

*Intel® Math Kernel Library 9.1 Cluster Edition for Linux**

Release Notes

Contents

Overview

Directory Structure

New in Intel® MKL 9.1

System Requirements

Installation Notes

Documentation

Known Limitations

Technical Support and Feedback

Related Products and Services

Disclaimer and Legal Information

Overview

The Intel® Math Kernel Library (Intel® MKL) provides developers of scientific, engineering and financial software with a set of linear algebra routines, fast Fourier transforms, and vectorized math and random number generation functions, all optimized for the latest Intel® Pentium® 4 processors, Intel® Xeon® processors with Streaming SIMD Extensions 3 (SSE3) and Intel® Extended Memory 64 Technology (Intel® EM64T), and Intel® Itanium® 2 processors. This software also performs well on non-Intel (x86) processors.

Intel MKL provides linear algebra functionality with LAPACK (solvers and eigensolvers) plus level 1, 2, and 3 BLAS offering the vector, vector-matrix, and matrix-matrix operations needed for complex mathematical software. Users who prefer the FORTRAN 90/95 programming language may call LAPACK driver and computational subroutines via specially designed interfaces with reduced numbers of arguments. The Cluster Edition of Intel MKL provides ScaLAPACK (SCALable LAPACK) and support functionality including the Parallel Basic Linear Algebra Subprograms (PBLAS). For solving sparse systems of equations, Intel MKL provides direct and iterative sparse solvers as well as a supporting set of sparse BLAS (levels 1, 2, and 3). Intel MKL offers multidimensional fast Fourier transforms (1D, 2D, 3D) with mixed radix support (not limited to sizes of powers of 2). The Cluster Edition of Intel MKL also provides distributed versions of these functions for use on clusters. For the solution of Partial Differential Equations, Intel MKL provides some useful tools to create efficient preconditioners. Optimization solvers provide efficient routines for solution nonlinear least square problems with and without boundary constraints. Intel MKL also includes a set of vectorized transcendental functions (called the Vector Math Library (VML)) offering both greater performance and excellent accuracy compared to the libm (scalar) functions for most of the processors. The Vector Statistical Library (VSL) offers high performance vectorized random number generators for a number of probability distributions as well as convolution and correlation routines. Intel MKL also includes a set of functions which act on intervals of floating point numbers. This

interval arithmetic package includes solvers for interval systems of linear equations, interval matrix inversion, as well as functions for testing regularity/singularity of interval matrices. The BLAS, LAPACK, direct sparse solver (DSS), FFT, VML, Poisson library functions, and optimization solvers in Intel MKL are threaded using OpenMP*. All of Intel MKL is thread-safe.

Directory Structure

An orientation to the directory structure of Intel® MKL can be found in the User's Guide (filename: userguide.pdf).

New in Intel® MKL 9.1

Performance improvements since Intel® MKL 9.0

- BLAS
 - Threading of DGEMM was improved for small and middle sizes - outer product sizes by 10%, square sizes by 80%
 - DTRSM, DTRMM, and DSYRK were improved by 5-30%
- LAPACK
 - Significant improvements in Nonsymmetric Eigenproblem due to improvements made on top of LAPACK 3.1
 - *HETRD improved by 75% on cache
 - *SYTRD improved by 40% on cache
 - Successive Bandwidth Reduction approach has been implemented in *HERDB/*SYRDB for a speedup of 2.8 times versus traditional *HETRD/*SYTRD on Dual-Core Xeon® 5100 series servers.
 - Improved CROT/ZROT performance by 80% for Intel® Core™2 Duo processors
 - Improved tridiagonal linear equations solver (DGTSV/SGTSV) by 8-10%
 - Improved tridiagonal symmetric positive definite linear equations solver (*PTSV) by 8-28%
 - Improved generalized non-symmetric eigenproblem (*GGEV) by 3-8%, and CGGEV by 30%.
 - Threaded upper triangular real*8 Cholesky decomposition
 - Improved performance of *POTRF functions by up to 1.5 times for small sizes on Intel® Core™2 Duo processors and Itanium® processors.
 - Performance of (D/S/Z/C)STEDC was improved by 30% and parallelized with OpenMP for up to 100% performance improvement on 4 threads.
- Sparse BLAS
 - Performance of triangular solver routines for the diagonal format was improved by 20-50% in serial mode affecting the following functions: mkl_ddiasm, mkl_ddiasv, and mkl_ddiatrsv
 - Level 3 triangular solver mkl_ddiasm was threaded.
- Direct Sparse Solver (DSS/PARDISO)
 - DSS/PARDISO – performance is improved by 20-30% from last release for symmetric, positive-definite matrices with one and multiple right-hand sides.
- FFTs
 - Single complex forward 1D FFT size greater than 2^{22} were sped up by up to 2 times on 4 threads and up to 2.4 times on 8 threads on Itanium®

- processors.
- o Double precision complex out-of-place 1D FFTs for power-of-two sizes was sped up by up to 2 times on 4 threads.
- o Complex 2D/3D FFTs were sped up for double precision by up to 1.25 times and for single precision up to 1.4 times on 1 thread and for double precision by up to 1.2 times and for single precision up to 1.3 times on 4 threads on systems with Intel® EM64T and running in 64-bit mode.
- o Parallel Real 2D/3D FFTs were sped up for double precision by up to 1.7 times and for single precision up to 1.6 times on systems with Intel® EM64T and running in 64-bit mode.
- VML
 - o Performance improvements 5-25% for double precision exp, single/double cube root, and single/double complex exp, ln, sqrt on systems with Intel® EM64T and running in 64-bit mode.
 - o Performance of the LA (low accuracy) ErfInv function was improved by 50-70% for all processors and floating-point precisions.

Other Improvements

- LAPACK
 - o LAPACK 3.1 inclusion for increased accuracy, stability, and performance.
 - o Merged LAPACK dynamic libraries into single one
- ScaLAPACK
 - o Version 1.7.5 of ScaLAPACK has been implemented
- FFTs
 - o Included parallel double precision complex out-of-place 1D FFT for power-of-two sizes
 - o FFTW wrappers are now placed in the lib directory corresponding to the architecture chosen
- ILU(0), an accelerator/preconditioner for the RCI FGMRES iterative solver, is now included for the CSR matrix format
- PDE Support: Added new fast Helmholtz and Poisson solvers on a sphere
- New Optimization solvers routines
 - o Added new solver for nonlinear least square problems without boundary constraints
 - o Added new solvers for nonlinear least square problems with boundary constraints
- New Conjugate Gradient solver with Multiple Right-Hand Sides was implemented and added
- Improvements to the Direct Sparse Solver
 - o Memory information was added to the dss_statistics call
- Improvements to ScaLAPACK
 - o Address error fixed for large problems (>2GB)
- Improvements to the VML and VSL
 - o Added new method of generation random numbers of the normal

- distribution, based on the inverse transformation
 - Fixed VSL bug when the quasi-random number generator's state working in dimensions 3 or 4 was incorrectly updated
 - Fixed VML threading logic bug that didn't allow to create more than two threads.
- The following two Linpack benchmarks has been added to the benchmarks directory:
 - Intel(R) Optimized Linpack Benchmark
 - Intel(R) Optimized MP Linpack Benchmark for Clusters
 Binary versions of these benchmarks are optimized for and run on only genuine Intel processors.
- A 64-bit integer (ILP64) interface for the library is now provided through addition of new library files in the main product package (see the User's Guide for further information)
- A serial version of the library with no threading library dependencies is now a part of the main package
 - A serial version of the direct sparse solver (DSS/PARDISO) available for the first time

Note: There are two options for downloading and installing Intel MKL 9.1 Cluster Edition for Linux:*

- *A standard package containing the Intel MKL product: `I_cluster_mkl_p_9.1.0xx.tgz`*
- *An extended package containing the Intel MKL product plus ILP64 and serial versions of the library: `I_cluster_mkl_enh_p_9.1.0xx.tgz`*

Unless you require serial or ILP64 versions of the libraries, it is recommended that you download the standard version due to package size.

- Fortran 95 interfaces: The default location for module and library files for the Fortran 95 interface to MKL has changed. When built these interface files will be placed in the appropriate lib directory (based on architecture).
- MKL Java examples are added to illustrate using MKL with Java. The examples support the following Java implementations:
 - J2SE SDK 1.4.2, and JDK 5.0 and 6.0 from Sun Microsystems (<http://www.sun.com>)
 - JRockit JDK 1.4.2 and 5.0 for IA32, Intel 64 and Itanium architectures from BEA (<http://www.bea.com>)
- Linux Standard Base (LSB): Intel MKL is now LSB compliant

- gfortran libraries have been added for compatibility in functions return real and complex results.
- The "Technical User Notes" and "Getting Started Guides" have been replaced by a "User's Guide" (userguide.pdf) document in the doc directory.
- The User Guide document has been restructured. Configuring of Eclipse CDT to link with Intel MKL is described. More aspects of ILP64 interface are discussed. Tips for selecting between static and dynamic linking are given. Changing the number of processors for threading at run time is described. The memory renaming feature is explained in detail with code examples. Coding techniques for calling LAPACK, BLAS, and CBLAS routines from C are presented and illustrated with examples.
- The MKL Reference manual now describes BLACS routines. LAPACK chapters fully reflect LAPACK 3.1 update. Chapters on BLAS (new SparseBLAS functions supporting zero-based indexing and BSR format have been documented) and FFT have been extended. Description of BSR format has been added to Appendix A and new FFT examples have been added to Appendix C.
- Support functions are now documented in the reference manual.

System Requirements

Hardware

To install and use Intel MKL you will need a system with a supported processor and 550 MB of free hard disk space plus an additional 300 MB during installation for download and temporary files (host system only). The requirement for the package with serial and ILP64 libraries is 1 GB plus 500 MB at installation.

Supported processors - The following is a list of processors on which Intel MKL is expected to run.

- Intel® Core™2 QuAD processor
- Dual-Core Intel® Xeon® processor MP 7000 sequence
- Dual-Core Intel® Xeon® processor 5100 sequence
- Intel® Core™ Duo processor
- Intel® Core™2 Duo processor
- 64-bit Intel® Xeon® processor
- Intel® Itanium® 2 processor
- Intel® Xeon® processors DP & MP
- Intel® Pentium® D processor
- Intel® Pentium® M processor
- Intel® Pentium® 4 processor with Streaming SIMD Extensions 3
- Intel® Pentium® 4 processor
- Intel® Celeron® D processor
- Intel® Celeron® processor
- Intel® Pentium® III processor
- AMD Athlon* and Opteron* processors

Software

To use Intel MKL you will need a supported compiler and MPI implementation.

Recommended software - The following software is recommended for use with Intel MKL:

- Intel® Fortran Compiler for Linux* version 10.0
- Intel® C++ Compiler for Linux* version 10.0
- Intel® MPI Library version 3.0
- Xcode 2.4.1

Following is the list of supported operating systems:

- Red Hat* EL3 (IA32/EM64T/Itanium)
- Red Hat* EL4 (IA32/EM64T/Itanium)
- Red Hat* EL5 (IA32/EM64T/Itanium)
- Red Hat* Fedora* Core 5 (IA32/EM64T)
- Red Hat* Fedora* Core 6 (IA32/EM64T)
- Red Flag* DC Server 5.0 (IA32/EM64T/Itanium)
- Suse* SLES* 9 (IA32/EM64T/Itanium)
- Suse* SLES* 10 (IA32/EM64T/Itanium)
- Mandriva/Mandrake Linux* 10.1 (IA32/EM64T)
- HaanSoft Linux* 2006 Server (IA32/EM64T/Itanium)
- Miracle Linux* 4.0 (IA32/EM64T/Itanium)
- Turbo Linux* 10 (IA32/EM64T/Itanium)

Following is the list of supported C/C++ and Fortran compilers:

- Intel® Fortran Compiler for Linux* version 8.1
- Intel® Fortran Compiler for Linux* version 9.1
- Intel® Fortran Compiler for Linux* version 10.0
- Intel® C++ Compiler for Linux* version 8.1
- Intel® C++ Compiler for Linux* version 9.1
- Intel® C++ Compiler for Linux* version 10.0
- GNU Compiler Collection

Following is the list of MPI implementations that Intel MKL has been validated against:

- Intel® MPI Library Version 3.0
- Intel® MPI Library Version 2.0
- SGI MPT
- Open MPI 1.1.2, 1.1.5, and 1.2 found at <http://www.open-mpi.org>
- MPICH version 1.2.5 configured with device ch_vapi available at <http://www.topspin.com/>
- MPICH version 1.2.5.10 (Myricom*'s designation) available at <http://www.myri.com/>
- MPICH version 1.2.5.2 available at <http://www-unix.mcs.anl.gov/mpi/mpich>
- MPICH version 2.0 available at <http://www-unix.mcs.anl.gov/mpi/mpich>

Note: See the User's Guide in the doc directory for linking instructions.

Note:

- Intel MKL has parts which have Fortran interfaces, and are Fortran in their data structures, and parts which have C interfaces and have C data structures. The User Guide in the doc directory contains advice on how to link to Intel MKL with different compilers.
- The above lists of processor model names are not exhaustive - other processor models correctly supporting the same instruction set as those listed are expected to work. Please contact Intel® Premier Support if you have questions regarding a specific processor model.
- Adobe Acrobat* Reader 6.0 or later is required to view some of the reference documentation.

Installation Notes

Guidance on the installation of Intel MKL is provided at install time. Links will be provided to a file with step by step instructions (filename: Install.txt). This file can also be found in the doc directory.

Documentation

The Documentation Index (Doc_Index.htm in the doc directory) has a list of the principal MKL documents. For a complete list, see Table 3-8 on p.3-14 of the User's Guide. An HTML WebHelp version of the MKL Reference Manual is available at <http://www.intel.com/software/products/mkl/docs/WebHelp/mkl.htm>.

Known Limitations

Limitations to the sparse solver and optimization solvers in Intel MKL 9.1:

- The default number of threads (when OMP_NUM_THREADS is not set) is equal to the number of processors in the system. This differs from the default OpenMP mode in Intel MKL. By default the number of threads is set to one.
- Only statically linkable sparse and optimization solver library files will be available with this release.
- Enhanced precision accumulation is implemented in long doubles (10 bytes real precision) instead of in quad doubles (16 bytes real precision) as in the original version of PARDISO.

Limitations to the FFT functions in Intel MKL 9.1:

- The function DftiCopyDescriptor is not implemented.
- Mode DFTI_TRANSPOSE is implemented only for the default case.
- Mode DFTI_REAL_STORAGE can have the default value only and are not changeable by the DftiSetValue function (i.e. DFTI_REAL_STORAGE = DFTI_REAL_REAL)
- Some limitations exist on arrays sizes for Cluster FFT functions. See mklman.pdf for a detailed description.

Limitations to the LAPACK functions in Intel MKL 9.1:

- The ILAENV function, which is called from the LAPACK routines to choose problem-

dependent parameters for the local environment, can not be substituted by user's version.

Limitations to the Vector Statistical Library (VSL) functions in Intel MKL 9.1:

- The version of the Convolution and Correlation functions for Intel® EM64T fail on AMD* processors that do not support SSE3 instructions.

Limitations to the interval arithmetic functions in Intel MKL 9.1:

- The interval libraries will require the libifcore library from Intel® Fortran compiler.
- There is no "default" version of the interval arithmetic functions. In other words, all versions contain SSE instructions.

Limitations to the multi-precision functions in Intel MKL 9.1:

- If your application is dynamically linked to Intel MKL and contains calls to GMP functions you will have to link with the libvml and libm libraries. For example: `$CC prog.c -L$MKL_LIB_PATH -lmkl -lvml -lguide -lpthread -lm`

Limitations to the ScaLAPACK functions in Intel MKL 9.1:

- The user can not substitute PJLAENV for their own version. This function is called by LAPACK routines to choose problem-dependent parameters for the local environment.
- Only statically linkable libraries files are available

Limitations to the ILP64 version of Intel MKL:

- The ILP64 version of Intel MKL does not contain the complete functionality of the library. For a full listing of what is in the ILP64 version refer to the user's guide in the doc directory.
- gfortran can not be used with the ILP64 libraries.

Limitations to the Fortran 95 interface to LAPACK

- If you are compiling the Fortran 95 interface to LAPACK with the GNU gfortran compiler you must manually remove the "pure" attribute from all subroutines containing a procedure argument: ?GEES, ?GEESX, ?GGES, ?GGESX (where ? can be S, D, C, or Z).

On Intel® processors with Intel® EM64T enabled, user programs compiled with the GNU Fortran compiler (version 3.2.3) will likely get incorrect results from those functions in Intel® MKL that return single precision values, if `-fno-f2c` GNU Fortran compiler flag isn't used. The GNU Fortran compiler by default expects REAL*4 values in the first 8 bytes of the return register (just as a double precision value would be represented) while the Intel® Fortran compiler expects REAL*4 values in the first 4 bytes of the return register. The behavior of Intel MKL is compatible with that of the Intel Fortran compiler. GNU Fortran compiler behavior could be changed to be compatible with the Intel Fortran

compiler by using the `-fno-f2c` flag.

FFT, VML, VSL, and PDE support functions can not be used with Fortran 77 compilers.

We recommend that `-Od` be used for the 10.0 Intel® compilers when compiling test source code available with Intel MKL. Current build scripts do not specify this option and default behavior for these compilers has changed to provide vectorization.

All VSL functions return error status of current operation, i.e., default VSL API is a function style now rather than a subroutine style used in earlier MKL versions. This means that Fortran users should call VSL routines as functions. For example:

```
errstatus = vslnrggaussian(method, stream, n, r, a, sigma)
```

rather than subroutines:

```
call vslnrggaussian(method, stream, n, r, a, sigma)
```

Nevertheless, MKL provides subroutine-style interface for backward compatibility also. To use subroutine-style interface, manually include `mkl_vsl_subroutine.fi` file instead of `mkl_vsl.fi` by changing the line `include 'mkl_vsl.fi'` in `include\mkl.fi` with the line `include 'mkl_vsl_subroutine.fi'`. VSL API changes don't affect C/C++ users.

Limitations to the Java examples:

- The Java examples don't work with static MKL libraries. Please use the dynamic MKL libraries for running the Java examples.

Hyper-Threading Technology (HT Technology) is especially effective when each thread is performing different types of operations and when there are under-utilized resources on the processor. Intel MKL fits neither of these criteria as the threaded portions of the library execute at high efficiencies (using most of the available resources) and perform identical operations on each thread. You may obtain higher performance when using Intel MKL without HT Technology enabled.

Memory Allocation: In order to achieve better performance, memory allocated by Intel MKL is not released. This behavior is by design and is a one time occurrence for Intel MKL routines that require workspace memory buffers. Even so, the user should be aware that some tools may report this as a memory leak. Should the user wish, memory can be released by the user program through use of a function (`MKL_FreeBuffers()`) made available in Intel MKL or memory can be released after each call by setting an environment variable (`MKL_DISABLE_FAST_MM`) (see User's Guide in the `doc` directory for more details). Using one of these methods to release memory will not necessarily stop programs from reporting memory leaks, and in fact may increase the number of such reports should you make multiple calls to the library thereby requiring new allocations with each call. Memory not released by one of the methods described will be released by the system when the program ends. The maximum number of buffers allocated in each thread is 32. To avoid this restriction disable memory management as described above.

Memory management has a restriction for the number of allocated buffers in each thread. Currently this number is 32. To avoid this restriction disable memory management as described above.

On Red Hat* Enterprise Linux 3.0, in order to ensure that the correct support libraries are linked, the environment variable `LD_ASSUME_KERNEL` must be set: For example: `'export LD_ASSUME_KERNEL=2.4.1'`

Technical Support and Feedback

Self Help and User Forums

A rich repository of self-help product information such as tutorials, getting started tips, known product issues, product errata, compatibility information and answers to frequently asked questions can be found at the Intel® Software Development Products Technical Support site (<http://www.intel.com/software/products/support/index.htm>). It's a great place to find answers quickly or to gain insight in using our products effectively.

The Intel MKL User Forum (<http://softwareforums.intel.com/ids/board?board.id=MKL>) is the place to ask questions of and share information with other users of Intel® MKL.

Submitting Issues

Your feedback is very important to us. To receive technical support and product updates for the tools provided in this product you need to register at the Intel® Registration Center (<https://registrationcenter.intel.com/>).

If you have questions or problems getting started with the Intel® Math Kernel Library please contact support at <https://registrationcenter.intel.com/support/>.

Note: Please notify your support representative prior to submitting source code where access needs to be restricted to certain countries to determine if this request can be accommodated.

To submit an issue via the Intel® Premier Support website, please perform the following steps:

1. Ensure that Java* and JavaScript* are enabled in your browser.
2. Go to <http://www.intel.com/software/products/support>.
3. Type in your Login and Password. Both are case-sensitive.
4. Click the "Submit Issues" button.
5. Read the Confidentiality Statement and click the "I Accept" button.
6. Click on the "Go" button next to the "Product" drop-down list.
7. Click on the "Submit Issue" link in the left navigation bar.
8. Choose "Development Environment (tools,SDV,EAP)" from the "Product Type" drop-down list.
9. If this is a software or license-related issue choose "**Intel(R) MKL Cluster Edition for Linux***" from the "Product Name" drop-down list.
10. Enter your question and complete the fields in the windows that follow to successfully submit the issue.

Please follow these guidelines when forming your problem report or product suggestion:

1. Describe your difficulty or suggestion:
For problem reports please be as specific as possible (e.g., including compiler and link command line options), so that we may reproduce the problem. Please include

a small test case if possible.

2. Describe your system configuration information:

Be sure to include specific information that may be applicable to your setup: operating system, name and version number of installed applications, and anything else that may be relevant to helping us address your concern.

Related Products and Services

Information on Intel® software development products is available at <http://www.intel.com/software/products>. Some of the related products include:

- The [Intel® Software College](#) provides interactive tutorials, documentation, and code samples that teach Intel® architecture and software optimization techniques.
- The [VTune™ Performance Analyzer](#) allows you to evaluate how your application is utilizing the CPU and helps you determine if there are modifications you can make to improve your application's performance.
- The [Intel® C++ and Fortran Compilers](#) are an important part of making software run at top speeds and fully support the latest Intel IA-32 and Itanium® processors.
- The [Intel® Performance Library Suite](#) provides a set of routines optimized for various Intel® processors. The Intel® Math Kernel Library, which provides developers of scientific and engineering software with a set of linear algebra, fast Fourier transforms and vector math functions optimized for the latest Intel Pentium and Intel Itanium® processors. The Intel® Integrated Performance Primitives consists of cross platform tools to build high performance software for several Intel architectures and several operating systems.

Attribution

As referenced in the End User License Agreement, attribution requires, at a minimum, prominently displaying the full Intel product name (e.g. "Intel® Math Kernel Library") and providing a link/URL to the Intel MKL homepage (www.intel.com/software/products/mkl) in both the product documentation and website.

The original versions of the BLAS from which that part of Intel MKL was derived can be obtained from <http://www.netlib.org/blas/index.html>. The original versions of LAPACK from which that part of Intel MKL was derived can be obtained from <http://www.netlib.org/lapack/index.html>. The authors of LAPACK are E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. Our FORTRAN 90/95 interfaces to LAPACK are similar to those in the LAPACK95 package at <http://www.netlib.org/lapack95/index.html>. All interfaces are provided for pure procedures. The original versions of ScaLAPACK from which that part of Intel MKL was derived can be obtained from <http://www.netlib.org/scalapack/index.html>. The authors of ScaLAPACK are L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley.

Some FFT functions in this release of Intel MKL have been generated by the SPIRAL software generation system (<http://www.spiral.net/>) under license from Carnegie Mellon

University. Some FFT functions in this release of the Intel MKL DFTI have been generated by the UHFFT software generation system under license from University of Houston. The Authors of SPIRAL are Markus Puschel, Jose Moura, Jeremy Johnson, David Padua, Manuela Veloso, Bryan Singer, Jianxin Xiong, Franz Franchetti, Aca Gacic, Yevgen Voronenko, Kang Chen, Robert W. Johnson, and Nick Rizzolo.

Disclaimer and Legal Information

The information in this manual is subject to change without notice and Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document. This document and the software described in it are furnished under license and may only be used or copied in accordance with the terms of the license. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. The information in this document is provided in connection with Intel products and should not be construed as a commitment by Intel Corporation.

EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The software described in this document may contain software defects which may cause the product to deviate from published specifications. Current characterized software defects are available on request.

Intel, the Intel logo, Intel SpeedStep, Intel NetBurst, Intel NetStructure, MMX, i386, i486, Intel386, Intel486, Intel740, IntelDX2, IntelDX4, IntelSX2, Celeron, Intel Centrino, Intel Xeon, Intel XScale, Itanium, Pentium, Pentium II Xeon, Pentium III Xeon, Pentium M, and VTune are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

* Other names and brands may be claimed as the property of others.

Copyright (C) 2000 - 2007, Intel Corporation.