



# Quick-Reference Guide to Optimization with Intel® Compilers

For IA-32 processors, processors supporting Intel® Extended Memory  
64 Technology (Intel® 64) and Intel® Itanium® (IA-64) processors.

## Application Performance

### A Step-by-Step Approach to Application Tuning with Intel® Compilers

Before you begin performance tuning, you may want to check correctness of your application by building it with the basic set of optimization options using `/Od (-O0)`, or building in debug mode using the `/Zi (-g)` option.

1. Use the Automatic Optimization Options (Windows `/O1`, `/O2` or `/O3`; Linux and Mac OS `-O1`, `-O2`, or `-O3`) and determine which one works best for your application by measuring performance with each. Most users should start at `/O2 (-O2)` (default) (`/O3 (-O3)` on IA-64-based systems), then try the processor-specific optimization option before adding interprocedural optimization (IPO).
2. Fine-tune performance with processor-specific options to target systems based on IA-32 and Intel 64 using `/O2 (-O2)` first, then `/QxP (-xP)`. On IA-64-based systems, try `/O3 (-O3)` first. For Intel® Itanium® 5100 series processors, set `/G2-p9000 (-mtune=itanium2-p9000)`.
3. Fine-tune performance with processor-specific options to target systems based on the latest Intel 64 systems: Intel® Core™2 Duo processors, Intel® Core™2 Extreme processors, or Dual-Core Intel® Xeon® 5100 series processors using `/O2 (-O2)` first, then `/QxT (-xT)`.
4. Step 2 (or 3) works best by identifying performance "hotspots" with the Intel® VTune™ Performance Analyzer so you know which specific parts of your application need tuning. The Intel Compilers' optimization reports also help by showing where the compiler could use more of your attention.
5. Add in interprocedural optimization (IPO) (`/Qipo (-ipo)`) and/or profile-guided optimization (PGO) (`/Qprof-gen (-prof-gen)`, `/Qprof-use (-prof-use)`), and measure performance again to determine whether your application benefits from either of them.
6. Run your applications on multi-core, multi-processor, or processors with Hyper-Threading Technology (HT Technology)-capable systems using the Parallel Performance options (`/Qparallel (-parallel)`, `/Qopenmp (-openmp)`).

Please consult the Compilers' User Documentation and the "Optimizing Applications with the Intel® C++ & Fortran Compilers" white paper for more details.

## Included in this Guide:

### General Optimization Options

Begin performance tuning with **/O1**, **/O2**, or **/O3 (-O1, -O2, or -O3)**. Then try different options and measure your performance before proceeding with more advanced options. Before you begin performance tuning, you may want to check correctness of your application by building it with the basic set of optimization options using **/Od (-O0)**, or building in debug mode using the **/Zi (-g)** option. These are general optimization options that should be at the heart of any application tuning for all 32-bit and 64-bit Intel processors.

### Interprocedural Optimization (IPO) and Profile-Guided Optimization (PGO) Options

IPO includes function-inlining to reduce function call overhead and improve data layout across functions. PGO provides runtime feedback to guide optimization decisions about data and code layout to improve instruction-cache efficiency, paging and branch prediction. However, IPO can increase code size. Be sure to measure your execution performance, compile time, and code size tradeoffs with these options. IPO is best used in conjunction with PGO to guide which functions to inline.

### Floating-Point Arithmetic Options

The Intel® compilers provide options for enhancing the consistency or precision of floating-point results on all Intel® architectures. Refer to the Compiler Options section of the Intel Compiler documentation for detailed information on floating-point options.

### Optimizations Specific to IA-32 Processors and Processors Supporting Intel® 64

These options allow you to tune performance specifically for the Intel processor-based systems you are targeting. As with each previous step, measure the performance benefit of each option to guide your decisions. Use the Intel compilers' optimization reports to assist in determining whether you can provide more help to the compiler to resolve possible dependencies or aliases.

**Intel 64-Specific Optimization Recommendation:** We recommend **/QxP (-xP on Linux\*)** for best performance across all Intel processors with Intel 64. We recommend **/QxW /QaxP (-xW -axP on Linux\*)** for best performance on all Intel 64 capable systems, including those from AMD\*. For Intel's latest Intel 64 processors, the Intel® Core™2 Duo, Intel® Core™2 Extreme, and the Dual-Core Intel® Xeon® 5100 series processors, we recommend **/QxT (-xT)** for these specifically. Use **/QxW /QaxT (-xW -axT)** to include other Intel 64 and like AMD\* processors as well.

**IA-32-Specific Optimization Recommendation:** Use the **/QxN (-xN on Linux\*)** for best performance across all Pentium 4 processors and the Pentium® M processor. (You may also want to experiment with **/QxB (-xB on Linux\*)** on Pentium M processors.)

### Intel® Itanium® (IA-64) Processor-Specific Optimization Options

In general, using **/O3 (-O3)**, IPO, and/or PGO, in conjunction with the optimization reports (described in the Fine-Tuning section of this document) to help resolve possible aliases and improve memory utilization provides the best performance for IA-64 based systems.

### Fine-Tuning (All Processors)

Once you have identified performance hot-spots, you may need to provide the compiler with more information to fine-tune specific functions. The optimization and vectorization reports may show places where loops could not be optimized fully due to pointer aliasing or memory-access overlaps, for example. The *Intel® C++ and Fortran Compiler Documentation* includes details on other #pragmas, directives, and intrinsics that can be used to control software-pipelining, unrolling, vectorization, and prefetching for further fine-tuning within your application code.

### Parallel Performance

These options allow the compiler to help you parallelize your application for multi-processor, multi-core or processors with Hyper-Threading Technology<sup>1</sup>.

### Recommended Optimization Options for Intel® Processors

Recommended optimization settings for best performance on Intel processor-based systems. As with each previous step, measure the performance benefit of each option to guide your decisions. Use the Intel compilers' optimization reports to assist in determining whether you can provide more help to the compiler to resolve possible dependencies or aliases and improve memory utilization.

## General Optimization Options

Windows*	Linux* Mac OS*	Comment
/Od	-O0	<b>No optimization.</b> Used during the early stages of application development and debugging. Use a higher setting when the application is working correctly.
/O1	-O1	<b>Optimize for size.</b> Omits optimizations that tend to increase object size. Creates the smallest optimized code in most cases.  This option is useful in many large server/database applications where memory paging due to larger code size is an issue.
/O2	-O2	<b>Maximize speed.</b> Default setting. Creates the fastest code in most cases, but may increase code size significantly over /O1 (-O1).
/O3	-O3	Enables /O2 (-O2) optimizations plus more aggressive optimizations, such as prefetching, scalar replacement, and loop and memory access transformations. Enables optimizations for maximum speed, such as loop unrolling, code replication to eliminate branches, and padding the size of certain power-of-two arrays to allow more efficient cache use.  On IA-32 and Intel 64 processors, when /O3 (-O3) is used with options /Qax or /Qx (-ax or -x), the compiler performs more aggressive data dependency analysis than for /O2 (-O2), which may result in longer compilation times.  On IA-64 processors, the /O3 (-O3) option enables optimizations for technical computing applications (loop-intensive code): loop optimizations and data prefetch.  The /O3 (-O3) optimizations may not cause higher performance unless loop and memory access transformations take place. The optimizations may slow down code in some cases compared to /O2 (-O2) optimizations. The /O3 (-O3) option is recommended for applications that have loops that heavily use floating-point calculations and process large data sets.
/fast	-fast	The /fast (-fast) option maximizes speed across the entire program. For IA-64-based systems, /fast (-fast) is equivalent to /O3 (-O3), /Qipo (-ipo), and -static (Linux only).  For IA-32 and Intel 64 based systems, /fast (-fast) is equivalent to /O3 (-O3), /Qipo (-ipo), -static (Linux only), and /QxP (-xP).  Note: For Mac OS, -fast is equivalent to -O3 -ipo. The -static option is not supported.
/Zi	-g	Generates debug information for use with any of the common platform debuggers. This option turns off /O2 (-O2) and makes /Od (-O0) the default unless /O2 (-O2) (or another option) is specified.

## Interprocedural Optimization (IPO) and Profile-Guided Optimization (PGO)

Windows*	Linux* Mac OS*	Comment
/Qip	-ip	Single file optimization. Interprocedural optimizations, including selective inlining, within the current source file.  Caution: For large files, this option may sometimes significantly increase compile time and code size.
/Qipo[value]	-ipo[value]	Permits inlining and other optimizations among multiple source files. The optional <b>value</b> argument controls the maximum number of link-time compilations (or number of object files) spawned. Default for <b>value</b> is 0 (the compiler chooses).  Caution: This option can in some cases significantly increase compile time and code size.
/Ob2	-finline-functions	This option enables function inlining at the compiler's discretion. This option is enabled by default at /O2 and /O3 (-O2 and -O3).  Caution: For large files, this option may sometimes significantly increase compile time and code size. It can be disabled by /Ob0 (-finline-functions on Linux* and Mac OS*).
/Qprof-gen	-prof-gen	Instruments a program for profiling.
/Qprof-use	-prof-use	Enables the use of profiling information during optimization.
/Qprof-dir <i>dir</i>	-prof-dir <i>dir</i>	Specifies a directory for the profiling output files, *.dyn and *.dpi.

## Floating-Point Arithmetic Optimizations

Windows*	Linux* Mac OS*	Comment
/fp:name	-fp- model name	This method of controlling the consistency of floating point results by restricting certain optimizations is recommended in preference to the /Op (-mp) and /Qprec (-mp1) switches. The possible values of <b>name</b> are:  <b>precise</b> – Enables only value-safe optimizations on floating point code. <b>double/extended/source</b> – Implies <b>precise</b> and causes intermediates to be computed in double, extended or source precision. The <b>double</b> and <b>extended</b> options are not available for Intel® Fortran. <b>fast=[1 2]</b> – Allows aggressive optimizations at a slight cost in accuracy or consistency. ( <b>fast=1</b> is the default) <b>except</b> – Enables floating point exception semantics. <b>strict</b> – Strictest mode of operation, enables both the <b>precise</b> and <b>except</b> options and disables contractions.  <b>Recommendation:</b> /fp:source (-fp-model source) is the recommended form for the majority of situations on IA-64 processors, on processors supporting Intel® 64, and on IA-32 processors when SSE are enabled with /Qxw (-xw) or higher.

## Optimizations Specific to IA-32 Processors and Processors Supporting Intel® 64

<p><b>/Qax</b> (T P W N B K)</p>	<p><b>-ax</b> (T P W N B K)</p>	<p>Automatic Processor Dispatch. Generates specialized code and enables vectorization for the indicated processors while also generating non-processor-specific code. You can use more than one letter to tune for multiple processors in the same executable.</p> <p><b>T</b> – Generate Supplemental Streaming SIMD Extensions 3 (SSSE3) instructions for Intel Core 2 Duo processor, Intel Core 2 Extreme processor and the Dual-Core Intel Xeon processor.</p> <p><b>P</b> – Generate SSE3, SSE2 and SSE instructions for Intel processors and optimize for Intel Core Duo processor, Intel Core Solo processor, Intel Pentium 4 processor with SSE3, and Intel Xeon processor with SSE3. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>W</b> – Generate SSE2 and SSE instructions and optimize for the Intel Pentium 4 processor, Intel Xeon processor with SSE2, and other compatible processors that include SSE2 and SSE such as AMD* processors.</p> <p><b>N</b> – Generate SSE2 and SSE instructions for Intel processors and optimize for the Intel Pentium 4 and Intel Xeon processor with SSE2. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>B</b> – Generate SSE2 and SSE instructions for Intel processors and optimize for the Intel Pentium M processor. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>K</b> – Generate SSE instructions for Intel processors and optimize for the Intel® Pentium® III processor, the Intel® Pentium® III Xeon® processor and other compatible processors that include SSE such as AMD* processors. Note: For Mac OS*, this option is not supported.</p>
<p><b>/Qx</b> (T P W N B K)</p>	<p><b>-x</b> (T P W N B K)</p>	<p>Processor-specific targeting. Generates specialized code for the indicated processor and enables vectorization. The executable should only be run on the targeted compatible processors.</p> <p><b>T</b> – Generate Supplemental Streaming SIMD Extensions 3 (SSSE3) instructions for Intel Core 2 Duo processor, Intel Core 2 Extreme processor and the Dual-Core Intel Xeon processor.</p> <p><b>P</b> – Generate SSE3, SSE2 and SSE instructions for Intel processors and optimize for Intel Core Duo processor, Intel Core Solo processor, Intel Pentium 4 processor with SSE3, and Intel Xeon processor with SSE3. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>W</b> – Generate SSE2 and SSE instructions and optimize for the Intel Pentium 4 processor, Intel Xeon processor with SSE2, and other compatible processors that include SSE2 and SSE such as AMD* processors.</p> <p><b>N</b> – Generate SSE2 and SSE instructions for Intel processors and optimize for the Intel Pentium 4 and Intel Xeon processor with SSE2. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>B</b> – Generate SSE2 and SSE instructions for Intel processors and optimize for the Intel Pentium M processor. The processor-specific code path may contain features that are not supported on other processors. The generic code path does not contain such features.</p> <p><b>K</b> – Generate SSE instructions for Intel processors and optimize for the Intel Pentium III processor, the Intel Pentium III Xeon processor and other compatible processors that include SSE such as AMD* processors. Note: For Mac OS*, <b>-xP</b> is default and the other options are not supported.</p>
<p><b>/Qvec-report [n]</b></p>	<p><b>-vec-report [n]</b></p>	<p>n = 0: no information n = 1: indicates vectorized loops (default) n = 2: indicates vectorized and non-vectorized loops n = 3: indicates vectorized loops and explains why non-vectorized loops were not vectorized</p>

## IA-64 Processor-Specific Optimization Options

<code>/G2</code>	<code>-mtune=itanium2</code>	Targets optimization for the Intel Itanium 2 processor. Generated code is also compatible with the older IA-64 processor (default).
<code>/G2-p9000</code>	<code>-mtune=itanium2-p9000</code>	Targets optimizations for Dual-Core Intel® Itanium® 2 9000 Sequence processors. Generated code is also compatible with all IA-64 processors, unless the user program calls intrinsic functions specific to the Dual-Core Intel Itanium 2 9000 Sequence processors.
<code>/QIPF-fma[-]</code>	<code>[-no-]IPF-fma</code>	Enables [disables] the combining of floating-point multiply operations and add/subtract operations. (Enabled by default; however, if you specify <code>/fp:strict (-fp-model strict)</code> and do not specifically specify this option, the default is disabled.)
<code>/QIPF-fp-speculation mode</code>	<code>-IPF-fp-speculation mode</code>	Enables floating-point speculations with one of the following modes: <b>fast</b> – Speculate floating-point operations. (default) <b>off</b> – Disables speculation of floating-point operations. <b>safe</b> – Do not speculate if this could expose a floating-point exception. <b>strict</b> – This is the same as specifying off.
<code>/Qivdep-parallel</code>	<code>-ivdep-parallel</code>	Indicates that there is no forward or backward loop-carried memory dependency in the loop where the IVDEP directive is specified. Typically used in conjunction with <code>/Qparallel (-parallel)</code> .
<code>/Qprefetch[-]</code>	<code>[-no-]prefetch</code>	Enables or disables prefetch insertion.
<code>/Qftz[-]</code>	<code>[-no-]ftz</code>	When the main program or dll main is compiled with this option, denormal results are flushed to zero for the whole program (dll). This option is enabled by default at <code>/O3 (-O3)</code> , but not at <code>/O2 (-O2)</code> .

## Fine-Tuning (All Processors)

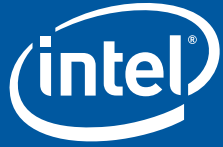
Windows*	Linux* Mac OS*	Comment
<code>/Qunroll[n]</code>	<code>-unroll[n]</code>	Sets the maximum number of times to unroll loops. <code>/Qunroll0 (-unroll0)</code> disables loop unrolling. The default is <code>/Qunroll (-unroll)</code> , which uses default heuristics.
<code>/Qrestrict[-]</code>	<code>-[no]restrict</code>	Enables [disables] pointer disambiguation with the <b>restrict</b> qualifier. Default is OFF, i.e. no pointer disambiguation.
<code>/Oa</code>	<code>-fno-alias</code>	Assumes no aliasing in the program.
<code>/Ow</code>	<code>-fno-fnalias</code>	Assumes no aliasing within functions, but assumes aliasing across calls.
<code>/Qauto-ilp32</code>	<code>-auto-ilp32</code>	Specifies that the application cannot exceed a 32-bit address space, which allows the compiler to use 32-bit pointers whenever possible.  To use this option, you must also specify <code>/Qipo [value] (-ipo[value])</code> . Using the <code>-auto-ilp32</code> option on programs that can exceed 32-bit address space ( $2^{**32}$ ) may cause unpredictable results during program execution. This option has no effect on systems with Intel 64 unless the <code>/QaxP (-axP)</code> , <code>/QxP (-xP)</code> , <code>/QaxT (-axT)</code> , or <code>/QxT (-xT)</code> option is also used.  Note: For IA-32 processors, this option is not supported.
<code>/Qalias-args[-]</code>	<code>-alias-args[-]</code>	Implies arguments may be aliased [are not aliased]. Default is ON, i.e., arguments may be aliased.
<code>/Qopt-report</code>	<code>-opt-report</code>	Generates an optimization report directed to stderr.
<code>/Qopt-report-levellevel</code>	<code>-opt-report-levellevel</code>	Specifies the verbosity level of the output. Valid <b>level</b> settings are <b>min</b> (default), <b>med</b> , and <b>max</b> .
<code>/Qopt-report-phasename</code>	<code>-opt-report-phasename</code>	Reports are generated. The option can be used multiple times in the same compilation to get output from multiple phases.  Valid <b>name</b> arguments are as follows:  <b>all</b> – All possible optimization reports for all phases <b>ipo</b> – Interprocedural Optimizer <b>ipo_inl</b> – Gives only the report on function inlining <b>hlo</b> – High Level Optimizer <b>hlo_prefetch</b> – Gives only the report on compiler-generated prefetching <b>ilo</b> – Intermediate Language Scalar Optimizer <b>ecg</b> – Code Generator (Windows* and Linux* on IA-64 only) <b>ecg_swp</b> – Gives only the report on software pipelining component of the Code Generator (Windows* and Linux* on IA-64 only) <b>pgo</b> – Profile Guided Optimizer
<code>/Qopt-report-routine[rtn]</code>	<code>-opt-report-routine[rtn]</code>	Specifies a routine <b>rtn</b> . Generates reports from all routines or functions with names that include <b>rtn</b> as part of the name.  By default, generates reports for all routines or functions.
<code>/Qopt-report-help</code>	<code>-opt-report-help</code>	Displays all possible settings for <code>/Qopt-report-phase (-opt-report-phase)</code> . No compilation is performed.

## Parallel Performance

<code>/Qopenmp</code>	<code>-openmp</code>	Enables the parallelizer to generate multithreaded code based on OpenMP* directives.
<code>/Qopenmp-report{0 1 2}</code>	<code>-openmp-report{0 1 2}</code>	Controls the OpenMP parallelizer's diagnostic levels. The default is <code>/Qopenmp-report1</code> .
<code>/Qparallel</code>	<code>-parallel</code>	Detects simply structured loops capable of being executed safely in parallel and automatically generates multithreaded code for these loops.
<code>/Qpar-report{0 1 2 3}</code>	<code>-par-report{0 1 2 3}</code>	Controls the auto-parallelizer's diagnostic levels as follows: <b>0</b> – Displays no diagnostic information. <b>1</b> – Indicates loops successfully parallelized (default). <b>2</b> – Loops successfully and unsuccessfully parallelized. <b>3</b> – Adds information about any proven or assumed dependencies inhibiting parallelization.
<code>/Qpar-threshold[n]</code>	<code>-par-threshold[n]</code>	Sets a threshold for the autoparallelization of loops based on the probability of profitable execution of the loop in parallel, <b>n=0</b> to <b>100</b> . Default: <b>n=75</b> .  <b>0</b> – Parallelize loops regardless of computation work volume. <b>100</b> – Parallelize loops only if profitable parallel execution is almost certain.  Must be used in conjunction with <code>/Qparallel (-parallel)</code> .
<code>/Qopt-mem-bandwidth&lt;n&gt;</code> (IA-64 only)	<code>-opt-mem-bandwidth&lt;n&gt;</code> (IA-64 only)	Restricts certain optimizations that may increase memory bandwidth requirements.  <code>/Qopt-mem-bandwidth0 (-opt-mem-bandwidth0)</code> - no restriction (default for serial compilation)  <code>/Qopt-mem-bandwidth1 (-opt-mem-bandwidth1)</code> - restricts optimizations for loops in OpenMP parallel regions (default with <code>/Qparallel (-parallel)</code> or <code>/Qopenmp (-openmp)</code> )  <code>/Qopt-mem-bandwidth2 (-opt-mem-bandwidth2)</code> - restricts optimizations for all loops. May be useful for MPI or other parallel applications.  Note: For Mac OS*, this option is not supported.

## Recommended Optimization Options for Intel® Processors

For Best Performance on Processor	Windows*	Mac OS*	Linux*
Intel® Core™2 Duo processor Intel® Core™2 Extreme processor Dual-Core Intel® Xeon® 5100 series processor	/QxT /QaxT		-xT -axT
Other Intel® processors supporting Intel Extended Memory 64 Technology (Intel 64 ) such as: Intel® Core™ Duo, Intel® Core Solo processor Intel® Celeron® processor (only on processors that support SSE3) <sup>1</sup> Intel Celeron D processor Intel Celeron M processor Intel® Pentium® 4 processor with Streaming SIMD Extension 3 (SSE3) instruction support Intel® Pentium® D processor Intel® Xeon® processor (only on processors that support SSE3) <sup>1</sup>	/QxP /QaxP <sup>1</sup>		-xP -axP <sup>1</sup>
Intel® Core Duo processor	/QxP /QaxP	-xP -axP	-xP -axP
Non-Intel processor-based systems such as AMD Athlon 64 and AMD Opteron processors supporting SSE and SSE2	/QxW /QaxW		-xW -axW
Intel Pentium 4 processor	/QxN /QaxN		-xN -axN
Intel Pentium M processor	/QxB /QaxB		-xB -axB
Intel Pentium III processors and non-Intel x86 processor-based systems such as AMD Athlon processors supporting SSE.	/QxK /QaxK		-xK -axK
Intel Itanium 2 processor	/G2		-mtune= itanium2
Dual-core Intel Itanium 2 9000 sequence processors	/G2-p9000		-mtune= itanium2- p9000



For product and purchase information, visit the  
Intel® Software Development Products site at:  
[www.intel.com/software/products/compilers](http://www.intel.com/software/products/compilers)

<sup>1</sup> Hyper-Threading Technology requires a computer system with an Intel® Pentium® 4 processor supporting HT Technology and a Hyper-Threading Technology enabled chipset, BIOS, and operating system. Performance will vary depending on the specific hardware and software you use. See <http://www.intel.com/info/hyperthreading> for more information including details on which processors support HT Technology.

Intel, the Intel logo, Itanium, Pentium, Intel Centrino, Intel Xeon, Intel XScale, 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 © 2004-2006, Intel Corporation. All Rights Reserved.

