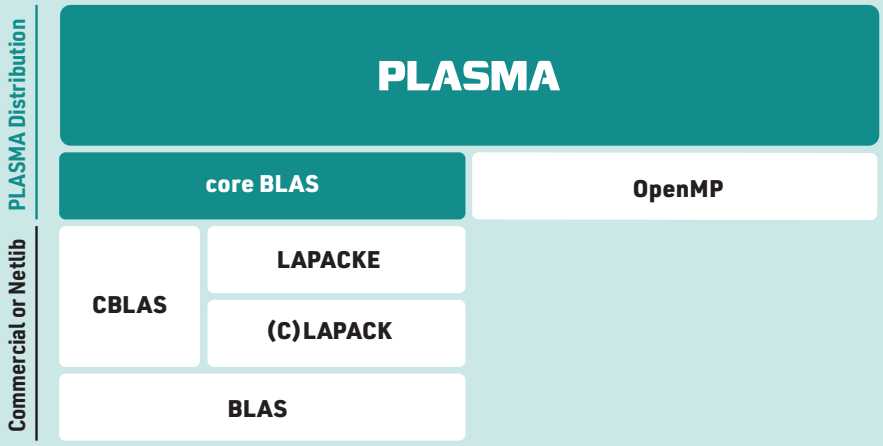
PLASMA

PARALLEL LINEAR ALGEBRA SOFTWARE FOR MULTICORE ARCHITECTURES

PLASMA (Parallel Linear Algebra Software for Multicore Architectures) is a software package implementing a set of fundamental linear algebra routines using the OpenMP standard. PLASMA includes routines for solving linear systems of equations and linear least square problems, parallel BLAS, parallel matrix norms, etc. PLASMA has been deployed to systems based on Intel processors (including the Xeon Phi family), IBM POWER processors, and ARM processors. For the last decade, PLASMA served as a tremendous research vehicle for the design of new dense linear algebra algorithms, and paved the way for new developments, such as the ECP SLATE project, which will ultimately deliver exascale capabilities. FIND OUT MORE AT **http://icl.utk.edu/slate**

MODERN SOFTWARE STACK



STATE-OF-THE-ART SOLUTIONS

Tile Matrix Layout

PLASMA lays out matrices in square tiles of relatively small size, such that each tile occupies a continuous memory region. Tiles are loaded to the cache memory efficiently with little risk of eviction while being processed. The use of tile layout minimizes conflict cache misses, TLB misses, and false sharing, and maximizes the potential for prefetching. PLASMA contains parallel and cache efficient routines for converting between the conventional LAPACK layout and the tile layout.

Tile Algorithms

PLASMA introduces new algorithms redesigned to work on tiles, which maximize data reuse in the cache levels of multi-core systems. Tiles are loaded to the cache and processed completely before being evicted back to the main memory. Operations on small tiles create fine-grained parallelism, providing enough work to keep a large number of cores busy.

Dynamic Scheduling

PLASMA relies on runtime scheduling of parallel tasks. Runtime scheduling is based on the idea of assigning work to cores based on the availability of data for processing at any given point in time, and thus is also referred to as data-driven scheduling. The concept is related closely to the idea of expressing computation through a task graph, often referred to as the DAG (Directed Acyclic Graph), and the flexibility of exploring the DAG at runtime. This is in direct opposition to the fork-and-join scheduling, where artificial synchronization points expose serial sections of the code and multiple cores are idle while sequential processing takes place.

LAPACK WORKING NOTES http://www.netlib.org/lapack/lawns/

LAWN293
PLASMA 17.1 Functionality Report
Maksims Abalenkovs, Negin Bagherpour, Jack Dongarra, Mark Gates, Azzam Haidar, Jakub Kurzak, Piotr Luszczek, Samuel Relton, Jakub Sistek, David Stevens, Panruo Wu, Ichitaro Yamazaki, Asim YarKhan, and Mawussi Zounon
UT-EECS-17-751 June 2017

LAWN292
PLASMA 17 Performance Report
Maksims Abalenkovs, Negin Bagherpour, Jack Dongarra, Mark Gates, Azzam Haidar, Jakub Kurzak, Piotr Luszczek, Samuel Relton, Jakub Sistek, David Stevens, Panruo Wu, Ichitaro Yamazaki, Asim YarKhan, and Mawussi Zounon
UT-EECS-17-750 June 2017

PLASMA

FIND OUT MORE AT **https://bitbucket.org/icl/plasma**

IN COLLABORATION WITH
Berkeley University of California
University of Colorado Denver
WITH SUPPORT FROM
FUJITSU
MathWorks
Microsoft
SPONSORED BY
U.S. Department of Energy
ECP
National Science Foundation