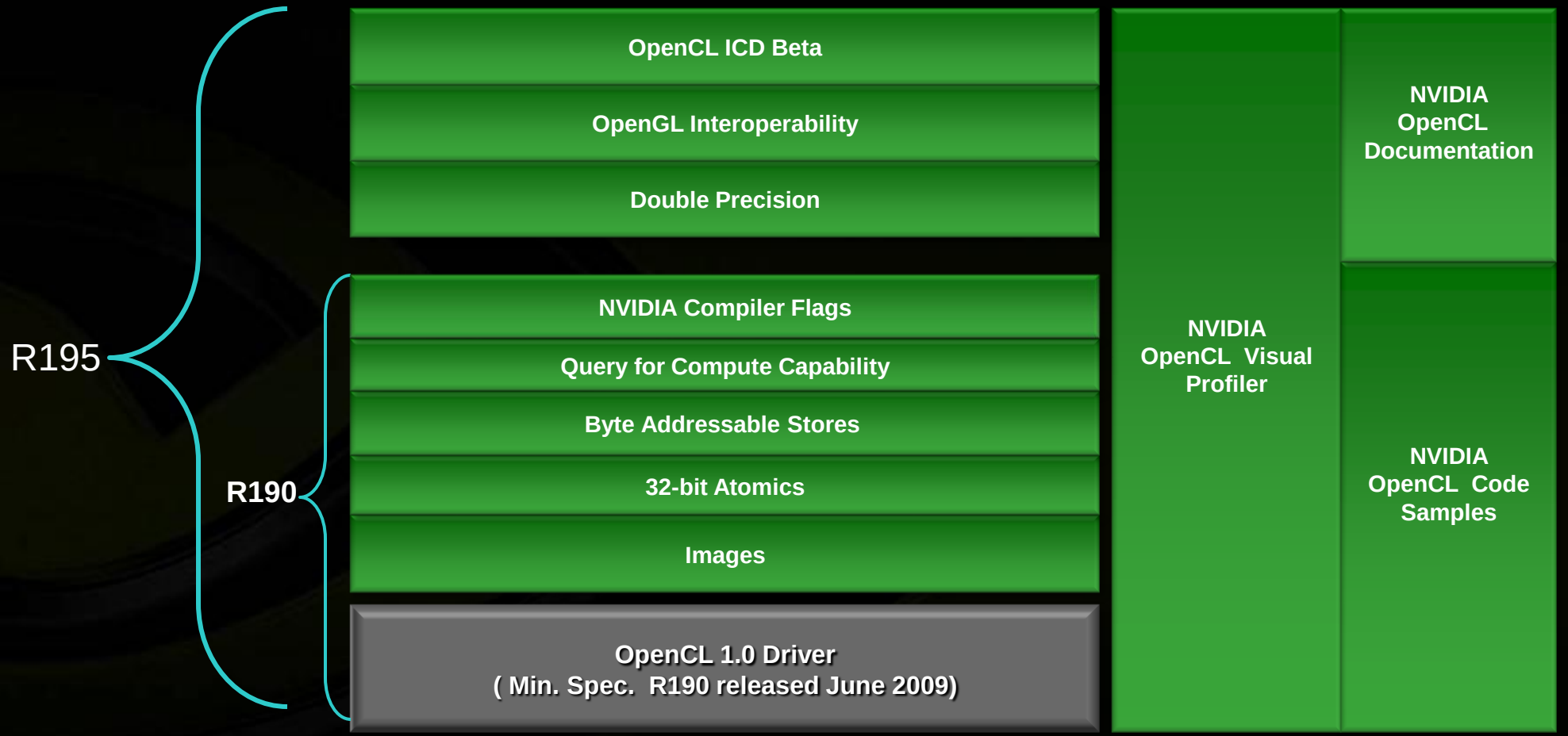
CUDA C/C++ 3.0
Beta

- **C++ Functionality**
- **Fermi Arch support**
- **Tools**
- **Driver and RT**


OpenCL
OpenCL Update
December ‘09
NVIDIA

NVIDIA OpenCL Leadership

NVIDIA OpenCL Leadership



CUDA: Most Widely Taught Parallel Processing Model

269 Universities Teach CUDA Parallel Programming Model