高性能计算II(B)

基于图形处理器的并行计算及CUDA编程

Ying Liu, Associate Prof., Ph.D

School of Computer and Control, University of Chinese Academy of Sciences

Key Lab of Big Data & Knowledge Management, Chinese Academy of Sciences

Welcome

- Education
  - 1999/07, Computer Science, BS., Peking University
  - 2001/12, Computer Engineering, MS., Northwestern University, USA
  - 2005/06, Computer Engineering, Ph.D., Northwestern University, USA
- Work Experience
  - 2005/06 – 2005/11，Research Associate, Northwestern University, USA
  - 2006/01 – Present，Associate Professor, University of Chinese Academy of Sciences

Welcome

- Research interests
  - Data mining
  - High performance computing
  - Business Intelligence
  - Performance evaluation
- Contact: yingliu@ucas.ac.cn

Welcome

- Class schedule: Mon Wed 10:30 – 12:10pm

  教1-406
- Class website: http://www2.ucas.ac.cn
- Teaching assistant

Name	Email
Yang, Jiajun	cudaucas@163.com

Objectives of This Course

- Introduce a new emerging paradigm
- Present parallel programming principles, the parallelism models
- Introduce successful cases
- Provide students with knowledge and hands-on experience in developing multi-threaded code for GPUs using CUDA
- Enhance independent research capability

Syllabus (Tentative)

- Introduction
- Parallel Computing
- CUDA Programming Model
- CUDA Memory
- CUDA Threads
- Performance Optimization
- Case Study: Typical Examples
- Advanced Topics
- Final Project Presentation & Conclusion

References

- Websites
  - http://www.nvidia.com/object/cuda_home.html
  - http://www.mpi-forum.org/
  - http://www.openmp.org/
- Documentation
  - NVIDIA CUDA Programming Guide (v5.0), NVIDIA

References

- Books
  - CUDA范例精解--通用GPU编程(英文影印版)，Jason Sanders, Edward Kandrot,清华大学出版社, 2010
  - GPU高性能运算之CUDA，张舒，褚艳利，中国水利水电出版社，2009
  - Programming Massively Parallel Processors: A Hands-on Approach, David B. Kirk, Wen-mei W. Hwu, Morgan Kaufmann, 2010
  - Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, Introduction to Parallel Computing (2nd Edition), Addison Wesley, 2003
  - Barry Wilkinson, Michael Allen, Parallel Programming 2nd edition, Pearson Education (Prentice Hall)
  - Quinn, Michael J., Parallel Programming in C with MPI and OpenMP McGraw-Hill Science, 2004

Grading Scheme

- Homework assignments (30%)
  - 2 assignments (3-4 students/group)
- Course Project (40%)
  - Group project (3-4 students/group)
  - Select a topic from a recommended topic list
  - Implement it using CUDA on GPU
  - Hand in a project report
  - To be evaluated in technical innovation, thoroughness of the work, clarity of presentation
- Final Exam (30%)
  - Open book, open notes

Policies

- Students are expected to attend all classes
- No late homework will be accepted
- All work must be efforts of your own

University CUDA Courses (2010)

加州理工学院	美国伊利诺伊大学厄本那—香槟分校	斯图加特大学
斯坦福大学	美国北卡罗来纳州立大学	马里兰州大学
美国哈佛大学	美国东北大学	普渡大学
苏黎世理工学院	威斯康星大学	犹他大学
国佐治亚理工学院	美国俄勒冈州立大学	德国埃尔兰根大学
麻省理工学院	宾夕法尼亚大学	京都大学
印度国际信息技术学院	加拿大滑铁卢大学	瑞典兰德大学
印度理工学院（德里）	加拿大麦吉尔大学	东京大学
美国杜克大学	纽约州立大学	维也纳技术大学
法国国立信息与自动化研究院	华盛顿大学	威廉姆斯大学
澳大利亚西澳大学	弗吉尼亚大学	圣克拉拉大学
墨西哥蒙特雷理工大学	美国北卡罗来纳大学	美国格洛夫城市学院
约翰霍普金斯大学	美国南卡来罗纳大学	
肯特州立大学	美国爱荷华大学	

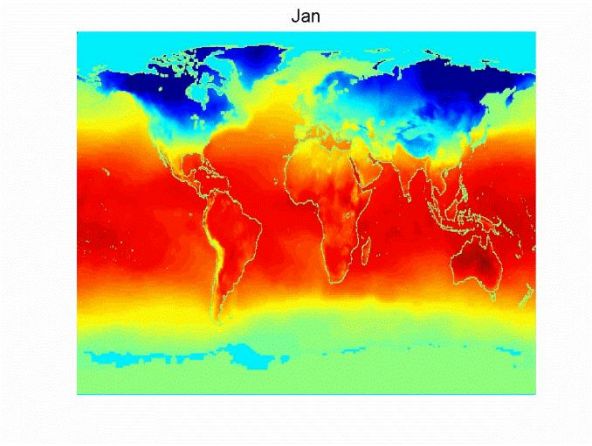
CUDA Teaching Center

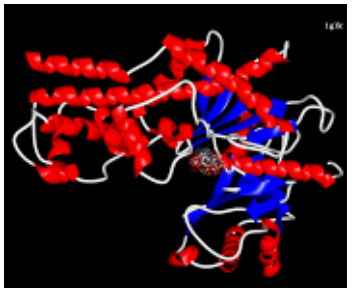
Outline

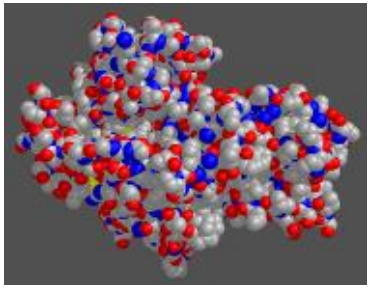
- CUDA Introduction
  - High Performance Computing
  - GPU
  - Why CUDA?
  - What is CUDA?
  - Successful Cases
  - Personal Supercomputer

Why High Performance Computing?

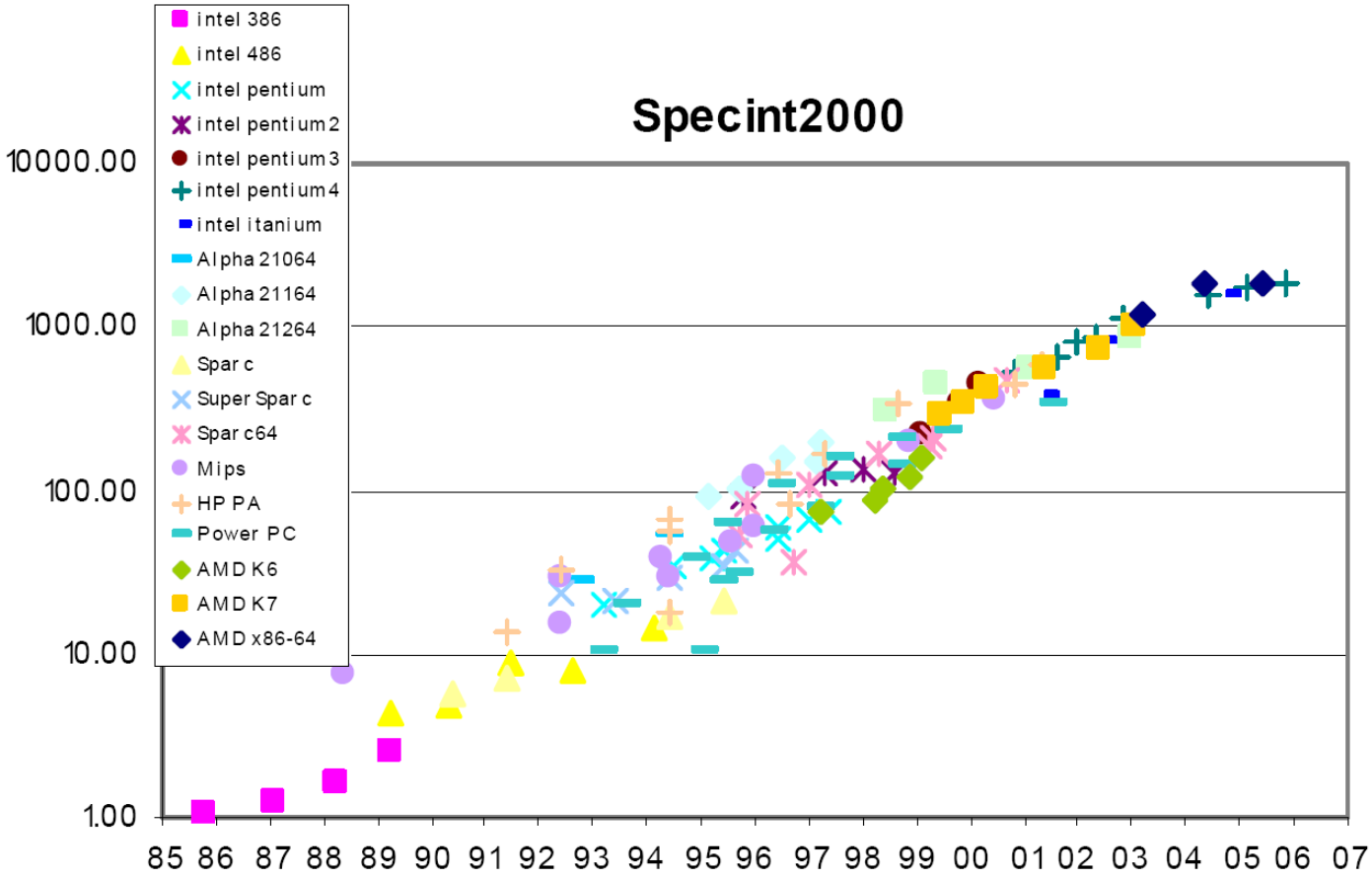
- Lots of data being collected in commercial and scientific world, massive data sets
- Strong competitive pressure to extract and use the information from the data, e.g.
  - Climate simulation
  - Astrophysics
  - Molecular biology







Performance of Single-Core CPU



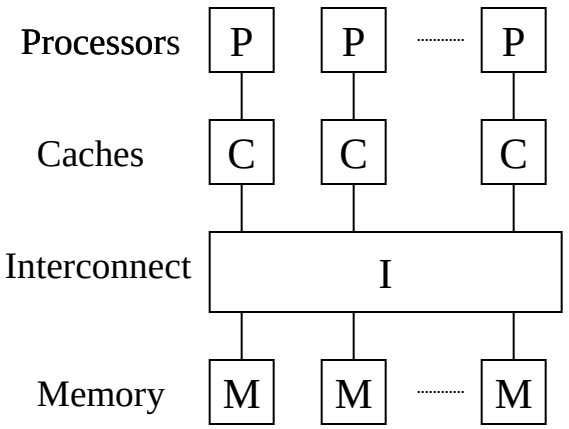
Why High Performance Computing?

- Computing power limits are being approached
- Memory limitations of sequential computers cause sequential algorithms to make multiple expensive I/O passes over data
- Lacking of capability to solve bigger and more realistic distributed applications

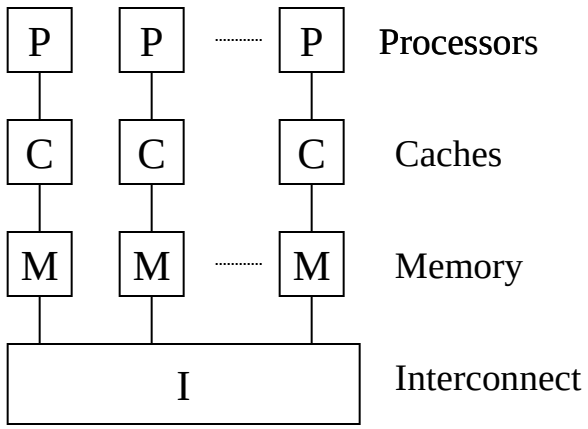
Accelerate the computation
Use more memory from multiple machines

Architectures

- Shared Address Space
  - All processors share a single global address space
  - Single address space facilitates a simple programming model
  - Examples: SGI Origin 3000, IBM SP2



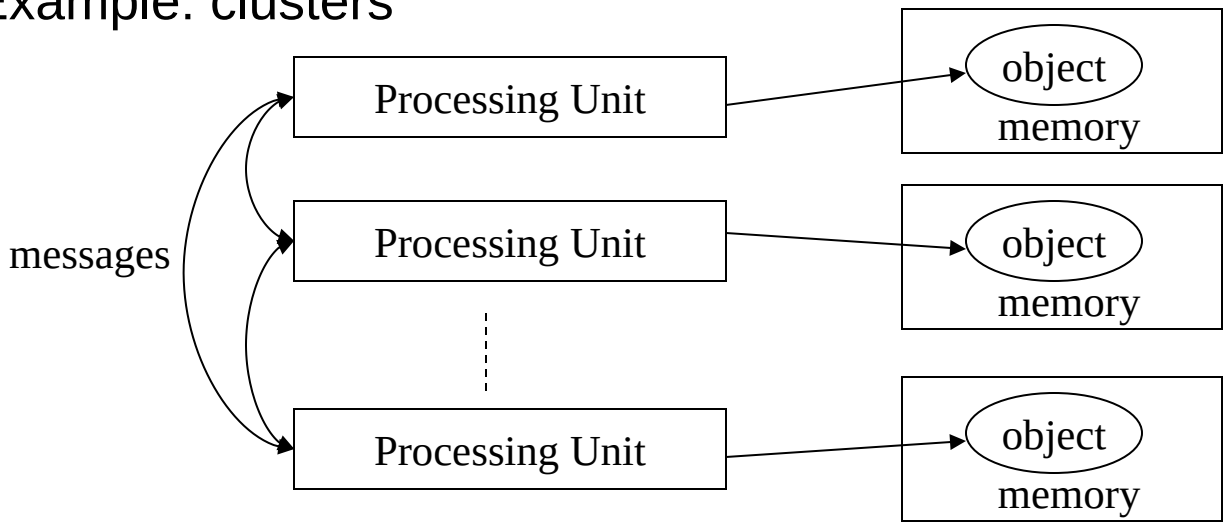
(a) UMA



(b) NUMA

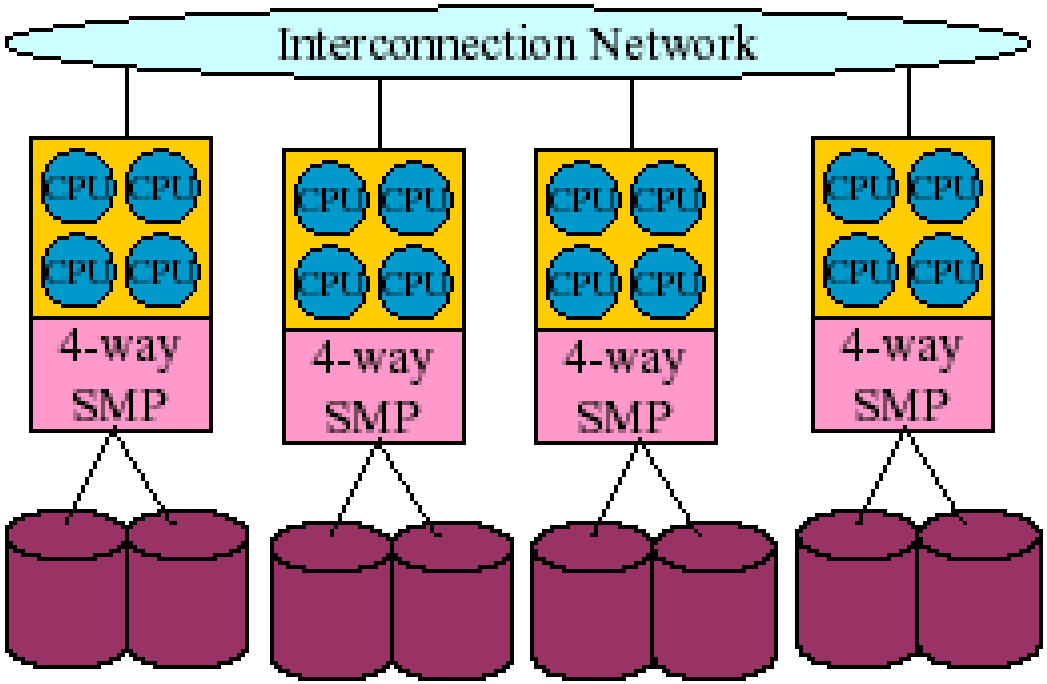
Architectures

- Message passing platform
  - Each processor has local memory with local address space
  - Only way to exchange data is using explicit message passing
  - Time taken for message depends on the relative locations of the source and destination processors
  - Performance of a parallel program determined by how well the location of data matches its use
  - Example: clusters



Architectures

- Hybrid
  - Most popular



Clusters of 4-way SMPs

TOP 10 Machines 11/2014

Rank	Name	Site	System	#core	R_{max} TF/s	Architecture	Country
1	Tianhe-2	National Super Computer Center in Guangzhou	NUDT Cluster, Intel Xeon Phi 31S1P	3120000	33862.7	cluster	China
2	Titan	DOE/SC/Oak Ridge National Laboratory	Cray XK7, Nvidia K20x	560640	17590.0	MPP	USA
3	Sequoia	DOE/NNSA/LLNL	BlueGene/Q	1572864	17173.2	MPP	USA
4	K	RIKEN Advanced Institute for Computational Science (AICS)	SPARC64 VIIIfx	705024	10510	Cluster	Japan
5	Mira	DOE/SC/Argonne National Laboratory	BlueGene/Q	786432	8586.6	MPP	USA

TOP 10 Machines 11/2014

Rank	Name	Site	System	#core	R_{max} TF/s	Architecture	Country
6	Piz Daint	Swiss National Supercomputing Centre	Cray XC30, Nvidia K20x	115984	6271.0	MPP	Switzerland
7	Stampede	Texas Advanced Computing Center/Univ. of Texas	Dell PowerEdge Intel Xeon Phi SE10P	462462	5168.1	cluster	USA
8	JUQUEEN	Forschungszentrum Juelich (FZJ)	BlueGene/Q	458752	5008.9	MPP	Germany
9	Vulcan	DOE/NNSA/LLNL	BlueGene/Q	393216	4293.3	MPP	USA
10		Government	Cray Nvidia K40	72800	3577	MPP	USA

TOP 6 Machines in China 11/2013

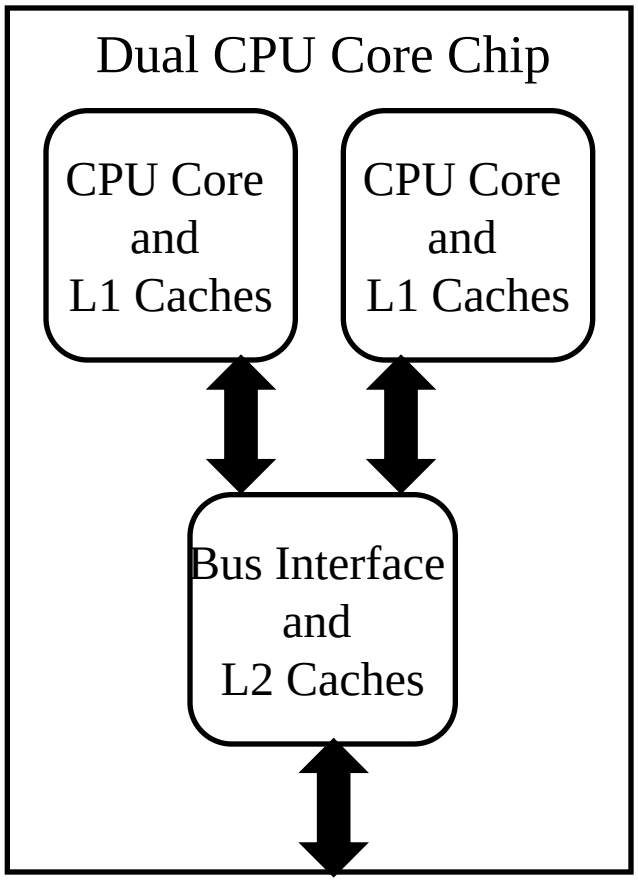
Rank	Name	Site	System	#core	R_{max} TF/s
1	Tianhe-2	National Super Computer Center in Guangzhou	NUDT Cluster, Xeon Phi	3120000	33862.7
12	Tianhe-1A	National Supercomputing Center in Tianjin	NUDT MPP, NVIDIA 2050	186368	2566
21	Nebulae	National Supercomputing Centre in Shenzhen	Dawning Cluster, NVIDIA 2050	120640	1271
40	Sunway Blue Light	National Supercomputing Center in Jinan	MPP	137200	795.9
42	Tianhe-1A Hunan Solution	National Supercomputing Center in Hunan	NUDT MPP, NVIDIA 2050	53248	771.7
63	Mole-8.5	Institute of Process Engineering, Chinese Academy of Sciences	Cluster, NVIDIA 2050	29440	496.5

TOP 6 Machines in China 11/2013

Rank	Name	Site	System	#core	R_{max} TF/s
1	Tianhe-2	National Super Computer Center in Guangzhou	NUDT Cluster, Xeon Phi	3120000	33862.7
12	Tianhe-1A	National Supercomputing Center in Tianjin	NUDT MPP, NVIDIA 2050	186368	2566
21	Tianhe-2 LvLiang Solution	LvLiang Cloud Computing Center	NUDT MPP, Intel Xeon Phi 31S1P	174720	2071.3
35	Nebulae	National Supercomputing Centre in Shenzhen	Dawning Cluster, NVIDIA 2050	120640	1271
65	Sunway Blue Light	National Supercomputing Center in Jinan	MPP	137200	795.9
67	Tianhe-1A Hunan Solution	National Supercomputing Center in Hunan	NUDT MPP, NVIDIA 2050	53248	771.7

Multi-Core

- A *multi-core* processor combines two or more independent cores into a single package composed of a single integrated circuit (IC), called a die.
- Each individual core is a CPU



Multi-Core

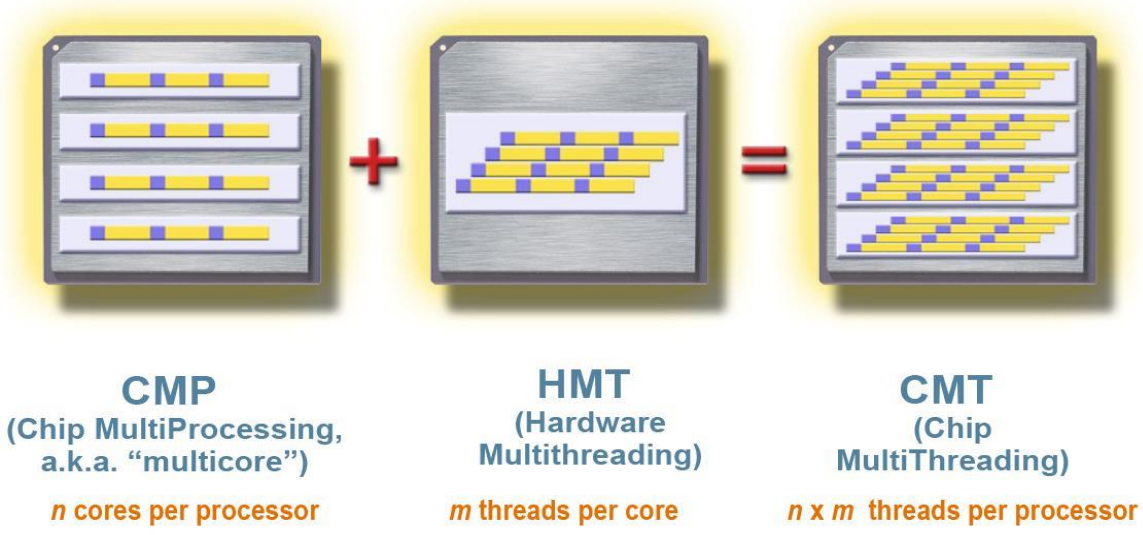
- Pros
  - Allow many users to connect to a site simultaneously and have independent threads
  - Cores in a die share a single coherent cache
  - Lower cost for higher performance
  - Low power consumption
- Cons
  - Design difficulty, 2-core => 4-core => 8 core

Multi-Core

- Hardware
  - AMD
    - Athlon 64, Athlon 64 FX and Athlon 64 X2 family, dual-core desktop processors.
    - Opteron, dual- and quad-core server/workstation processors.
    - Phenom, triple- and quad-core desktop processors.
    - Turion 64 X2, dual-core laptop processors.
  - IBM
    - POWER4, the world's first dual-core processor.
    - POWER5, a dual-core processor.
    - POWER6, a dual-core processor.
    - PowerPC 970MP, a dual-core processor, in the Apple Power Mac G5.
    - Sony/Toshiba/IBM Cell, 9 cores.
  - Intel
    - Celeron Dual Core, the first dual-core processor for the budget/entry-level market.
    - Core Duo, a dual-core processor.
    - Itanium 2, a dual-core processor.
    - Pentium D, a dual-core processor.
    - Nehalem, a quad-core processor, a 6-core processor, a 8-core processor.

Programming on Multi-Core

- Multithreading
  - Parallel execution
  - Common data
    - Lock and barrier guarantees synchronization and data consistence



Programming on Multi-Core

- Problem
  - Heavy weight thread, 1000 cycles, launching, communication, synchronization, etc.
  - Poor scalability on 8+ cores
- Good at
  - Coarse grain
  - Less communication
  - Task parallelism

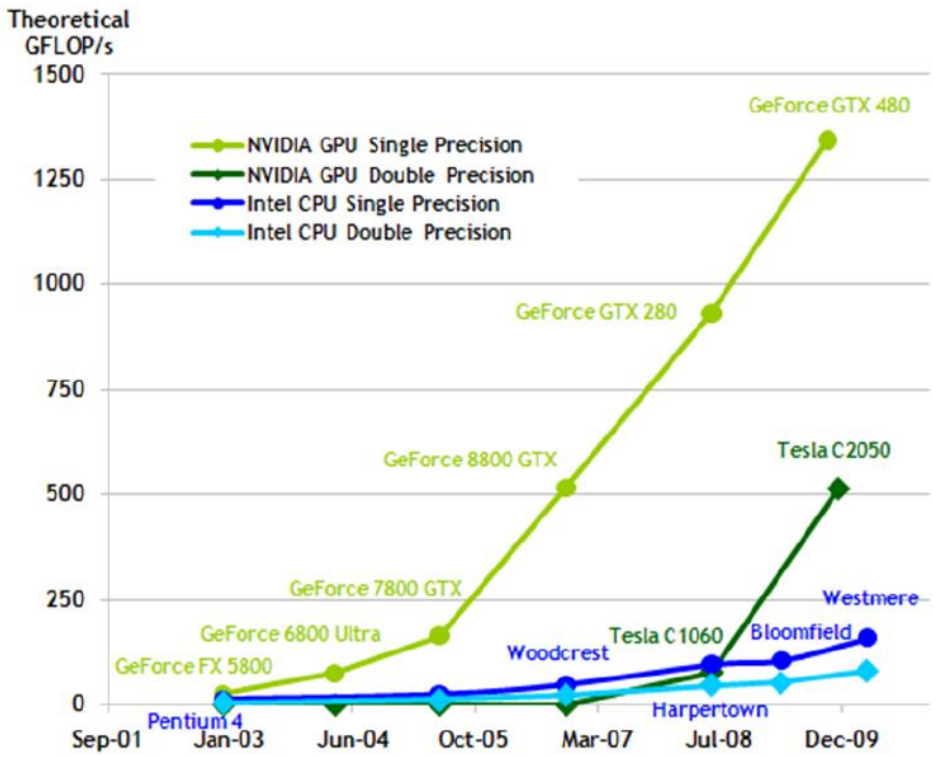
Parallel Computing with DSPs

- DSPs:
  - Signal Processing
    - Communication, Radar, Sonar, Video/Audio
  - Real time computing
  - Embedded system
  - Low power consumption
  - Products
    - TI: TMS320C6000/C5000/C2000
    - ADI: SHARC/TigerSHARC
  - Weakness
    - Difficult to co-program with HOST/CPU
  - Multi-Chips in parallel, low parallelism

Parallel Computing with FPGA

- FPGA Vendors
  - Altera
  - Xilinx
- Embedded Reconfigurable System
  - Multi-channel data/signal processing in parallel
  - Standard/user defined interfaces with external system
  - Hard to co-program with HOST/CPU
  - Multi-blocks and multi-chips in parallel
  - Difficult to develop and debug

GPU (Graphic Processing Unit)



GPU

- Architecture difference between GPU and CPU
  - More transistors for data processing
  - Many-core (hundreds of cores)

GPU

- Significant application-level speedup over uniprocessor execution
- Easy entrance
  - An initial, naïve code typically get at least 2-3X speedups
- Wide availability to end users
  - Available on laptops, desktops, clusters, supercomputers
- Numerical precision and accuracy
  - IEEE floating-point and double precision
- Strong scalability

Historic GPGPU Movement

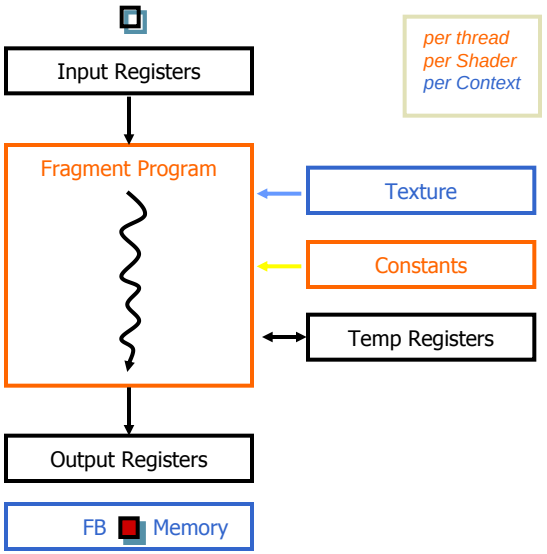
- General Purpose computation using GPU in applications other than 3D graphics
  - GPU accelerates critical path of application
- Data parallel algorithms leverage GPU attributes
  - Large data arrays, streaming throughput
  - Fine-grain SIMD parallelism
  - Low-latency floating point (FP) computation
- Applications – see http://GPGPU.org
  - Game effects (FX) physics, image processing
  - Physical modeling, computational engineering, matrix algebra, convolution, correlation, sorting

What is GPU Good at?

- Advantages of GPU
  - Low cost (hundreds of US dollars)
  - Many threads (hundreds or thousands of threads)
- Good at data-parallel processing
  - The same computation executed on many data elements in parallel – low control flow overhead with high SP floating point arithmetic intensity
  - Many calculations per memory access
- High floating-point arithmetic intensity and many data elements mean that memory access latency can be hidden with calculations instead of big data caches – *Still need to avoid bandwidth saturation!*

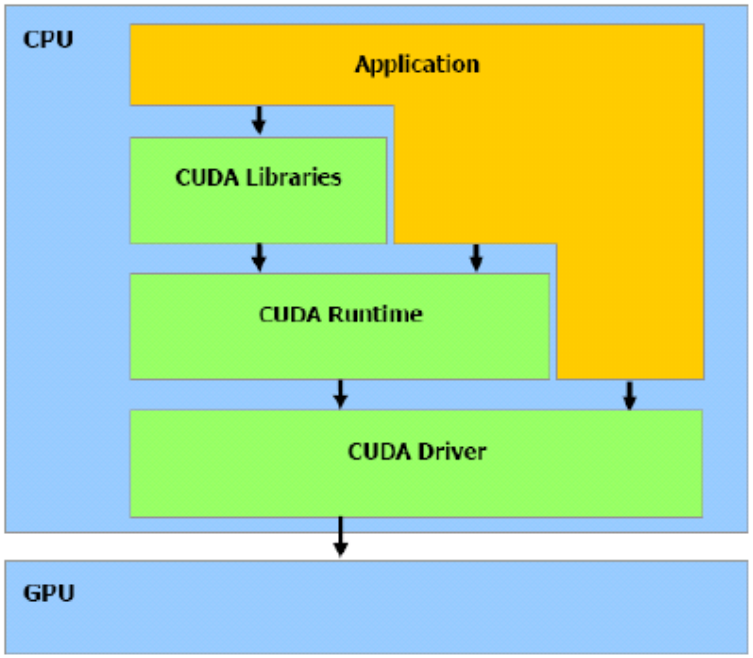
Historic GPGPU Constraints

- Dealing with graphics API
  - Working with the corner cases of the graphics API
- Addressing modes
  - Limited texture size/dimension
- Communication limited
  - No interaction between pixels



CUDA (Compute Unified Device Architecture)

- Write programs for GPU on C language with minimum extensions
- Ease of programming
- Single Program Multiple Data (SPMD)
- No need of graphics APIs

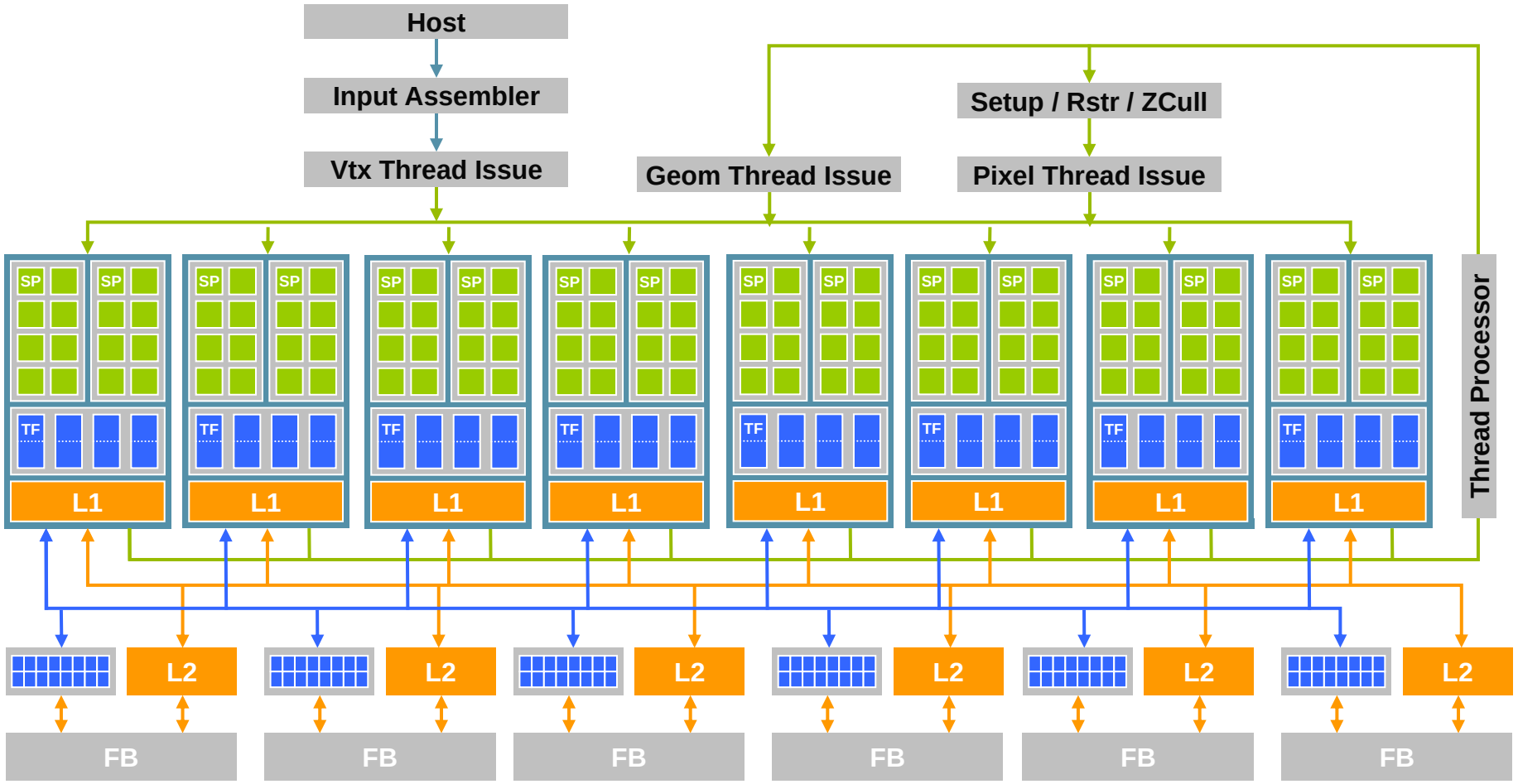


Software Stack

CUDA API Highlights: Easy and Lightweight

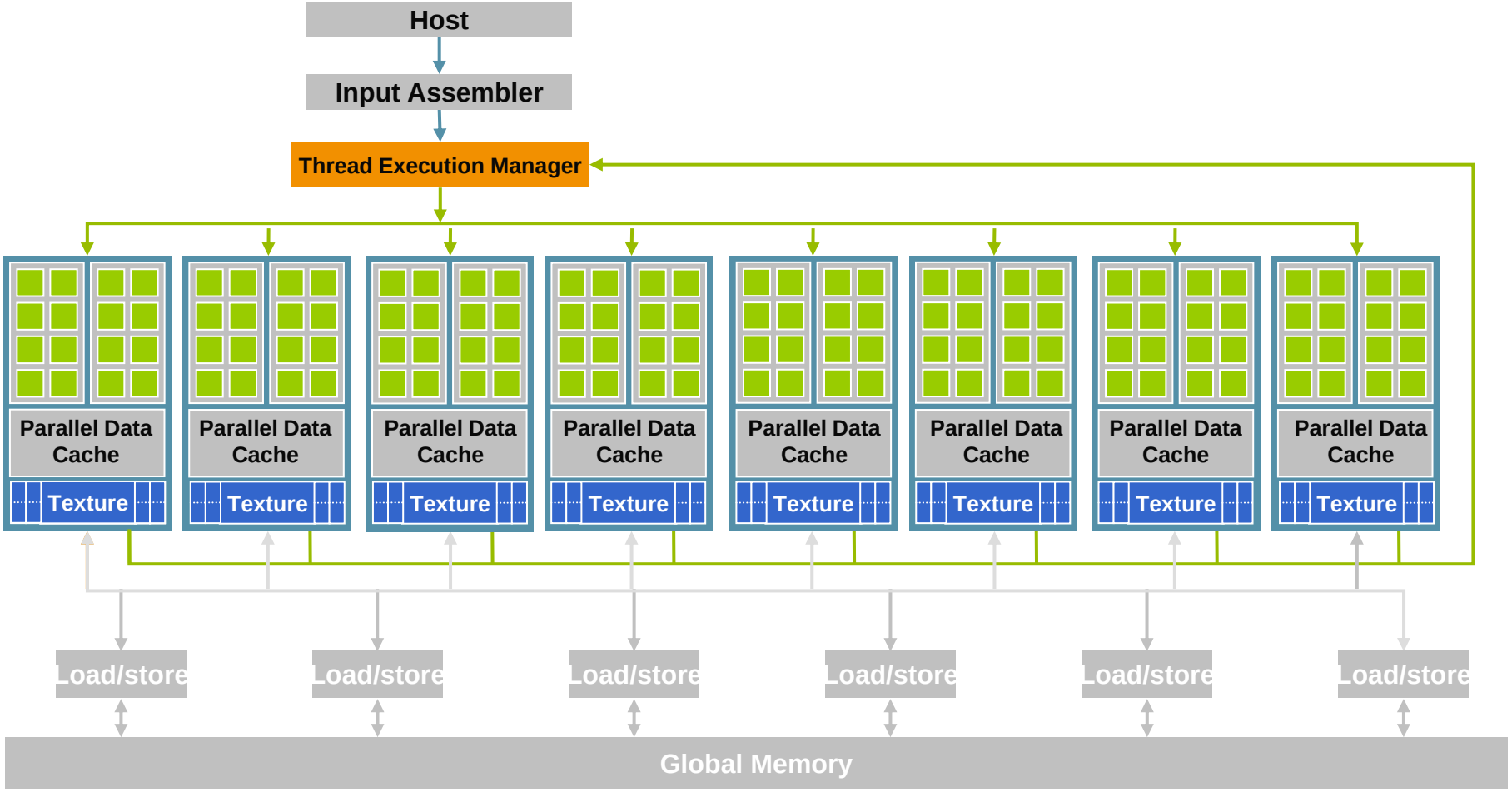
- The API is an extension to the ANSI C programming language
  - Low learning curve
- The hardware is designed to enable lightweight runtime and driver
  - High performance

G80 – Graphics Mode



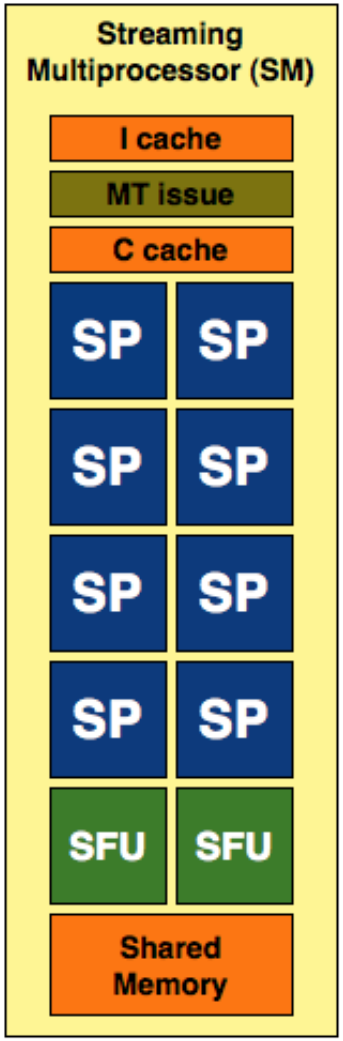
Block Diagram of the GeForce 8800

G80 CUDA Mode

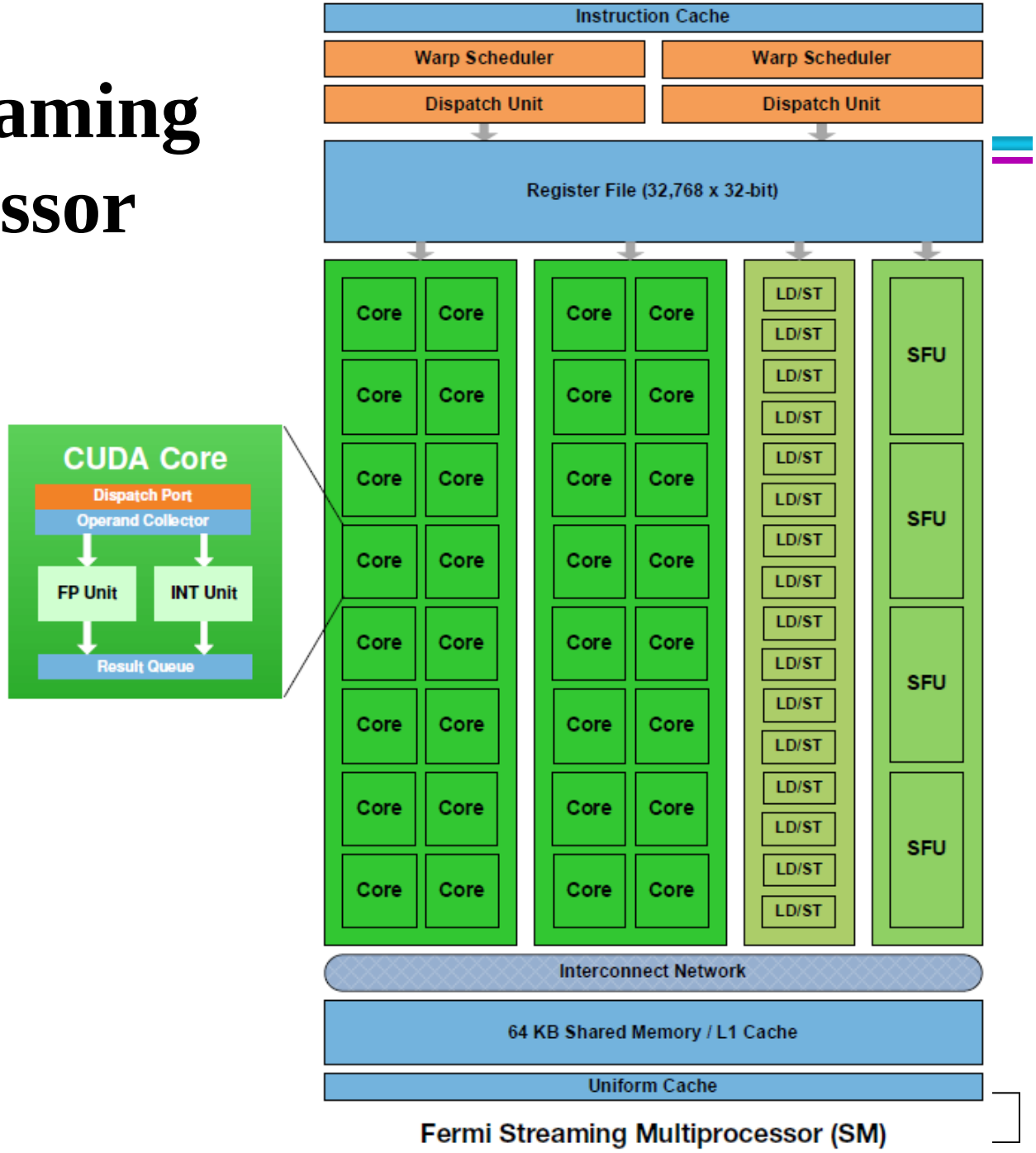


Streaming Multiprocessor (SM)

- An array of SPs
  - 8 streaming processors
  - 2 Special Function Units (SFU)
    - Transcendental operations (e.g. sin, cos) and interpolation
  - A 16KB read/write shared memory
    - Not a cache, but a software-managed data store
  - Multithreading issuing unit
    - Dispatch instructions
  - Instruction cache
  - Constant cache



Fermi Streaming Multiprocessor (SM)



CUDA Device

- A compute device
  - Is a coprocessor to the CPU or host
  - Has its own DRAM (device memory)
  - Runs many threads in parallel
  - Is typically a GPU but can also be another type of parallel processing device
- Kernel — Data-parallel portions of an application which run on many threads

GPU+CUDA

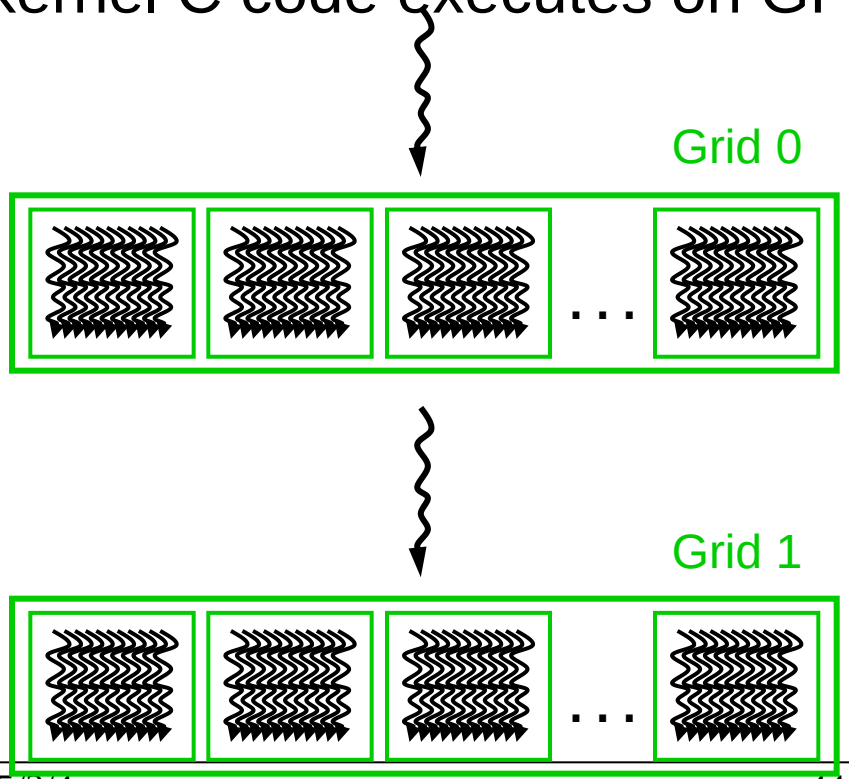
- CUDA integrated CPU+GPU application C program
  - Serial or modestly parallel C code executes on CPU
  - Highly parallel SPMD kernel C code executes on GPU

CPU Serial Code

GPU Parallel Kernel
KernelA<<< nBlk, nTid >>>(args);

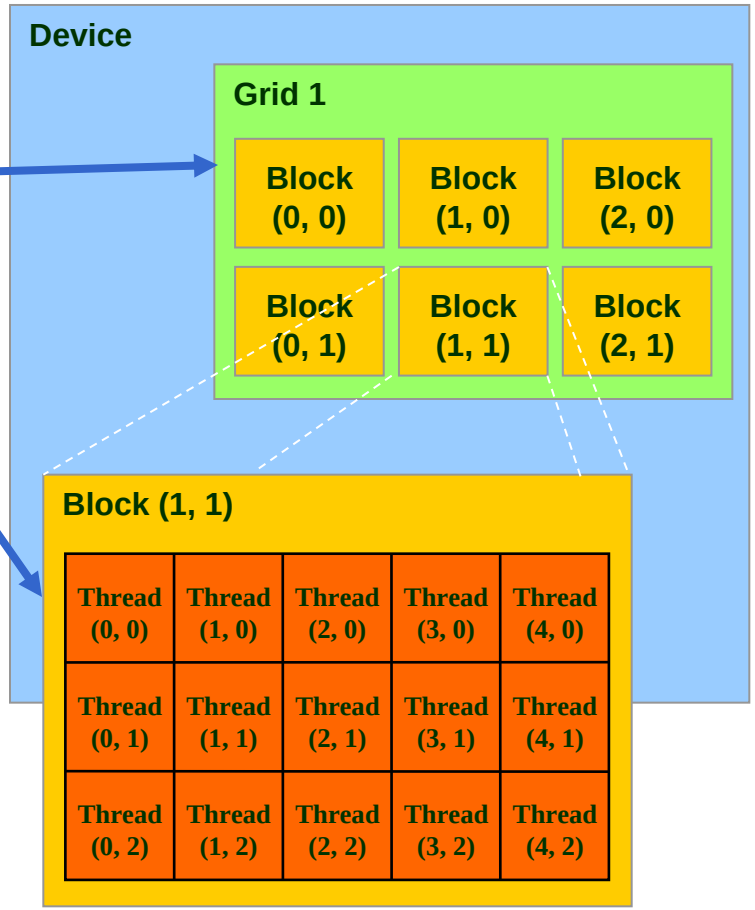
CPU Serial Code

GPU Parallel Kernel
KernelB<<< nBlk, nTid >>>(args);



Block IDs and Thread IDs

- Each thread uses IDs to decide what data to work on
  - Block ID: 1D or 2D
  - Thread ID: 1D, 2D, or 3D
- Simplify memory addressing when processing multidimensional data
  - Image processing
  - Solving PDEs on volumes
  - …

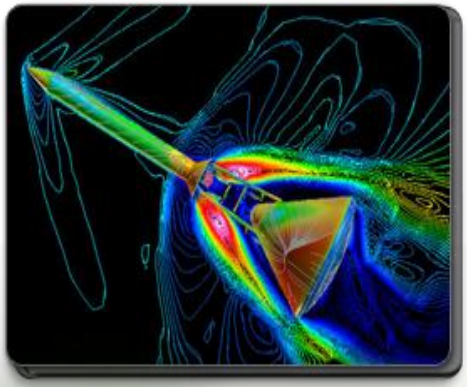


支持CUDA的NVIDIA硬件

GeForce®
娱乐

Quadro®
设计和创作

Tesla™
高性能计算



Tegra
嵌入式系统

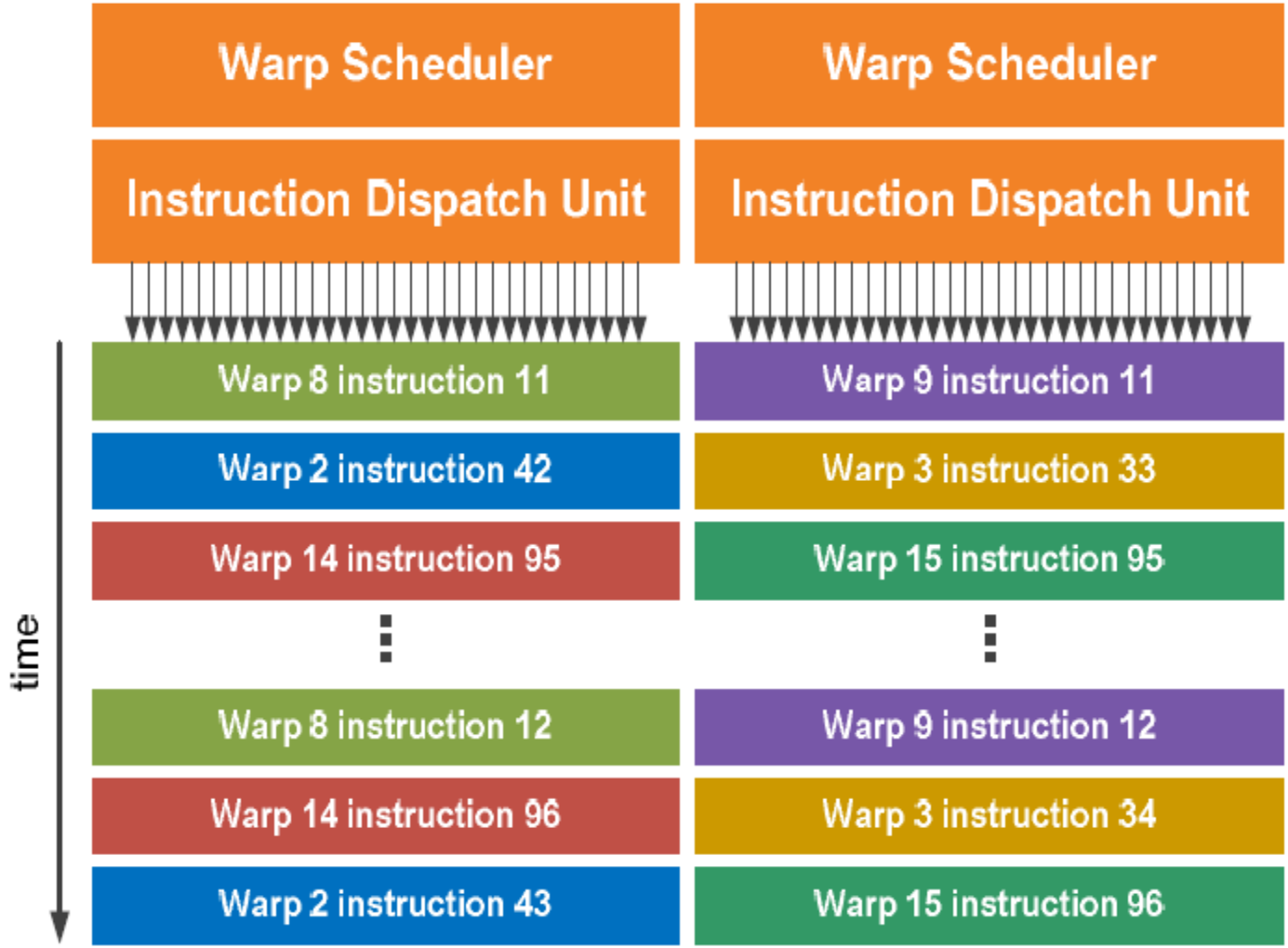
Configuration

	GeForce 8800 GTX	Tesla C1060 (GTX280)
# stream processor	128	240
# stream multiprocessor	16	30
# registers per SM	8192 (32KB)	16384(64KB)
# threads per block	Up to 512	Up to 512
# threads per SM	Up to 768	Up to 768
# blocks per SM	Up to 8	Up to 8
# blocks per grid	Up to 65535 each dim	Up to 65535 each dim
global memory	768MB, 1.8GHz, 384-bit	4GB, 1.6GHz, 512-bit
constant memory	64KB	64KB
shared memory per SM	16KB	32KB
clock	1.35GHz	1.296GHz
peak	346.5 GFlops/s	936 GFlops/s
memory bandwidth	86.4 GB/s	102 GB/s

New Features in Fermi

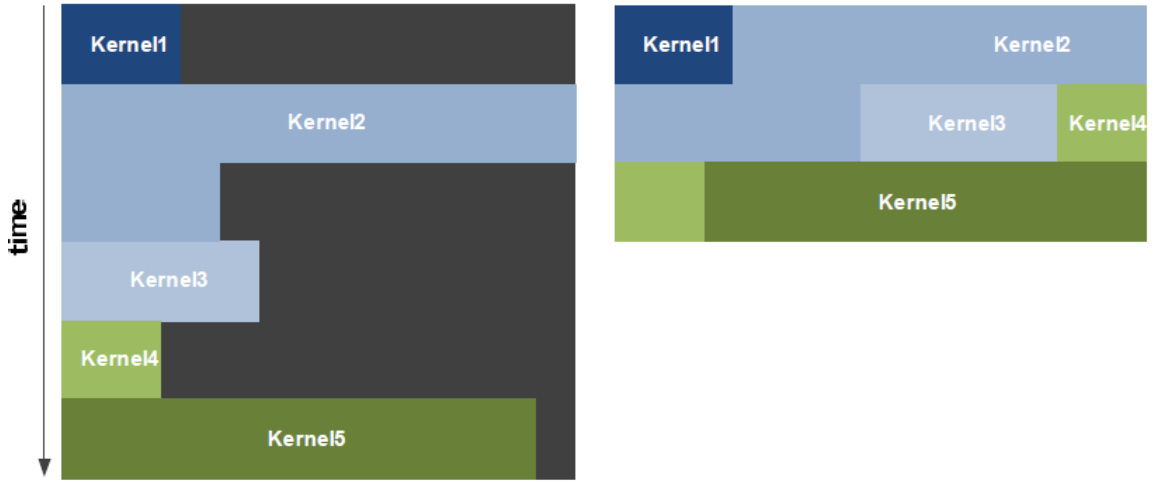
- 512 SPs in total
- 32 SPs per SM, 4x over GT200
- 4 SFUs per SM
- 64 KB on-chip memory per SM
  - Can be configured as 48 KB of Shared memory with 16 KB of L1 cache or as 16 KB of Shared memory with 48 KB of L1 cache
- Dual Warp Scheduler simultaneously schedules and dispatches instructions from two independent warps

Fermi Dual Warp Scheduler



New Features in Fermi

- Support full C++
- Up to 20x faster atomic memory operations
- Concurrent kernel execution
  - Different kernels of the same application context can execute on the GPU at the same time



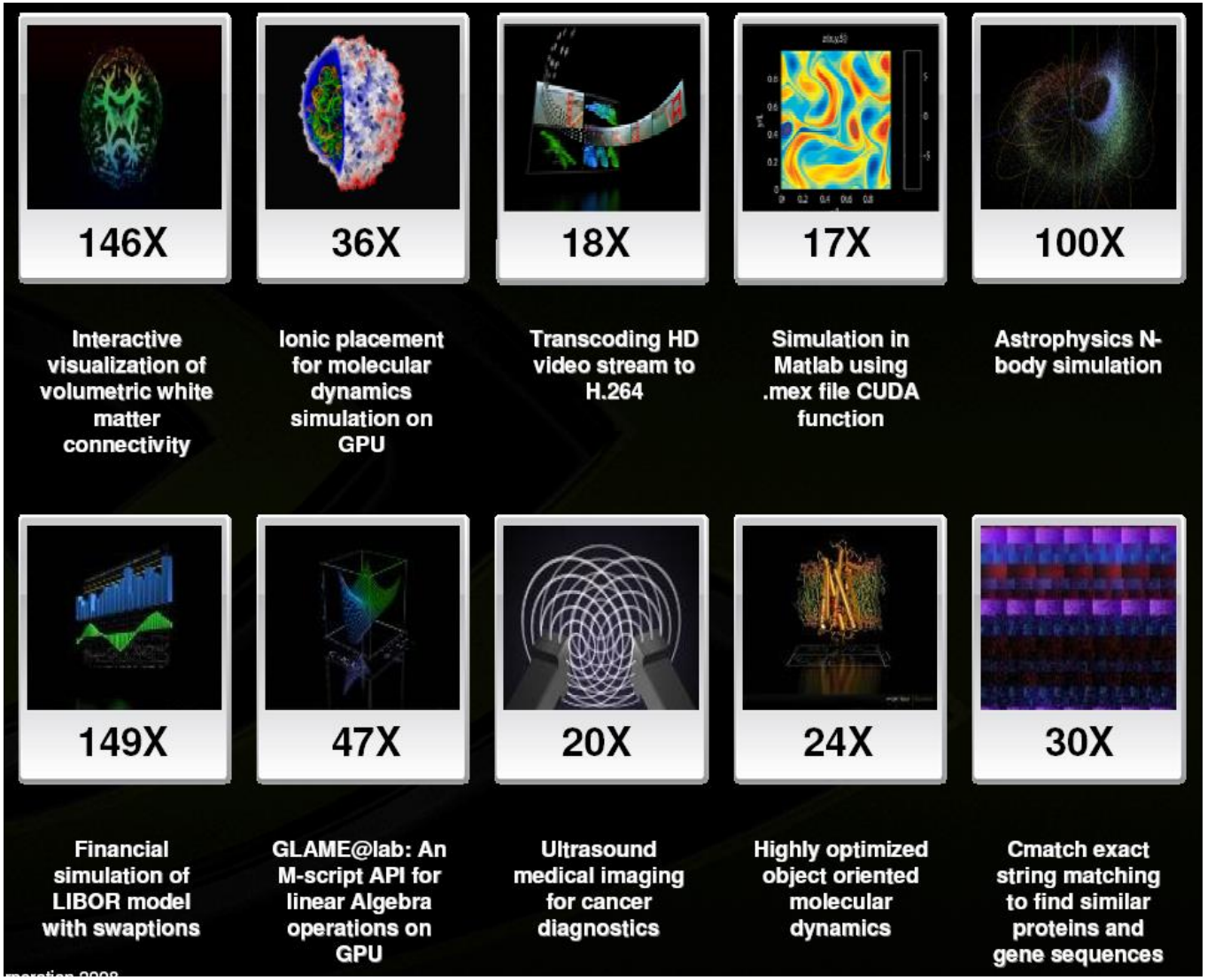
Configuration

GPU	G80	GT200	Fermi
Transistors	681 million	1.4 billion	3.0 billion
CUDA Cores	128	240	512
Double Precision Floating Point Capability	None	30 FMA ops / clock	256 FMA ops /clock
Single Precision Floating Point Capability	128 MAD ops/clock	240 MAD ops / clock	512 FMA ops /clock
Special Function Units (SFUs) / SM	2	2	4
Warp schedulers (per SM)	1	1	2
Shared Memory (per SM)	16 KB	16 KB	Configurable 48 KB or 16 KB
L1 Cache (per SM)	None	None	Configurable 16 KB or 48 KB
L2 Cache	None	None	768 KB
ECC Memory Support	No	No	Yes
Concurrent Kernels	No	No	Up to 16
Load/Store Address Width	32-bit	32-bit	64-bit

Free Downloadable CUDA Software

- http://www.nvidia.cn/object/cuda_get_cn.html
  - CUDA driver
  - CUDA toolkit
  - CUDA SDK
  - CUDA Visual Profiler

Successful Applications

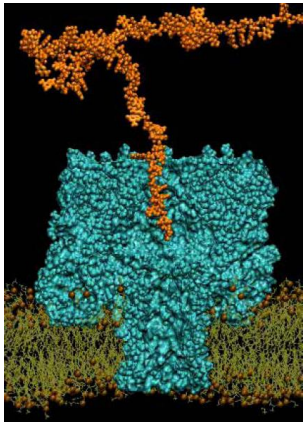


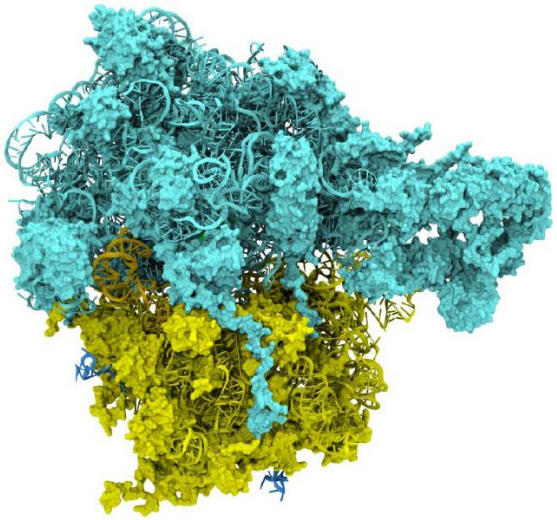
Science and Engineering Application Speedup

App.	Simult. T	Kernel X	App X
LBM	3,200	12.5	12.3
FEM	4,096	11.0	10.1
RPES	4,096	210.0	79.4
PNS	2,048	24.0	23.7
LINPACK	12,288	19.4	11.8
TRACF	4,096	60.2	21.6
FDTD	1,365	10.5	1.2
MRI-FHD	8,192	23.0	23.0

VMD

Calculation / Algorithm	Algorithm class	Speedup vs. Intel QX6700 CPU core
Fluorescence microphotolysis	Iterative matrix / stencil	12x
Pairlist calculation	Particle pair distance test	10-11x
Pairlist update	Particle pair distance test	5-15x
Molecular dynamics non-bonded force calculation	N-body cutoff force calculations	10x 20x (w/ pairlist)
Cutoff electron density sum	Particle-grid w/ cutoff	15-23x
Cutoff potential summation	Particle-grid w/ cutoff	12-21x
Direct Coulomb summation	Particle-grid	44x



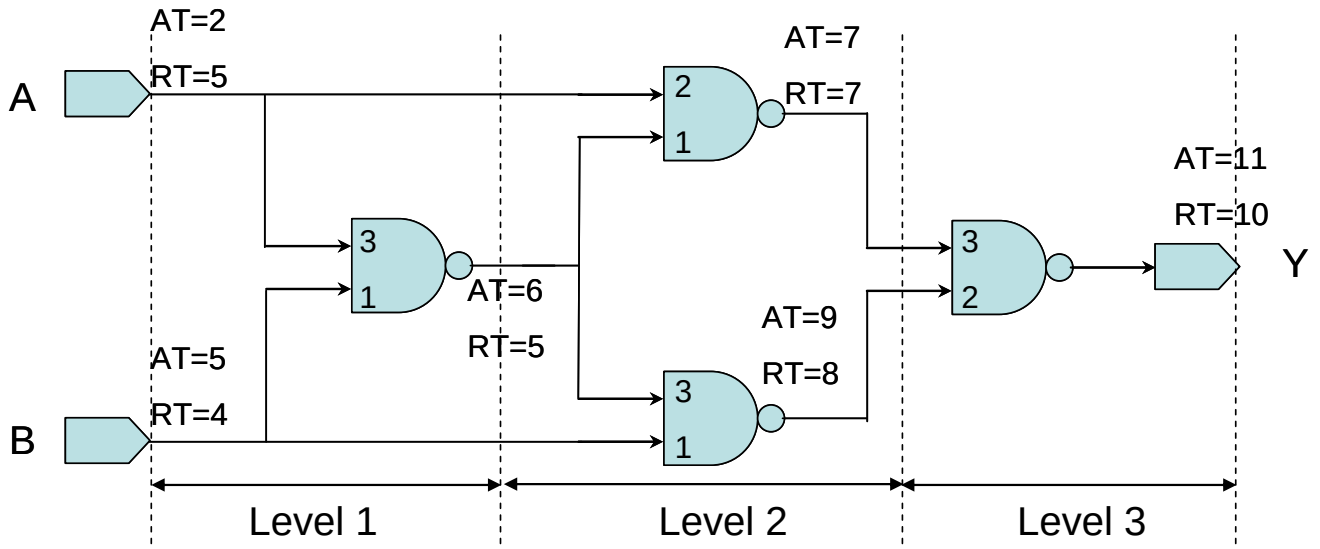


Theoretical and Computational Biophysics Group, UIUC

http://www.ks.uiuc.edu/Research/gpu/

EDA

- Graph theory algorithms (20-50X)
  - breadth first traversal
  - single source shortest path
  - all pair shortest path
  - maximum flow, Push re-label



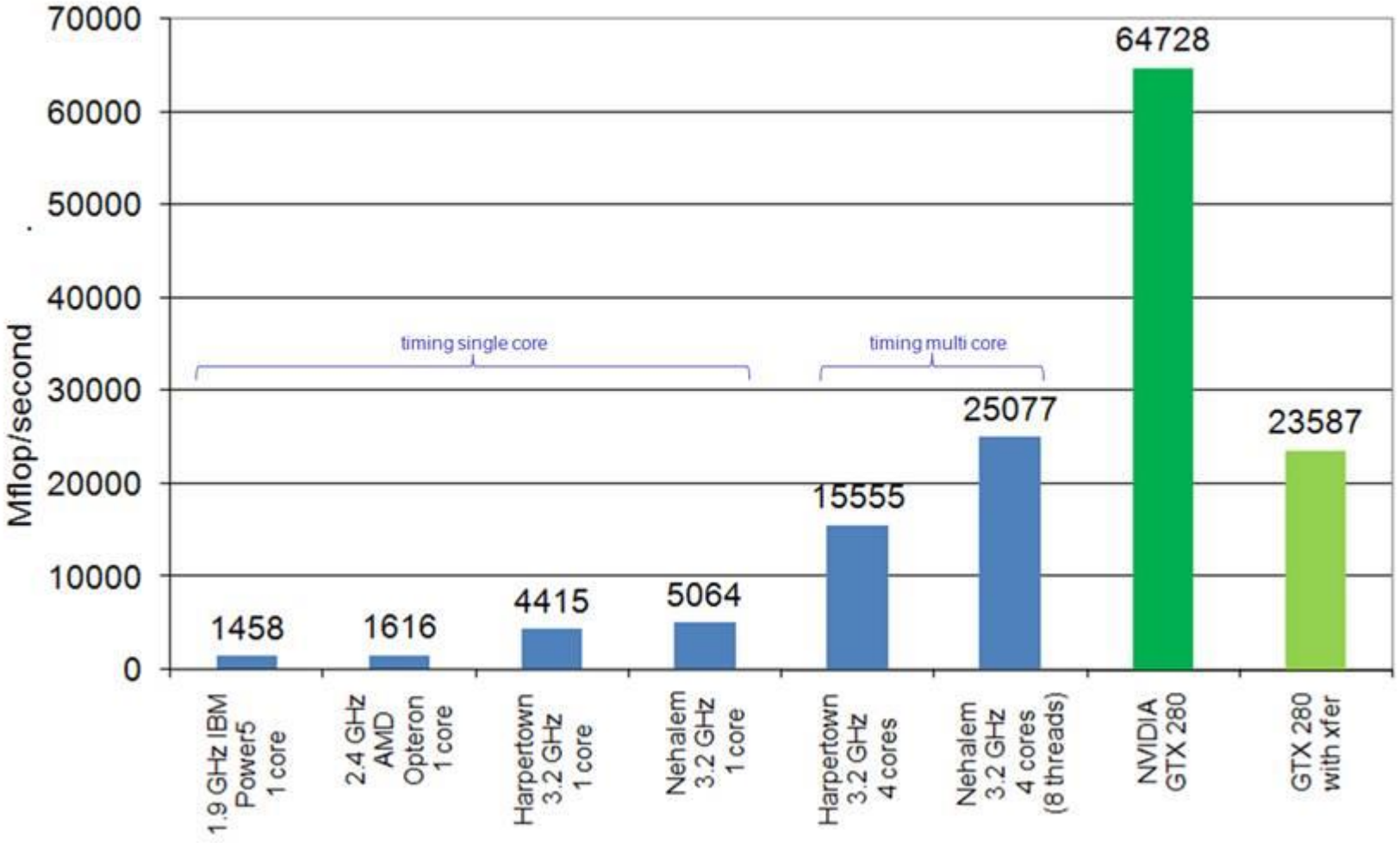
Numerical Weather Prediction (NWP)

- Weather Research and Forecast Model (WRF)
- WRF Single Moment 5-tracer (WSM5)
  - 0.4% of the WRF code but consumes 25% total run time

CUDA Implementation

- Geographical region partitioned in a 2D grid parallel to the ground
- Multiple levels along vertical height in the atmosphere for each grid
- 2400 floating point multiply-equivalent operation per cell per invocation
- Use -use_fast_math option to nvcc compiler
  - Square root, log, exponent to be computed by SFUs on the GPU
- Eliminate temporary arrays that store results between successive loops over k, the vertical dimension of WRF domain

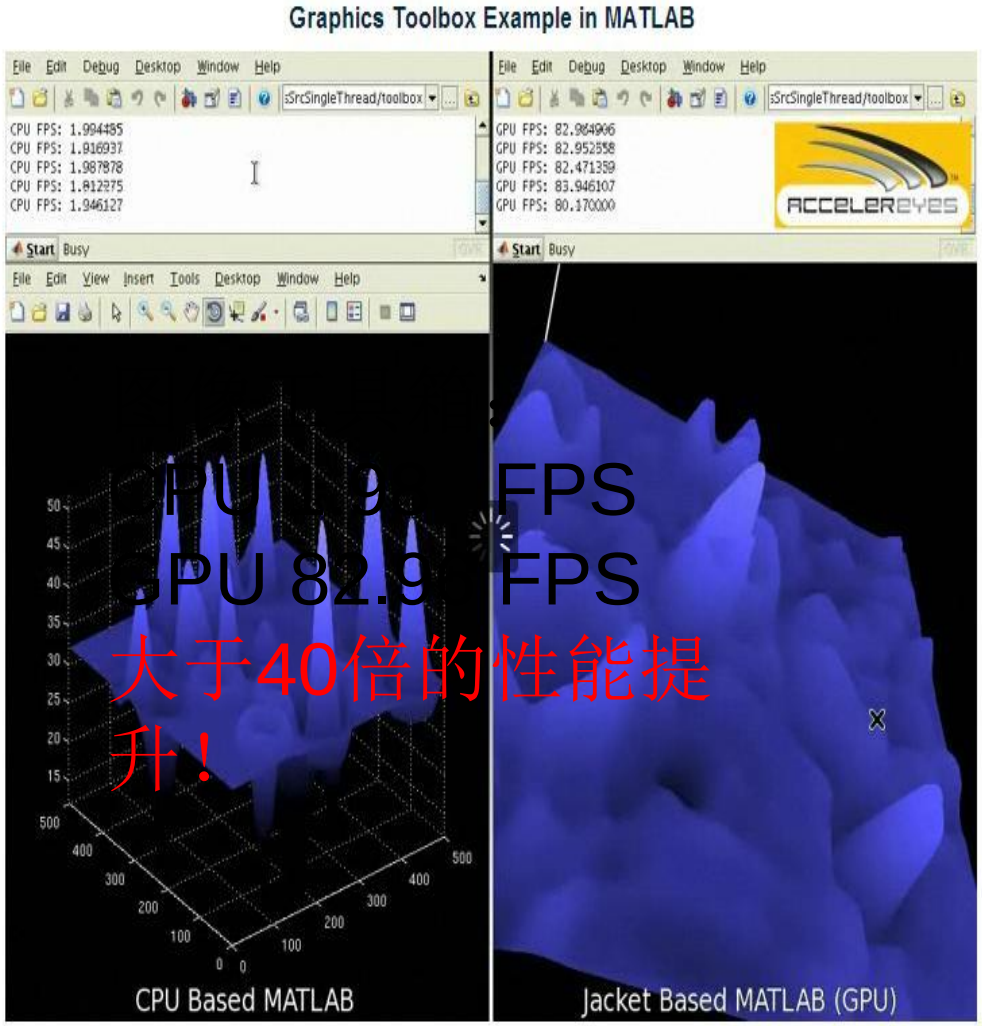
Evaluation


70000
60000
50000
40000
30000
20000
10000
0
Mflop/second
timing single core
timing multi core
1458
1616
4415
5064
15555
25077
64728
23587
1.9 GHz IBM Power5 1 core
2.4 GHz AMD Opteron 1 core
Harpertown 3.2 GHz 1 core
Nehalem 3.2 GHz 1 core
Harpertown 3.2 GHz 4 cores
Nehalem 3.2 GHz 4 cores (8 threads)
NVIDIA GTX 280
GTX 280 with xfer

GPU 加速MATLAB — Jacket 插件

- MATLAB被广泛地应用于科学计算、控制系统、信息处理、医疗成像仿真等领域的分析、仿真和设计工作
- Jacket是由AccelerEyes公司开发的一个强大的基于CUDA的MATLAB 插件
- 性能提高 40 倍以上
- 支持 Win 32/64, Linux 32/64, Mac 32

http://www.accelereyes.com



FPS is the number of frames per second processed by the computation and visualization pipelines, not the video frame rate.

(Last Updated: 07 September 2008)

GPU 加速矢量信号图像处理库 VSIPL 20 至 350 倍

- GPU VSIPL 实现了 VISPL Core Lite Profile
- 用CUDA 2.3 和 Visual Studio 2005 实现
- 在 GeForce 8800GTX 上 20 至350 倍的加速
- 支持 Windows XP/Vista, Linux

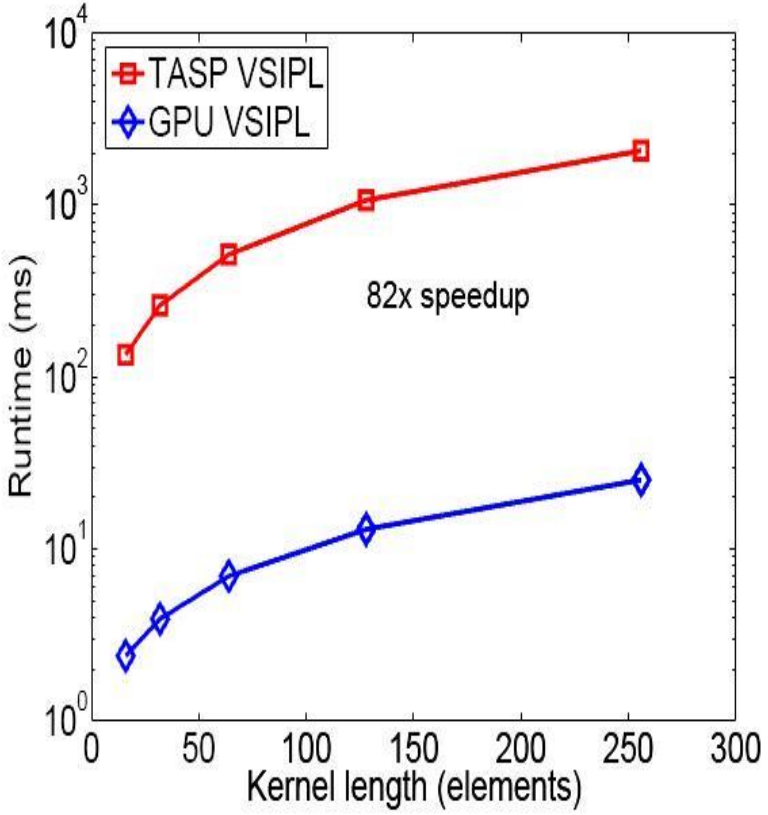
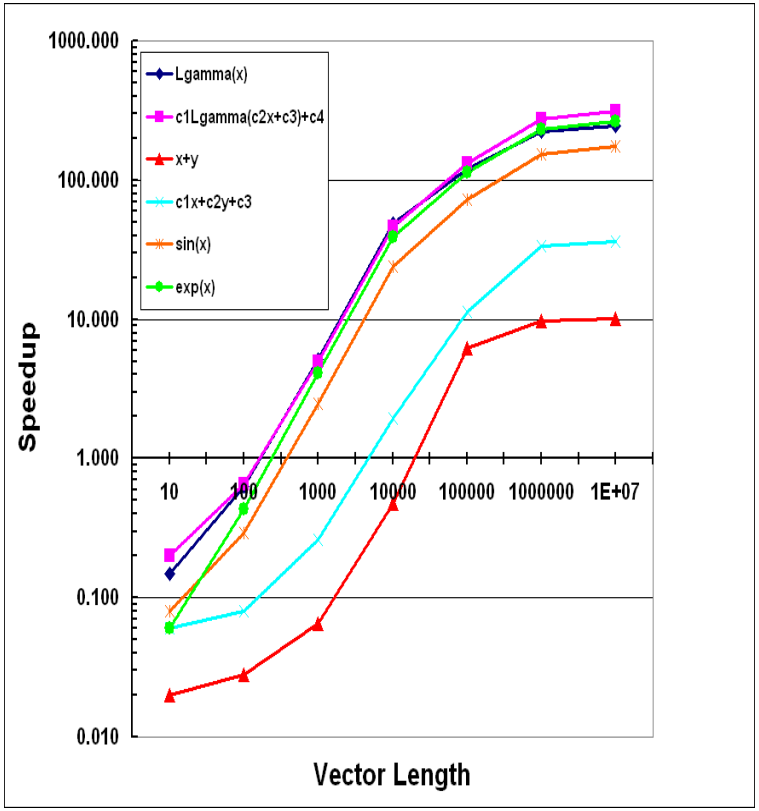


Figure 1: Time-domain FIR filtering runtime.

http://gpu-vsipl.gtri.gatech.edu/

GPU 加速数学库 — Tech-X 的 GPULib

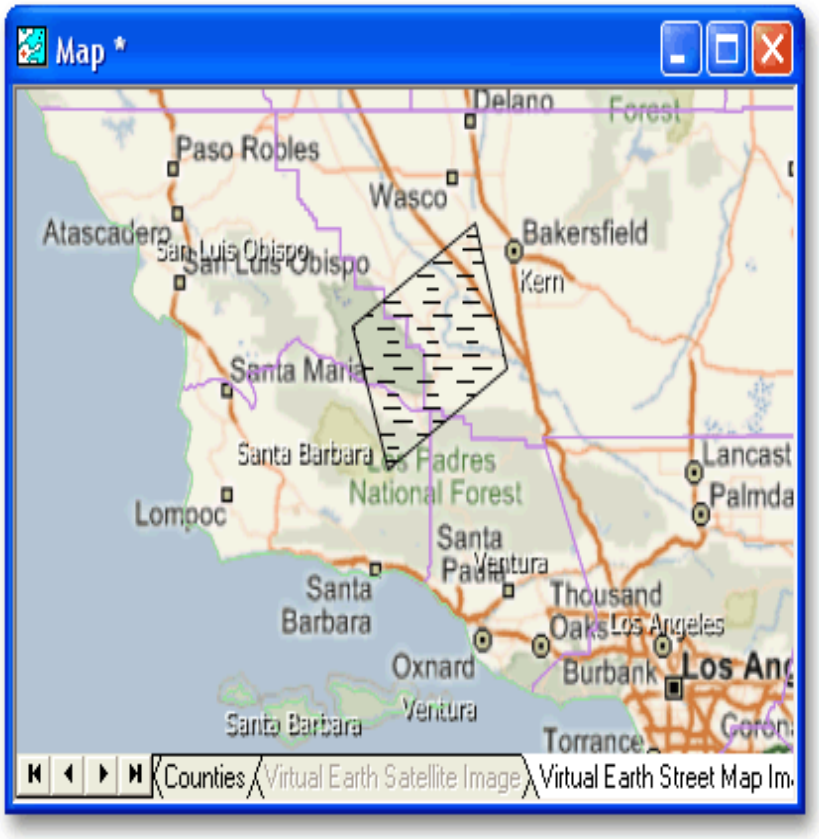
- GPULib 提供了基于GPU的数学库
- 不需要GPU编程的知识
- 30 倍的加速



http://www.txcorp.com/products/GPULib/

GPU 加速 GIS 应用 — Manifold 8

- Manifold 8 是第一个支持GPU的地理信息系统
- 利用GPU，GIS任务和分析比以前快几百倍



http://www.manifold.net/index.shtml

GPU加速分子建模应用软件 — OpenMM

- OpenMM 在 GPU 上加速 GROMACS
- 高达 10 至 1000 的加速
- 支持 MacOS，Windows，Linux

Molecule	# atoms	ns/day	speedup*	GFLOPS (GPU)	GFLOPS (x86)
fip35	544	576	128	311	657
villin	582	529	136	328	692
lambda	1254	202	255	547	1153
a-spectrin	5078	17	735	805	1702

(*comparing a GTX280 to a single core of a 3GHz core 2 duo using the AMBER code)

https://simtk.org/project/xml/downloads.xml?group_id=161

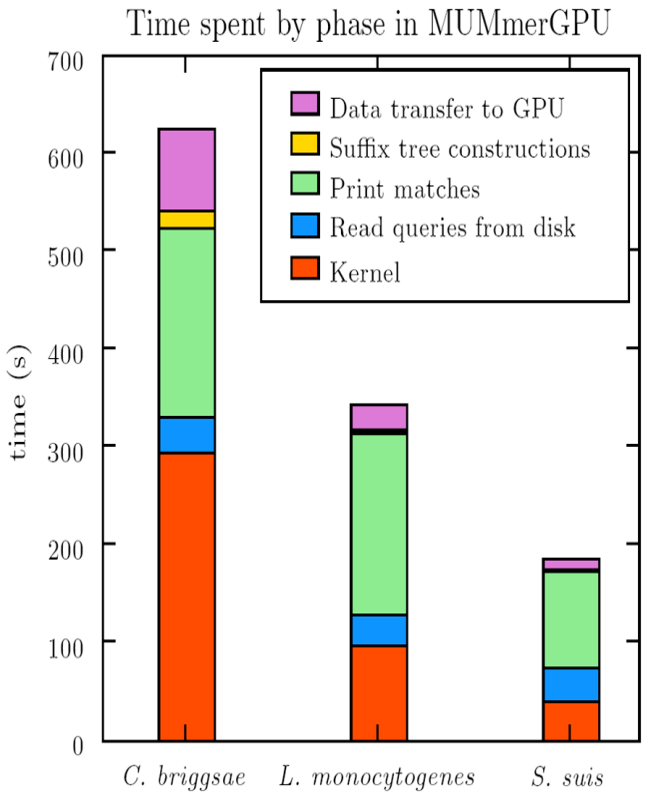
GPU加速分子建模应用软件 – 100x
伊利诺依大学NAMD / VMD

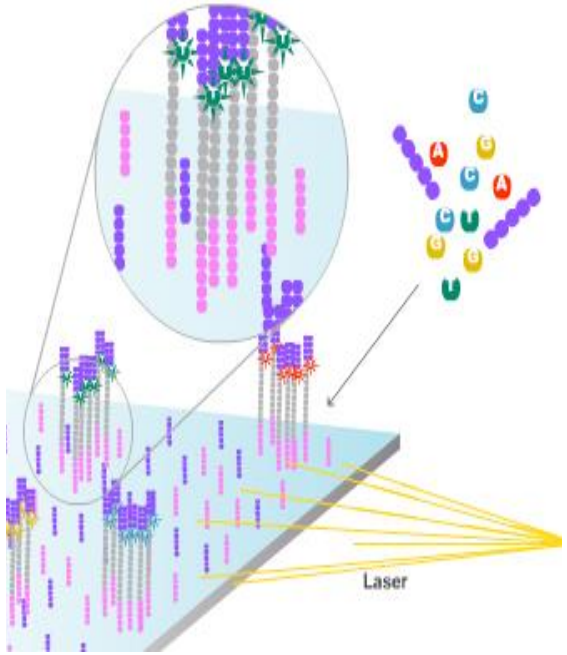
- 117 billions evals/sec
- 863 GFLOPS
- 131 倍的加速 （相对于CPU核）

Quad-core Intel QX6700
3 块 NVIDIA GeForce 8800GTX

基因序列排列软件 — MUMmerGPU

- 下载：
  http://sourceforge.net/projects/mummergpu/
- 加速 3 倍多

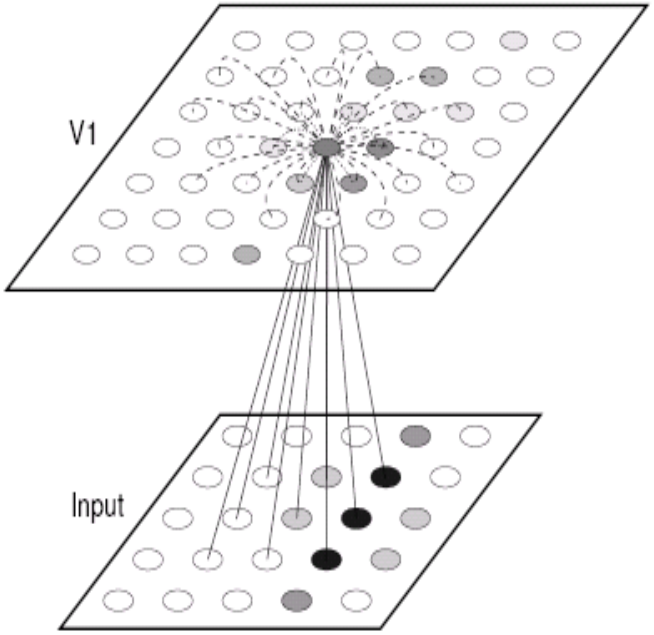




人类大脑模拟软件 — RF-LISSOM

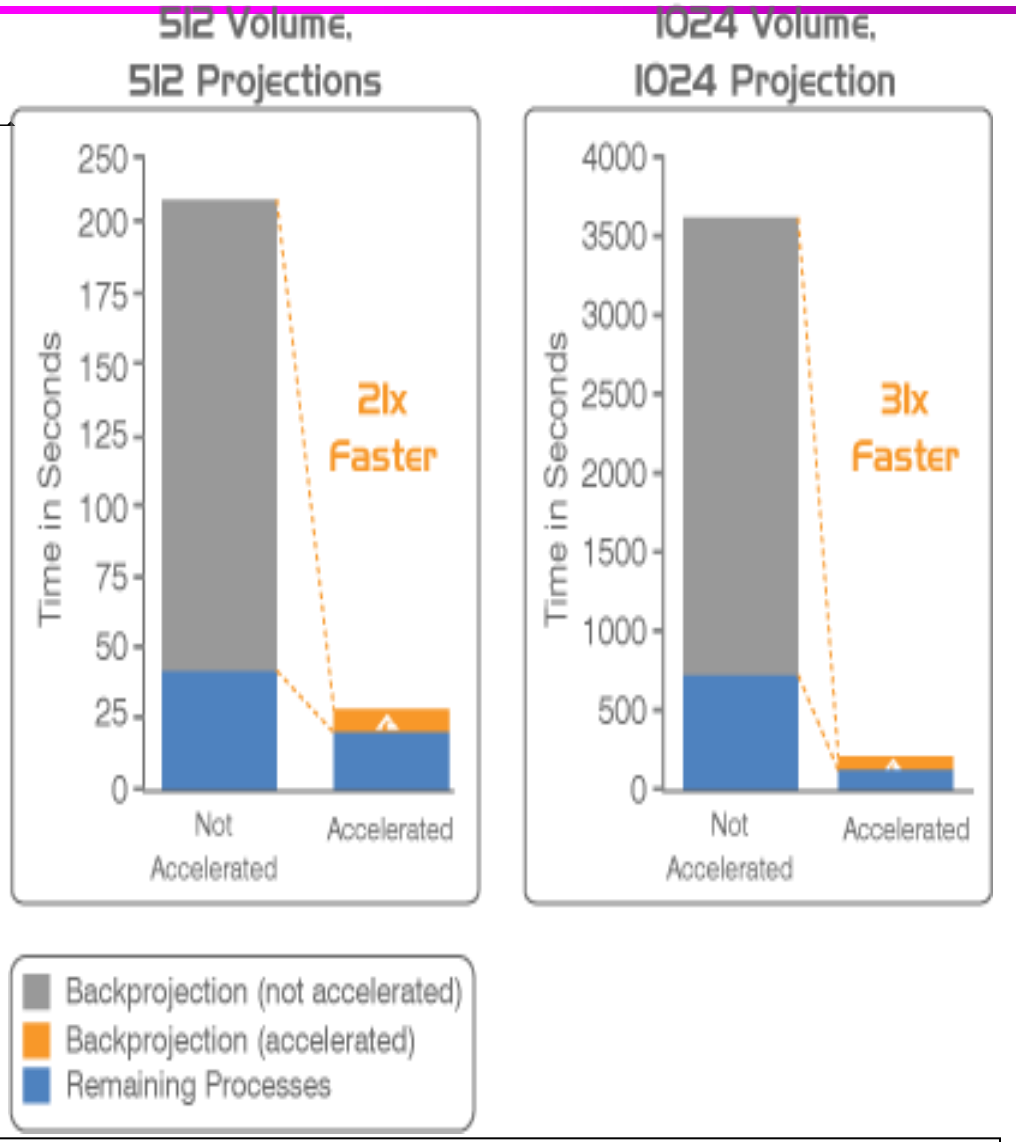
- 参见：
  http://homepages.inf.ed.ac.uk/jbednar/rflissom_small.html

- GT200 加速 5x 以上



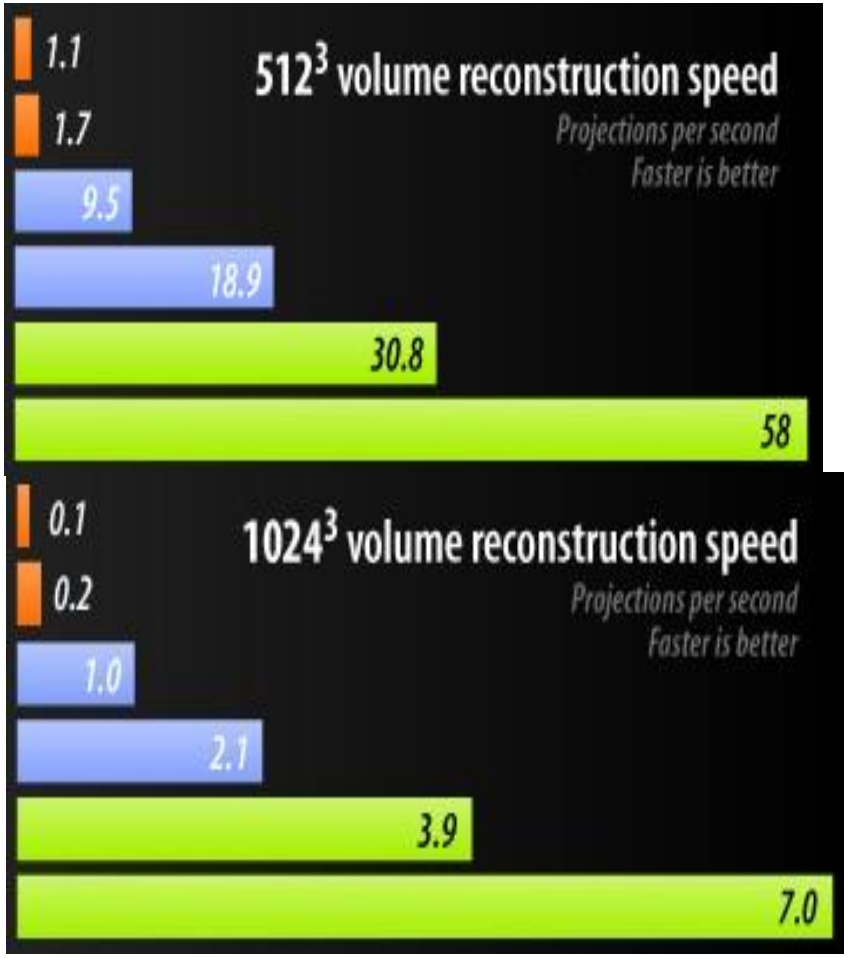
GPU 加速CT机的成像 — AxRecon

- 强大的性能，不再需要计算集群
- 节省电力
- 无损图像质量



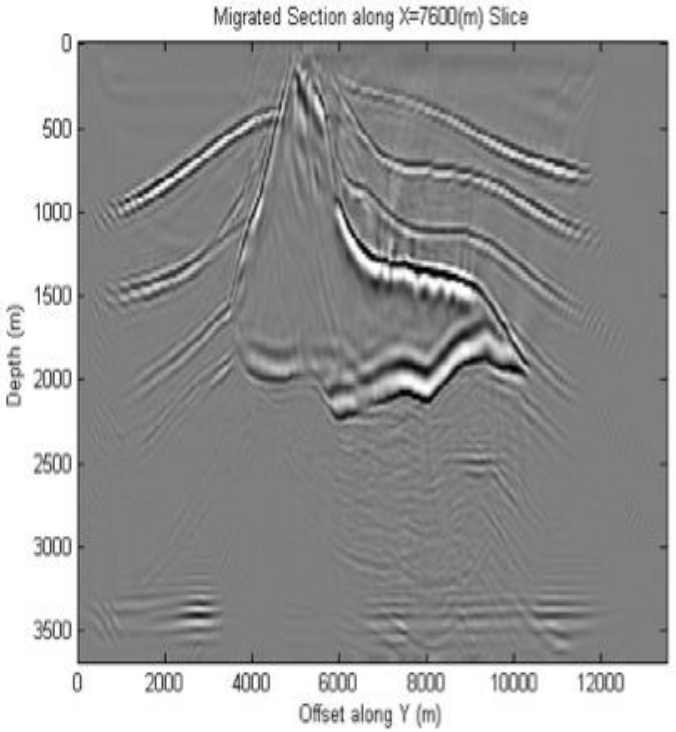
GPU 加速CT机的成像 — SnapCT

- 性能提高 20～50 倍



GPU加速 RTM 和 KTM

- 基尔霍夫叠前时间偏移（KTM）和反向时间偏移（RTM ）是石油天然气行业常用的数据处理手段
- Acceleware的GPU 方案节省70%的电力
- 性能提高 20倍以上
- 参见：http://www.acceleware.com/default/index.cfm/solutions/seismic-solutions/



GPU加速叠前深度偏移的3D 地震成像

- SeismicCity 现在在利用 NVIDIA Tesla S1070 进行叠前深度偏移的成像处理
- 性能提高 60倍以上
- 参见：http://www.seismiccity.com/index.html

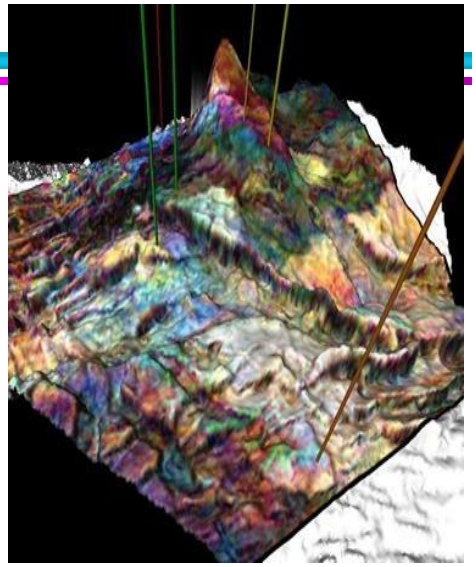
GPU加速频谱分析和反演

- OpenGeoSolutions公司专门使用一项叫做“光谱分析”的技术来提供地质信息，这些信息超越了传统的地震资料分辨率以及检测方式，在处理巨大的地区数据集时提高了数据质量。更重要的是，这种技术在逆向转换数据时还能够将其转换为真实的地质构造。

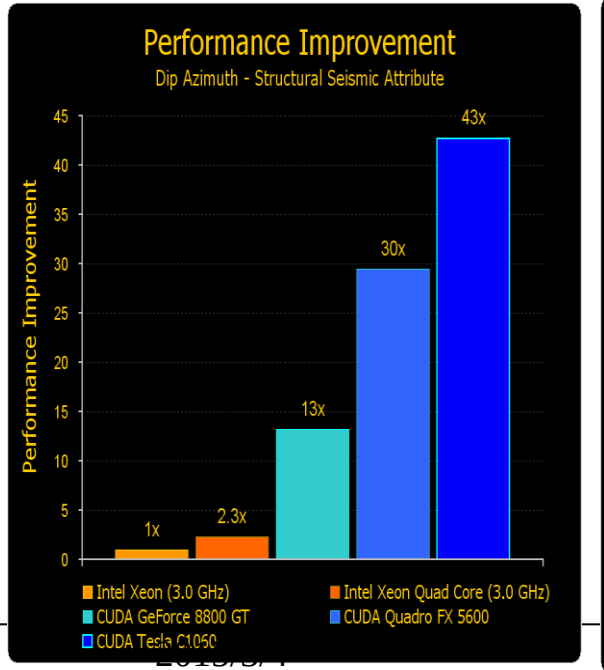
- OpenGeoSolutions利用NVIDIA Tesla C1060 进行光谱分析和反演，性能提高了数十倍。

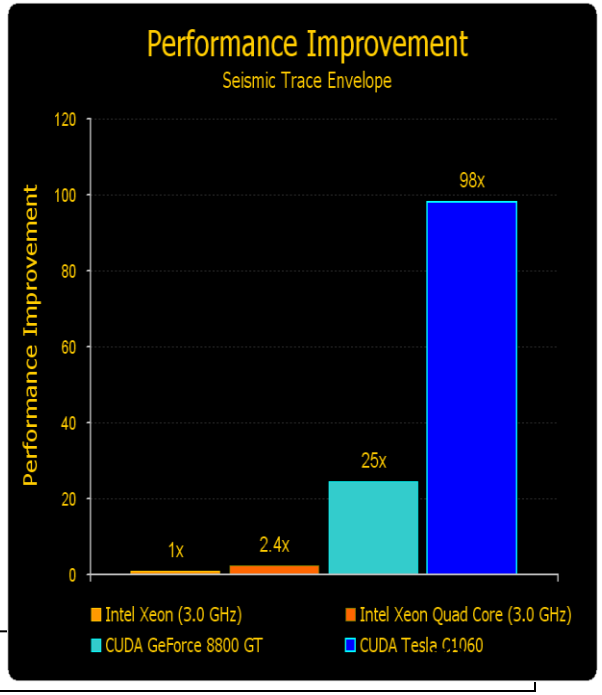
- http://www.opengeosolutions.com/

GPU加速地震属性的计算



- ffA是英国一家专门从事3D地震成像处理技术的公司。
- SEA 3D & SVI Pro 是ffA 公司3D地震成像分析和可视化的软件。
- 利用NVIDIA GPU性能提高了10 ~ 100 倍。
- **参见：** http://www.ffa.co.uk/index.html



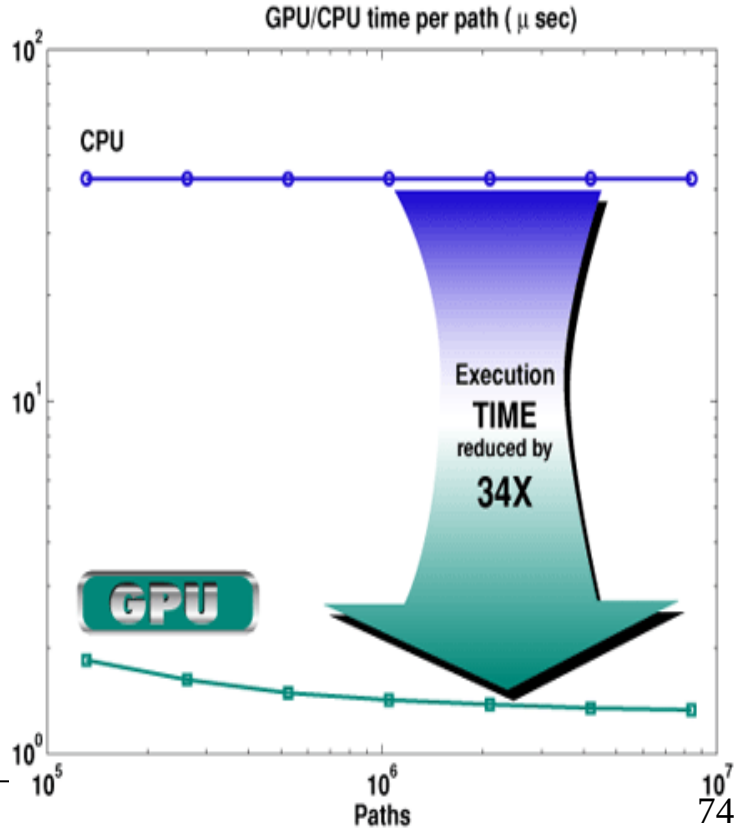


GPU加速定价模型

- SciFinance 是专门从事建造衍生定价和风险模型的公司

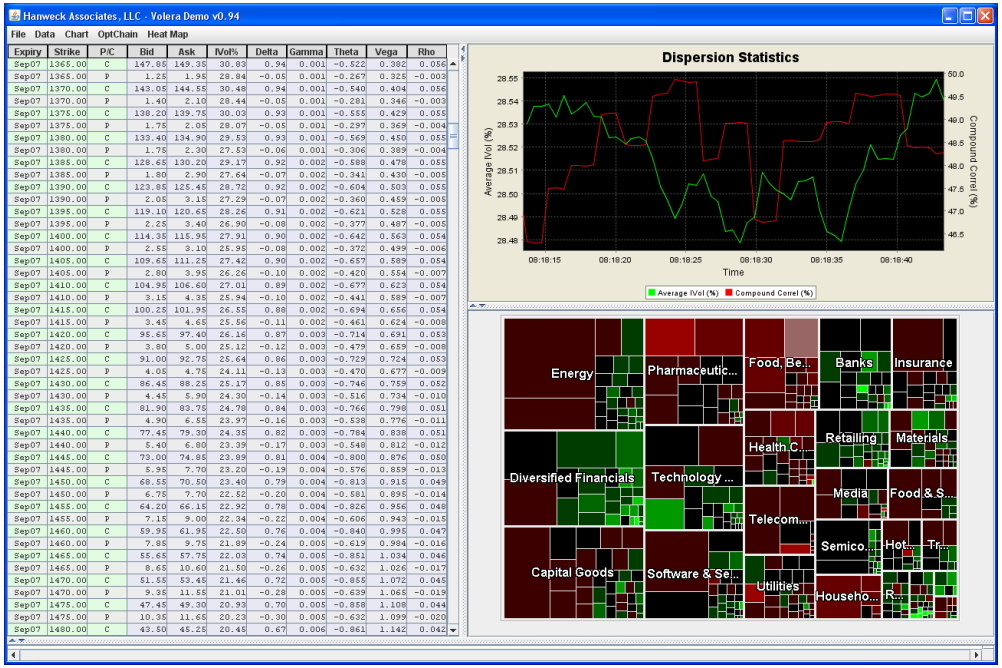
- 利用CUDA技术， Monte Carlo定价模型性能提高30~100 倍

Serial	OpenMP (quad-core PC)	Single GPU	Dual GPU
43.3 sec	11.0 sec (x 3.94)	1.27 sec (x 34)	0.77 sec (x 56)



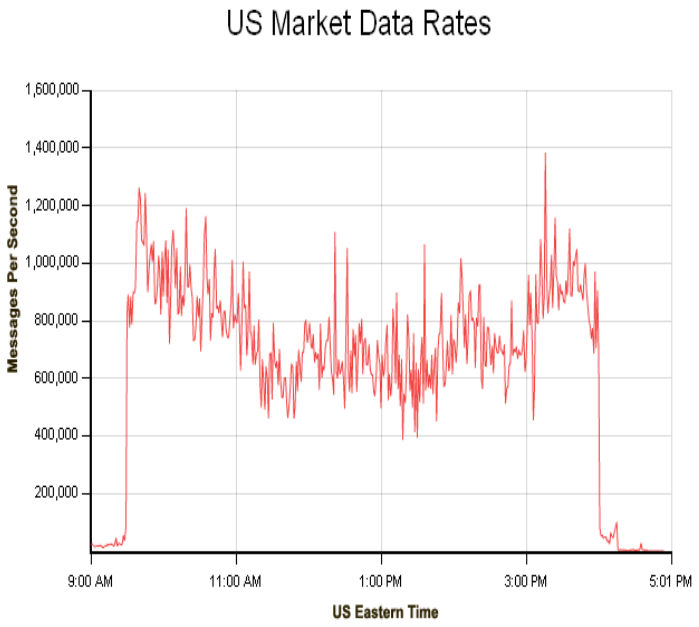
GPU加速期权定价

- VOLERA、实时期权隐含波动引擎
- 单精度的准确结果
- 在不到1秒钟的时间内，评估所有美国上市的股票期权
- 参见：www.hanweckassoc.com



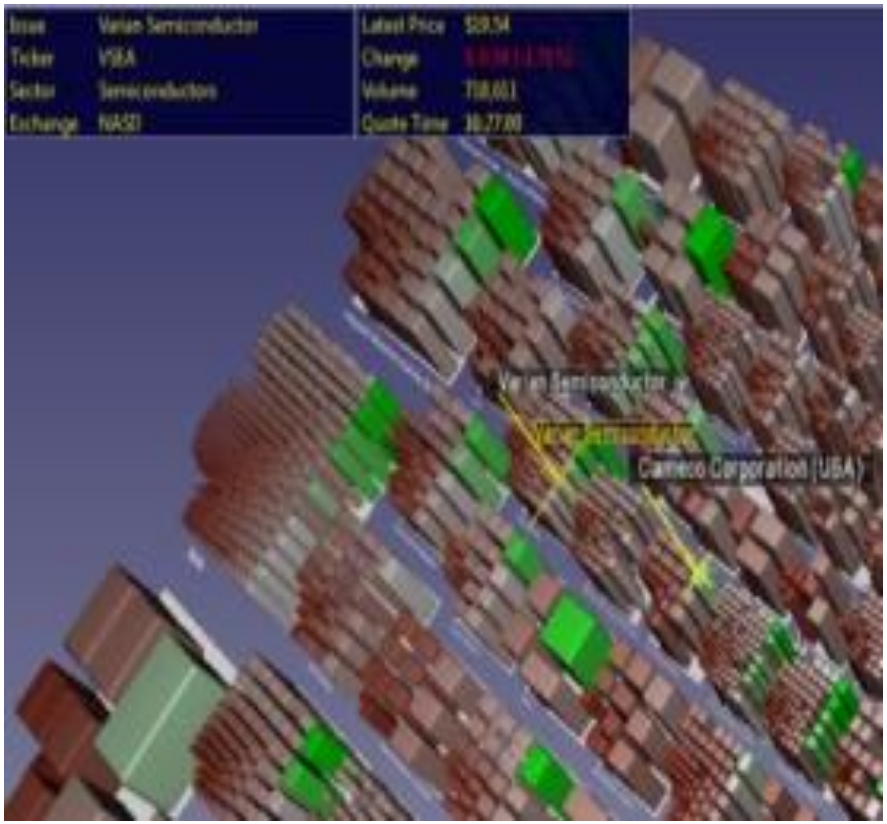
GPU加速风险分析

- Exegy 是专门为实时数据处理提供硬件加速的公司
- 利用 Nvidia GPU， 性能提高 180 倍
- 参见：http://www.exegy.com/PDFs/WHT-0001-A_final.pdf



GPU加速市场数据三维可视化

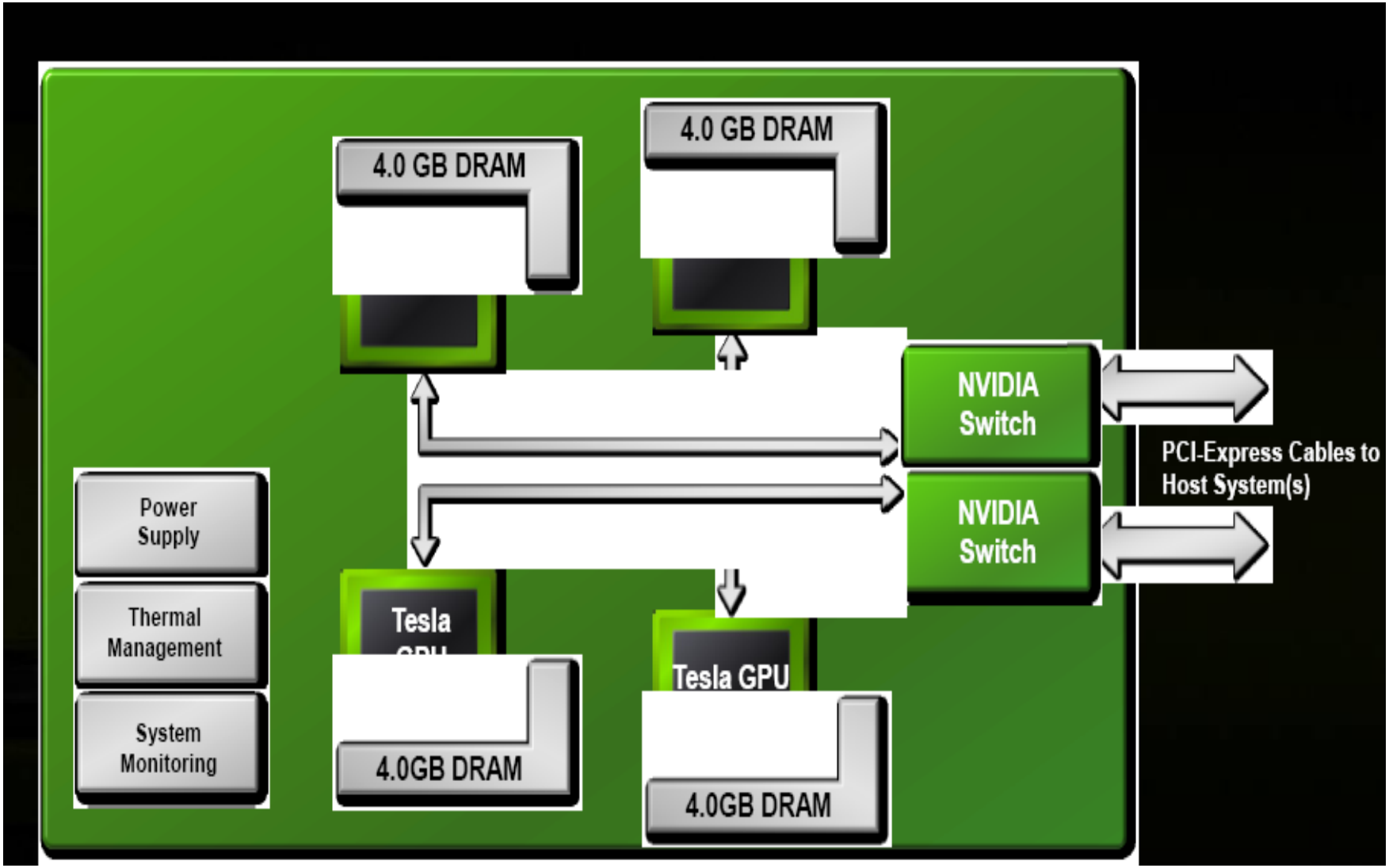
- AQUMIN 是一家金融工具提供商
- Aqumin 的 AlphaVision 把金融数字转换为三维模式，以更加直观的方式显示
- 参见：https://www.aqumin.com/Home/tabid/36/Default.aspx



Tesla S1070 1U系统集群解决方案

Processors	4xTesla T10
Number of cores	960
Core Clock	1.296GHz
Performance	4 Teraflops
Total system memory	16.0 GB (4.0GB per T10)
Memory bandwidth	408 GB/sec peak (102 GB/sec per T10)
Memory I/O	2048-bit, 1.6GHz GDDR3 (512-bit per T10)
Form factor	1U (EIA 19’’ rack)
System I/O	2 PCIe x16 Gen2
Typical power	700W

Tesla S1070 1U系统架构



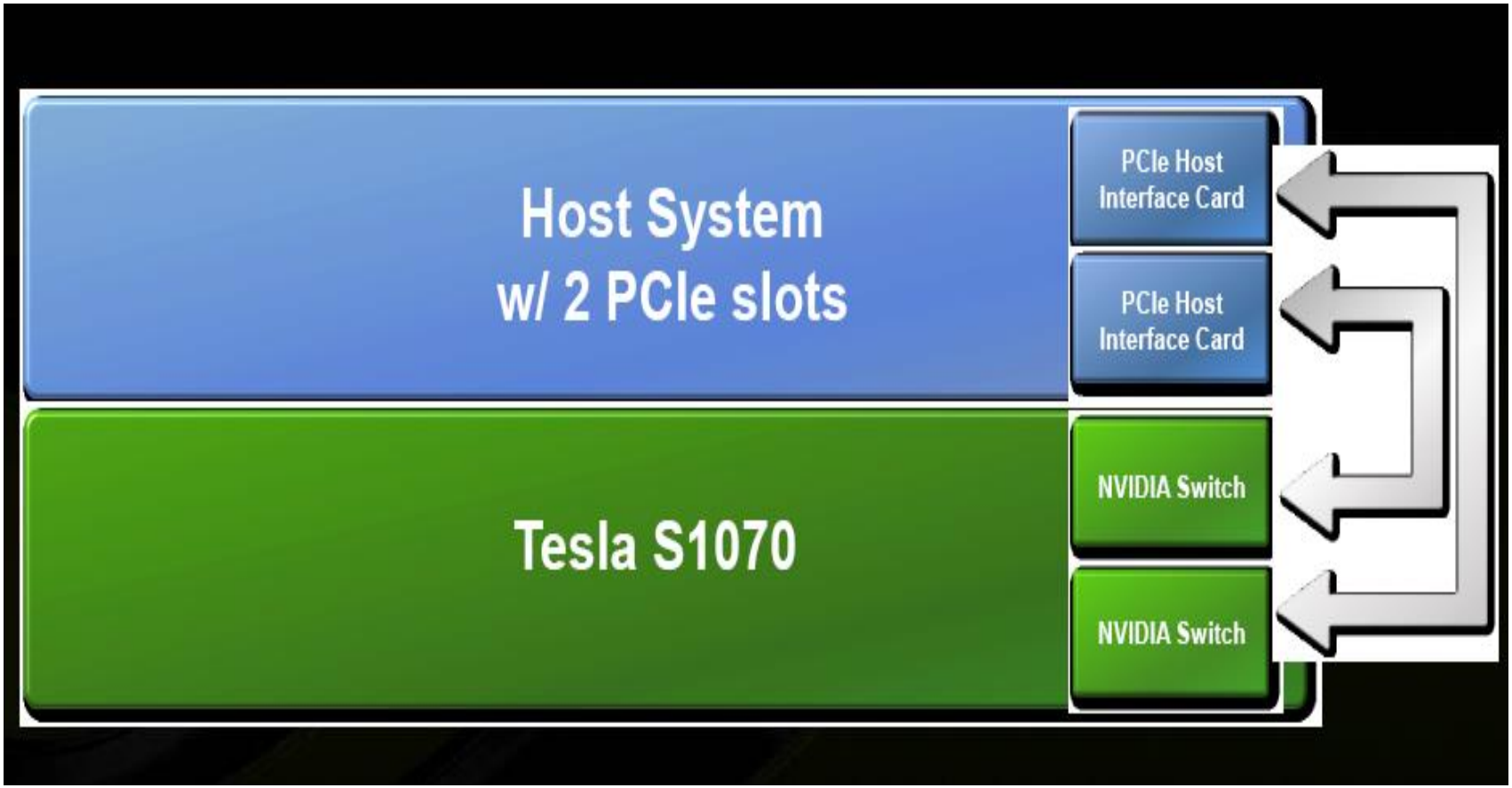
Tesla S1070 与服务器节点的连接

S1070

PCI-E Gen2 Cable(0.5m length)

PCI-E Gen2 Host Interface Card in Host

Tesla S1070 与节点服务器的连接

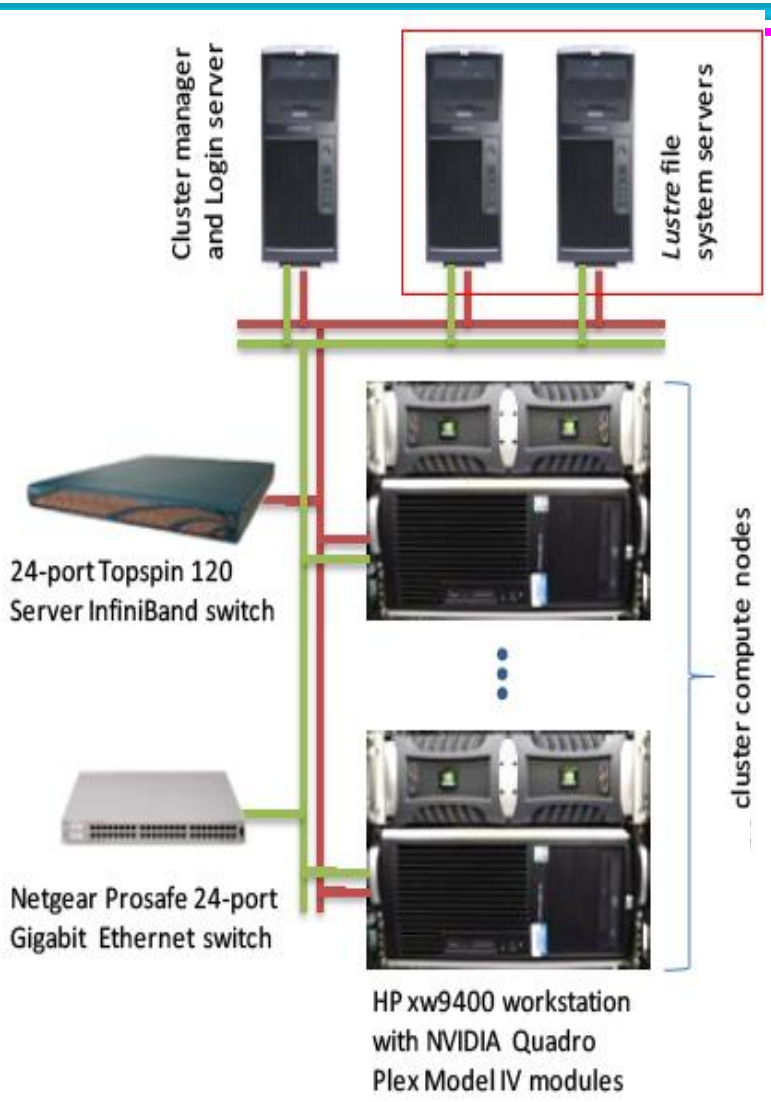


UIUC Accelerator Cluster

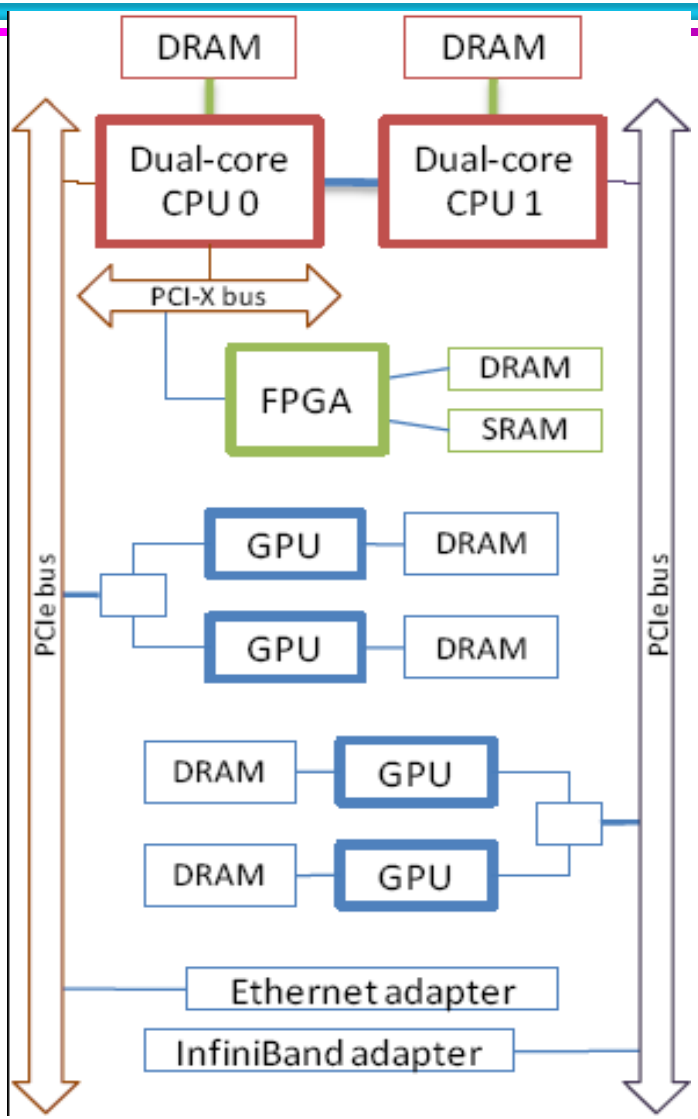
- Combining GPU and FPGA
- 32 compute nodes:
  - 2 dual-core 2.4 GHz AMD Opterons
  - 8 GB host memory
  - 1 NVIDIA Tesla S1070 containing 4 GT200 GPUs, each with 4 GB memory
  - PCI-E GEN 2 cable
  - Nallatech H101-PCIX FPGA accelerator, 16 MB SRAM, 512 MB SDRAM
- 2GB/sec InfiniBand connection
- Red Hat Enterprise Linux 5
- GNU C/Fortran and Intel C/Fortran compilers
- CUDA 2.0

http://www.ncsa.uiuc.edu/Projects/GPUcluster

UIUC Accelerator Cluster



Multi-core/GPU/FPGA cluster architecture



Compute node architecture

集群的网络连接

- 服务器节点之间的连接:InfiniBand
  - 目前全球带宽最高的高速网络互联技术
  - 专门针对服务器端的连接而设计
  - 高扩展性：在每个子网内支持上万个节点
  - 高吞吐量： 2008年已达到40Gb/sec
  - 低延迟： 1 微秒，以太网的1/10
- 集群子系统之间的连接：千兆以太网

集群软件

- 操作系统：
  - Red Hat Enterprise Linux等
- 集群管理软件：
  - 开源或商业版的Rocks
- 集群编程：
  - Message Passing Programming (MPI)

NVIDIA TESLA打造性能强大的个人超级计算机

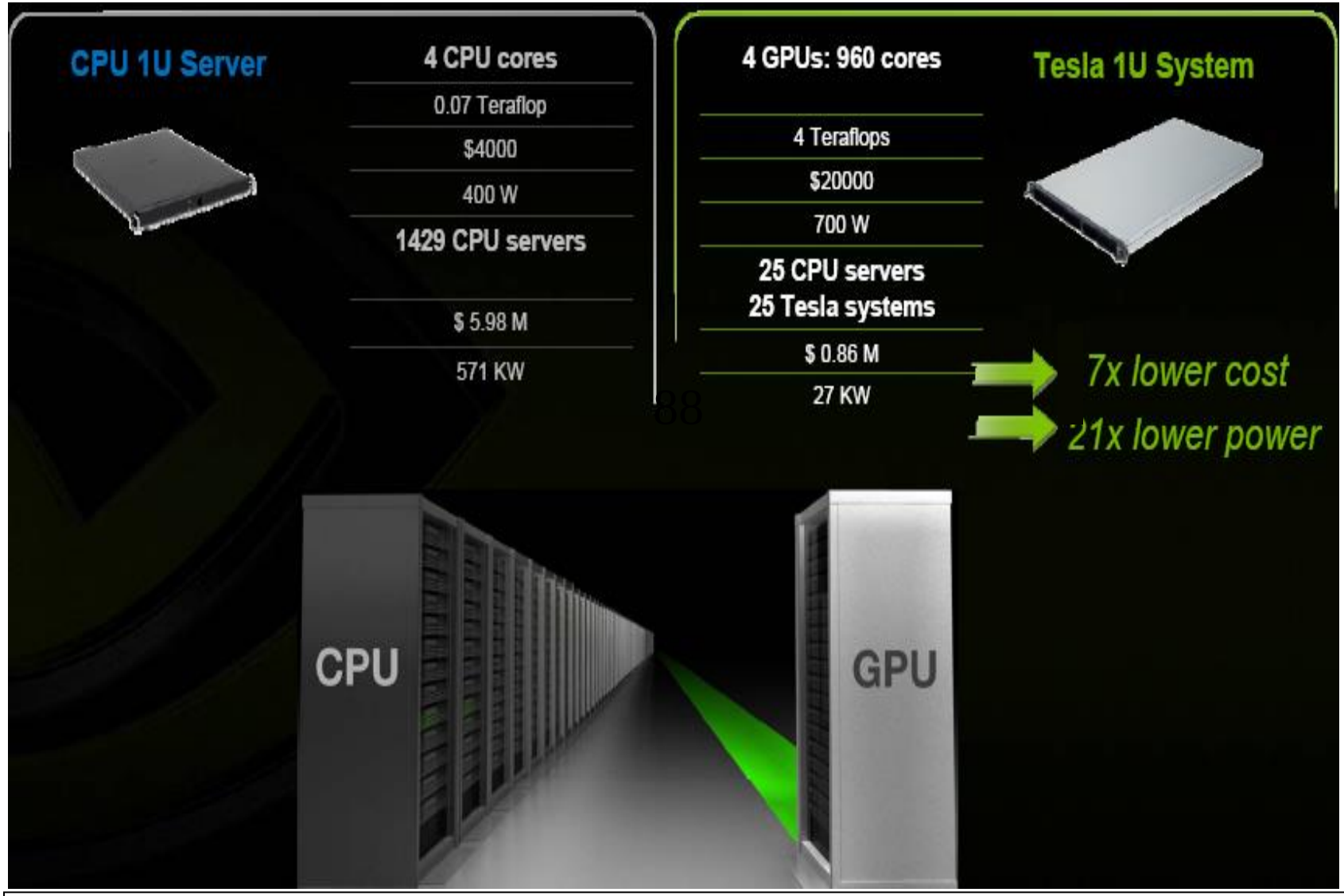
AMAX（美国）、Armari（英国）、华硕（全球）、Azken Muga（西班牙）、Boxx（美国）、CAD2（英国）、CADnetwork（德国）、Carri（法国）、Colfax （美国）、Comptronic（德国）、Concordia（意大利）、Connoisseur（印度）、戴尔（全球）、Dospara （日本）、E-Quattro（意大利）、Founder（中国），Inspur (中国），JRTI（美国）、联想（全球）、Littlebit（瑞士）、Meijin （俄罗斯）、Microway（美国）、Sprinx（捷克）、Sysgen（德国）、Transtec（德国）、Tycrid （美国）、Unitcom（日本）、Ustar（乌克兰）、Viglen（英国）、Western Scientific（美国）

“Democratization” of Power

Name	Year	# Processors	Tflops/$1 million
ILLIAC IV	1976	64	0.00000048
CRAY Y-MP	1988	8 vector processors	0.000115
ASCI RED	1997	4510	0.01818182
EARTH SIMULATOR	2002	5120	0.0175
BLUE GENE/L	2004	65536	2.8
PLAYSTATION 3 CLUSTER	2007	8 PlayStation 3s	375
ROADRUNNER	2008	19440	8.3
NVIDIA TESLA	2008	960 cores	439.1

Personal Supercomputer!

搭建一个100 TF的数据中心

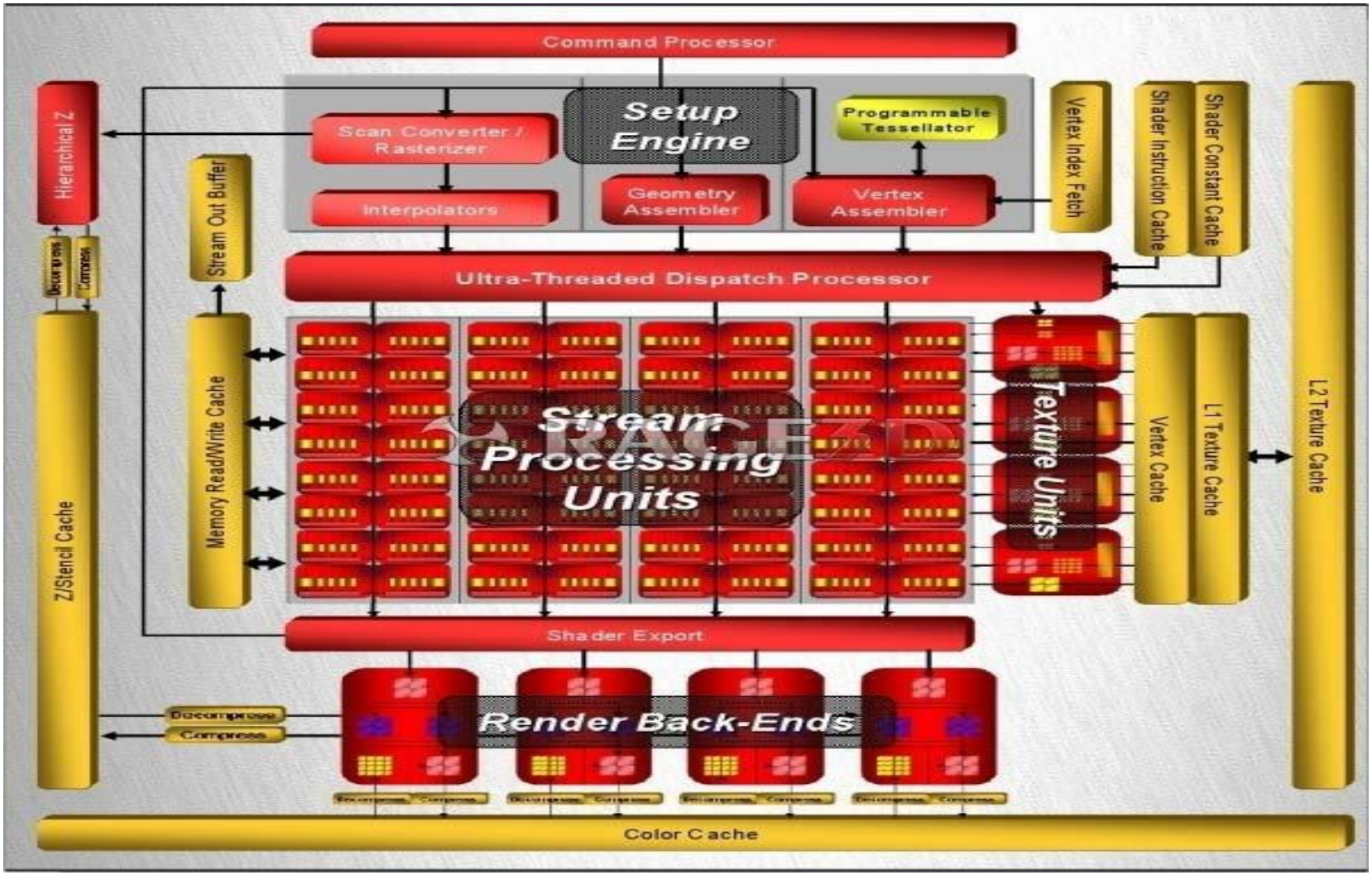


A Great Opportunity for Many

- GPU parallel computing allows
  - Drastic reduction in “time to discovery”
  - 1st principle-based simulation at meaningful scale
  - New, 3rd paradigm for research: computational experimentation

- The “democratization” of power to discover
  - $2000/Teraflops in personal computers today
  - Cost will no longer be the main barrier for big science

AMD(ATI) R600, RV770



R600

Comparison of Hardware Specifications

Factor	Nvidia(GT200)	AMD(RV770)
Components of a Processing Unit	8SP+2SFU+1DU	5ALU+1SFU+1DU
Num of Processing Units	240	160
Thread	Thread: SP 1:1	Thread: ALU 1:5
Frequency of Processor Cores	1296MHz	750MHz
DRAM	GDDR3	GDDR5
On-chip cache	Yes	No
Global Memory	Yes	Yes

ATI's Stream Computing

- Brook: General purpose streaming language (Stanford University, 2003)
  - Implementation for stream programming
  - An extension to C-language
  - Demonstrate GPU streaming coprocessor
  - Make programming GPUs easier
    - Hide texture/pbuffer data management
    - Hide graphics based constructs in CG/HLSL
    - Hide rendering passes
- Brook+
  - An implementation by AMD of Brook on AMD's compute abstraction layer with some enhancements

Example

```
kernel void sum(float a<>, float b<>, out float c<>)
{
    c = a + b;
}

int main(int argc, char** argv)
{
    int i, j;
    float a<10, 10>;
    float b<10, 10>;
    float c<10, 10>;

    float input_a[10][10];
    float input_b[10][10];
    float input_c[10][10];

    for(i=0; i<10; i++) {
        for(j=0; j<10; j++) {
            input_a[i][j] = (float) i;
            input_b[i][j] = (float) j;
        }
    }

    streamRead(a, input_a);
    streamRead(b, input_b);

    sum(a, b, c);

    streamWrite(c, input_c);
    ...
}
```

Kernels – Program functions that operate on stream elements

Streams – collection of data elements of the same type which can be operated on in parallel.

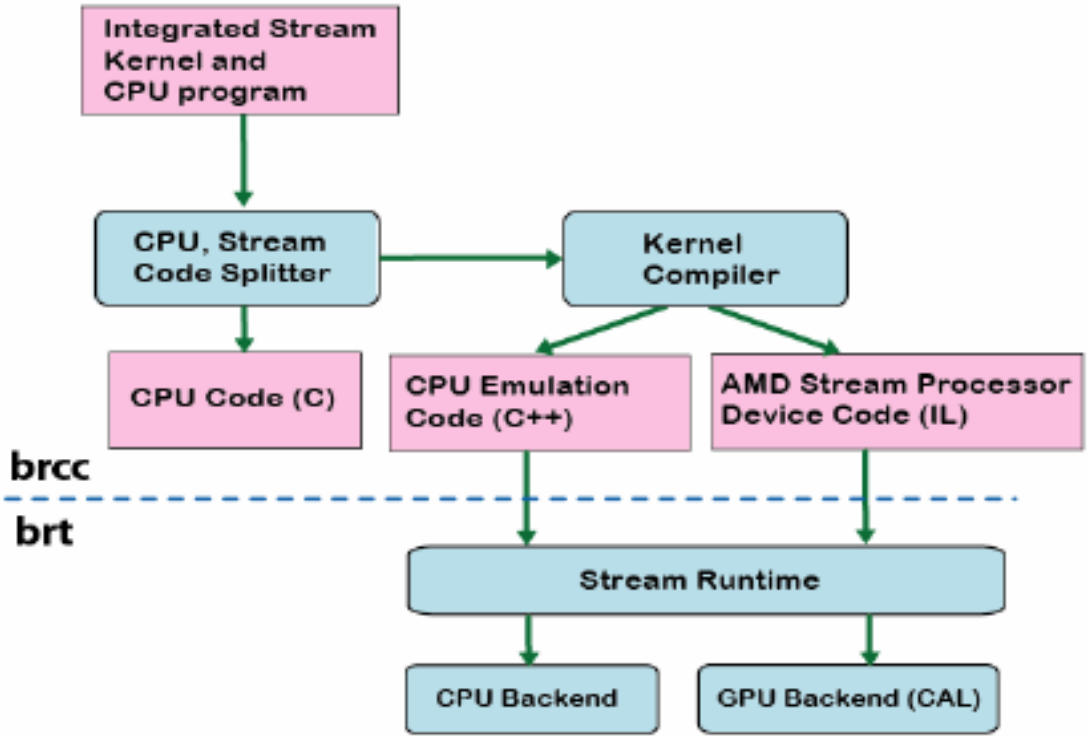
Brook+ access functions

3

Brook+ Compiler

AMD Smarter Choice

Converts Brook+ files into C++ code. Kernels, written in C, are compiled to AMD's IL code for the GPU or C code for the CPU.



OpenCL (Open Computing Language)

- Open standard for heterogeneous parallel programming
  - Suitable to various processors
    - Multi-core CPU, GPU, IBM Cell, DSP, FPGA, …
- Implementable on a range of embedded, desktop, and server systems
  - HPC, desktop, and handheld profiles in one specification
- Cross-vendor software portability
  - API abstractions just high enough to hide implementation specifics
- Compute models
  - Both data-parallel and task-parallel
- Low learning curve
  - C-based cross-platform programming interface