
Accelerating Science with
Kepler and CUDA 5
Jonathan Bentz, NVIDIA
NVIDIA

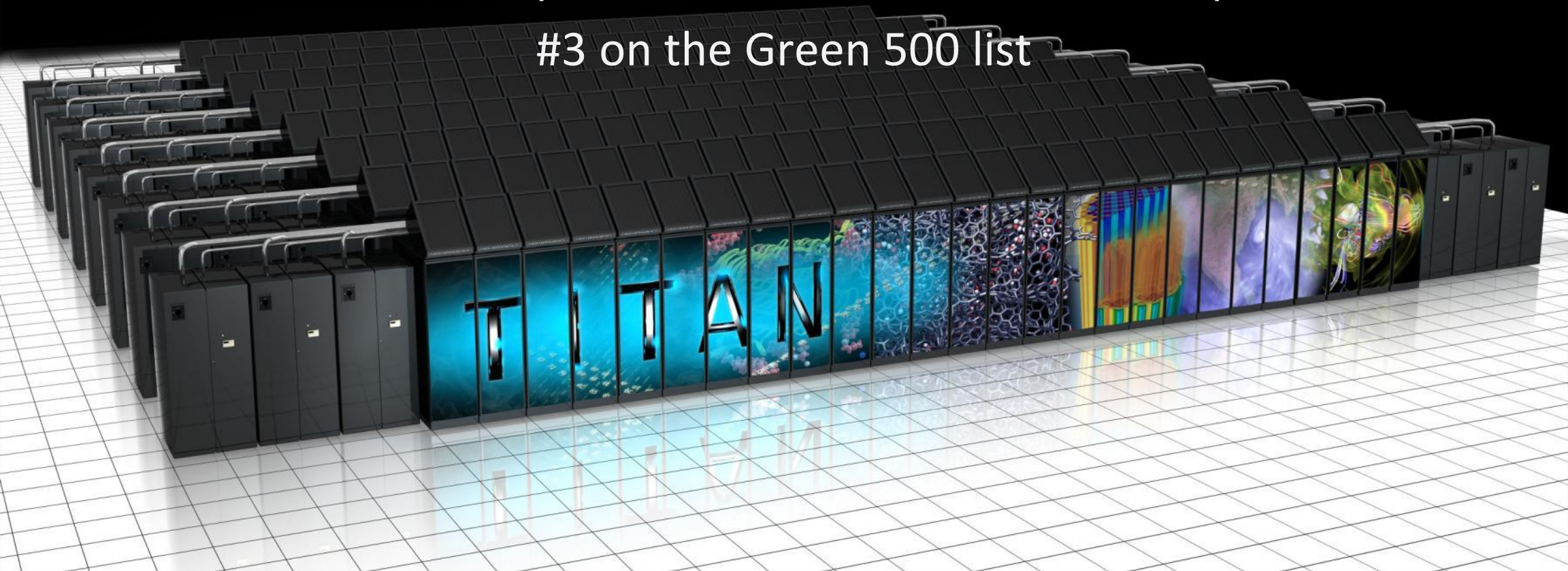
Titan: World's Fastest Supercomputer 2012

18,688 Tesla K20X GPUs

27 Petaflops Peak: 90% of Performance from GPUs

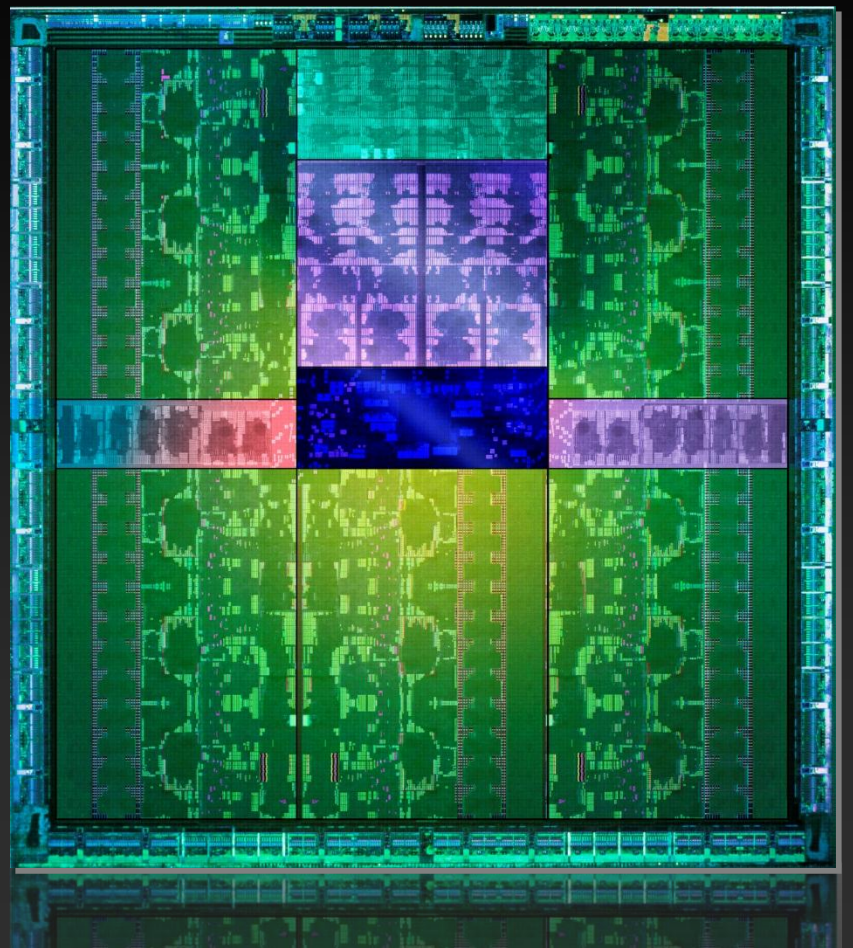
17.59 Petaflops Sustained Performance on Linpack

#3 on the Green 500 list



Kepler

Fastest, Most Efficient HPC Architecture Ever



SMX

Hyper-Q

Dynamic Parallelism

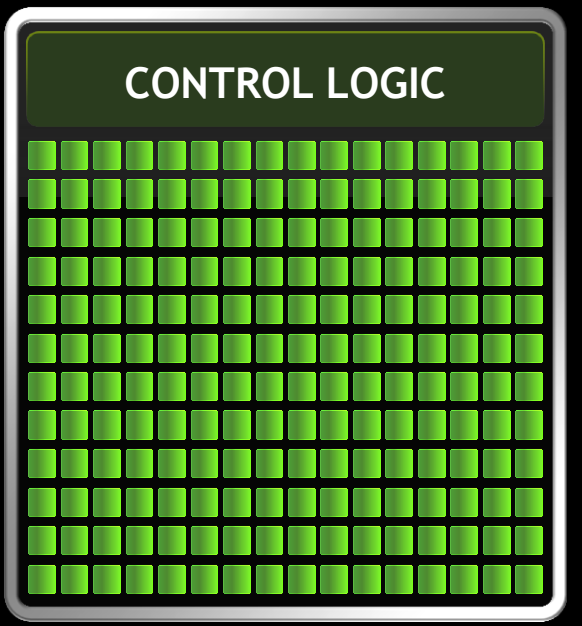
Kepler: Fast & Efficient

SM
M2090

32 cores

3x
Perf / Watt

SMX
K20



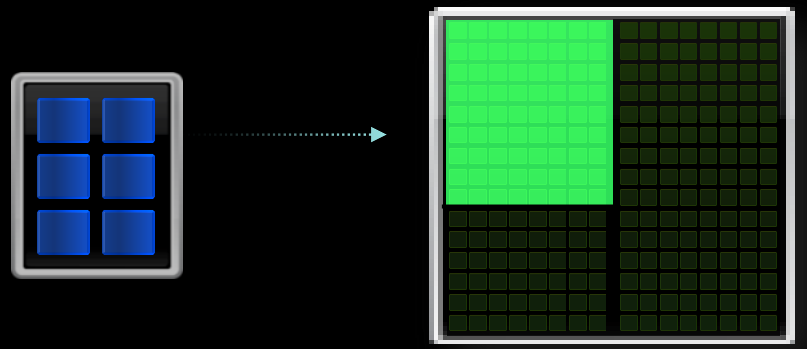
192 cores

Hyper-Q

CPU Cores Simultaneously Run Tasks on Kepler

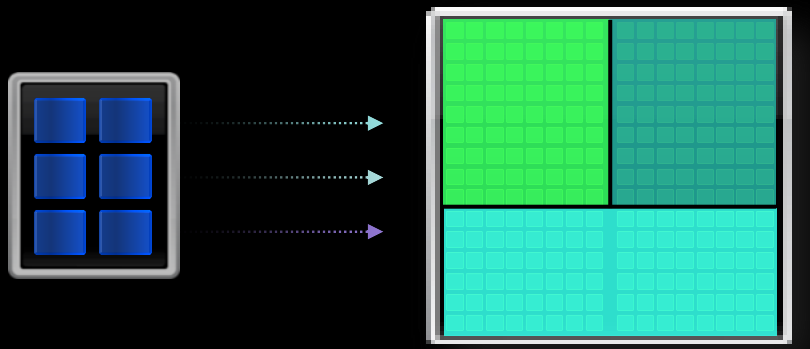
FERMI

1 MPI Task at a Time



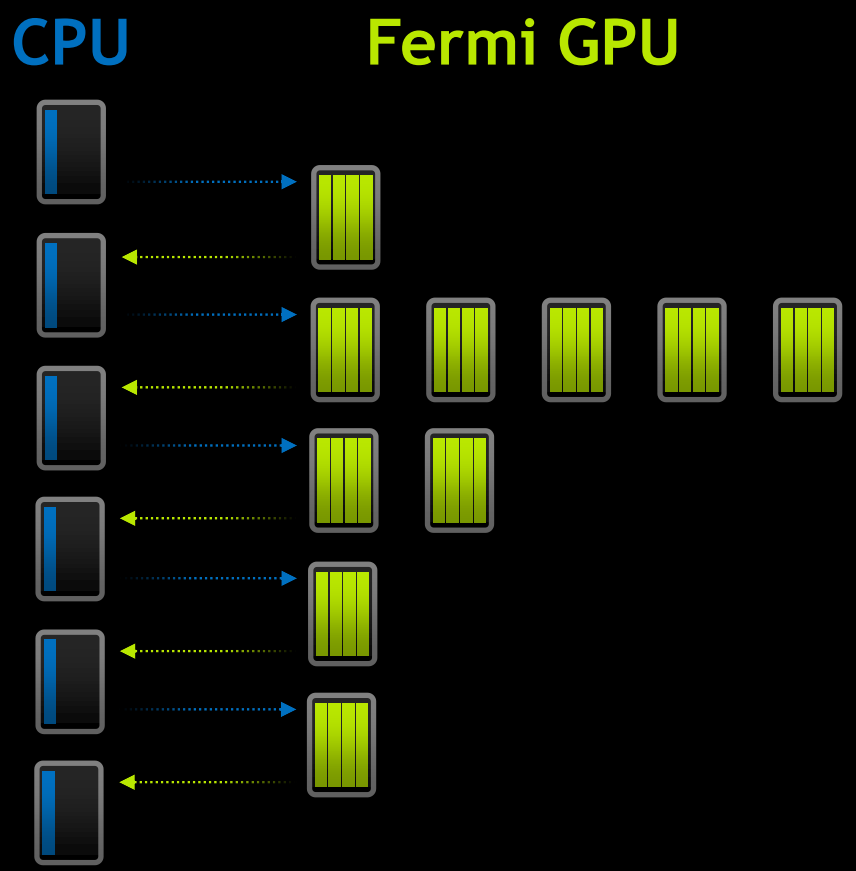
KEPLER

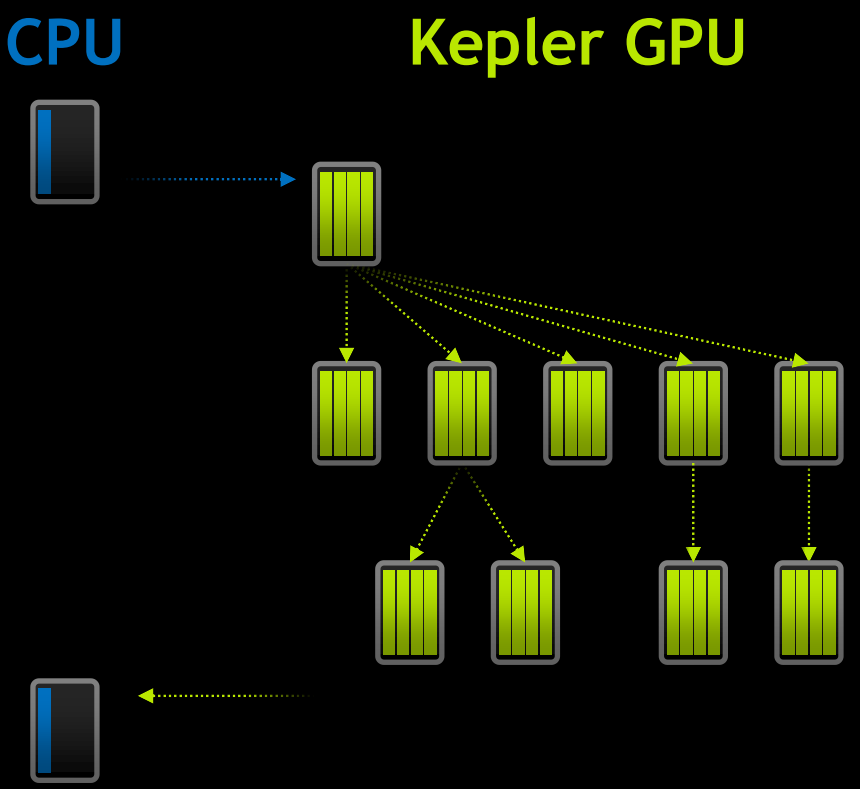
32 Simultaneous MPI Tasks



Dynamic Parallelism

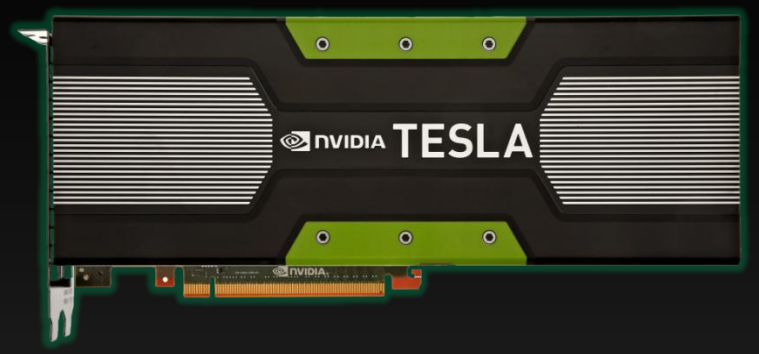
GPU Adapts to Data, Dynamically Launches New Threads

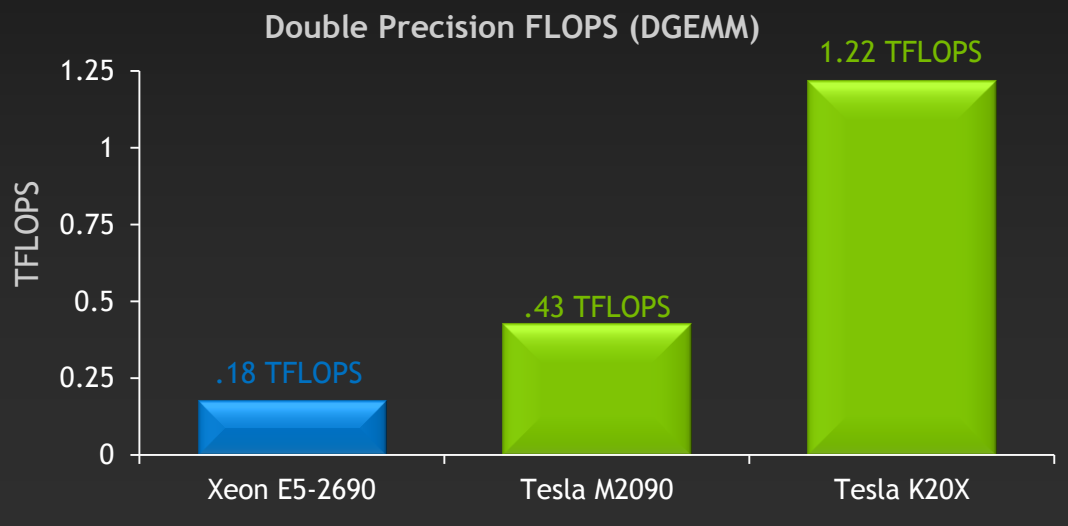




Tesla K20 Family: 3x Faster Than Fermi

Tesla K20X





	Tesla K20X	Tesla K20
# CUDA Cores	2688	2496
Peak Double Precision	1.32 TF	1.17 TF
Peak DGEMM	1.22 TF	1.10 TF
Peak Single Precision	3.95 TF	3.52 TF
Peak SGEMM	2.90 TF	2.61 TF
Memory Bandwidth	250 GB/s	208 GB/s
Memory size	6 GB	5 GB
Total Board Power	235W	225W

SMX Balance of Resources

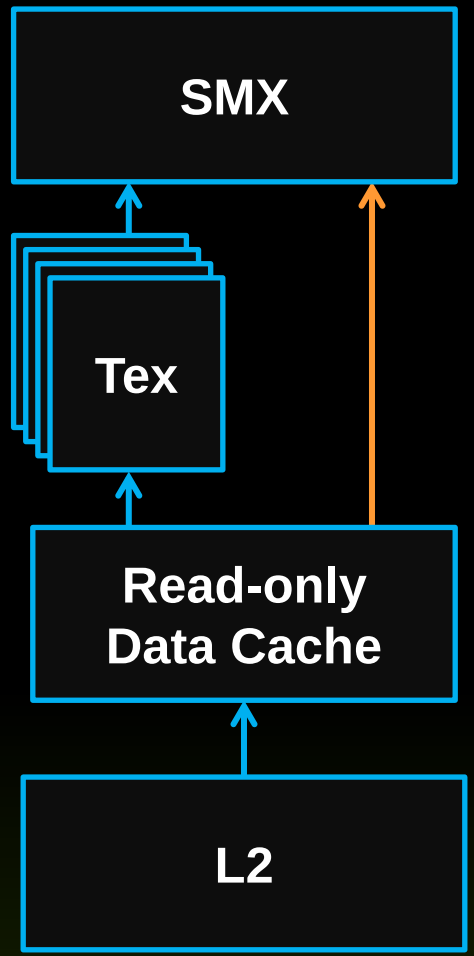
Resource	Kepler GK110 vs Fermi
Floating point throughput	**2-3x**
Max Blocks per SMX	**2x**
Max Threads per SMX	**1.3x**
Register File Bandwidth	**2x**
Register File Capacity	**2x**
Shared Memory Bandwidth	**2x**
Shared Memory Capacity	**1x**

New ISA Encoding: 255 Registers per Thread

- Fermi limit: 63 registers per thread
  - A common Fermi performance limiter
  - Leads to excessive spilling
- Kepler : Up to 255 registers per thread
  - Especially helpful for FP64 apps
- Ex. Quda QCD fp64 sample runs 5.3x faster
  - Spills are eliminated with extra registers

Texture Cache Unlocked

- Added a new path for compute
  - Avoids the texture unit
  - Allows a global address to be fetched and cached
  - Eliminates texture setup
- Why use it?
  - Separate pipeline from shared/L1
  - Highest miss bandwidth
  - Flexible, e.g. unaligned accesses
- Managed automatically by compiler
  - “const __restrict” indicates eligibility



Kepler GK110 Memory System Highlights

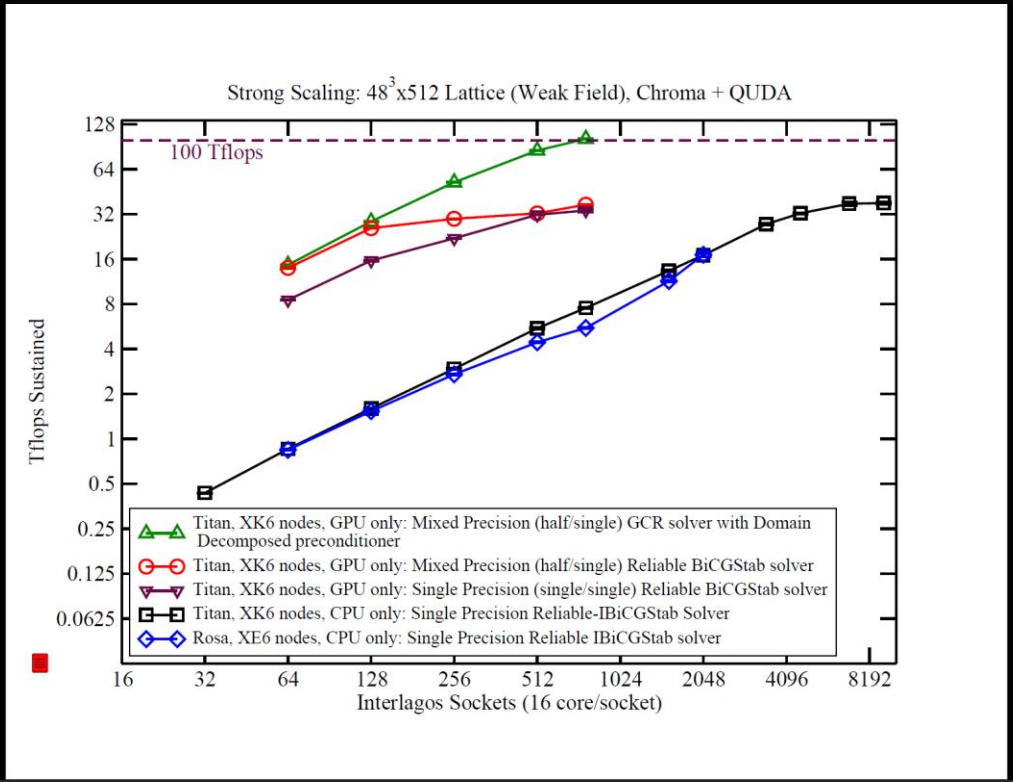
- Efficient memory controller for GDDR5
  - Peak memory clocks achievable
- More L2
  - Double bandwidth
  - Double size
- More efficient DRAM ECC Implementation
  - DRAM ECC lookup overhead reduced by 66%
    (average, from a set of application traces)

Physics Applications

Application	Features Supported	GPU Perf	Release Status	Notes
Chroma	Wilson-clover fermions, Krylov solvers, Domain-decomposition	Up to 9x 768 GPUs vs. 768 (16-core) CPUs	Multi-GPU support November 2009 Available now	http://usqcd.jlab.org/usqcd-docs/chroma/
GTC	Electron push and shift (accounting for >80% of run time)	**2-3x** 128 GPUs vs. 128 (16-core) CPUs	Multi-GPU support Available now	Simulates microturbulence & transport in magnetically confined fusion plasma
MILC	Improved staggered fermions, Krylov solvers, Guage-link fattening	5-6x	Multi-GPU support November 2009 Available now	http://physics.indiana.edu/~sg/milc.html
QUDA	Wilson, Wilson-clover, Twisted mass, Improved staggered (asqtad or HISQ) and Domain wall	Up to 9x at various scales	Available Multi GPU	http://lattice.github.com/quda/

Accessible Capability-class Supercomputing

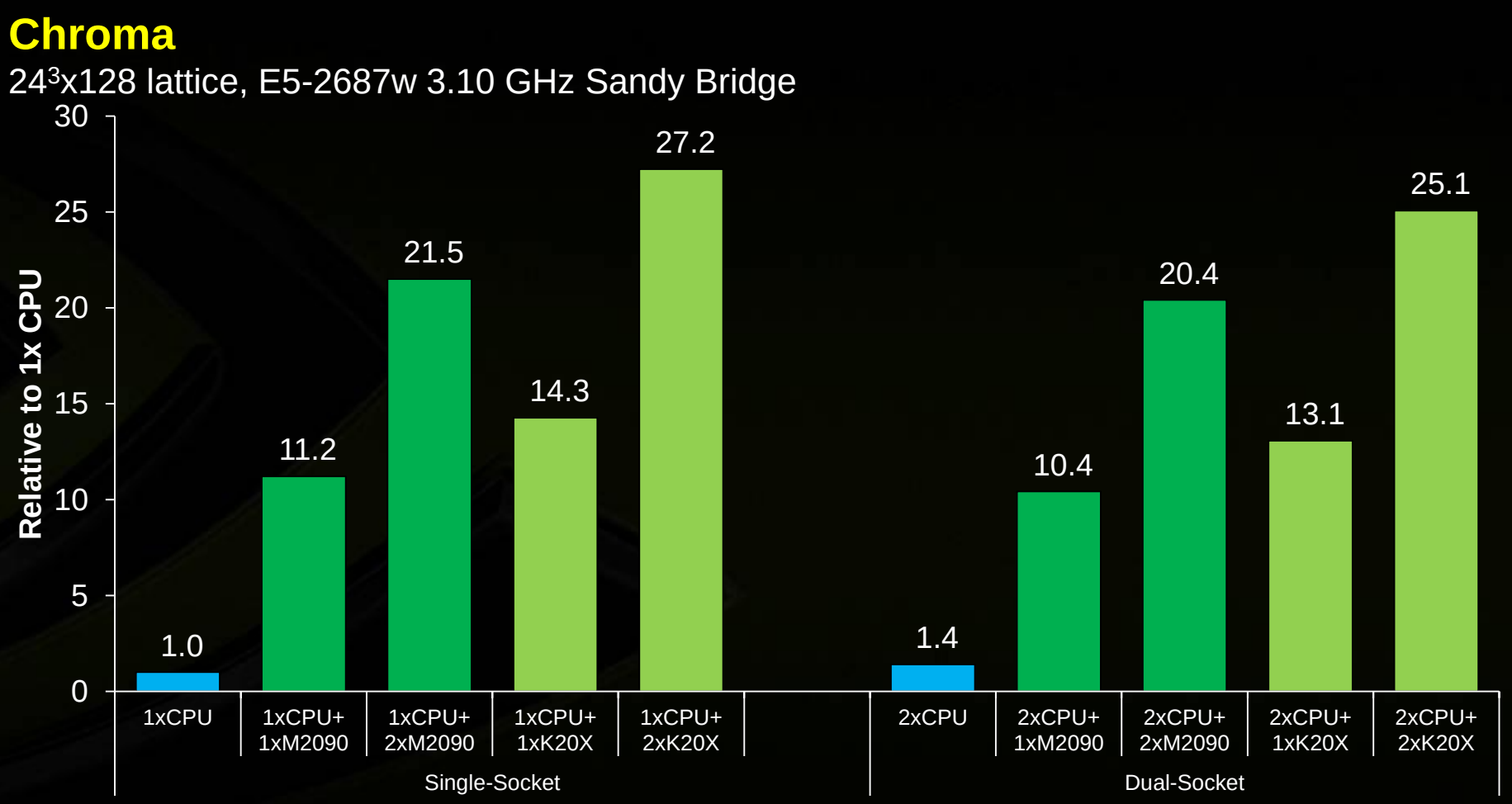
Scaling to 1000s of GPUs



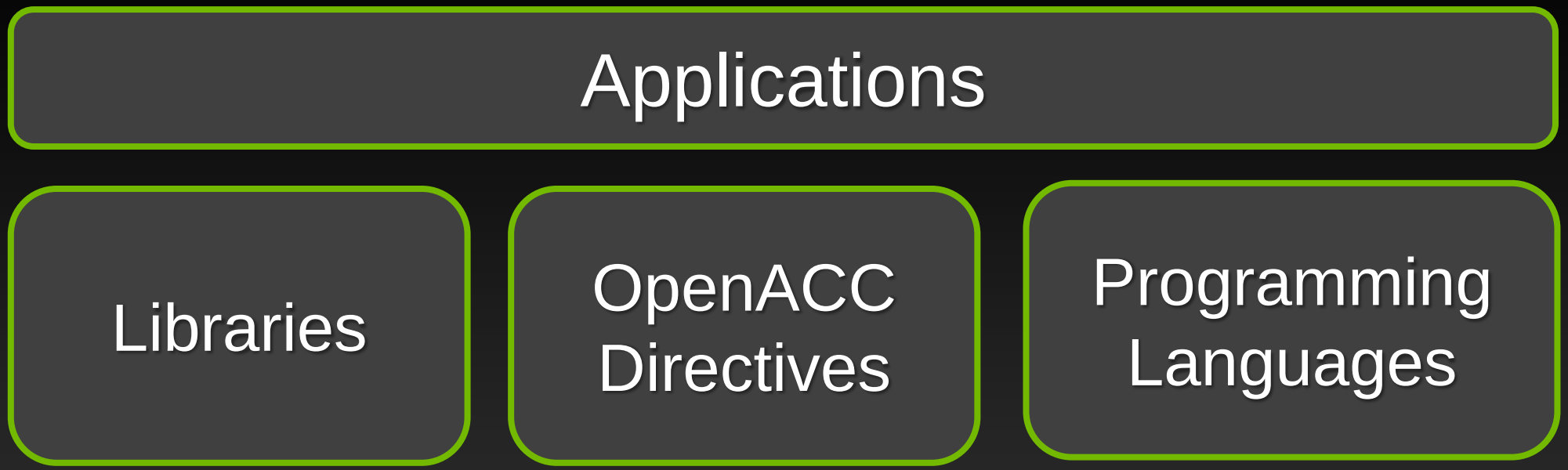
- 768 Tesla X2090 delivers >100 TFLOPS
- 3x better performance than ~8000 CPU cores

- M.A. Clark, R. Babich, “High-efficiency Lattice QCD computations on the Fermi architecture,” InPar 2012
- GPU performance scaled down to provide equivalent BiCGStab solver FLOPS. All solves performed to double-precision accuracy. Cray solvers used mixed (double-single) precision, BG/P solver used only double precision.

Chroma – High Energy & Nuclear Physics



3 Ways to Accelerate Applications



CUDA Math Libraries

High performance math routines for your applications:

- cuFFT Fast Fourier Transforms Library
- cuBLAS Complete BLAS Library
- cuSPARSE Sparse Matrix Library
- cuRAND Random Number Generation (RNG) Library
- NPP Performance Primitives for Image & Video Processing
- Thrust Templated Parallel Algorithms & Data Structures
- math.h C99 floating-point Library

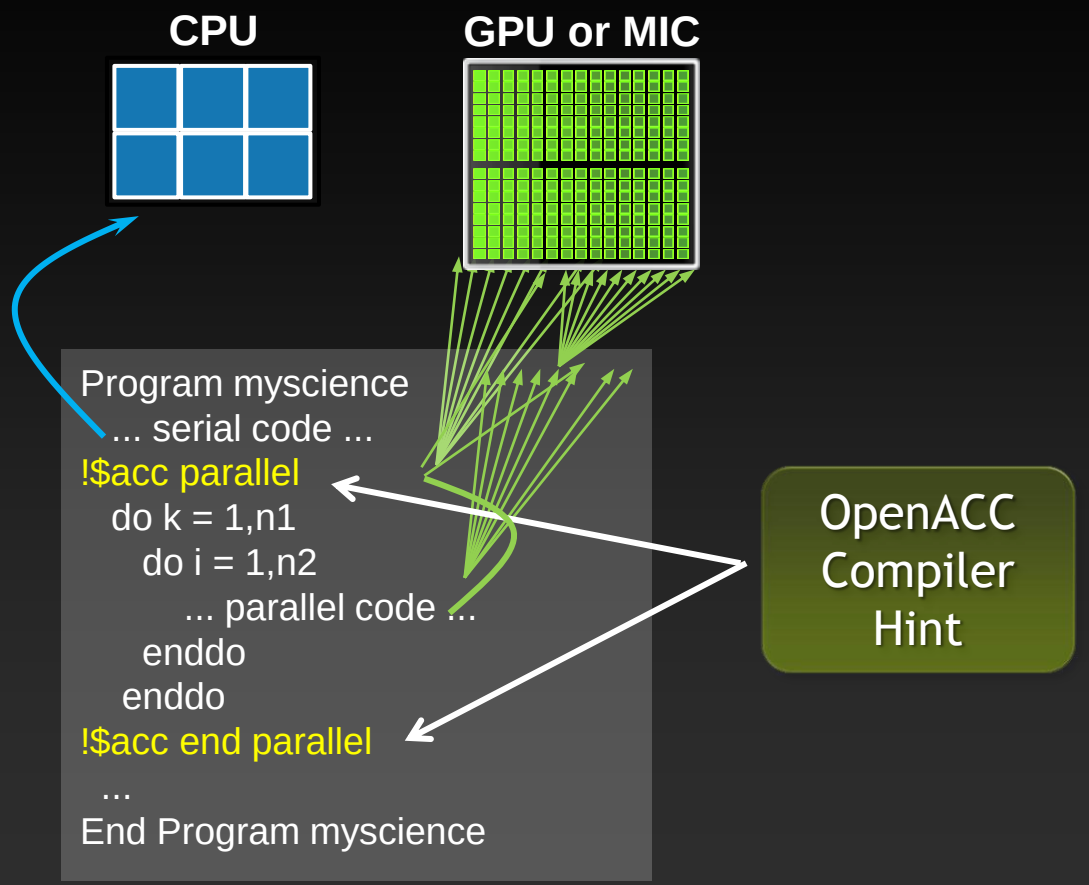
Included in the CUDA Toolkit: www.nvidia.com/getcuda (free download)

For more information on CUDA libraries:

http://developer.download.nvidia.com/GTC/PDF/GTC2012/PresentationPDF/S0629-Monday-CUDA-Accelerated.pdf

OpenACC Directives

Portable for All Accelerators



Directives = Simple hints

Compiler Parallelizes code

Works on multicore CPUs, GPUs, and Intel MIC

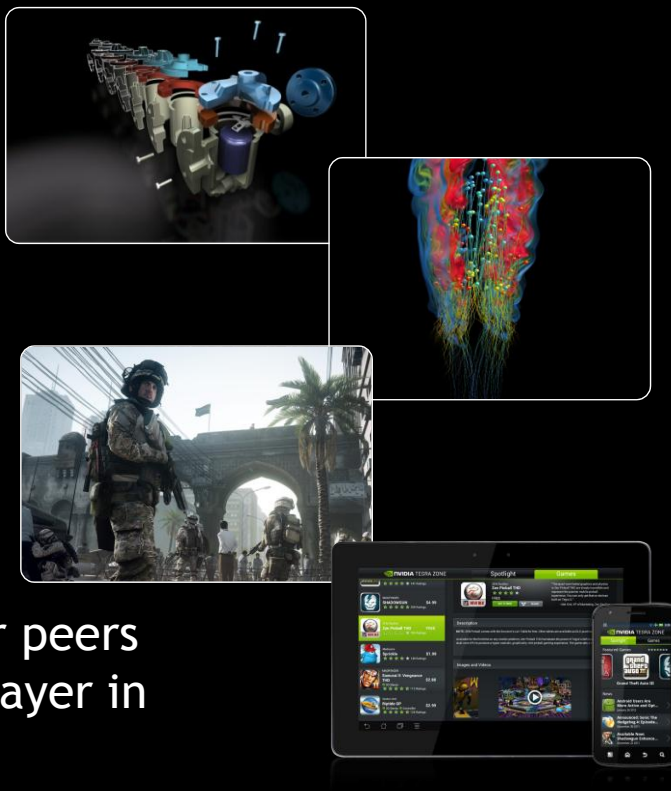
Open, Industry Standard

http://developer.nvidia.com/openacc

GPU Technology Conference 2013
March 18-21 | San Jose, CA

Why attend GTC?

GTC advances global awareness of the dramatic changes we're seeing in science and research, graphics, cloud computing, game development, and mobile computing, and how the GPU is central to innovation in all areas.



Ways to participate

- Submit a Research Poster – share your work and gain exposure as a thought leader
- Register – learn from the experts and network with your peers
- Exhibit/Sponsor – promote your organization as a key player in the GPU ecosystem

Visit www.gputechconf.com

GPU Technology Conference 2013
March 18-21 | San Jose, CA

- Felice Pantaleo, CERN – “Real-time Triggering Using GPUs in High Energy Physics”
- Valerie Halyo, Princeton – “GPUs for the Large Hadron Collider (LHC) Trigger System to Extend the Physics Discovery Reach”
- Mathias Wagner, U. Bielefeld – “GPUs Immediately Relating Lattice QCD to Collider Experiments”
- Balint Joo, JLab - “Lattice QCD on GPUs Using Chroma and QUDA”
- Frank Winter, U. Edinburgh – “QDP-JIT: A C++ Embedded Domain-Specific Language for Lattice Field Theory for GPU-Enabled Parallel Systems”

www.gputechconf.com


CUDA Toolkit 5.0
Performance Report
January 2013
NVIDIA

CUDA Math Libraries

High performance math routines for your applications:

- cuFFT Fast Fourier Transforms Library
- cuBLAS Complete BLAS Library
- cuSPARSE Sparse Matrix Library
- cuRAND Random Number Generation (RNG) Library
- NPP Performance Primitives for Image & Video Processing
- Thrust Templated Parallel Algorithms & Data Structures
- math.h C99 floating-point Library

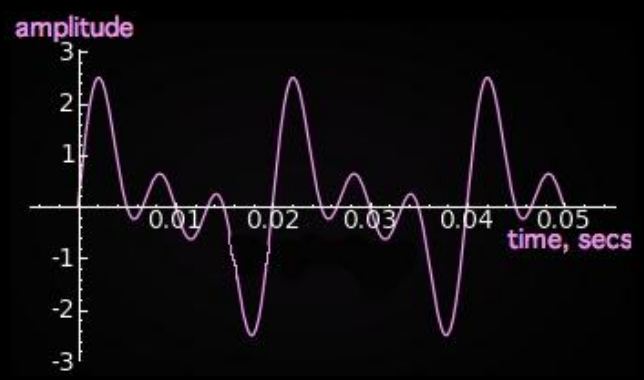
Included in the CUDA Toolkit: www.nvidia.com/getcuda (free download)

For more information on CUDA libraries:

http://developer.download.nvidia.com/GTC/PDF/GTC2012/PresentationPDF/S0629-Monday-CUDA-Accelerated.pdf

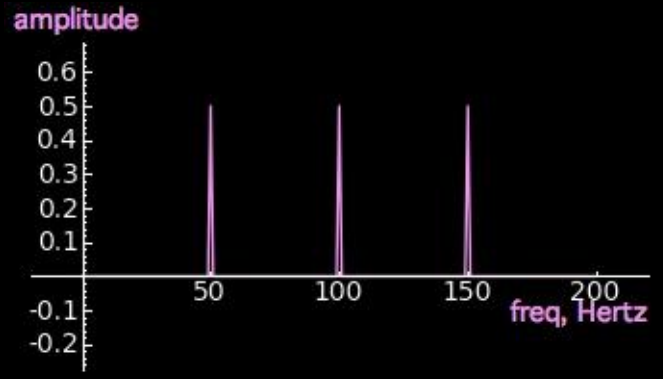
cuFFT: Multi-dimensional FFTs

- Real and Complex data types
- Single- and double-precision
- 1D, 2D and 3D batched transforms
- Flexible input and output data layouts
  - Similar to the FFTW “Advanced Interface”



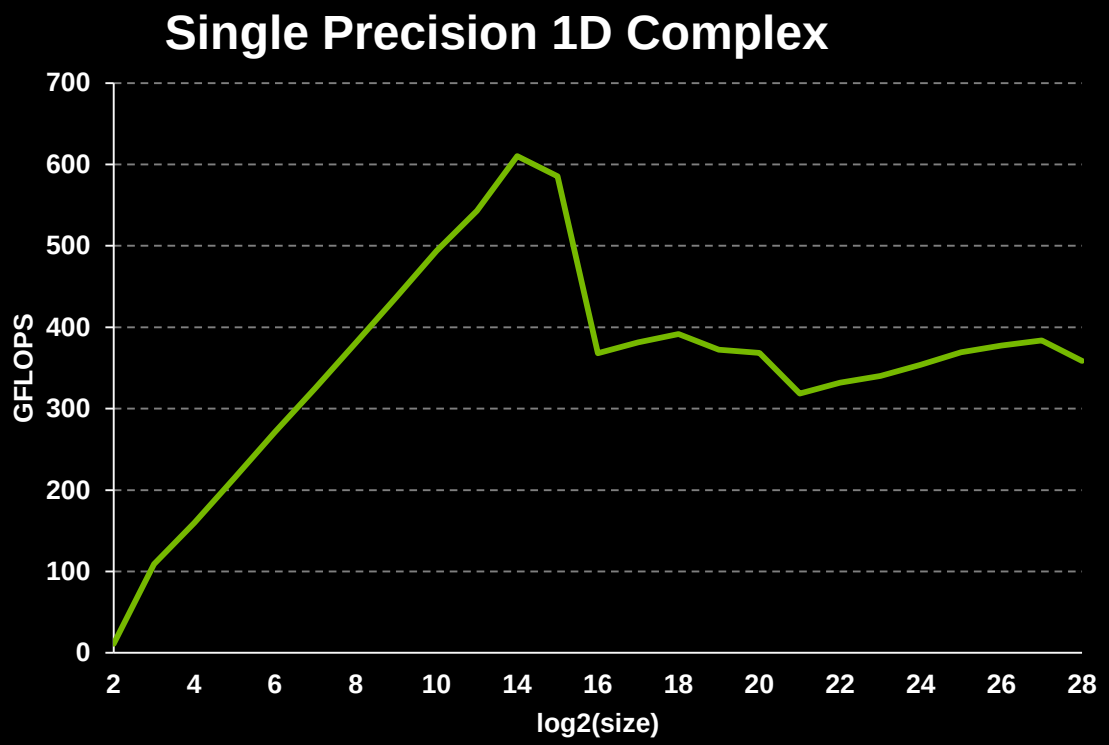
$$F(x) = \sum_{n=0}^{N-1} f(n) e^{-j2\pi(x\frac{n}{N})}$$

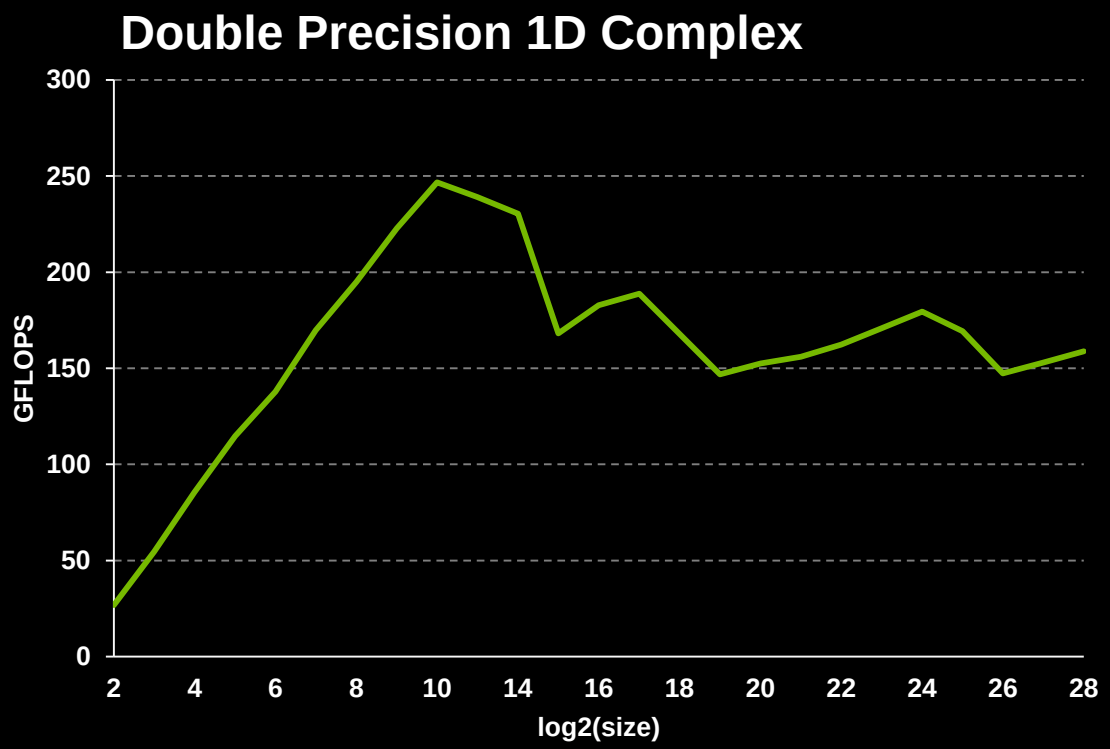
$$f(n) = \frac{1}{N} \sum_{n=0}^{N-1} F(x) e^{j2\pi(x\frac{n}{N})}$$



cuFFT: up to 600 GFLOPS

1D used in audio processing and as a foundation for 2D and 3D FFTs





Performance may vary based on OS version and motherboard configuration

• cuFFT 5.0 on K20X, input and output data on device

cuBLAS: Dense Linear Algebra on GPUs

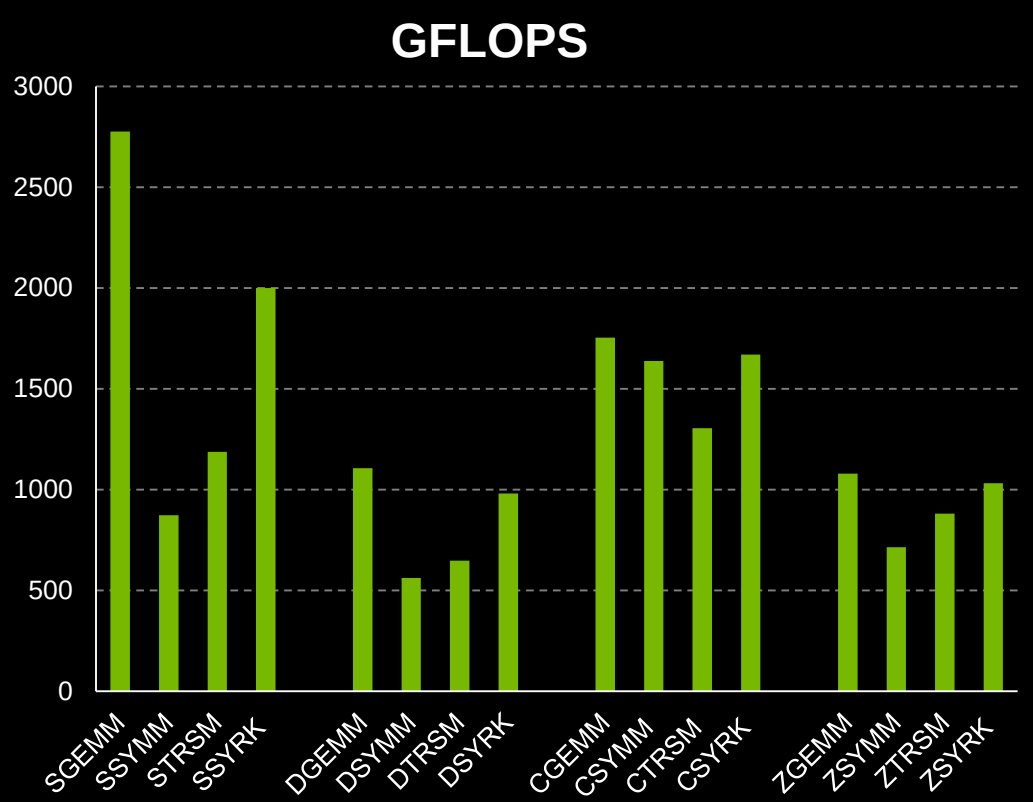
- Complete BLAS implementation plus useful extensions
  - Supports all 152 standard routines for single, double, complex, and double complex

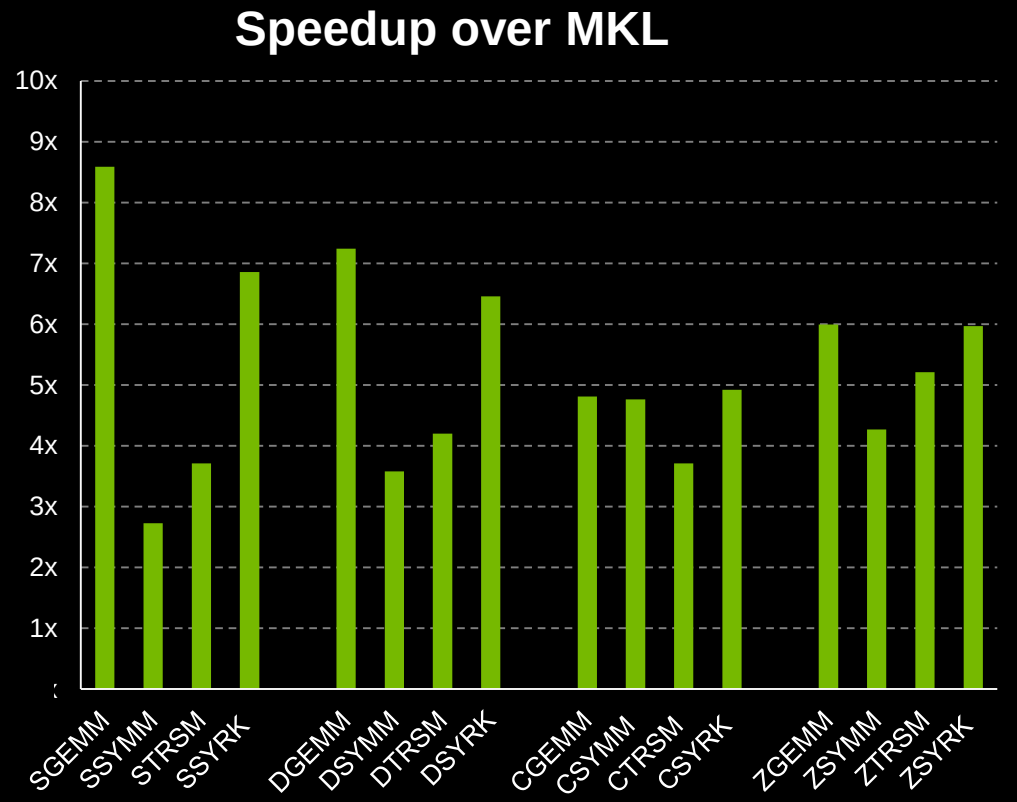
- New in CUDA 5.0:
  - cuBLAS interface callable from device kernels on K20, K20X
  - Full list of new features and optimizations:

    http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#cublas

    http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#cublas-performance-improvements

cuBLAS: >1 TFLOPS double-precision

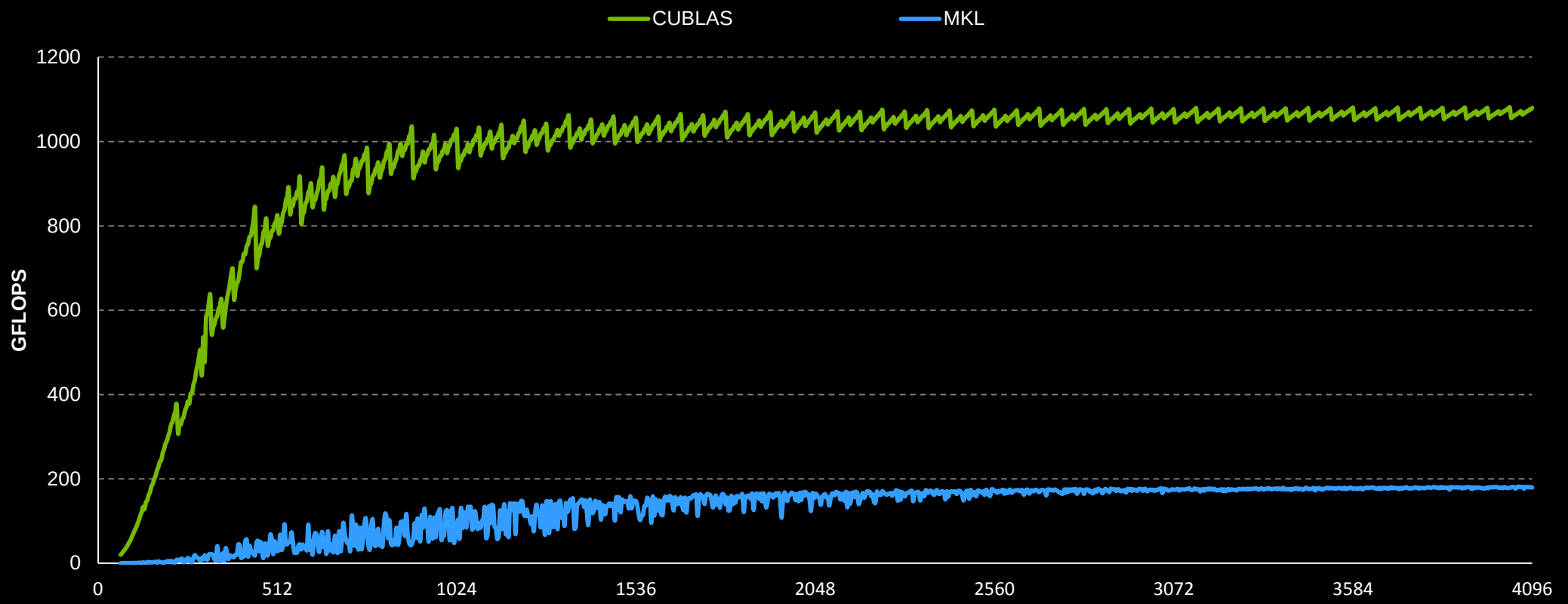




- cuBLAS 5.0 on K20X, input and output data on device
- MKL 10.3.6 on Intel SandyBridge E5-2687W @ 3.10GHz

Performance may vary based on OS version and motherboard configuration

ZGEMM Performance vs. Matrix Size



- cuBLAS 5.0 on K20X, input and output data on device
- MKL 10.3.6 on Intel SandyBridge E5-2687W @ 3.10GHz

Performance may vary based on OS version and motherboard configuration

cuSPARSE: Sparse linear algebra routines

- Format conversion: dense, COO, CSR, CSC, HYB, BlockCSR
- Optimized sparse linear algebra for CSR and HYB formats
- New in CUDA 5.0:
  - Incomplete-LU & -Cholesky preconditioners (ilu0 and ic0)
  - BlockCSR storage format (bsr)
  - Complete list of new feature and optimizations:

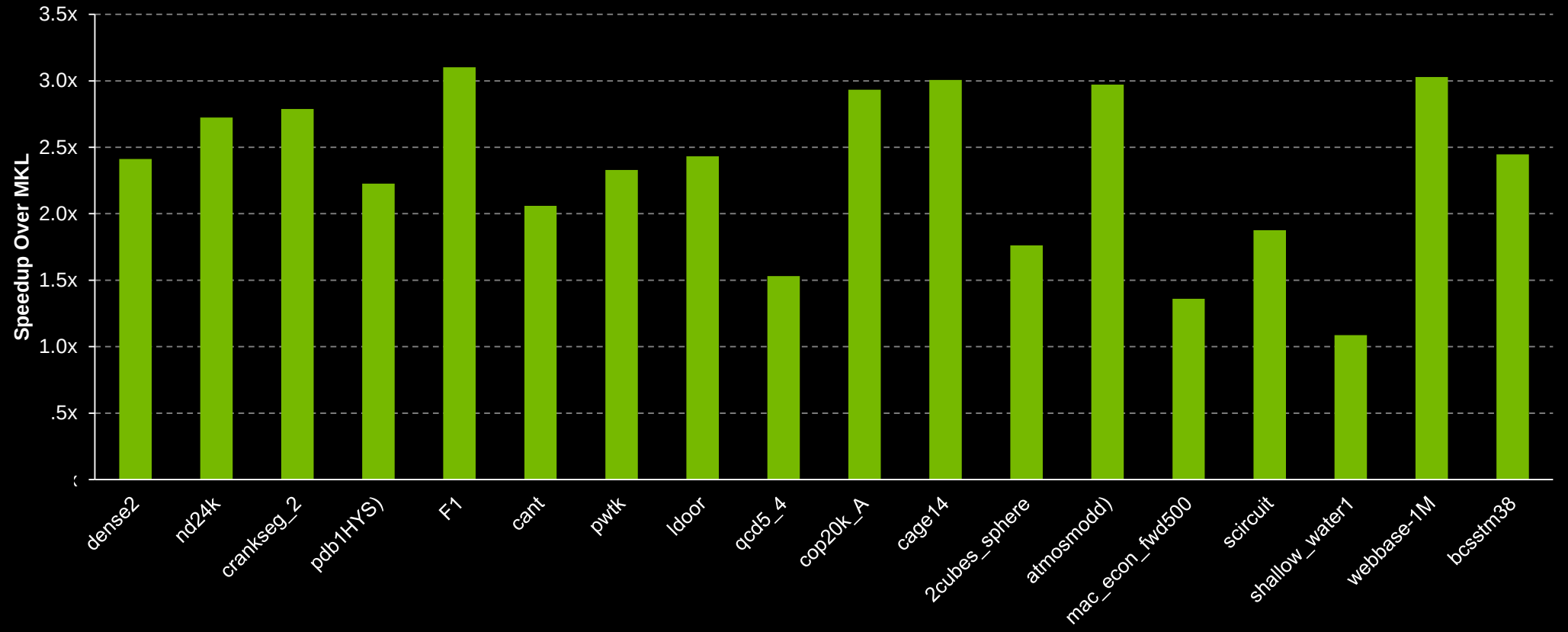
http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#cusparse

http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#cusparse-performance-improvements

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \alpha \begin{bmatrix} 1.0 & & & \\ 2.0 & 3.0 & & \\ & & 4.0 & \\ 5.0 & & 6.0 & 7.0 \end{bmatrix} \begin{bmatrix} 1.0 \\ 2.0 \\ 3.0 \\ 4.0 \end{bmatrix} + \beta \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

cuSPARSE performance

Sparse Matrix x Dense Vector



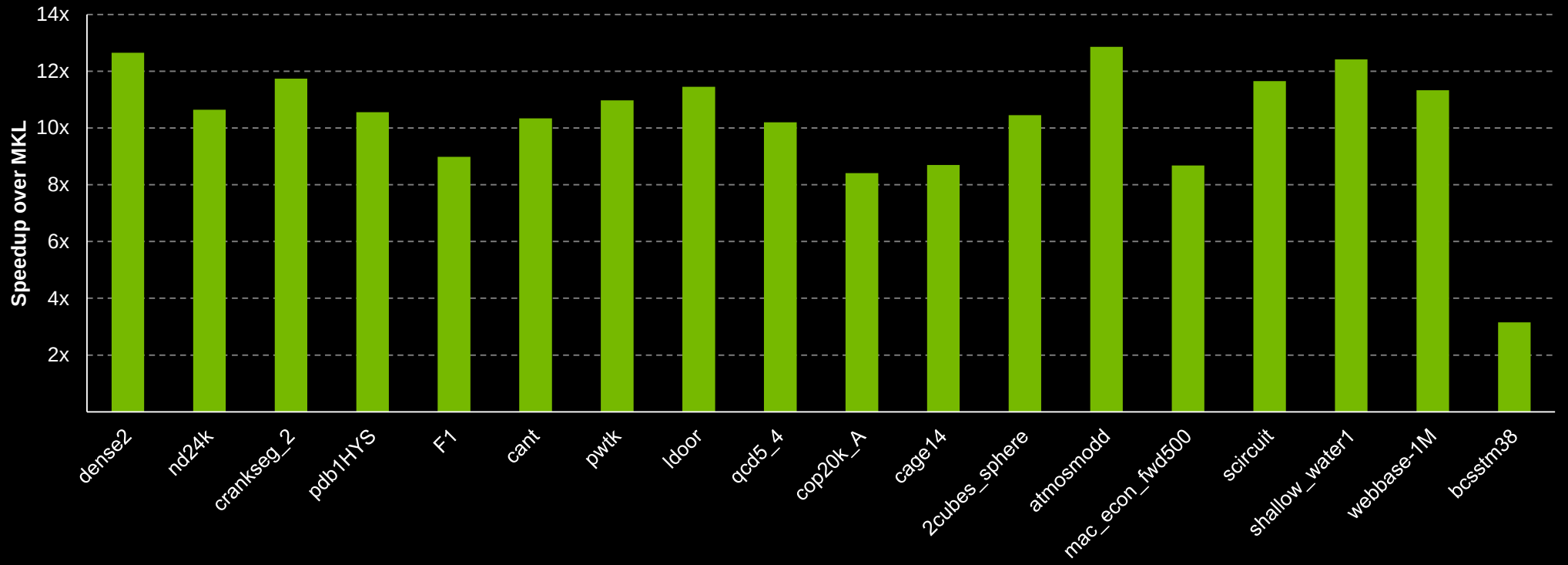
- Average of s/d/c/z routines
- cuSPARSE 5.0 on K20X, input and output data on device
- MKL 10.3.6 on Intel SandyBridge E5-2687W @ 3.10GHz

Performance may vary based on OS version and motherboard configuration

cuSPARSE: up to 12x Faster than MKL

Sparse Matrix x 6 Dense Vectors

(useful for block iterative solvers)

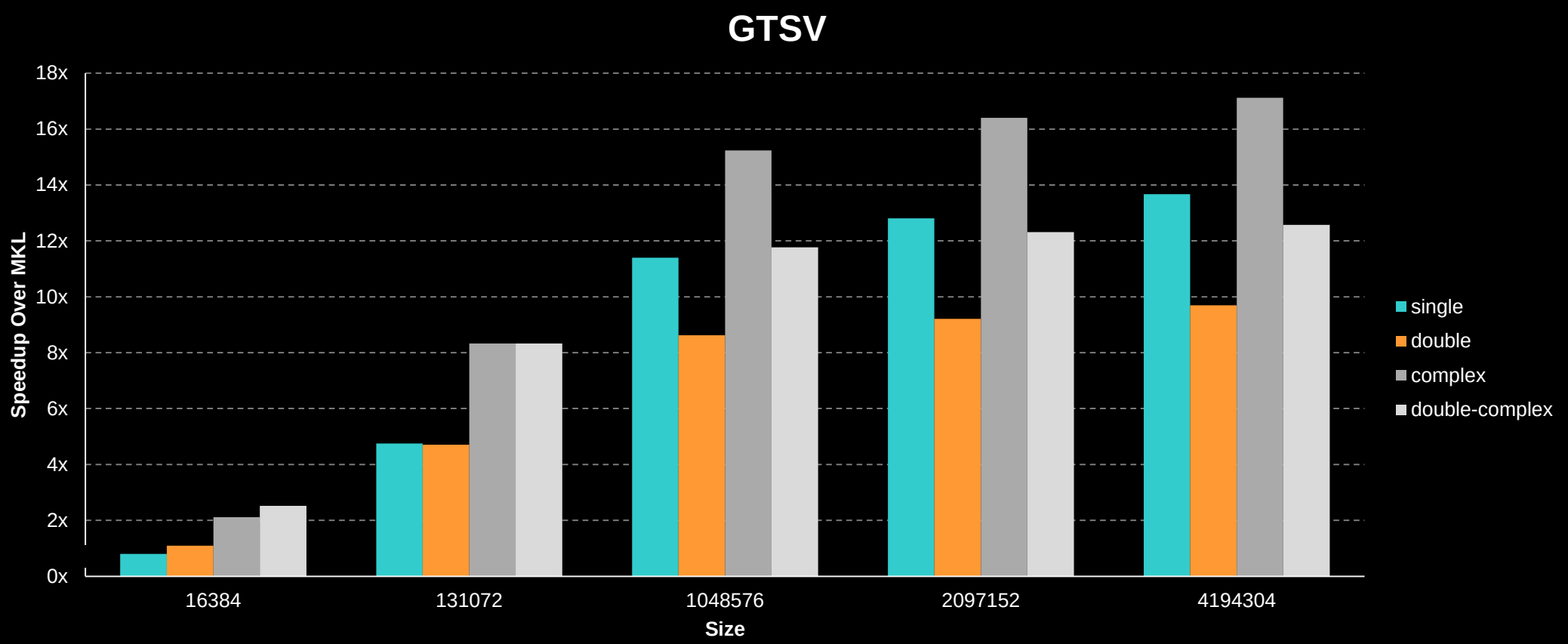


- Average of s/d/c/z routines
- cuSPARSE 5.0 on K20X, input and output data on device
- MKL 10.3.6 on Intel SandyBridge E5-2687W @ 3.10GHz

Performance may vary based on OS version and motherboard configuration

Tri-diagonal Solver Performance vs. MKL

Speedup for Tri-Diagonal solver (gtsv)



- cuSPARSE 5.0 on K20X, input and output data on device
- MKL 10.3.6 on Intel SandyBridge E5-2687W @ 3.10GHz

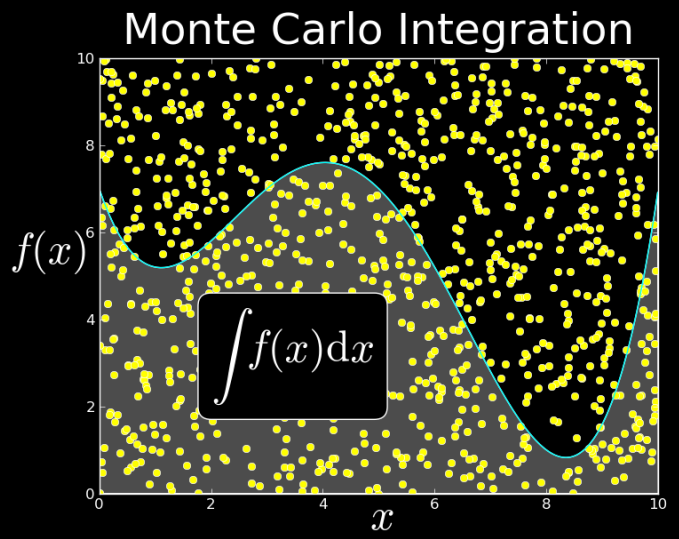
Performance may vary based on OS version and motherboard configuration

cuRAND: Random Number Generation

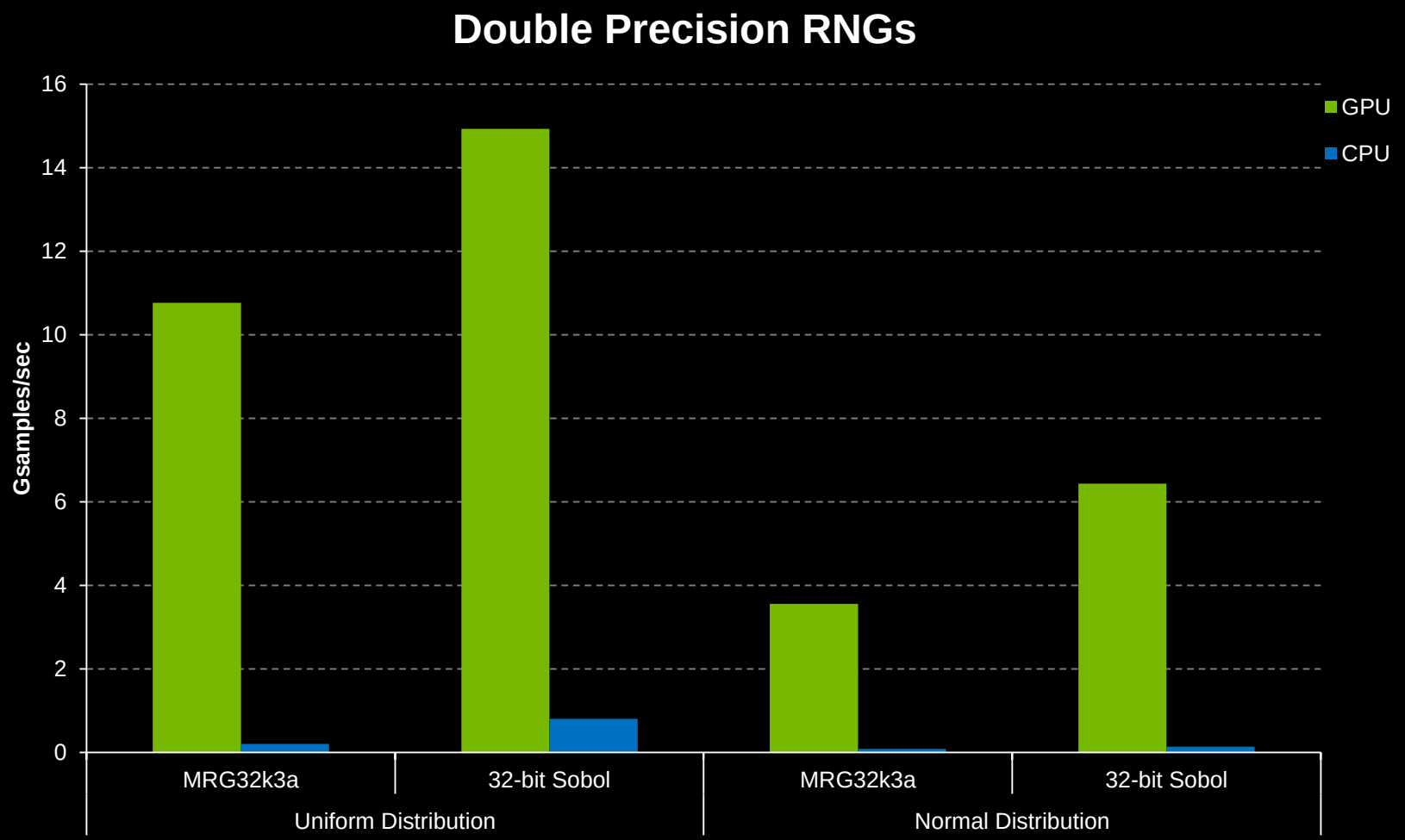
- Generating high quality random numbers in parallel is hard
  - Don’t do it yourself, use a library!

- Pseudo- and Quasi-RNGs
- Supports several output distributions
- Statistical test results in documentation

- New in CUDA 5.0: Poisson distribution



cuRAND Performance



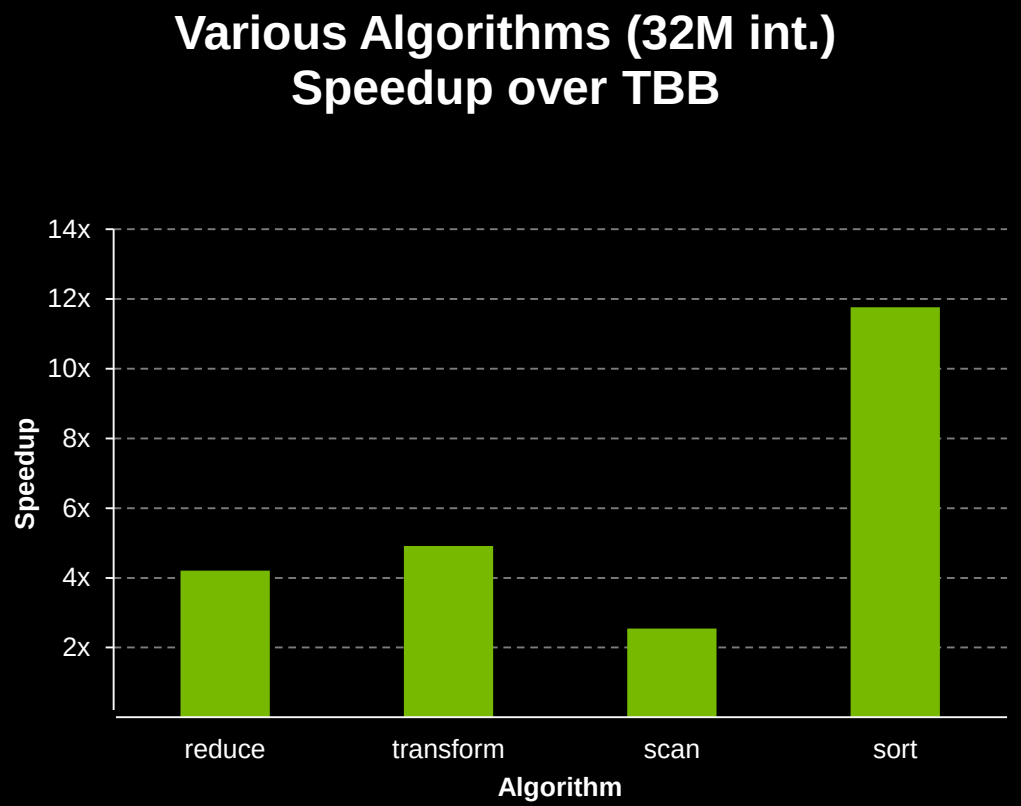
- cuRAND 5.0 on K20X, input and output data on device
- MKL 10.2.3 on Intel SandyBridge E5-2687W @ 3.10GHz

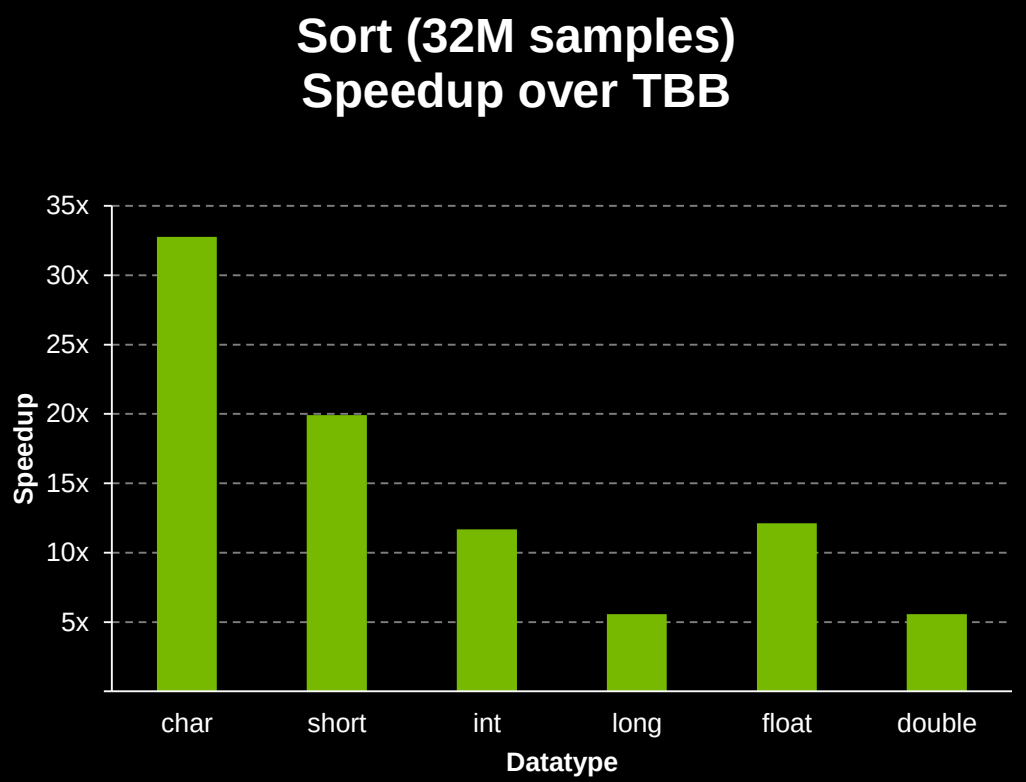
Performance may vary based on OS version and motherboard configuration

CUDA C++ Template Library

- Template library for CUDA
  - Host and Device Containers that mimic the C++ STL
  - Optimized algorithms for sort, reduce, scan, etc.
  - OpenMP backend for portability
- Also available on github: http://thrust.github.com/
- Allows applications and prototypes to be built *quickly*

Thrust Performance





- Thrust 5.0 on K20X, input and output data on device
- TBB 4.1 on Intel SandyBridge E5-2687W @3.10GHz

Performance may vary based on OS version and motherboard configuration

math.h: C99 floating-point library + extras

CUDA math.h is industry proven, high performance, accurate

- Basic: +, *, /, 1/, sqrt, FMA (all IEEE-754 accurate for float, double, all rounding modes)
- Exponentials: exp, exp2, log, log2, log10, ...
- Trigonometry: sin, cos, tan, asin, acos, atan2, sinh, cosh, asinh, acosh, ...
- Special functions: lgamma, tgamma, erf, erfc
- Utility: fmod, remquo, modf, trunc, round, ceil, floor, fabs, ...
- Extras: rsqrt, rcbrt, exp10, sinpi, sincos, cospi, erfinv, erfcinv, ...

- New in CUDA 5.0
  - sincospi[f]() and normcdf[inv][f]()
  - sin(), cos() and erfcinvf() more accurate and faster
  - Full list of new features and optimizations:

    http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#math

    http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html#math-performance-improvements