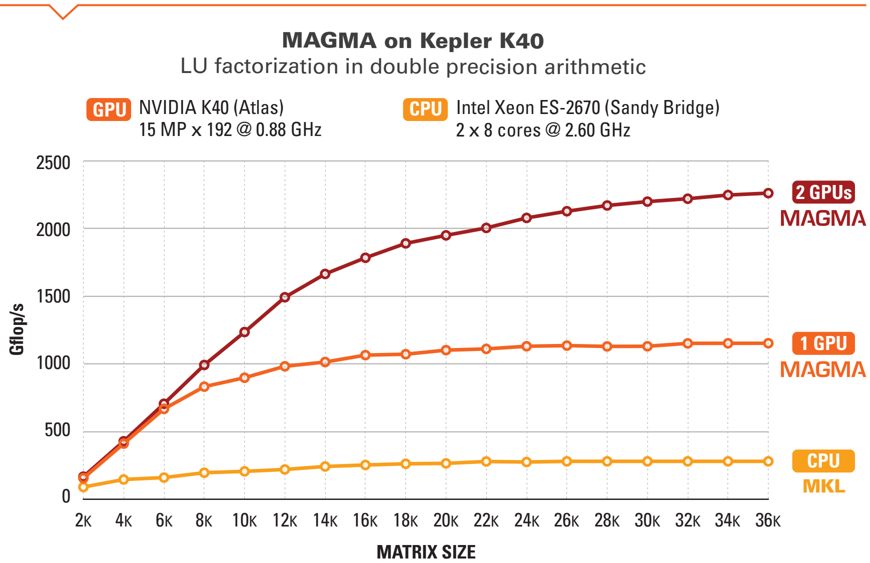
MAGMA

MAGMA (Matrix Algebra on GPU and Multicore Architectures) is a collection of next generation linear algebra libraries for heterogeneous architectures. MAGMA is designed and implemented by the team that developed LAPACK and ScaLAPACK, incorporating the latest developments in hybrid synchronization- and communication-avoiding algorithms, as well as dynamic runtime systems. Interfaces for the current LAPACK and BLAS standards are supported to allow computational scientists to seamlessly port any linear algebra reliant software components to heterogeneous architectures. MAGMA allows applications to fully exploit the power of current heterogeneous systems of multi/many-core CPUs and multi-GPUs to deliver the fastest possible time to accurate solution within given energy constraints.

HYBRID ALGORITHMS

MAGMA uses a hybridization methodology where algorithms of interest are split into tasks of varying granularity and their execution scheduled over the available hardware components. Scheduling can be static or dynamic. In either case, small non-parallelizable tasks, often on the critical path, are scheduled on the CPU, and larger more parallelizable ones, often Level 3 BLAS, are scheduled on the GPU.

PERFORMANCE



FEATURES AND SUPPORT

- MAGMA 1.4.1 FOR CUDA
- clMAGMA 1.1 FOR OpenCL
- MAGMA MIC 1.1 FOR Intel Xeon Phi

CUDA	OpenCL	Intel Xeon Phi	
●	●	●	Linear system solvers
●	●	●	Eigenvalue problem solvers
●	●		Auxiliary BLAS
●	●	●	CPU Interface
●	●	●	GPU Interface
●	●	●	Multiple precision support
●			Non-GPU-resident factorizations
●	●	●	Multicore and multi-GPU support
●	●	●	LAPACK testing
●	●	●	Linux
●			Windows
●			Mac OS

NVIDIA CUDA CENTER OF EXCELLENCE

NVIDIA's CUDA Center of Excellence Program recognizes universities expanding the frontier of massively parallel computing using CUDA.

KEENELAND

MAGMA provides scientific libraries that power **the Keeneland Project**, an NSF-funded partnership to enable large-scale computational science on heterogeneous architectures.

FIND OUT MORE AT **http://keeneland.gatech.edu/**

SUPPORT FROM THE SILAS **NSF** AWARD

ICL UT INNOVATIVE COMPUTING LABORATORY THE UNIVERSITY of TENNESSEE

IN COLLABORATION WITH
Berkeley University of California; University of Colorado Denver; Inria; KAUST

WITH SUPPORT FROM
AMD; intel; MathWorks; NVIDIA

SPONSORED BY
U.S. Department of Defense; U.S. Department of Energy; National Science Foundation

MAGMA

MAGMA 1.4.1 ROUTINES & FUNCTIONALITIES

	SINGLE GPU	MULTI-GPU
One-sided Factorizations (LU, QR, Cholesky)	✓	✓
Linear System Solvers	✓	
Linear Least Squares (LLS) Solvers	✓	
Matrix Inversion	✓	
Singular Value Problem (SVP)	✓	
Non-symmetric Eigenvalue Problem	✓	✓
Symmetric Eigenvalue Problem	✓	✓
Generalized Symmetric Eigenvalue Problem	✓	✓

SINGLE GPU
Hybrid LAPACK algorithms with static scheduling and LAPACK data layout

MULTI-GPU
Hybrid LAPACK algorithms with 1D block cyclic static scheduling and LAPACK data layout

MAGMA 1.4.1 DRIVER ROUTINES

	MATRIX	OPERATION	ROUTINE	INTERFACES CPU	INTERFACES GPU
LINEAR EQUATIONS	GE	Solve using LU	{sdcz}gesv	✓	✓
		Solve using MP	{zc,ds}gesv		✓
	SPD/HPD	Solve using Cholesky	{sdcz}posv	✓	✓
		Solve using MP	{zc,ds}posv		✓
LLS	GE	Solve LLS using QR	{sdcz}geqrs		✓
		Solve using MP	{zc,ds}geqrsv		✓
STANDARD EVP	GE	Compute e-values, optionally e-vectors	{sdcz}geev	✓	
	SY/HE	Computes all e-values,	{sd}syevd	✓	✓
		optionally e-vectors	{cz}heevd	✓	✓
		Range (D&C)	{cz}heevdx		✓
		Range (B&I It.)	{cz}heevx	✓	✓
		Range (MRRR)	{cz}heevr	✓	✓
STAND. SVP	GE	Compute SVD,	{sdcz}gesvd	✓	
		optionally s-vectors		✓	
GENERALIZED EVP	SPD/HPD	Compute all e-values,	{sd}sygvd	✓	
		optionally e-vectors	{cz}hegvd	✓	
		Range (D&C)	{cz}hegvdx	✓	
		Range (B&I It.)	{cz}hegvx	✓	
		Range (MRRR)	{cz}hegvr	✓	

ABBREVIATIONS

GE	General
SPD/HPD	Symmetric/Hermitian Positive Definite
TR	Triangular
D&C	Divide & Conquer
B&I It	Bisection & Inverse Iteration
MP	Mixed-precision Iterative Refinement

NAMING CONVENTION

magma_{routine name}[_gpu]

MAGMA 1.4.1 COMPUTATIONAL ROUTINES

	MATRIX	OPERATION	ROUTINE	INTERFACES CPU	INTERFACES GPU
LINEAR EQUATIONS	GE	LU	{sdcz}getrf	✓	✓
		Solve	{sdcz}getrs		✓
		Invert	{sdcz}getri		✓
	SPD/HPD	Cholesky	{sdcz}potrf	✓	✓
		Solve	{sdcz}potrs		✓
		Invert	{sdcz}potri		✓
	TR	Invert	{sdcz}trtri	✓	
ORTHOGONAL FACTORIZATIONS	GE	QR	{sdcz}geqrf	✓	
		QR w/ pivoting	{sdcz}geqp3		✓
		Generate Q	{sd}orgqr	✓	✓
			{cz}ungqr	✓	✓
		Multiply matrix by Q	{sd}ormqr	✓	✓
			{cz}unmqr	✓	✓
		LQ factorization	{sdcz}gelqf	✓	✓
		QL factorization	{sdcz}geqlf	✓	
		Multiply matrix by Q	{sd}ormql	✓	✓
			{cz}unmql	✓	✓
STANDARD EVP	GE	Hessenberg reduction	{sdcz}gehrd	✓	
		Generate Q	{sd}orghr	✓	
			{cz}unghr	✓	
	SY/HE	Tridiagonalization	{sd}sytrd	✓	
			{cz}hetrd	✓	
		Generate Q	{sd}orgtr		✓
			{cz}ungtr		✓
		Multiply by Q	{sd}ormtr	✓	✓
			{cz}unmtr	✓	✓
SVD	GE	Bidiagonalization	{sdcz}gebrd	✓	
GENERALIZED EVP	SPD/HPD	Reduction to standard form	{sd}sygst	✓	✓
			{cz}hegst	✓	✓

ICL UT INNOVATIVE COMPUTING LABORATORY THE UNIVERSITY of TENNESSEE

IN COLLABORATION WITH
Berkeley University of California · University of Colorado Denver · Inria · KAUST

WITH SUPPORT FROM
AMD · intel · MathWorks · NVIDIA

SPONSORED BY
U.S. Department of Defense · U.S. Department of Energy · National Science Foundation