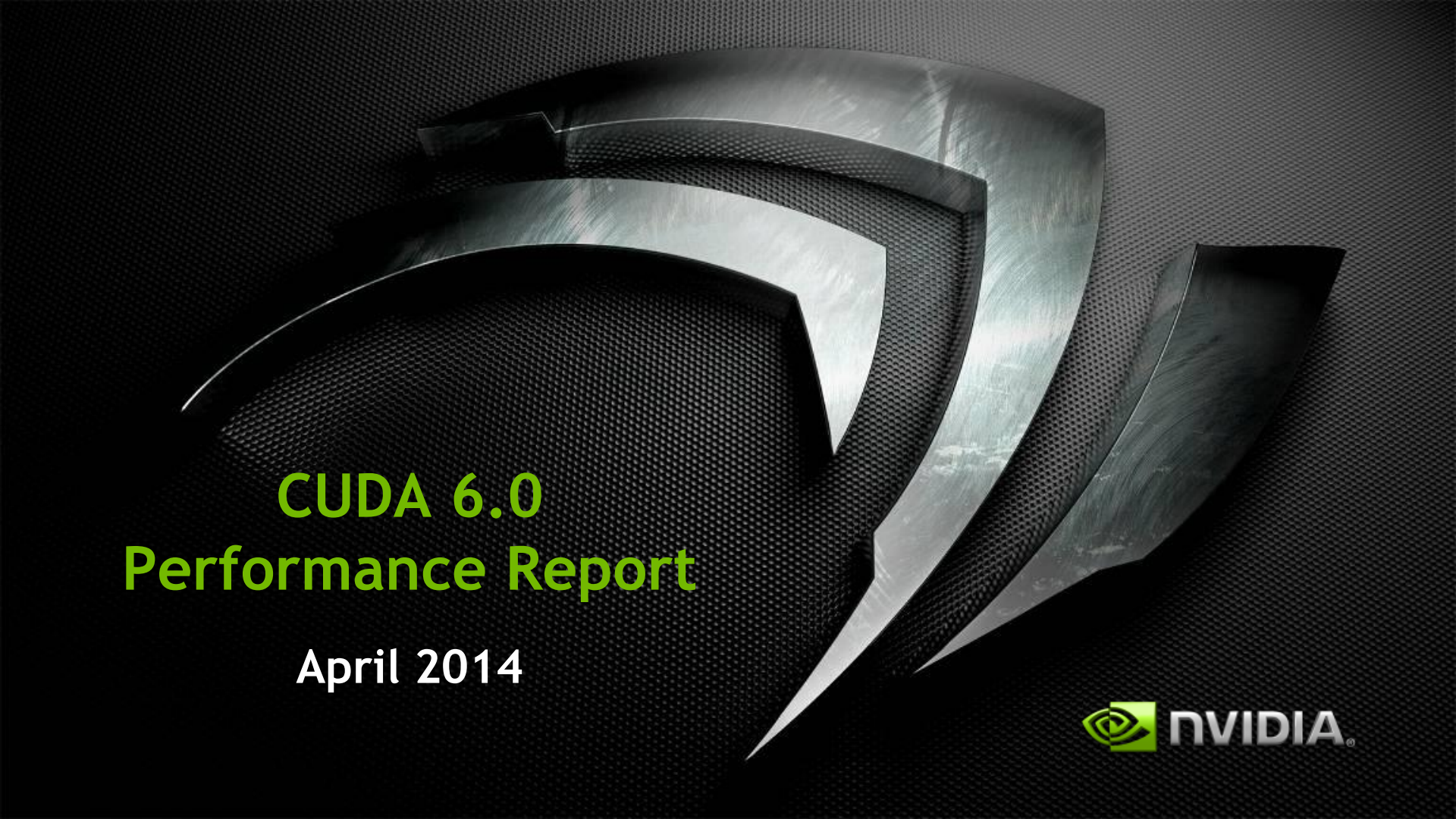

CUDA 6.0
Performance Report
April 2014
NVIDIA

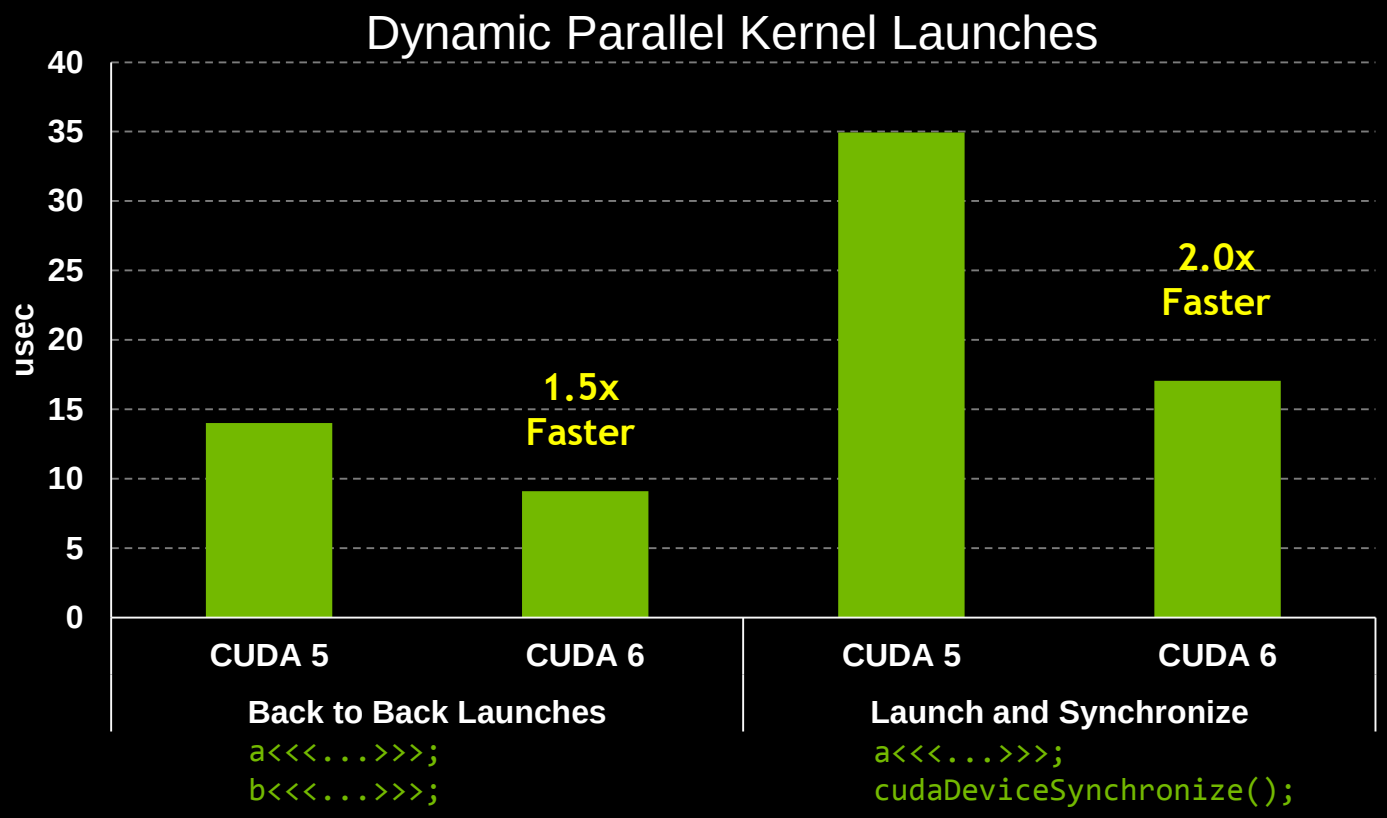
CUDA 6 Performance Report

- **CUDART** CUDA Runtime Library
- **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 (free download):
developer.nvidia.com/cuda-toolkit

For more information on CUDA libraries:
developer.nvidia.com/gpu-accelerated-libraries

CUDA 6: 2x Faster GPU Kernel Launches



Performance may vary based on OS version and motherboard configuration

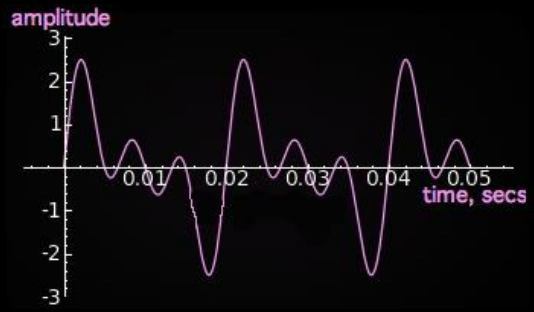
- CUDA 5.0 and CUDA 6.0 on Tesla K20

cuFFT: Multi-dimensional FFTs

- Real and complex
- Single- and double-precision data types
- 1D, 2D and 3D batched transforms
- Flexible input and output data layouts

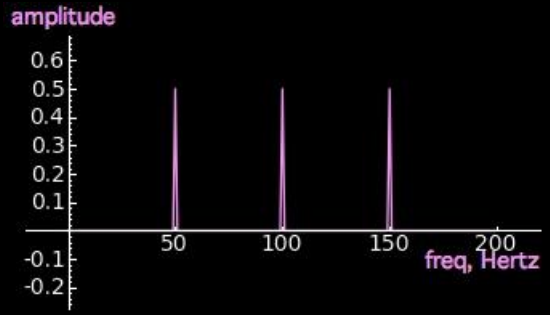
New in CUDA 6

- XT interface supports dual-GPU cards
  (Tesla K10, GeForce GTX690, …)



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

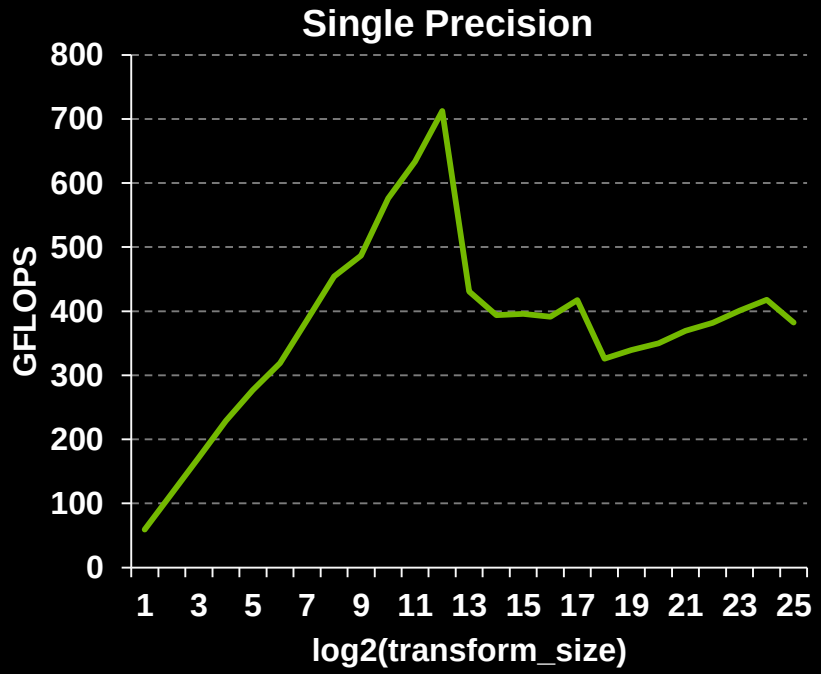
$$f(n) = \frac{1}{N} \sum_{n=0}^{N-1} F(x) e^{j2\pi(x\frac{n}{N})}$$

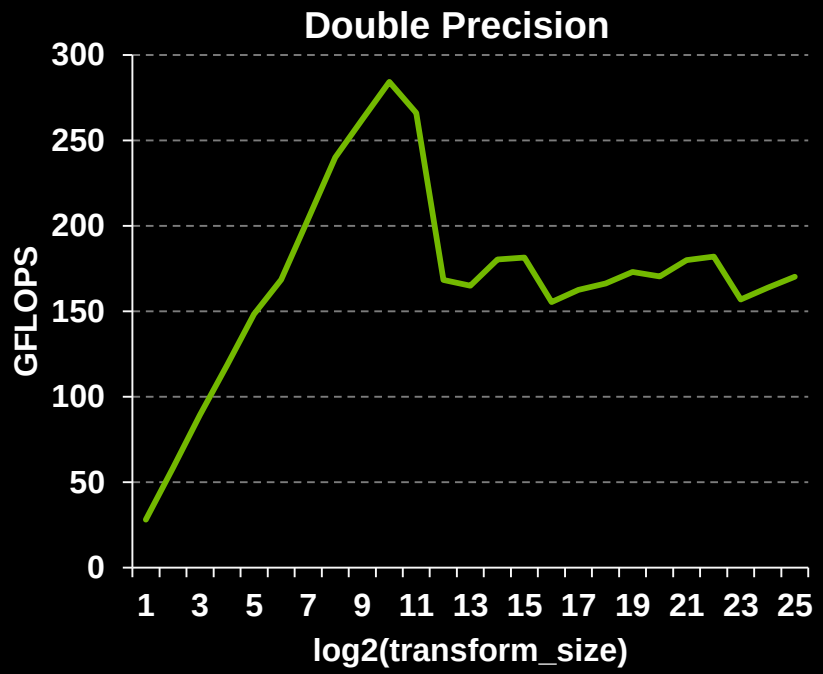


cuFFT: up to 700 GFLOPS

1D Complex, Batched FFTs

Used in Audio Processing and as a Foundation for 2D and 3D FFTs





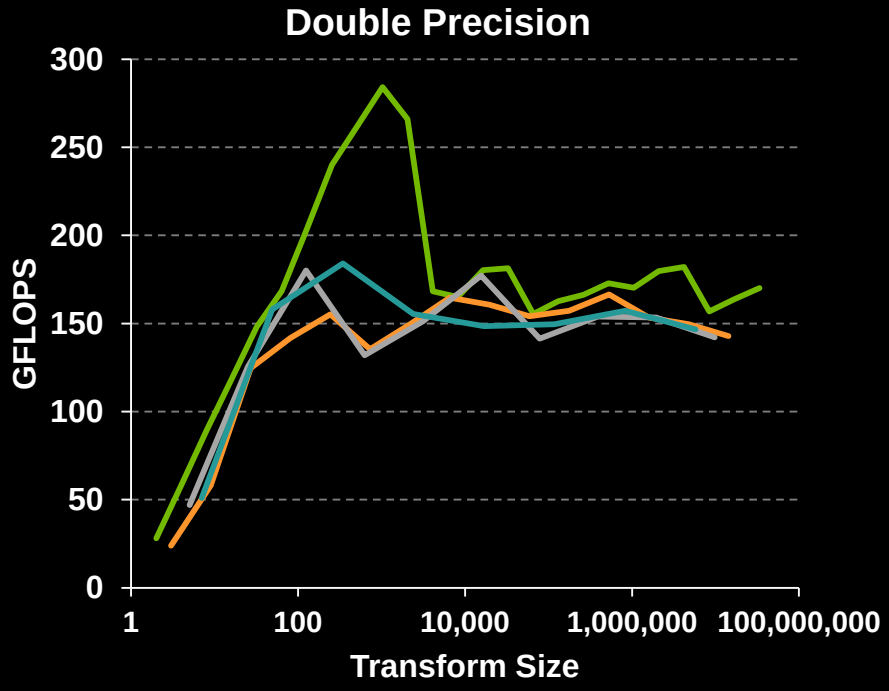
Performance may vary based on OS version and motherboard configuration

• cuFFT 6.0 on K40c, ECC ON, 32M elements, input and output data on device

cuFFT: Consistently High Performance

1D Complex, Batched FFTs

Used in Audio Processing and as a Foundation for 2D and 3D FFTs

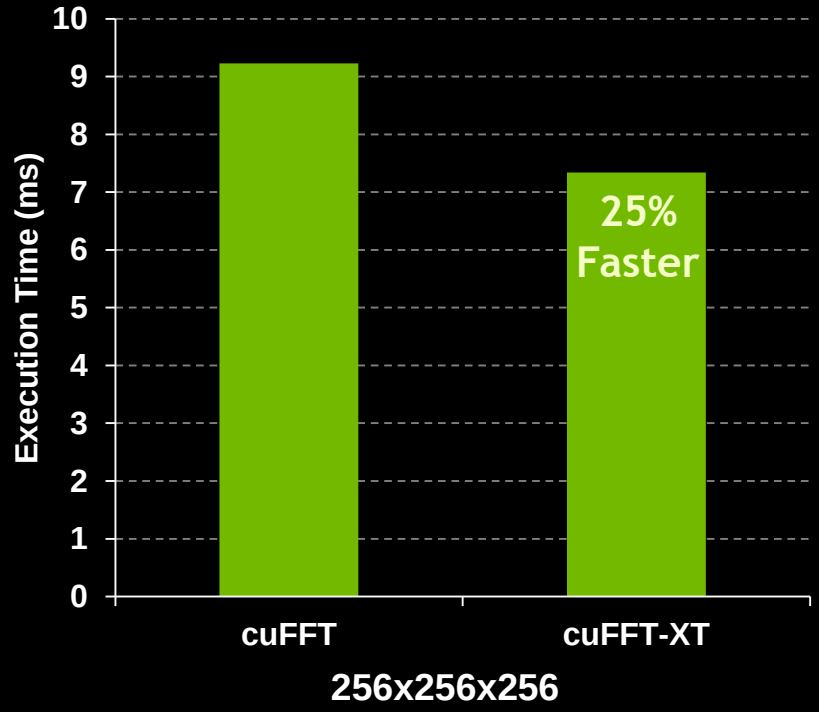


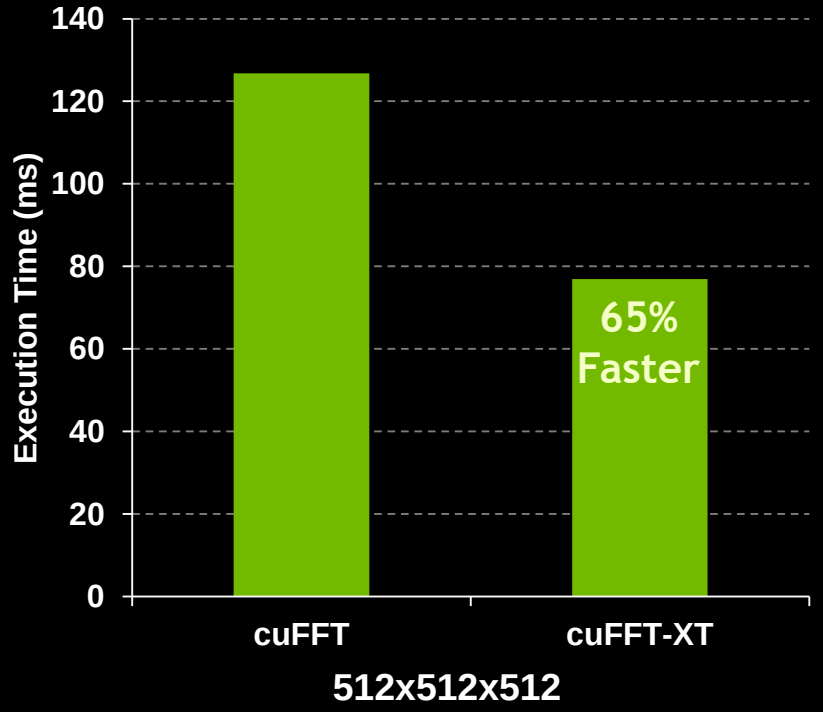
Powers of 2 Powers of 3 Powers of 5 Powers of 7

Performance may vary based on OS version and motherboard configuration • cuFFT 6.0 on K40c, ECC ON, 28M-33M elements, input and output data on device

New in CUDA 6

cuFFT-XT: Boosts Performance on K10





Performance may vary based on OS version and motherboard configuration

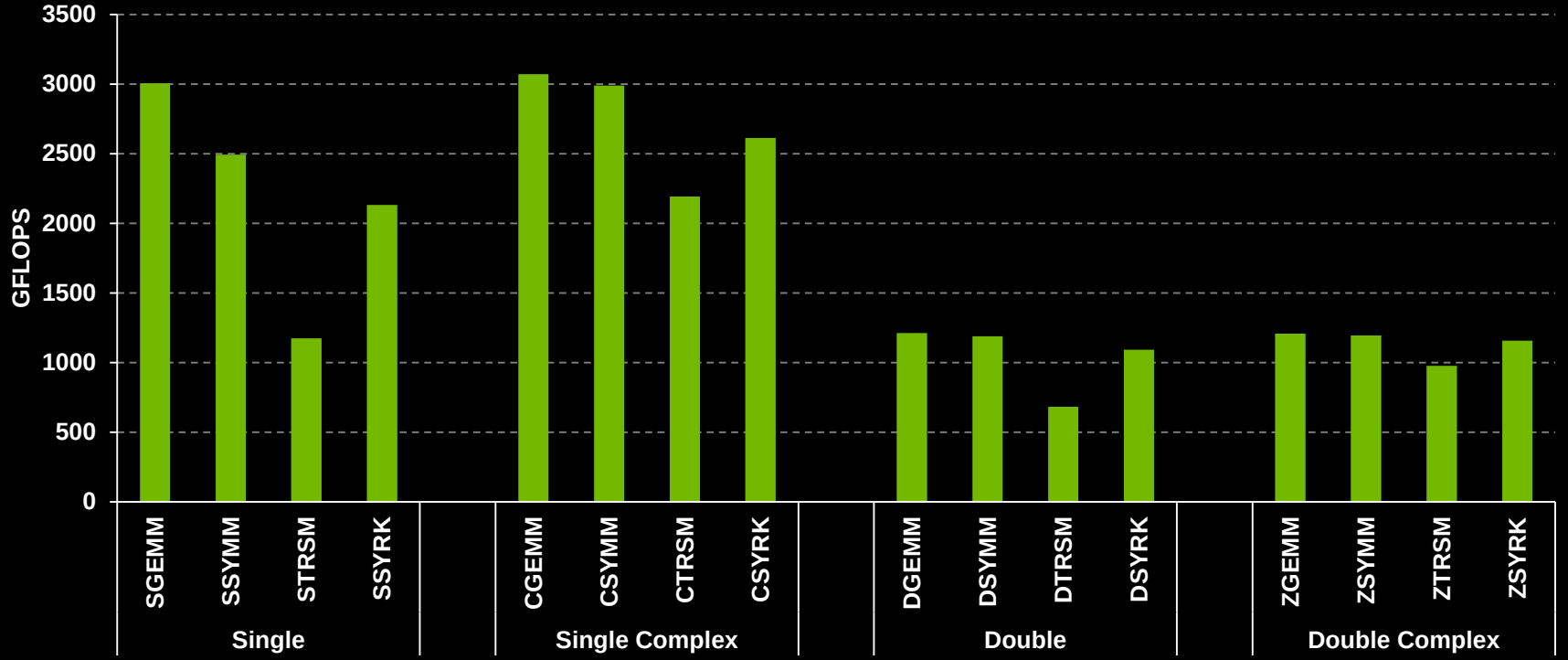
• cuFFT 6.0 on K10, ECC ON, input and output data on device

cuBLAS: Dense Linear Algebra on GPUs

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

- XT Interface for Level 3 BLAS
  - Distributed computations across multiple GPUs
  - Out-of-core streaming to GPU, no upper limit on matrix size
  - “Drop-in” BLAS intercepts CPU BLAS calls, streams to GPU

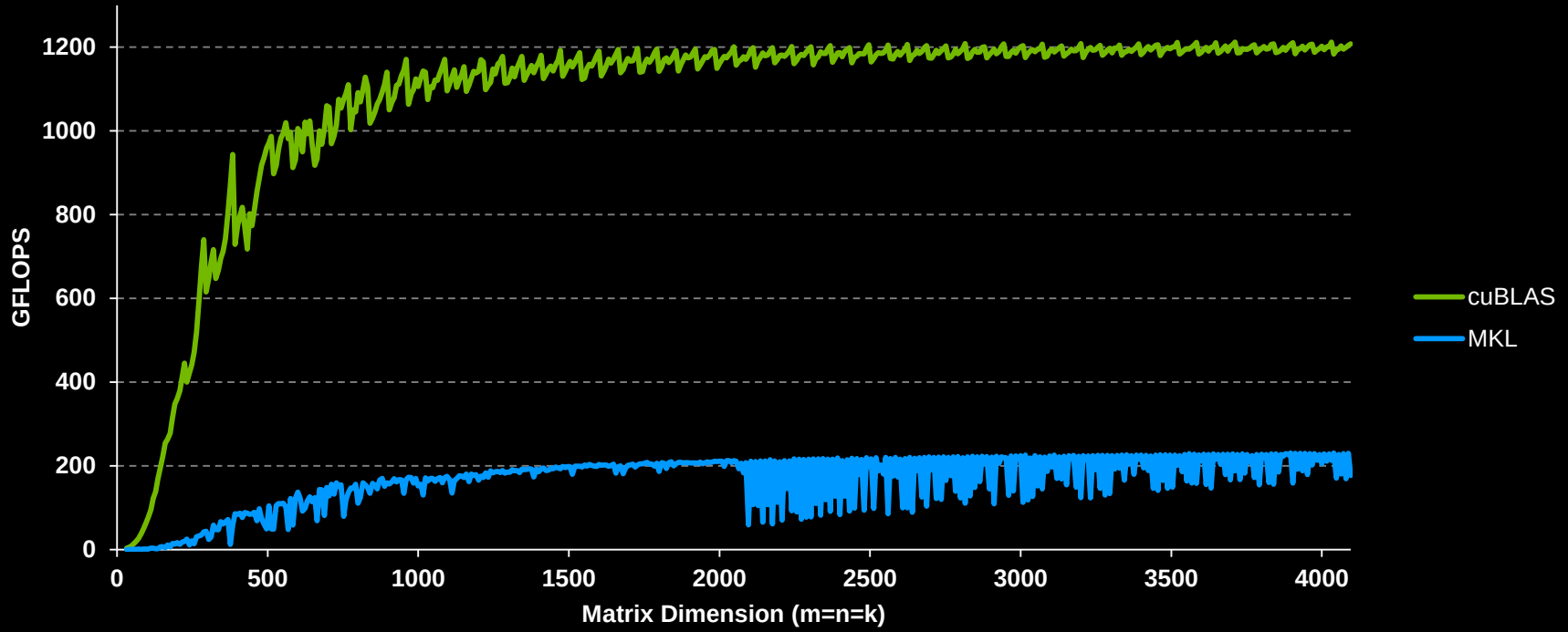
cuBLAS: >3 TFLOPS single-precision >1 TFLOPS double-precision



- cuBLAS 6.0 on K40m, ECC ON, input and output data on device
- m=n=k=4096, transpose=no, side=right, fill=lower

Performance may vary based on OS version and motherboard configuration

cuBLAS: ZGEMM 5x Faster than MKL

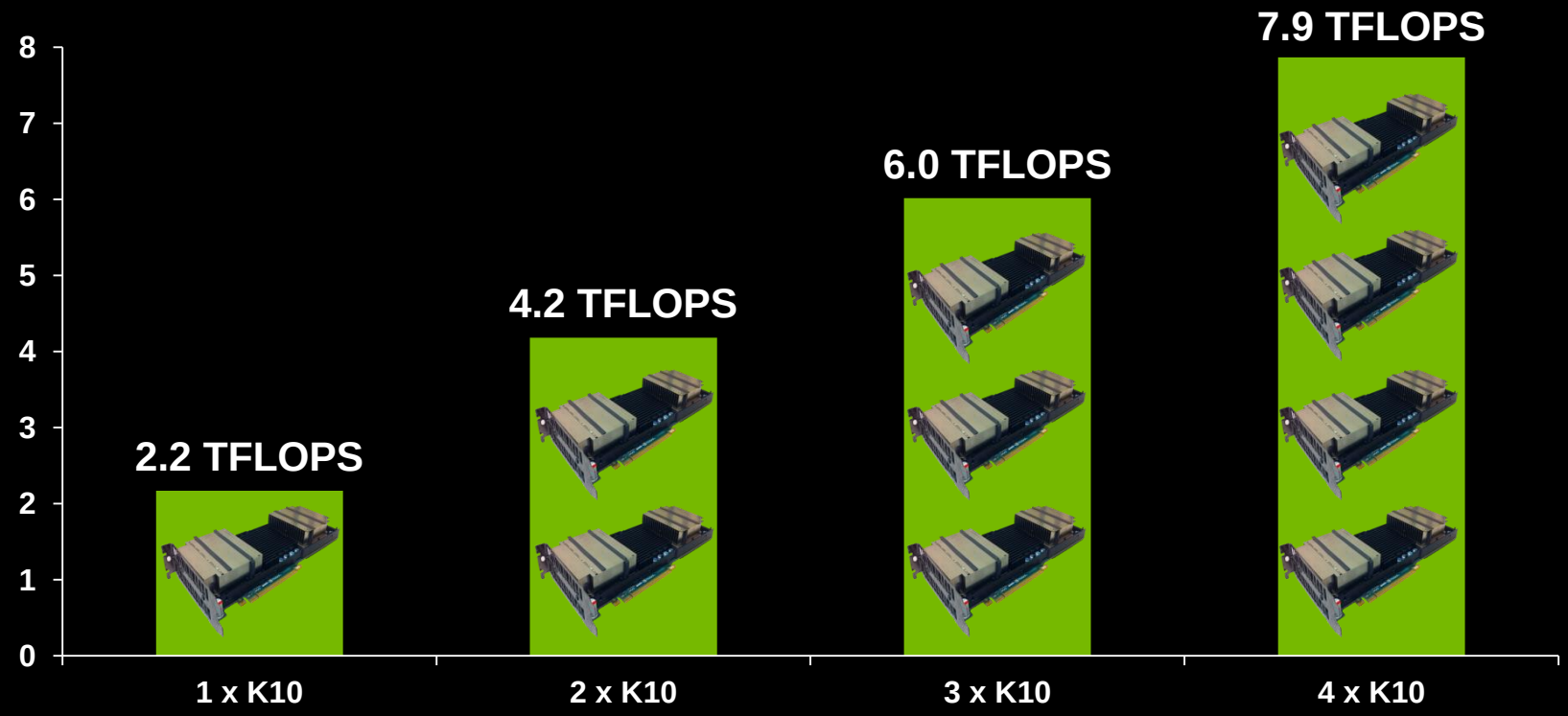


- cuBLAS 6.0 on K40m, ECC ON, input and output data on device
- MKL 11.0.4 on Intel IvyBridge 12-core E5-2697 v2 @ 2.70GHz

Performance may vary based on OS version and motherboard configuration

New in CUDA 6

cuBLAS-XT: Multi-GPU Performance Scaling



16K x 16K SGEMM on Tesla K10

Performance may vary based on OS version and motherboard configuration

• cuBLAS-XT 6.0 on K10, ECC ON, input and output data on host

cuSPARSE: Sparse linear algebra routines

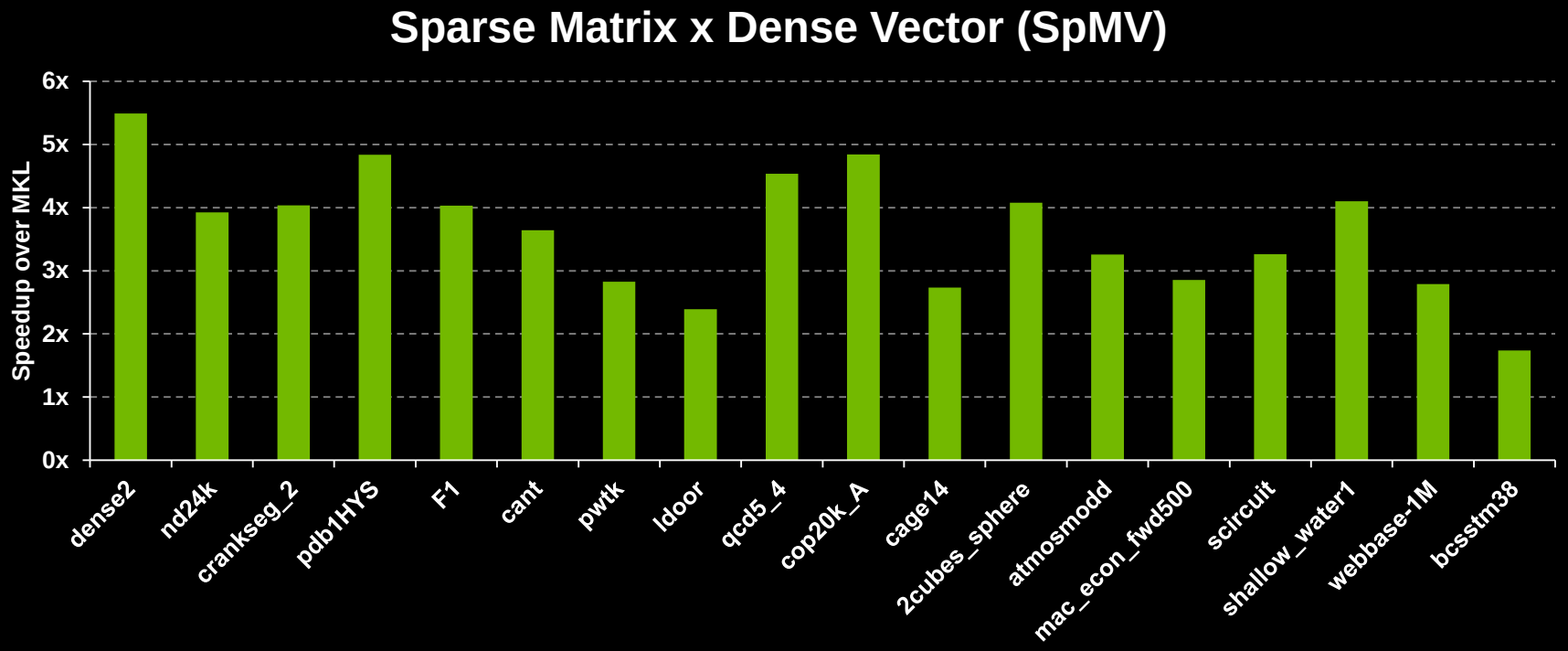
- Optimized sparse linear algebra BLAS routines - matrix-vector, matrix-matrix, triangular solve
- Support for variety of formats (CSR, COO, block variants)

New in CUDA 6

- Many improvements to triangular solvers, Incomplete-LU, and Cholesky preconditioners

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

cuSPARSE: 5x Faster than MKL



- Average of s/c/d/z routines
- cuSPARSE 6.0 on K40m, ECC ON, input and output data on device
- MKL 11.0.4 on Intel IvyBridge 12-core E5-2697 v2 @ 2.70GHz
- Matrices obtained from: http://www.cise.ufl.edu/research/sparse/matrices/

Performance may vary based on OS version and motherboard configuration

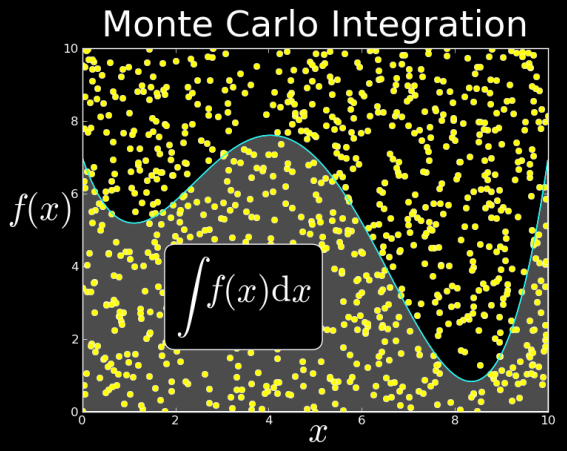
cuRAND: Random Number Generation

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

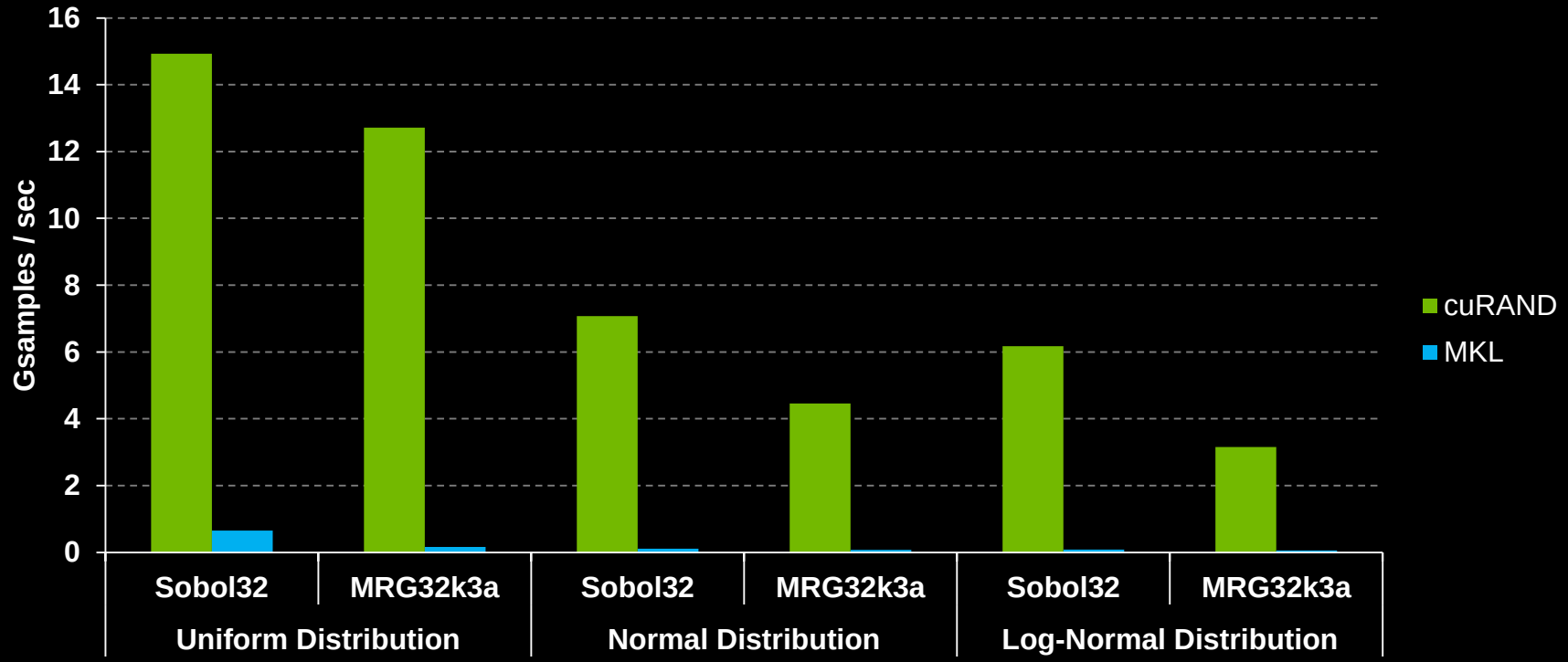
- Pseudo- and Quasi-RNGs
- Supports several output distributions
- Statistical test results in documentation

New in CUDA 6

- Mersenne Twister 19937



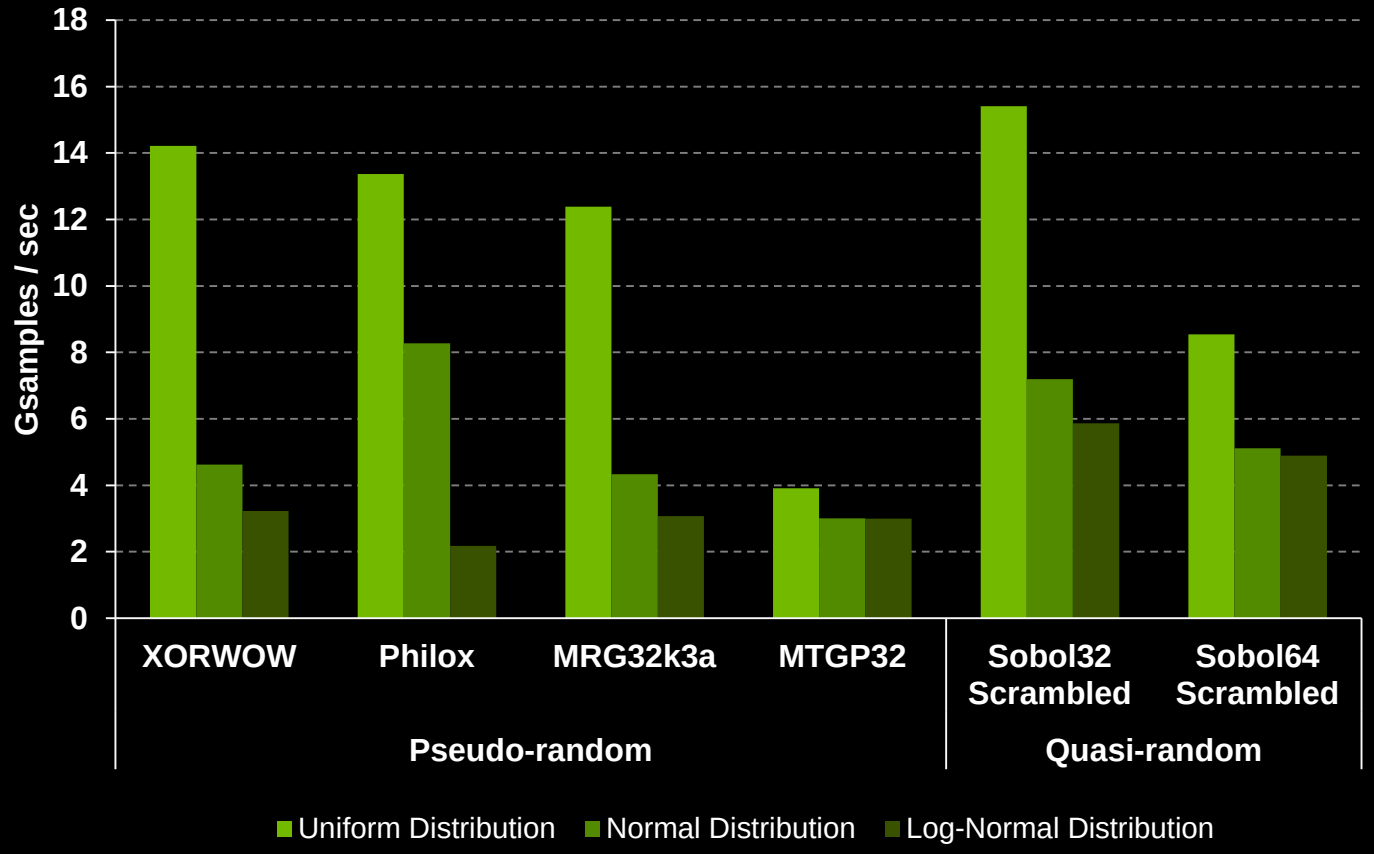
cuRAND: Up to 75x Faster vs. Intel MKL



- cuRAND 6.0 on K40c, ECC ON, double-precision input and output data on device
- MKL 11.0.1 on Intel SandyBridge 6-core E5-2620 @ 2.0 GHz

Performance may vary based on OS version and motherboard configuration

cuRAND: High Performance RNGs



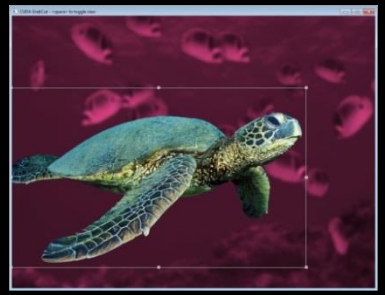
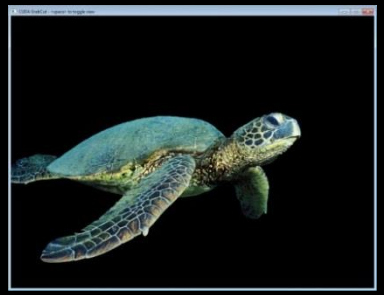
Performance may vary based on OS version and motherboard configuration • cuRAND 6.0 on K40m, ECC ON, double precision input and output data on device

NPP: NVIDIA Performance Primitives

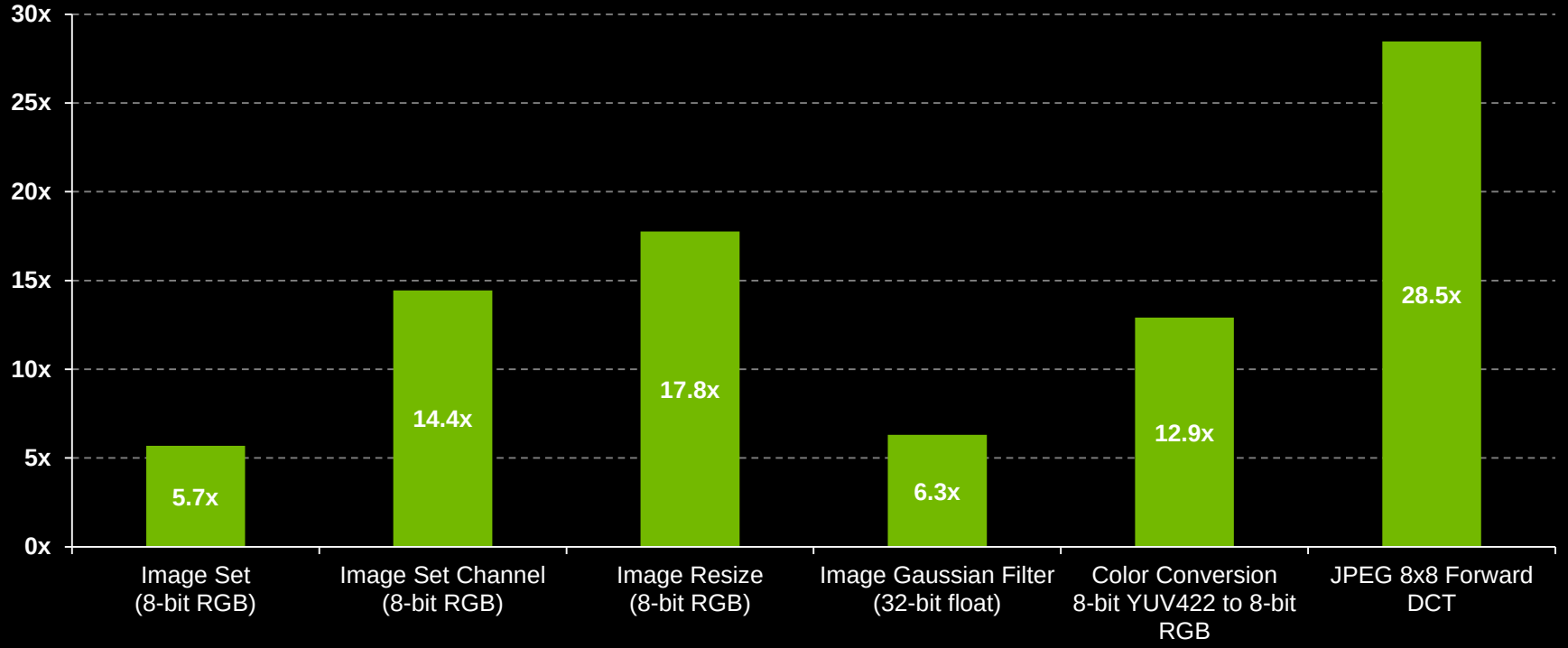
- Over 5000 image and signal processing routines:
  - color transforms, geometric transforms, move operations, linear filters, image & signal statistics, image & signal arithmetic, JPEG building blocks, image segmentation

New in CUDA 6

- Over 500 new routines, including:
  - median filter, BGR/YUV conversion, 3D LUT color conversion, improvements to JPEG primitives, plus many more

NPP Speedup vs. Intel IPP



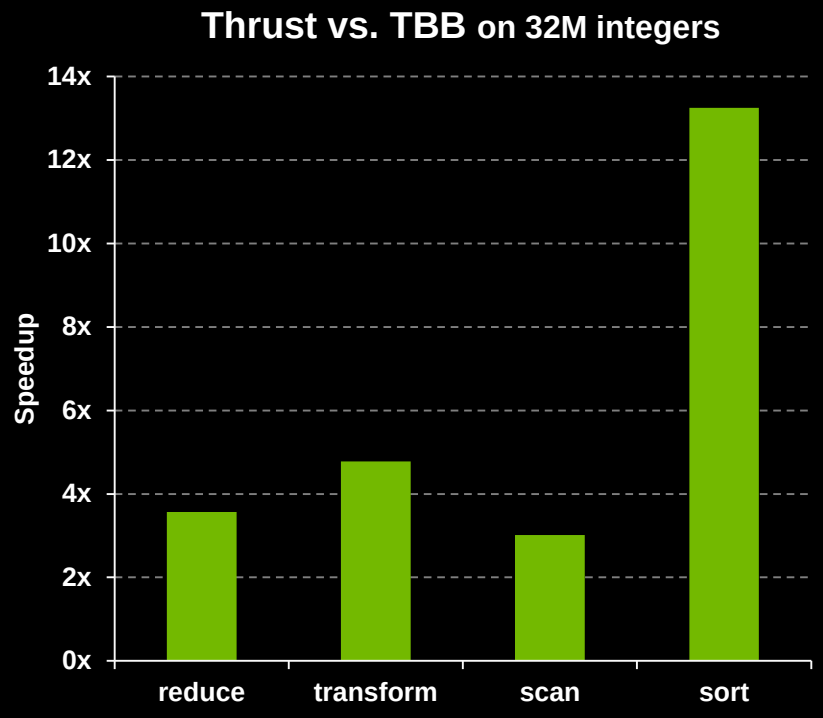
- NPP 6.0 on K40m, input and output data on device
- IPP 7.0 on Intel IvyBridge 12-core E5-2697 v2 @ 2.70GHz

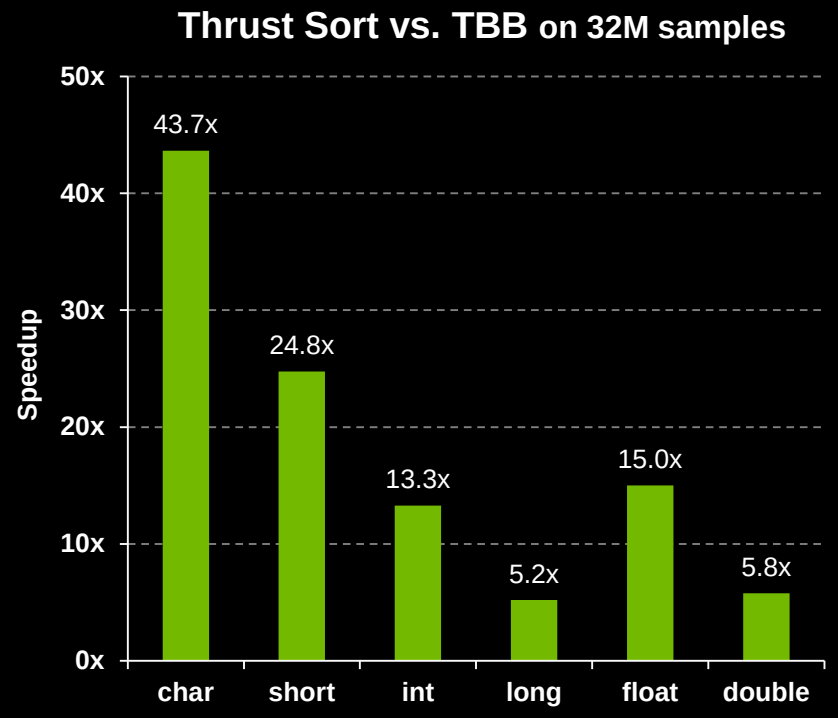
Performance may vary based on OS version and motherboard configuration

CUDA C++ Template Library

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

Thrust Performance vs. Intel TBB





- Thrust v1.7.1 on K40m, ECC ON, input and output data on device
- TBB 4.2 on Intel IvyBridge 12-core E5-2697 v2 @ 2.70GHz

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[pi], cospi, erfinv, erfcinv, normcdf[inv],...

New in CUDA 6

- Over 80 new SIMD instructions
  - Useful for video processing: _v*2, _v*4
- Cylindrical bessel: cyl_i{0,1}
- 1/hypotenuse: rhypot