

Datasheet

# RapidMind Scientific Library

The RapidMind Scientific Library is an open-source distribution that provides a real-world programming example using the RapidMind platform. The version 1.0 library performs basic linear algebra operations using a variety of data access patterns to illustrate design alternatives with different target processors. The library demonstrates techniques for development of algorithms that can be tuned for any available processors (x86 CPU, GPU or Cell BE).

A script-based auto-tuner is provided to leverage dynamic programming with RapidMind. This allows you to further optimize algorithms for your target processor.

The RapidMind library includes a cross section of Level 1 to Level 3 BLAS (Basic Linear Algebra Subprograms) routines using both single and double precision on x86 processors: sdot, ddot, saxpy, daxpy, sgemv, dgemv, sgemm, and dgemm.

These BLAS primitives are typically used to solve systems of linear equations. BLAS is commonly used in medical imaging, digital media, cryptography, and finance.

## Who Should Use the RapidMind Scientific Library?

- Developers working with vector spaces, linear maps or systems of linear equations
- Developers interested in tuning their algorithms for specific backends and specific processors
- Developers interested in learning more about best practices for programming with the RapidMind platform

## Benefits

The RapidMind Library meets or exceeds Intel MKL and ATLAS for problems sizes of 512 MB or larger that do not exceed L2 cache, or that can be parsed in smaller matrices that do not exceed L2 cache, which is well suited to leading parallel solvers that typically break down large matrices into smaller blocks. As the complexity of the computation increases, the performance increases to 1600 MFlops on problem sizes of one million or larger using dual quad-core Intel Xeon E5345 processors with 2GB of RAM. The RapidMind library offers the following benefits:

- Compared to off-the-shelf BLAS libraries, the RapidMind Library scales better as the number of cores in a multi-core processor increases (e.g. 8-core x86 CPU).
- The RapidMind programming model simplifies the implementation of many different algorithms reducing development cost and timelines.
- The same algorithms can be custom tuned for different target processors increasing performance gains.

## Using the RapidMind Library

The RapidMind library is binary compatible with reference BLAS implementations. As a result, developers can substitute the RapidMind library for an existing BLAS library in C, C++ and Fortran applications. The RapidMind library is not a replacement for the full BLAS standard. Instead, it provides a specific subset of BLAS for benchmarking and to illustrate best practices in software development using RapidMind.

Each algorithm has been tested with single precision and, when supported by the target processors, with double precision. The RapidMind library includes a variety of programs for each algorithm with different tunables to illustrate a variety of data access patterns that leverage the capabilities of the full range of supported processors. To take advantage of SIMD instructions, algorithms in the library are typically benchmarked with 4-tuples of floats or 2-tuples of doubles.

## RapidMind Library Auto-Tuner

The auto-tuner is a set of Perl scripts that run the testing application (test\_blas.exe) while varying tunable parameters. When the parameters are modified, the algorithm used in the RapidMind programs will change with the goal of adjusting for competing resources such as registers, threads, cache and local store. The set of tunables is pre-defined and hard-coded; however, we provide the source code so that you can extend or customize the list of tunables when applying this technique to your algorithms.

### Using the Auto-Tuner

The performance of any algorithm is analyzed as each tuning parameter is varied. The fastest performance is used to pick the best value of a tuning parameter. The best set of tunables are written to a configuration file that can be loaded as the default values when the RapidMind programs are run subsequently. In some cases, one tunable affects another. As a result, the auto-tuner can also vary two tunables and generate a 3D graph of the results in order to identify dependencies and to pick the sweet spot.

### Benefits

Auto-tuning allows developers to tune a production algorithm by leveraging the dynamic programming fundamental to run-time compilation. Typically, users tune their software after installation or after upgrading their hardware. Note that the best values for tunables is affected by highly variable characteristics that depend not only on the processor, but the available resources at runtime and the specific algorithm under development. So the key points are that the actual production algorithm can be experimentally tuned and that automating this kind of tuning is far easier with dynamic programming.

In many cases, performance is flat because the tuning is already implemented at some level such as the backend, drivers, low-level code emitters or even on-chip logic. It's useful to prove this because it allows a developer to write a simpler algorithm if the target hardware is already known.

---

RapidMind provides the award-winning RapidMind Multi-Core Development Platform that simplifies the development of parallel applications, minimizing the impact on traditional lifecycle costs and timelines. Developers of HPC and enterprise software are using RapidMind today to create manageable, single-threaded applications that leverage the full potential of multi-core processors from AMD® and Intel® and to seamlessly take advantage of the application acceleration available from GPUs and the Cell Broadband Engine™. For more information on RapidMind, visit <http://www.rapidmind.com>.