

最適化クイック・リファレンス・ガイド

インテル[®] C++、Fortran コンパイラー 17.0 のオプション紹介
およびインテル[®] VTune[™] Amplifier XE 2017 の概要



PARALLEL STUDIO XE

IA-32 プロセッサ、インテル[®] 64 プロセッサ、互換プロセッサ（非インテル製プロセッサ）、
インテル[®] Xeon Phi[™] プロセッサ / コプロセッサ、およびインテル[®] グラフィックス・テクノロジー向け。

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

目次

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.....	2
一般的な最適化オプション**	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
きめ細かなチューニング (すべてのプロセッサ)**	6
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.....	7
浮動小数点演算オプション	8
オフロード向けコンパイル [§]	9
オフロード向け環境変数 [§]	9
インテル® Xeon Phi™ コプロセッサ x100 製品ファミリー向けの最適化 [§]	10
インテル® メニー・インテグレートッド・コア (インテル® MIC) アーキテクチャー向けの最適化.....	10
インテル® VTune™ Amplifier XE を使用したパフォーマンス解析.....	11
デバッグオプション	12
‡ 最適化に関する注意事項	13

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:

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

§ インテル® Xeon Phi™ プロセッサおよびコプロセッサは、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 OS X* 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、第4世代インテル® Core™ プロセッサ・ファミリー向けには **/QxCORE-AVX2 (-xcore-avx2)** 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** および **/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. 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。
5. インテル® Advisor とインテル® VTune™ Amplifier XE^{††} 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を支援します。
6. また、**/Qopenmp-simd (-qopenmp-simd)** による OpenMP* 4.0 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。**
7. インテル® MIC アーキテクチャーでは、ステップ5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)。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』には、インテル® MIC アーキテクチャー向けのアプリケーションのコンパイルに関する節が含まれています。

** 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。†

†† 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。

インテル® コンパイラー『ユーザー・リファレンス・ガイド』日本語版ダウンロード：
無償評価版登録ページ: <https://www.xlsoft.com/jp/products/download/intelj.html>
より、インテル® Parallel Studio XE Cluster Editionをお申し込みいただくことで入手いただけます。

一般的な最適化オプション**

Windows®	Linux*/OS X*	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 [: n]	-qopt-report [n]	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 .opt rpt 拡張子を持つファイルに出力されます。n 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-fi le:name	-qopt-report-fi le=name	最適化レポートを stderr、stdout、またはファイル name に出力します。
/Qopt-report-p hase:name1, n ame2, ...	-qopt-report-p hase=name1, n ame2, ...	最適化フェーズ name1、name2 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。 all - 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) loop - ループの入れ子とメモリーの最適化 vec - 自動ベクトル化と明示的なベクトル・プログラミング par - 自動並列化 openmp - OpenMP* によるスレッド化 cg - コード生成 ipo - インライン展開を含むプロシージャー間の最適化 pgo - プロファイルに基づく最適化 offload - インテル® MIC アーキテクチャーへのデータ/実行のオフロード
/Qopt-report-h elp	-qopt-report-h elp	上記の /Qopt-report-phase (-qopt-report-phase) の name に指定可能なすべてのキーワードを表示します。コンパイルは実行されません。
/Qopt-report-r outine: substri ng	-qopt-report-r outine= substri ng	substring (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-fi lter: "string"	-qopt-report-fi lter="string"	"string" (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-a nnotate:fmt	-qopt-report-a nnotate=fmt	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)。fmt には、text (デフォルト) または html を指定できます。
/Qstd:cval	-std=cval	より新しい C++ または C 言語標準の機能をサポートします。cval には、c++14、c++11、c99 などを指定できます。

** 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*/OS X*	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。
/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。
/Qpar-threshold[:n]	-par-threshold [n]	パフォーマンス向上の可能性に基づいて、ループの自動並列化のしきい値を設定します。n=0 から 100 が指定でき、デフォルトは 100 です。 n=0(計算量にかかわらずループを並列化します) から n=100(デフォルト、パフォーマンス上の利点があると思われる場合にのみループを並列化します)。/Qparallel (-parallel) オプションを併用する必要があります。
/Qpar-affinity: name	-par-affinity= name	OpenMP* や自動並列化アプリケーション向けにスレッドとプロセッサのアフィニティーを指定します。name に指定できる値は、none(デフォルト)、scatter、compact などです。メインプログラムをコンパイルする場合にのみ効果があります。設定と詳細な情報については、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。
/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[:keyword]	-coarray [=keyword]	Fortran 2008 の Co-Array 機能を有効にします (Fortran のみ)。keyword 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:n	-coarray-num-images=n	n 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)。
/Qcoarray-config-file:filename	-coarray-config-file=filename	filename 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。
/Qmkl:name	-mkl=name	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。name 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 OS X* では利用できません)
/Qtbb	-tbb	インテル® スレッディング・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[:lib]	-daal[=lib]	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 (lib 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*/OS X*	説明
<code>/Qxtarget</code>	<code>-xtarget</code>	<p><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。実行可能ファイルは、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>に指定可能な命令セットは次のとおりです (上位から下位)。</p> <p>CORE-AVX512、MIC-AVX512、COMMON-AVX512、CORE-AVX2、AVX、SSE4.2、ATOM_SSE4.2、SSE4.1、ATOM_SSSE3、SSSE3、SSE3、SSE2</p> <p>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 OS X* では、SSE3 と SSE2 はサポートされません。</p> <p>注: MIC-AVX512 (インテル® Xeon Phi™ プロセッサ x200 製品ファミリー向けの命令セット) は CORE-AVX512 のサブセットではありません。COMMON-AVX512 は両方の共通のサブセットです。</p>
<code>/arch: target</code>	<code>-mtarget</code>	<p><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。</p> <p><code>target</code>に指定可能な値: avx、sse4.2、sse4.1、ssse3、sse3、sse2、ia32</p> <p>注: ia32 オプションは、専用ではない、x86/x87 汎用コードを生成します。これは、IA-32 アーキテクチャーでのみサポートされます。OS X* ではサポートされません。</p>
<code>/QxHOST</code>	<code>-xhost</code>	<p>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 <code>/Qx (-x)</code> 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 <code>/arch (-m)</code> オプションに相当します。</p>
<code>/Qaxtarget</code>	<code>-axtarget</code>	<p><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。</p> <p><code>target</code>に指定可能な値: CORE-AVX512、MIC-AVX512、COMMON-AVX512、CORE-AVX2、AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2</p> <p>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: <code>/QaxAVX,SSE4.2</code>。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、<code>/Qx (-x)</code> や <code>/arch (-m)</code> オプションを使用して変更することができます。</p> <p>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、<code>/QaxCORE-AVX2/arch:SSE3</code> (Linux* では <code>-axcore-avx2 -msse3</code>) を使用します。</p> <p>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。</p> <p>注: 64 ビット OS X* では、sse3 と sse2 オプションはサポートされません。</p>

オンライン記事「[インテル® SSE およびインテル® AVX 世代 \(SSE2、SSE3、SSSE3、ATOM_SSSE3、SSE4.1、SSE4.2、ATOM_SSE4.2、AVX、AVX2\) 向けのインテル® コンパイラ・オプションとプロセッサ固有の最適化](#)」で、プロセッサ専用の推奨最適化オプションを参照してください。

** 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。†

きめ細かなチューニング (すべてのプロセッサ)**

Windows®	Linux*/OS X*	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。 <i>n=0</i> 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。 <i>n=0</i> (デフォルト、プリフェッチなし) から <i>n=5</i> (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-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 <i>n</i> (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>	ストリーミング・ストアの生成を有効/無効にします。 <i>mode</i> に指定可能な値は次のとおりです。 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ーの一義化を有効 [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 および ISO C 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 OS X* 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 (ポインター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 OS X*) 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: <i>n</i>	-vec-threshold= <i>n</i>	パフォーマンス向上の可能性に基づいて、ループの自動並列化のしきい値 <i>n</i> を設定します。デフォルトは <i>n=100</i> です。 <i>n=0</i> (計算量にかかわらずループをベクトル化します) から <i>n=100</i> (パフォーマンス上の利点があると思われる場合にのみループをベクトル化します)。
/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。
/align: array <i>nnb</i> yte	-align array <i>nnb</i> yte	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、 <i>nn</i> で割り切れるメモリー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)。
	-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。

** 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。*

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*/OS X*	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、リンク時コンパイルの最大数 (オブジェクト・ファイル数) 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 -finline-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。/O2 と /O3 (-O2 と -O3) 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な場合、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。/Ob0 (Linux* と OS X* では -fno-inline-functions) で無効にできます。
/Qinline-factor: <i>n</i>	-finline-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 1 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。
PROF_DIR	PROF_DIR	プロファイル出力ファイルのディレクトリーを設定します (/Qprof-dir や -prof-dir で使用します)。
/Qprofile-functions	-profile-functions	各関数の実行時間を生成するようにインストルメント関数を追加します。
/Qprofile-loops	-profile-loops	関数をインストルメントし、各ループまたはループの入れ子のシリアルコードのプロファイルを生成します。プロファイルの表示方法やその他の詳細については、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』で、「関数またはループの実行時間のプロファイル」をご覧ください。

浮動小数点演算オプション

Windows®	Linux*/OS X*	説明
/fp:name	-fp-model name	<p>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。</p> <p>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)。</p> <p>precise – FMA (Fused Multiplty 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。</p> <p>double/extended/source – 中間結果をそれぞれ倍精度、拡張精度、ソースの精度で丸めます。オーバーライドされない限り precise も適用されます。インテル® Fortran コンパイラーでは、double と extended はサポートされません。precise、strict、consistent が指定された場合、Intel64 のデフォルトは source です。</p> <p>except – 浮動小数点例外セマンティクスを強制します。</p> <p>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 Multiplty Add) 命令を生成しないように強制します。</p> <p>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。</p>
/Qopt-dynamic-align[-]	-q[no-]opt-dynamic-align	<p>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。</p>
/Qftz[-]	-[no-]ftz	<p>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) 全体で実行時にインテル® SSE やインテル® 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。</p>
/Qimf-precision: name	-fimf-precision: name	<p>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)。name 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。</p>
/Qimf-arch-consistency: true	-fimf-arch-consistency= true	<p>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。</p>
/Qprec-div[-]	-[no-]prec-div	<p>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。</p>
/Qprec-sqrt[-]	-[no-]prec-sqrt	<p>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。</p>
/Qprotect-parens[-]	-f[no-]protect-parens	<p>括弧で指定した順序で式が評価されます。</p>
/Qfma[-]	-[no]fma	<p>コンパイラーが、FMA (Fused Multiplty 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)。</p>

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

オフロード向けコンパイル^S

Windows [®]	Linux [*]	説明
/Qoffload[-] /Qoffload[: <i>kywd</i>]	-q[no-]offload -qoffload= <i>kywd</i>	インテル [®] MIC アーキテクチャーにオフロードする (OpenMP [*] 以外の) 言語構造を制御します。 <i>kywd</i> には次の値を指定できます。 <i>none</i> : オフロード構文は無視され、すべてのコードはホスト上でのみ実行するようにコンパイルされます。 /Qoffload- (-qno-offload) と同じです。 <i>mandatory</i> (デフォルト): status 句が指定されておらずターゲットが利用できない場合、プログラムは失敗しオフロードはスキップされます。 <i>optional</i> : ターゲットが利用できない場合、すべてのコードはホストで実行されます。
/Qopenmp-offload: <i>device</i> /Qopenmp-offload-	-qopenmp-offload= <i>device</i> -qno-openmp-offload	<i>device</i> にオフロードする TARGET のような OpenMP [*] 構造を制御します。 <i>device</i> に指定可能な値は以下のとおりです。 <i>mic</i> (デフォルト): インテル [®] MIC アーキテクチャーにオフロードします。 <i>host</i> : オフロード構文は無視され、すべてのコードはホスト上でのみ実行するようにコンパイルされます。 /Qopenmp-offload- または -qno-openmp-offload と同じです。
/Qoffload-arch: <i>arch</i>	-qoffload-arch= <i>arch</i>	ターゲット・アーキテクチャーを指定します。 <i>arch</i> には次の値を指定できます。 <i>mic</i> : インテル [®] Xeon Phi™ コプロセッサ x100 製品ファミリー (-qoffload-arch が指定されていない場合のデフォルト) <i>mic-avx512</i> : インテル [®] Xeon Phi™ プロセッサ x200 製品ファミリー
/Qoffload-option, <i>target</i> , <i>tool</i> , "option-list"	-qoffload-option, <i>t</i> <i>arget</i> , <i>tool</i> , "option-list"	ターゲット向けのコンパイルにのみ使用するオプションを指定します。インテル [®] MIC アーキテクチャーの場合、 <i>target</i> には <i>mic</i> を指定します。 <i>tool</i> には、 <i>compiler</i> 、 <i>ld</i> 、 <i>link</i> 、 または <i>as</i> を指定できます。
/Qoffload-attribute-target: <i>target-name</i>	-qoffload-attribute-target= <i>target-name</i>	ファイルスコープ関数やデータ・オブジェクトにオフロード属性 <i>target</i> (<i>t</i> <i>arget-name</i>) のフラグを付けます。 <i>target-name</i> には、 <i>mic</i> (インテル [®] MIC アーキテクチャー) を指定できます。
/Qopt-report-phase: <i>offload</i>	-qopt-report-phase= <i>offload</i>	ホストとコプロセッサ間でコピーされる変数のレポートをコンパイル時に生成します。
__INTEL_OFFLOAD	__INTEL_OFFLOAD	ホスト上のオフロード・プログラミングで利用できる事前定義マクロ。

オフロード向け環境変数^S

変数	説明
OFFLOAD_REPORT=< <i>n</i> >	オフロード・アプリケーションのランタイムレポートを生成します。 <i>n</i> =1 ホストとコプロセッサ上の実行時間をレポートします。 <i>n</i> =2 ホストとコプロセッサ間のデータ転送もレポートします。 <i>n</i> =3 デバイスの初期化と個々の変数の転送を含む詳細をレポートします
OFFLOAD_DEVICES=< <i>n1</i> , <i>n2</i> ,...>	ホスト上のプロセスが使用する物理コプロセッサを <i>n1</i> 、 <i>n2</i> などに制限します。コプロセッサの番号は <i>0</i> から始まります。
MIC_STACKSIZE=< <i>n</i> >M	オフロード・アプリケーション向けにコプロセッサ上の最大スタックサイズを設定します。この例では、 <i>n</i> はメガバイトです。
MIC_ENV_PREFIX=< <i>name</i> >	オフロード・アプリケーション向けに、ホストとコプロセッサ上の環境変数を区別するためプリフィックスを指定します。例えば、 <i>name</i> =MIC の場合、 MIC_OMP_NUM_THREADS がコプロセッサ上の OpenMP [*] スレッドの数を制御します。
MIC_USE_2MB_BUFFERS=< <i>n</i> >M	実行時に <i>n</i> MB を越えるオフロードポインター変数は、2MB のラージページに割り当てられます。

§ インテル® Xeon Phi™ プロセッサおよびコプロセッサは、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せんが、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。

インテル® Xeon Phi™ コプロセッサ x100 製品ファミリー向けの最適化[§] (開発コード名 Knights Corner)

Windows®	Linux*	説明
/Qmic	-mmic	インテル® Xeon Phi™ コプロセッサ x100 製品ファミリーでネイティブに実行するアプリケーションをビルドします。(デフォルトはオフです)。
/Qopt-streaming-cache-evict:n	-qopt-streaming-cache-evict=n	ストリーミング・ストアの後にキャッシュ・エビクション (追い出し) 命令を生成するかどうかを制御します。 $n=0$ 追い出しなし、 $n=1$ L1 のみ追い出し、 $n=2$ L2 のみ追い出し、 $n=3$ L1 と L2 を追い出し。
/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。
/Qopt-gather-scatter-unroll:n	-qopt-gather-scatter-unroll=n	集約 (Gather)/分散 (Scatter) ループの代替アンロールシーケンスを指定します。ほとんどのデータ項目が異なるキャッシュラインにある場合、アンロールファクター n の値を大きくするとパフォーマンスが向上します。
__MIC__	__MIC__	インテル® Xeon Phi™ コプロセッサ x100 製品ファミリー向けのプログラミングで利用できる事前定義済みマクロです。

インテル® メニー・インテグレートッド・コア (インテル® MIC) アーキテクチャー向けの最適化

(インテル® Xeon Phi™ コプロセッサ x100 製品ファミリーとインテル® Xeon Phi™ プロセッサ x200 製品ファミリーを含む)

Windows®	Linux*	説明
/Qopt-threads-per-core:n	-qopt-threads-per-core=n	物理コアあたり n スレッド向けに最適化するようにコンパイラーに知らせます。 $n=1, 2, 3$ または 4 。
/Qopt-prefetch:n	-qopt-prefetch=n	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。インテル® Xeon Phi™ コプロセッサ x100 製品ファミリーでは、最適化レベル -O2 以上のデフォルトは $n=3$ です。
/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、 $n2$ ($\leq n1$) 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)。
/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)。

詳細は、<http://www.isus.jp/products/psxe/low-precision-optimizations/> をご覧ください。インテル® MIC アーキテクチャーについては、<http://www.isus.jp/mic-developer/> と <http://www.isus.jp/hpc/xeon-phi/> をご覧ください。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』は、<https://software.intel.com/intel-cplusplus-compiler-17.0-user-and-reference-guide> (C++、英語) と <https://software.intel.com/intel-fortran-compiler-17.0-user-and-reference-guide> (Fortran、英語) にあります。

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:

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

インテル® VTune™ Amplifier XE を使用したパフォーマンス解析

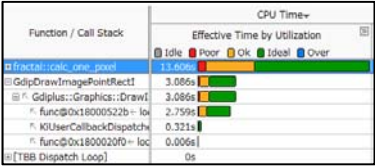
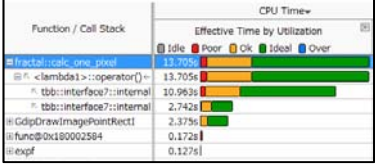
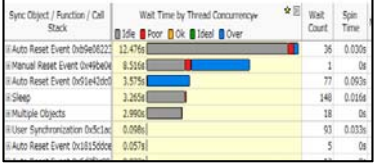
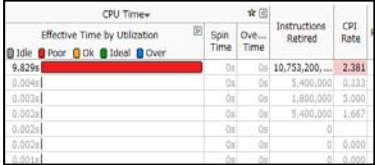
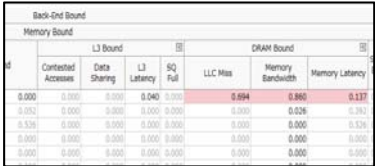
Hotspot: コードの中で CPU を使用してる時間が最も多い関数またはループの箇所です。

ボトルネック: コードの中で全体の最大の実行速度に影響する、最も遅い箇所です。

ボトルネックの原因: ジャンプするメモリー参照や、分岐予測ミスなどの処理が遅延の原因になります。

インテル® VTune™ Amplifier XE を用いてパフォーマンス解析を行うことで、

1. Hotspot を発見し、
2. その箇所がボトルネックであるかを検出し、
3. ボトルネックの原因を特定して修正することで、パフォーマンスを改善させることができます。

主要な解析タイプ	説明	
Basic Hotspots (コマンドライン解析タイプ: hotspots)	Hotspot (CPU 時間を要しているソースコードの箇所) を特定します。 実際の CPU 時間とともに、表の一番上に最も CPU 時間を要している Hotspot が表示されます。	
Concurrency (コマンドライン解析タイプ: concurrency)	プログラムのマルチスレッド動作を解析します。 実時間 (Elapsed Time) の内、プロセッサのすべてのコアを同時に利用できていた時間が長いほど、Ideal (望ましい) と分類されます。	
Locks & Waits (コマンドライン解析タイプ: locksandwaits)	待ち状態による CPU 時間の浪費状況を解析します。 マルチスレッドにおけるクリティカルセクションでの同期や、入出力待ちなどが性能に影響しているかどうかを確認できます。	
Advanced Hotspots (コマンドライン解析タイプ: advanced-hotspots)	プログラムの各場所での CPI 値を求めます。 CPI とは、実行に要した CPU サイクル数と、実行された CPU 命令数の比であり、プログラムの実行効率を表す基本的な評価基準です。	
General Exploration (コマンドライン解析タイプ: general-exploration)	キャッシュミスや分岐予測ミスなど、CPU の動作に関連する詳細な性能情報やボトルネックを取得します。 解析結果において性能劣化の原因と見られるボトルネックを特定します。ボトルネックの原因となる項目や値はピンク色で強調表示されます。	

デバッグオプション

Windows®	Linux*/OS X*	説明
/Zi /debug /debug:full /debug:all	-g -debug -debug full -debug all	最適化されていないコードの完全なシンボリック・デバッグのため、一般的な開発環境のデバッガ向けにデバッグ情報を生成します。最適化オプション /O2 (-O2) (またはほかの O オプション) を指定した場合に有効になります。デバッグシンボルの生成は、一般的にオブジェクト・モジュールのサイズを増加させ、最適化されたコードのパフォーマンスをわずかに低下させることがあります。
/debug:none	-debug none	デバッグ情報は生成されません。(デフォルト)。
/debug:minimal	-debug minimal	ローカルシンボルではなく、デバッグ用の行番号情報を生成します。
/debug:inline-debug-info	-debug inline-debug-info	このオプションを指定すると、インライン展開される関数のシンボルは、呼び出し元ではなく、呼び出される関数のソースに関連付けられます。 -O2 を指定しない限り、/debug:full (-debug full) だけでは有効になりません。
	-debug extended	最適化されたコードのシンボリック・デバッグを改善するため、追加情報を生成します。(Linux* のみ。debug full では有効になりません。)
	-debug parallel	スレッド化されたコードのデバッグのため、追加のシンボルとインストルメント・コードを生成します。(Linux* のみ。debug full では有効になりません。)
/Qsox[-]	-[no-]sox (Linux* のみ)	オブジェクト・ファイル (Windows® と Linux*) と実行可能ファイル (Linux*) に、コンパイラのバージョンとオプションを文字列で埋め込みます。デフォルトはオフです。
/Qtraceback	-traceback	ランタイム時に致命的なエラーが発生したとき、ソースファイルのトレースバック情報を表示できるように、オブジェクト・ファイル内に補足情報を生成します。最適化されたコードで利用します。(Fortran のみ)。
/Qinit:snan, arrays	-init=snan, arrays	実行時に初期化されていない浮動小数点変数を検出しやすくするため、特定の浮動小数点変数をシグナル型 NaN に初期化します。(Fortran のみ)。浮動小数点例外のマスクを解除するには、/fpe:0 (-fpe0) を指定します。
/Qcheck-pointer s:kywd	-check-pointers =kywd	ポインターによるメモリアクセスの範囲チェックを有効にします。 kywd には、none (デフォルト)、write または rw (読み取りと書き込み) を指定できます。(C/C++ のみ)。
/Qcheck-pointer s-mpx:kywd	-check-pointers-mpx=kywd	ポインターによるメモリアクセスのハードウェア支援範囲チェックを有効にします。 kywd には、none (デフォルト)、write (書き込みのみチェック) または rw (読み取りと書き込みをチェック) を指定できます。(C/C++ のみ)。

インテル® コンパイラーの無償評価版ダウンロードはこちら:

<https://www.xlsoft.com/jp/products/download/intelj.html>

上記の評価版登録ページで、「インテル® Parallel Studio XE」をお選びいただくことでインテル® コンパイラーをダウンロードいただけます。

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:

http://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 の商標です。

Microsoft および Windows は、米国 Microsoft Corporation の、米国およびその他の国における登録商標または商標です。

* 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。

© 2016 Intel Corporation. 無断での引用、転載を禁じます。

XLsoft のロゴ、XLsoft は XLsoft Corporation の商標です。Copyright © 2016 XLsoft Corporation.