

インテル® C++ および Fortran コンパイラー 19.1 最適化クイック・リファレンス・ガイド

IA-32 プロセッサ、インテル® 64 プロセッサ、インテル® Xeon Phi™ プロセッサおよび互換プロセッサ（非インテル製プロセッサ）向け。

目次

アプリケーション・パフォーマンス	2
一般的な最適化オプションとレポート**	3
並列パフォーマンス**	4
推奨するプロセッサ固有の最適化オプション**	5
インテル® Xeon Phi™ x200 製品ファミリー向けの最適化	6
プロシージャ間の最適化 (IPO) オプションとプロファイルに基づく最適化 (PGO) オプション	7
きめ細かなチューニング (すべてのプロセッサ)**	8
浮動小数点演算オプション	10
プロセッサ開発コード名と対応する拡張命令セット名のシノニム	11
コンパイラー・オプションでよく使用されるプロセッサ名	11
最新の LLVM フレームワーク・ベースのインテル® C++ コンパイラー	12
‡ 最適化に関する注意事項	12

製品と購入情報については、インテル® ソフトウェア開発ツールのサイトをご覧ください:

https://www.xlsoft.com/jp/products/category_intel.html

§ インテル® Xeon Phi™ プロセッサは、インテル® Parallel Studio XE に含まれるコンパイラーではサポートされますが、インテル® System Studio に含まれるコンパイラーではサポートされません。

** これらのオプションは、インテル® マイクロプロセッサおよびインテル製以外のマイクロプロセッサの両方で利用可能ですが、インテル® マイクロプロセッサ向けのほうが、より多くの最適化が適用される場合があります。‡

アプリケーション・パフォーマンス

インテル® コンパイラーでアプリケーションをチューニングする手順

パフォーマンス・チューニングを開始する前に、/Od (-O0) を使用して最適化を行わずにアプリケーションをビルドし、正常に動作することを確認してください。

1. 一般的な最適化オプション (Windows* では /O1、/O2、/O3。Linux* および macOS* では -O1、-O2、-O3) を使用してパフォーマンスを測定し、アプリケーションにとって最適なオプションを判断します。通常は、最初に /O2 (-O2) (デフォルト) を試してから、より高度な最適化を行うと効果的です。次に、ループを多用するアプリケーションに対しては /O3 (-O3) を試します。**
2. /Qx (-x) や /arch (-m) のようなプロセッサ専用のオプションを使用してきめ細かな最適化を行います。例えば、インテル® Xeon® スケラブル・プロセッサ向けには /QxCORE-AVX512 (-xcore-avx512) オプション、インテル® ストリーミング SIMD 拡張命令 3 (インテル® SSE3) 以上の命令セットをサポートするインテル製以外の互換プロセッサ向けには /arch:SSE3 (-msse3) オプションがあります。また、/QxHOST (-xhost) オプションを指定し、コンパイルするホストマシンに搭載されるプロセッサで利用可能な最上位の命令セットを使用して最適化することもできます。**
3. プロシージャー間の最適化 (IPO) を行う /Qipo (-ipo) やプロファイルに基づく最適化 (PGO) を行う /Qprof-gen と /Qprof-use (-prof-gen と -prof-use) オプションを追加して、パフォーマンスを測定し、これらの最適化がアプリケーションにとって効果的かどうかを確認します。
4. インテル® Advisor とインテル® VTune™ Amplifier†† を使用して、シリアルあるいは並列処理におけるパフォーマンスの「hotspot」を識別し、アプリケーション・コードでさらにチューニングが必要な部分を特定することができます。コンパイラーの最適化レポート /Qopt-report (-qopt-report) オプションは、個々の最適化の可能性を特定するのに支援します。
5. マルチスレッド、マルチコア、そしてマルチプロセッサ・システム上で並列実行を最適化するため、自動並列化オプション /Qparallel (-parallel)、/Qopenmp (-qopenmp) オプションによる OpenMP* プラグマやディレクティブ、または、製品に含まれるインテル® パフォーマンス・ライブラリーを使用します。
**インテル® Inspector を使用して、メモリーとスレッドのエラーを診断し、開発工程を早めてマルチスレッド・アプリケーションの市場投入までの時間を短縮します。
6. また、OpenMP* の SIMD 機能を使用して、明示的なベクトル・プログラミングを行い、アプリケーションを最適化します。追加した OpenMP* ディレクティブは、-O2 以上で自動的に認識されます。
7. インテル® Xeon Phi™ x200 製品ファミリーでは、ステップ 4 の結果に基づいてデータ・プリフェッチを追加することを検討してください。

詳細については、製品ドキュメントのページをご覧ください：

<https://software.intel.com/intel-software-technical-documentation> (英語)。

** これらのオプションは、インテル® マイクロプロセッサおよびインテル製以外のマイクロプロセッサの両方で利用可能ですが、インテル® マイクロプロセッサ向けのほうが、より多くの最適化が適用される場合があります。†

†† 製品の一部の機能は、非インテル製マイクロプロセッサ上では利用できません。

一般的な最適化オプションとレポート**

Windows*	Linux* macOS*	説明
/Od	-O0	最適化なし。アプリケーション開発の初期段階およびデバッグ時に使用します。
/Os /O1	-Os -O1	コードサイズの最小化。オブジェクト・サイズを増加させる最適化を無効にします。多くの場合に、最も小さな最適化されたコードを生成します。このオプションは、大きなコードサイズに起因するメモリーページングが問題となる大規模なサーバー/データベース・アプリケーションで有効です。
/O2	-O2	実行速度の最大化。デフォルト設定。ベクトル化とイントラファイルのプロシージャー間の最適化を含む多くの最適化を有効にします。多くの場合に、/O1 (-O1) よりも高速なコードを生成します。
/O3	-O3	/O2 (-O2) で提供される最適化に加え、キャッシュとデータ・プリフェッチをより効率良く使用するため、スカラー置換、ループアンロール、分岐を排除するコードの複製、ループ・ブロッキングなど、より積極的なループとメモリーの最適化を有効にします。 /O3 (-O3) オプションは、浮動小数点演算を多用するループや大きなデータセットを処理するループを含むアプリケーションに推奨します。これらの積極的な最適化は、アプリケーションの種類により /O2 (-O2) と比べて遅くなることがあります。
/Qopt-report[: <i>n</i>]	-qopt-report[= <i>n</i>]	最適化レポートを生成します。デフォルトでは、レポートは <i>.optprt</i> 拡張子を持つファイルに出力されます。 <i>n</i> には、0 (レポートなし) から 5 (最も詳しい) の詳細レベルを指定します。デフォルトは 2 です。
/Qopt-report-file: <i>name</i>	-qopt-report-file= <i>name</i>	最適化レポートを <i>stderr</i> 、 <i>stdout</i> 、またはファイル <i>name</i> に出力します。
/Qopt-report-phase: <i>name1</i> , <i>name2</i> , ...	-qopt-report-phase= <i>name1</i> , <i>name2</i> , ...	最適化フェーズ <i>name1</i> 、 <i>name2</i> 固有の最適化レポートを生成します。フェーズには、以下のキーワードを指定できます。 <i>all</i> - すべてのフェーズのすべての最適化レポート (デフォルト) <i>loop</i> - ループの入れ子とメモリーの最適化 <i>vec</i> - 自動ベクトル化と明示的なベクトル・プログラミング <i>par</i> - 自動並列化 <i>openmp</i> - OpenMP* によるスレッド化 <i>cg</i> - コード生成 <i>ipo</i> - インライン展開を含むプロシージャー間の最適化 <i>pgo</i> - プロファイルに基づく最適化
/Qopt-report-routine: <i>substring</i>	-qopt-report-routine= <i>substring</i>	<i>substring</i> (部分文字列) を含む関数やサブルーチンのみをレポートします。デフォルトでは、すべての関数とサブルーチンがレポートされます。
/Qopt-report-filter: " <i>string</i> "	-qopt-report-filter=" <i>string</i> "	" <i>string</i> " (文字列) で指定したファイル、関数、サブルーチン、行番号の範囲のみレポートします。 例: "myfile, myfun, line1-line2"
/Qopt-report-annotate[: <i>fmt</i>]	-qopt-report-annotate[= <i>fmt</i>]	最適化情報でソースリストにアノテーションを追加します (デフォルトはオフ)。 <i>fmt</i> には、 <i>text</i> (デフォルト) または <i>html</i> を指定できます。

** これらのオプションは、インテル® マイクロプロセッサおよびインテル製以外のマイクロプロセッサの両方で利用可能ですが、インテル® マイクロプロセッサ向けのほうが、より多くの最適化が適用される場合があります。‡

並列パフォーマンス**

Windows*	Linux* macOS*	説明
/Qopenmp	-qopenmp	OpenMP* プラグマ/ディレクティブが記述されている場合にマルチスレッド化されたコードが生成されます。Fortran では、ローカル配列が自動になり、スタックサイズの増加が必要なことがあります。
/Qopenmp-simd	-qopenmp-simd	OpenMP* SIMD ディレクティブが記述されている場合に SIMD コードが生成されます。 O2 以上のレベルで有効になります。
/Qopenmp-stubs	-qopenmp-stubs	OpenMP* ディレクティブを無視し、OpenMP* ランタイム・ライブラリー関数への参照をシングルスレッド処理と仮定してスタブ (ダミー) 関数にリンクします。
/Qparallel	-parallel	自動並列化は、安全に並列実行できる構造のループ (DO CONCURRENT 構文を含む) を検出し、ループのマルチスレッド・コードを自動生成します。
/Qopt-matmul [-]	-q[no-]opt-matmul	入れ子の行列乗算ループの特定およびループとコンパイラーが生成する matmul ライブラリー呼び出しの置換を有効 [無効] にします。 /O3 (-O3) と /Qparallel (-parallel) を指定すると、デフォルトで有効になります。このオプションは、 /O2 (-O2) 以上を指定しない限り、効果はありません。
/Qcoarray [: <i>kywd</i>]	-coarray [= <i>kywd</i>]	Fortran 2008 の Co-Array 機能を有効にします (Fortran のみ)。 <i>kywd</i> には、 shared 、 distributed 、 coprocessor 、 single オプションを指定できます。詳細は、インテル® コンパイラーの『 デベロッパー・ガイドおよびリファレンス 』をご覧ください。
/Qcoarray-num-images: <i>n</i>	-coarray-num-images= <i>n</i>	<i>n</i> には、Co-Array 実行ファイルを実行するイメージ数を指定します。デフォルトは オフ です (イメージ数は実行時に決定されます)。(Fortran のみ)
/Qcoarray-config-file: <i>filename</i>	-coarray-config-file= <i>filename</i>	<i>filename</i> には、MPI 構成ファイル (パスを含む) を指定します。デフォルトは オフ で、MPI のデフォルト設定が使用されます。(Fortran のみ)
/Qmkl: <i>name</i>	-mkl= <i>name</i>	インテル® マス・カーネル・ライブラリー (インテル® MKL) をリンクします。デフォルトは オフ です。 <i>name</i> に指定可能な値は以下のとおりです。 parallel – スレッド化されたインテル® MKL をリンク (デフォルト) sequential – スレッド化されていないインテル® MKL をリンク cluster – MPI が実装されたシーケンシャルなインテル® MKL をリンク (クラスター・ライブラリーは macOS* では利用できません)
/Qtbb	-tbb	インテル® スレディング・ビルディング・ブロック (インテル® TBB) をリンクします。(C++ のみ)
/Qdaal[: <i>lib</i>]	-daal[= <i>lib</i>]	インテル® データ・アナリティクス・アクセラレーション・ライブラリー (インテル® DAAL) をリンクします。 parallel (<i>lib</i> のデフォルト値) は、スレッド・ライブラリーにリンクします。 sequential は、スレッド化されていないライブラリーにリンクします。(C++ のみ)

** これらのオプションは、インテル® マイクロプロセッサおよびインテル製以外のマイクロプロセッサの両方で利用可能ですが、インテル® マイクロプロセッサ向けのほうが、より多くの最適化が適用される場合があります。‡

推奨するプロセッサ固有の最適化オプション**

Windows*	Linux* macOS*	説明
<code>/Qxtarget</code>	<code>-xtarget</code>	<code>target</code> で指定した命令セットをサポートするインテル® プロセッサ向けの専用コードを生成します。実行ファイルは、非インテル製プロセッサや、下位の命令セットをサポートするインテル製プロセッサ上では動作しません。 <code>target</code> に設定可能な値は、「 コンパイラ・オプションでよく使用されるプロセッサ名 」をご覧ください。 注: このオプションは、 <code>/arch</code> または <code>-m</code> オプションで有効にならない最適化を有効にします。64 ビット macOS* では、 SSE3 と SSE2 はサポートされません。
<code>/arch:target</code>	<code>-mtarget</code>	<code>target</code> で指定した命令セットをサポートするすべてのインテル® プロセッサまたは、非インテル製互換プロセッサ向けの専用コードを生成します。 指定した命令セットをサポートしていないインテル製プロセッサや、非インテル製互換プロセッサで実行可能ファイルを実行すると、ランタイムエラーが発生する場合があります。 <code>target</code> に設定可能な値は、「 コンパイラ・オプションでよく使用されるプロセッサ名 」をご覧ください。 注: ターゲット値として ia32 オプションを指定すると、専用ではない、x86/x87 汎用コードが生成されます。これは、IA-32 アーキテクチャベースのターゲットでのみサポートされます。macOS* ではサポートされません。
<code>/QxHOST</code>	<code>-xhost</code>	コンパイルを行うホストシステム上でサポートされる最上位の命令セットを利用するコードを生成します。インテル製プロセッサ上では /Qx (-x) オプションに相当します。非インテル製互換プロセッサ上では /arch (-m) オプションに相当します。
<code>/Qaxtarget</code>	<code>-axtarget</code>	<code>target</code> で指定した命令セットをサポートするインテル® プロセッサ向けの専用コードとデフォルトコード (SSE2) を生成します。 <code>target</code> に設定可能な値は、「 コンパイラ・オプションでよく使用されるプロセッサ名 」をご覧ください。カンマで区切った複数の値を指定することで、同じ実行ファイルにほかのインテル製プロセッサ向けにチューニングされたコードを含めることができます。例: /QaxAVX, SSE4.2 。 デフォルトのコードパスは、 SSE2 をサポートするすべてのインテル製または非インテル製互換プロセッサ上で実行できますが、 /Qx (-x) や /arch (-m) オプションを使用して変更することができます。 例えば、第 4 世代インテル® Core™ プロセッサ・ファミリー向けに最適化されたコードパスと、 SSE3 をサポートするインテル製プロセッサまたは非インテル製互換プロセッサ向けに最適化されたデフォルトのコードパスを生成するには、 /QaxCORE-AVX2 /arch:SSE3 (Linux* では -axcore-avx2 -msse3) を使用します。 アプリケーションは実行時にインテル® プロセッサで実行されているか自動検出し、最適なコードパスを選択します。インテル製プロセッサが検出されなかった場合、デフォルトのコードパスが選択されます。
<code>/Qvecabi: cmdtarget</code>	<code>-vecabi= cmdtarget</code>	上記の /Qx または /Qax (-x または -ax) オプションで指定されたターゲット向けに SIMD 関数のベクトルバージョンを作成します。
<code>/Qopt-zmm- usage:high</code>	<code>-qopt-zmm- usage=high</code>	/QxCORE-AVX512 (Windows*) または -xcore-avx512 (Linux* または macOS*) とともに使用すると、512 ビット SIMD 命令を積極的に生成します。

オンライン記事「[Intel® SSE および Intel® AVX 世代向けの Intel® コンパイラー・オプションとプロセッサー固有の最適化](#)」で、プロセッサー固有の最適化オプションに関する最新の推奨事項をご覧ください。

Intel® コンパイラーの『[デベロッパー・ガイドおよびリファレンス](#)』は、<https://software.intel.com/en-us/cpp-compiler-developer-guide-and-reference> (C++, 英語) および <https://software.intel.com/en-us/fortran-compiler-developer-guide-and-reference> (Fortran, 英語) から入手できます。

** これらのオプションは、Intel® マイクロプロセッサーおよび Intel 製以外のマイクロプロセッサーの両方で利用可能ですが、Intel® マイクロプロセッサー向けのほうが、より多くの最適化が適用される場合があります。†

Intel® Xeon Phi™ x200 製品ファミリー向けの最適化

Windows*	Linux*	説明
<code>/Qopt-threads-per-core:n</code>	<code>-qopt-threads-per-core=n</code>	物理コアあたり n スレッドで最適化するようにコンパイラーに指示します。 $n=1, 2, 3$ または 4 。
<code>/Qopt-prefetch:n</code>	<code>-qopt-prefetch=n</code>	ソフトウェア・プリフェッチのレベルを $n=0$ から 5 で指定します。
<code>/Qopt-prefetch-distance=n1[,n2]</code>	<code>-qopt-prefetch-distance=n1[,n2]</code>	データ・プリフェッチ前のベクトル化されるループ反復の数を指定します。 $n1$ は L2 キャッシュ、 $n2 (\leq n1)$ は L1 キャッシュ向けです。 デフォルトはオフです (コンパイラーが選択します)。
<code>/Qimf-domain-exclusion:n</code>	<code>-fimf-domain-exclusion=n</code>	数学関数が IEEE 標準に準拠する必要がない、特殊なケースの引数を指定します。 n のビットは以下に相当します。 0 - 極値 (非常に大きい、非常に小さい、特異値に近いなど) 1 - NaN 2 - 無限大 3 - デノーマル値 4 - ゼロ
<code>/align:array64byte</code>	<code>-align array64byte</code>	アライメントされたロードとベクトル化を支援するため、64 で割り切れるメモリアドレスに配列の先頭を配置するように指示します。(Fortran のみ)
<code>/Qopt-assume-safe-padding</code>	<code>-qopt-assume-safe-padding</code>	ユーザープログラムと同様に、コンパイラーが配列や動的に割り当てられたオブジェクトの終端から最大 64 バイトを超えて安全にアクセスできることを示します。 パディングの追加はユーザーの責任です。デフォルトはオフです。

詳細は、<https://software.intel.com/en-us/articles/intel-xeon-phi-coprocessor-code-named-knights-landing-application-readiness> (英語)、<https://www.isus.jp/products/psxe/low-precision-optimizations/>、および Intel® コンパイラーの『[デベロッパー・ガイドおよびリファレンス](#)』 (<https://software.intel.com/en-us/cpp-compiler-developer-guide-and-reference> (C++, 英語) および <https://software.intel.com/en-us/fortran-compiler-developer-guide-and-reference> (Fortran, 英語)) をご覧ください。

§ Intel® Xeon Phi™ プロセッサーは、Intel® Parallel Studio XE に含まれるコンパイラーではサポートされますが、Intel® System Studio に含まれるコンパイラーではサポートされません。

プロシージャー間の最適化 (IPO) オプションとプロファイルに基づく最適化 (PGO) オプション

Windows*	Linux* macOS*	説明
/Qip	-ip	現在のソースファイルを対象にしたインライン展開を含む、単一ファイルのプロシージャー間の最適化を行います。
/Qipo[<i>n</i>]	-ipo[<i>n</i>]	複数のソースファイルにまたがるインライン展開やその他のプロシージャー間の最適化を許可します。オプションの引数 <i>n</i> は、作成されるオブジェクト・ファイルの最大数を制御します。デフォルトは、 <i>n=0</i> です (コンパイラーが判断します)。 注: 状況によりコンパイル時間とコードサイズが大幅に増えることがあります。
/Qipo-jobs[<i>n</i>]	-ipo-jobs[<i>n</i>]	プロシージャー間の最適化 (IPO) のリンクフェーズで、同時に実行するコマンド (ジョブ) の数 <i>n</i> を指定します。デフォルトは <i>1</i> です。
/Ob2	-finline-functions または -inline-level=2	コンパイラーの判断により、現在のソースファイル内で関数のインライン展開を有効にします。 <i>/O2</i> と <i>/O3</i> (<i>-O2</i> と <i>-O3</i>) ではデフォルトで有効になります。 注: ファイルサイズが大きいか、コンパイル時間とコードサイズが大幅に増えることがあります。 <i>/Ob0</i> (Linux* と macOS* では <i>-fno-inline-functions</i>) で無効にできません。
/Qinline-factor: <i>n</i>	-inline-factor= <i>n</i>	インライン展開できる関数の合計サイズと最大サイズの係数を設定します。デフォルトは <i>n=100</i> で、100% またはスケーリング係数 <i>1</i> を示します。
/Qprof-gen[: <i>kywd</i>]	-prof-gen[= <i>kywd</i>]	プロファイル情報を生成するためインストルメントを行います。 <i>kywd=threadsafe</i> は、スレッド化されたアプリケーションのプロファイル生成を可能にします。 <i>kywd=srcpos</i> と <i>globdata</i> は、関数やデータの並び替えに役立つ追加情報を収集します。
/Qprof-use	-prof-use	最適化でプロファイル情報を使用するようにします。
/Qprof-dir <i>dir</i>	-prof-dir <i>dir</i>	プロファイル結果の出力ファイル (*.dyn および *.dpi) を格納するディレクトリーを指定します。(あるいは、 PROF_DIR 環境変数を使用してディレクトリーを指定します。)
/Qprof-gen-sampling	-prof-gen-sampling	インテル® VTune™ Amplifier を使用したハードウェア・イベント・ベースのプロファイル生成で使用する追加のデバッグ情報を生成します。
/Qprof-use-sampling: <i>file</i>	-prof-use-sampling= <i>file</i>	最適化で <i>file</i> のハードウェア・イベント・ベースのプロファイル情報を使用するようにします。

きめ細かなチューニング (すべてのプロセッサ)**

Windows*	Linux* macOS*	説明
/Qunroll[<i>n</i>]	-unroll[<i>n</i>]	ループアンロール回数の上限を設定します。 n=0 はループアンロールを無効にします。デフォルトは、デフォルトのヒューリスティックを使用する /Qunroll (-unroll) です。
/Qopt-prefetch: <i>n</i>	-qopt-prefetch= <i>n</i>	さまざまなレベルのソフトウェア・プリフェッチを有効にします。 n=0 (デフォルト、プリフェッチなし) から n=5 (最大限のプリフェッチ) で指定します。警告: 過度のプリフェッチは、リソースの競合によりパフォーマンスが低下することがあります。
/Qopt-prefetch-issue-excl-hint	-qopt-prefetch-issue-excl-hint	prefetchw 命令をサポートするプロセッサで過度のプリフェッチ (書き込みの予測) の生成を有効にします。
/Qopt-prefetch-distance:n1[,n2]	-qopt-prefetch-distance=n1[,n2]	データ・プリフェッチ前のベクトル化されるループ反復の数を指定します。 n1 は L2 キャッシュ、n2 (≤n1) は L1 キャッシュ向けです。 デフォルトは オフ です (コンパイラーが選択します)。
/Qopt-block-factor: <i>n</i>	-qopt-block-factor= <i>n</i>	デフォルトのヒューリスティックをオーバーライドするブロッキング係数 n (ブロック内のループ反復回数) を指定します。ループ・ブロッキングは、 /O3 (-O3) で有効になり、キャッシュのデータを再利用するように設計されています。
/Qopt-streaming-stores: <i>mode</i>	-qopt-streaming-stores <i>mode</i>	always - アプリケーションのメモリー再利用が少ないと想定し、キャッシュをバイパスするストリーミング・ストアの生成を促進します。 never - ストリーミング・ストアの生成を無効にします。 auto - デフォルトのコンパイラーのヒューリスティックを使用します。
/Qrestrict[-]	-[no]restrict	restrict キーワードによるポインタの一義化を有効 [無効] にします。 デフォルトは オフ です。(C/C++ のみ)
/Oa	-fno-alias	プログラムにエイリアシングがないことを想定します。デフォルトは オフ です。
/Ow	-fno-fnalias	関数内にエイリアシングがないことを想定します。デフォルトは オフ です。
/Qalias-args[-]	-fargument-[no]alias	関数の引数のエイリアス化を有効 [無効] にします。デフォルトは オン です。(C/C++ のみ) -fargument-noalias は、配列引数を持つ関数呼び出しを含む C または C++ のループのベクトル化に役立ちます。
/Qansi-alias[-]	-[no]-ansi-alias	ANSI、ISO C、および Fortran 標準の別名規則を有効 [無効] にします。 デフォルト: Windows* では無効、Linux* と macOS* では有効。
/Qopt-class-analysis[-]	-q[no]-opt-class-analysis	C++ クラス階層情報を使用して、コンパイル時に C++ 仮想関数の呼び出しを解決します。C++ アプリケーションに標準的ではない C++ 構造 (ポインタのダウンキャストリングなど) が含まれている場合、動作が異なることがあります。デフォルトでは オフ ですが、/Qipo (Windows*) や -ipo (Linux* と macOS*) オプションを指定すると、C++ の最適化を高めるため オン になります。(C++ のみ)
/Qvec-threshold:0	-vec-threshold=0	パフォーマンスの向上が見込めない場合でもループを自動ベクトル化するようにコンパイラーに指示します。
/Qvec[-]	-[no]-vec	ベクトル化を有効 [無効] にします。デフォルトは、 /O2 (-O2) で有効です。

Windows*	Linux* macOS*	説明
/Qstringop- strategy:alg	-mstringop- strategy=alg	memcpyや memsetのようなバッファ操作関数で使用するアルゴリズム algを設定します。 libcall: ライブラリー呼び出しを行うようにコンパイラーに指示します。 rep. rep movs または同様のシーケンスを使用してインライン展開します。 const_size_loop (デフォルト): コンパイル時にサイズが判明して [/Q -m] stringop-inline-threshold 未満の場合、ループにインライン展開します。
/align: arraynnbyte	-align arraynnbyte	アライメントされたロードとベクトル化を支援するため、nn で割り切れるメモリアドレスに配列の先頭を配置するように指示します。(Fortran のみ)
/assume:[no] buffered_io	-assume [no] buffered_io	I/O 効率を向上するため、連続するシーケンシャルな読み書きデータをバッファリングします。デフォルトでは、バッファリングしません。(Fortran のみ)
/assume: contiguous_ assumed_shape	-assume contiguous_ assumed_shape	形状引き継ぎ仮引数が常に連続している (ユニットストライドできる) と仮定します。(Fortran のみ)
/assume: contiguous_ pointer	-assume contiguous_ pointer	ポインター仮引数が常に連続している (ユニットストライドできる) と仮定します。(Fortran のみ)
なし	-f[no-] exceptions	C++ の場合、-fexceptions がデフォルトで、例外処理テーブルの生成を有効にします。ベクトル化が適切に行われなかったことがあります。-fno-exceptions は例外指定を解析しますが無視されます。呼び出しに -fno-exceptions でコンパイルされた関数が含まれている場合、try ブロックや throw 文はエラーになります。(Linux* のみ)
/Qfnsplit:n	-fnsplit=n	到達する確率が n% 未満の条件付きコードブロックを異なるコードセグメントに配置します。(Linux* および Windows* のみ)
/Qimf-domain- exclusion:n	-fimf-domain- exclusion=n	数学関数が IEEE 標準に準拠する必要がない、特殊なケースの引数を指定します。 n のビットは以下に相当します。 0 – 極値 (非常に大きい、非常に小さい、特異値に近いなど) 1 – NaN 2 – 無限大 3 – デノーマル値 4 – ゼロ
/align: array64byte	-align array64byte	アライメントされたロードとベクトル化を支援するため、64 で割り切れるメモリアドレスに配列の先頭を配置するように指示します。(Fortran のみ)
/Qopt-assume- safe-padding	-qopt-assume- safe-padding	ユーザープログラムと同様に、コンパイラーが配列や動的に割り当てられたオブジェクトの終端から最大 64 バイトを超えて安全にアクセスできることを示します。パディングの追加はユーザーの責任です。デフォルトはオフです。

** これらのオプションは、インテル® マイクロプロセッサおよびインテル製以外のマイクロプロセッサの両方で利用可能ですが、インテル® マイクロプロセッサ向けのほうが、より多くの最適化が適用される場合があります。‡

浮動小数点演算オプション

Windows*	Linux* macOS*	説明
<code>/fp:name</code>	<code>-fp-model name</code>	特定の最適化を制限することで、浮動小数点演算のパフォーマンス、精度、再現性を制御します。 <i>name</i> に指定可能な値は以下のとおりです。 fast[=1 =2] – 精度と一貫性を多少低くすることにより、さらに強力な最適化が可能になります。(デフォルトは fast=1) consistent – 異なる最適化レベルや同じアーキテクチャーの異なるプロセッサで、一貫した再現性のある結果を有効にします。 precise – FMA (Fused Multiply Add) 命令を生成する場合を除いて、異なる浮動小数点結果になるコンパイラーの最適化を無効にします。 except – 浮動小数点例外セマンティクスを強制します。 strict-precise と except オプションを有効にし、デフォルトの浮動小数点環境を仮定しません。コンパイラーが、FMA (Fused Multiply Add) 命令を生成しないように強制します。
<code>/Qopt-dynamic-align[-]</code>	<code>-q[no-]opt-dynamic-align</code>	実行時のデータ・アライメントに依存し、同じシリアル・アプリケーションを同じ入力データで繰り返し実行した際に、わずかな浮動小数点結果の違いを引き起こす可能性のある最適化を有効 [無効] にします。 /fp:precise または /fp:consistent (-fp-model precise または -fp-model consistent) が指定されない限り、デフォルトで有効です。
<code>/Qftz[-]</code>	<code>-[no-]ftz</code>	main プログラムや dll の main をこのオプションでコンパイルすると、プログラム (dll) 全体で実行時に Intel® SSE や Intel® AVX 命令によるデノーマル結果がゼロにフラッシュされます。 /Od (-O0) が指定されない限り、デフォルトで有効です。
<code>/Qimf-precision:name</code>	<code>-fimf-precision:name</code>	算術ライブラリー関数の精度を設定します。デフォルトはオフです (コンパイラーは、デフォルトのヒューリスティックを使用します)。 <i>name</i> には、 high 、 medium 、 low を指定できます。精度を下げると、特にベクトル化されたコードのパフォーマンスが向上する可能性があります。その逆もあり得ます。
<code>/Qimf-arch-consistency:true</code>	<code>-fimf-arch-consistency=true</code>	算術ライブラリー関数が、同じアーキテクチャーの異なる Intel 製プロセッサまたは非 Intel 製互換プロセッサにおいて一貫した結果を生成するようにします。これにより、ランタイム・パフォーマンスが低下することがあります。 /fp:consistent (-fp-model consistent) が指定されない限り、デフォルトは false (オフ) です。
<code>/Qprec-div[-]</code>	<code>-[no-]prec-div</code>	浮動小数点除算の精度を上げます [下げます]。パフォーマンスが低下 [向上] することがあります。デフォルトは オフ です。
<code>/Qprec-sqrt[-]</code>	<code>-[no-]prec-sqrt</code>	平方根計算の精度を上げ [下げ] ます。パフォーマンスが低下 [向上] することがあります。
<code>/Qprotect-parens[-]</code>	<code>-f[no-]protect-parens</code>	括弧で指定した順序で式が評価されます。 /fp:precise または /fp:consistent (-fp-model precise または -fp-model consistent) が指定されない限り、デフォルトは オフ です。
<code>/Qfma[-]</code>	<code>-[no]fma</code>	コンパイラーが、FMA (Fused Multiply Add) 命令を生成しないように強制します (ただし、ランタイム・ライブラリーでは FMA 命令が使用される可能性があります)。
<code>/Qimf-use-svml</code>	<code>-fimf-use-svml</code>	Intel の数学ライブラリー (LIBM) の代わりに、SVML (Short Vector Math Library) を使用してスカラー数学関数を実装するようにコンパイラーに指示します。
<code>/Qimf-force-dynamic-target</code>	<code>-fimf-force-dynamic-target</code>	プロセッサの種類に基づいて実行時に算術ライブラリー関数のコードパスを選択します。デフォルトは オフ です。
<code>/Qfp-speculation:safe</code>	<code>-fp-speculation=safe</code>	浮動小数点例外が起こる可能性がある場合は、特定の最適化を無効にするようにコンパイラーに指示します。浮動小数点例外がデバッグでマスク解除されている場合に設定すると便利です。

<https://www.isus.jp/products/c-compilers/consistency-of-floating-point-results/> もご覧ください。

プロセッサ開発コード名と対応する拡張命令セット名のシノニム

https://en.wikipedia.org/wiki/List_of_Intel_CPU_microarchitectures (英語)

インテル® マイクロアーキテクチャーの開発コード名	命令セット
ICELAKE-SERVER	シノニムなし
ICELAKE-CLIENT	シノニムなし
CANNONLAKE	シノニムなし
SKYLAKE-AVX512	CORE-AVX512
KNM	MIC-AVX512
KNL	MIC-AVX512
SKYLAKE	CORE-AVX2
BROADWELL	CORE-AVX2
HASWELL	CORE-AVX2
SILVERMONT	SSE4.2
IVYBRIDGE	CORE-AVX-I
SANDYBRIDGE	AVX

コンパイラ・オプションでよく使用されるプロセッサ名

インテル® マイクロアーキテクチャーの開発コード名または命令セット	<code>-xcode</code> <code>/Qxcode</code>	<code>-axcode</code> <code>/Qaxcode</code>	<code>-arch code</code> <code>/arch:code</code>	<code>-march=code</code> <code>code</code> は小文字	<code>-mtune=code</code> <code>/tune:code</code> <code>code</code> は小文字
ICELAKE-SERVER	✓	✓	✓	✓	✓
ICELAKE-CLIENT	✓	✓	✓	✓	✓
CANNONLAKE	✓	✓	✓	✓	✓
SKYLAKE-AVX512	✓	✓	✓	✓	✓
KNM †	✓	✓	✓	✓	✓
KNL †	✓	✓	✓	✓	✓
SKYLAKE	✓	✓	✓	✓	✓
BROADWELL	✓	✓	✓	✓	✓
HASWELL	✓	✓	✓	✓	✓
SILVERMONT †	✓	✓	✓	✓	✓
IVYBRIDGE	✓	✓	✓	✓	✓
SANDYBRIDGE	✓	✓	✓	✓	✓
COMMON-AVX512 Δ	✓	✓			
CORE-AVX512	✓	✓			
MIC-AVX512	✓	✓			
CORE-AVX2	✓	✓	✓	✓	✓
SSE4.2	✓	✓			
CORE-AVX-I	✓	✓	✓	✓	✓
AVX	✓	✓	✓		
corei7-avx				✓	✓

ATOM_SSE4.2	✓	✓			
SSE4.1	✓	✓	✓		
ATOM_SSSE3	✓	✓			
corei7					✓
atom				✓	✓
SSSE3	✓	✓	✓		
SSE3	✓	✓	✓		
SSE2	✓	✓	✓		
SSE			✓		
IA32			✓		
core2			✓		✓
pentiummx					✓
pentiumpro					✓
pentium-m			✓		
pentium4			✓		
pentium3			✓		
pentium			✓		

† Windows* および Linux* のみ

△ MIC-AVX512 と CORE-AVX512 のサブセット

最新の LLVM フレームワーク・ベースのインテル® C++ コンパイラー

最新の LLVM フレームワーク・ベースのインテル® C++ コンパイラー (ICC NextGen と呼ばれることもあります) は、-qnextgen コンパイラー・オプションを指定して起動します。このオプションは、Windows* または Linux* の icc/icl/icpc でのみ利用できます。このオプションおよび機能は、macOS* の icc では利用できません。

-qnextgen オプションを指定してコンパイルしたときに、このクイック・リファレンス・ガイドに記載されているすべてのコンパイラー・オプションが利用できるとは限りません。移植ガイドおよびその他の情報は、[インテル® C++ コンパイラー次世代コード・ジェネレーター](#)のドキュメント (英語) から入手できます。

製品と購入情報については、インテル® ソフトウェア開発ツールのサイトをご覧ください:

https://www.xlsoft.com/jp/products/category_intel.html

‡ 最適化に関する注意事項

インテル® コンパイラーでは、インテル® マイクロプロセッサに限定されない最適化に関して、他社製マイクロプロセッサ用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサに関して、いかなる最適化の利用、機能、または効果も保証いたしません。

本製品のマイクロプロセッサ依存の最適化は、インテル® マイクロプロセッサでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサ用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。

注意事項の改訂 #20110804

Intel、インテル、Intel ロゴ、Intel Core、Xeon、Intel Xeon Phi、VTune は、アメリカ合衆国および / またはその他の国における Intel Corporation またはその子会社の商標です。

* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

© 2020 Intel Corporation. 無断での引用、転載を禁じます。