



THINK PARALLEL



Parallel programming features supported by the Intel® C++ Compilers and Intel Fortran Compilers enable developers to increase application performance, accelerate the development of multi-threaded programs, and simplify the transformation of sequential programs into parallel programs.

The Intel C++ and Fortran Compilers support three major parallel programming features: parallelization with the OpenMP* application programming interface (API), auto-parallelization, and auto-vectorization. Each of these features contributes to application performance depending on the number of processors, the target architecture (Intel® IA-32 or Itanium® architecture), and the nature of the application. These parallel programming features can be combined to enhance application performance.

Parallel programming can be *explicit*, that is, defined by a programmer using OpenMP directives. Parallel programming can also be *implicit*, that is, detected automatically by the compiler. Implicit parallelism implements auto-parallelization of outermost loops and auto-vectorization of innermost loops (or both). See Figure 1.

Parallelism defined with OpenMP and auto-parallelization directives is based on thread-level parallelism (TLP). Parallelism defined with auto-vectorization techniques is based on instruction-level parallelism (ILP).

The Intel compilers support OpenMP and auto-parallelization on both IA-32 and Itanium architectures for multiprocessor systems as well as on single IA-32 processors with Intel® Hyper-Threading Technology. Auto-vectorization is supported on the families of the Intel® Pentium®, Pentium with MMX™ Technology, Pentium II, Pentium III, and Pentium 4 processors. To enhance the compilation of the code with auto-vectorization, programmers can also add vectorizer directives to their program. A closely related technique that is available on the Itanium-based systems is software pipelining.

Parallel program development

The Intel Fortran Compilers support the OpenMP FORTRAN version 2.0 API specification available from the www.openmp.org Web site. The OpenMP directives relieve the programmer from having to deal with the low-level details of iteration space partitioning, data sharing, and thread scheduling and synchronization.

The auto-parallelization feature of the Intel C++ and Fortran Compilers automatically translates serial portions of the input program into

semantically equivalent multi-threaded code. Automatic parallelization determines the loops that are good work-sharing candidates, performs the data-flow analysis to verify correct parallel execution, and partitions the data for threaded code generation as is needed in programming with OpenMP directives. The OpenMP and auto-parallelization applications provide the performance gains from shared memory on multiprocessor systems and IA-32 processors with Hyper-Threading Technology.

Auto-vectorization detects low-level operations in the program that can be done in parallel, and then converts the sequential program to process 2, 4, 8, or up to 16 elements in one operation, depending on the data type. In some cases auto-parallelization and auto-vectorization can be combined to enhance performance. For example, TLP can sometimes be exploited in the outermost loop, while ILP can be exploited in the innermost loop.

Auto-vectorization can help increase the performance of an application that runs on systems based on Intel Pentium, Pentium with MMX Technology, Pentium II, Pentium III, and Pentium 4 processors.

Choosing effective options enables programmers to increase the performance of their application with minimal effort and use compiler features to accelerate the development of multi-threaded programs. Additionally, with the relatively small effort of adding OpenMP directives to their code, programmers can transform a sequential program into a parallel program.

Parallelism	Description
Explicit	<ul style="list-style-type: none"> Implements OpenMP* (TLP) parallelization Programmed by the developer Supported on Intel® IA-32 or Itanium® architecture-based multiprocessor systems and IA-32 Intel® Hyper-Threading Technology-enabled systems
Implicit	<ul style="list-style-type: none"> Implements auto-parallelization (TLP) of outermost loops and auto-vectorization (ILP) of innermost loops Generated by the compiler and by developer-supplied hints Auto-parallelization supported on IA-32 or Itanium-based multiprocessor systems and IA-32 Hyper-Threading Technology-enabled systems Auto-vectorization supported on Intel® Pentium®, Pentium with MMX™ Technology, Pentium II, Pentium III, and Pentium 4 processors

Figure 1. Comparison of explicit and implicit parallelism

For more information

Intel software products:

www.dell.com/intelsoftware
www.intel.com/software/products

Intel software white papers:

www.intel.com/software/products/whitepapers