インテル® C++ コンパイラー 17.0 Update 1 for Windows® リリースノート (インテル® Parallel Studio XE 2017)

このドキュメントでは、新機能、変更された機能、注意事項、および製品ドキュメントに記述されていない既知の問題について説明します。

詳細は、パッケージに含まれるライセンスと本リリースノートの「著作権と商標について」を参照してください。本リリースのインテル® C++ コンパイラー 17.0 についての詳細は、次のリンクを参照してください。

変更履歴

Update 1 (インテル® C++ コンパイラー 17.0.1)

- インテル® コンパイラーのドキュメントおよび診断メッセージの日本語訳を追加
- 報告された問題を修正

Microsoft® Visual Studio® 2010 のサポートを終了および Visual Studio® 2012 のサポートを終了予定

- Visual Studio® 2010 のサポートはバージョン 16.0 で終了予定になり、17.0 で終了しました。
- Visual Studio® 2012 のサポートは終了予定です。

インテル® C++ コンパイラー 16.0 以降 (インテル® C++ コンパイラー 17.0 での変更)

- Windows® で long double 関数の使用をサポートする "*math.h*" を追加
- 第 2 世代インテル® Xeon Phi™ プロセッサー・ファミリー向けに新しい CPU 名 "*mic_avx512*" を追加
- インテル® C++ コンパイラーのヘッダーファイルのリストをサブフォルダーに移動 (17.0 RTM)

- SIMD Data Layout Templates (SDLT) に N 次元配列のサポートを追加 (17.0 RTM)
- OpenMP* 4.0 以降の新機能をサポート
- 新しいインテル® Xeon Phi™ プロセッサー/コプロセッサーへのオフロード機能
- アノテーション付きソースリスト
- コード・アライメント用の新しい属性、プラグマ、コンパイラー・オプション
- C++14 の機能をサポート
- C11 の機能をサポート
- 新規および変更されたコンパイラー・オプション

先頭へ戻る

動作環境

- インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) 対応のインテル® 64 アーキテクチャー・プロセッサーをベースとするコンピューター (第 2 世代以降のインテル® Core™ i3/i5/i7 プロセッサー、インテル® Xeon® プロセッサー E3/E5 ファミリー、または互換性のあるインテル以外のプロセッサー)
- RAM 2GB (4GB 推奨)
- 4GB のディスク空き容量 (すべての機能をインストールする場合)
- インテル® メニー・インテグレーテッド・コア (インテル® MIC) アーキテクチャー向けの開発/テスト:
    - インテル® Xeon Phi™ コプロセッサー
    - インテル® メニーコア・プラットフォーム・ソフトウェア・スタック (インテル® MPSS)
    - オフロードコードのデバッグには Microsoft® Visual Studio® 2012 以降が必要
- インテル® グラフィックス・テクノロジーへのオフロードまたはネイティブサポートの開発/テスト
    - 次のプロセッサー・モデルをサポートしています。
        - インテル® Xeon® プロセッサー E3-1285 v3 および E3-1285L v3 (インテル® C226 チップセット) (インテル® HD グラフィックス P4700)
        - 第 5 世代インテル® Core™ プロセッサー (インテル® Iris™ Pro グラフィックス、インテル® HD グラフィックス)
        - 第 4 世代インテル® Core™ プロセッサー (インテル® Iris™ Pro グラフィックス、インテル® Iris™ グラフィックス、またはインテル® HD グラフィックス 4200+ シリーズ)
        - 注: リストされているチップセットのインテル® Xeon® プロセッサーのみサポートしています。ほかのチップセットのインテル® Xeon® プロセッサーはサポートしていません。前世代のインテル® Core™ プロセッサーはサポートしていません。インテル® Celeron® プロセッサーおよびインテル® Atom™ プロセッサーとの互換性はありません。
    - インテル® グラフィックス・テクノロジー対応の最新の 32 ビットまたは 64 ビット・グラフィックス・ドライバー (インテル® ダウンロード・センター (英語) から入手できます)
    - Windows® 用 binutils (http://intel.ly/1fHX7xO (英語) から入手できます)
        - binutils をインストールした後、ld.exe を含むディレクトリーを PATH に追加する必要があります。
- Microsoft® Windows® 7 (SP1)、Microsoft® Windows® 8、Microsoft® Windows® 8.1、Microsoft® Windows® 10、Microsoft® Windows Server® 2008 (R2 SP1)、Microsoft® Windows® HPC Server 2008 (R2)、Microsoft® Windows Server® 2012 (R2) (エンベデッド・エディションはサポートされていません) [1]
    - Microsoft® Windows® 8、Microsoft® Windows® 8.1 および Microsoft® Windows Server® 2012 では、製品は「デスクトップ」環境にインストールされます。「Windows® 8 UI」アプリケーションの開発はサポートされていません。[2]

- IA-32 対応アプリケーションまたはインテル® 64 対応アプリケーションのビルドに、Microsoft® Visual Studio® 開発環境あるいはコマンドライン・ツールを使用する場合は、次のいずれか:
    - Microsoft® Visual Studio® 2015 Professional Edition 以上 (「Visual C++ 2015 用の共通ツール」コンポーネントがインストールされていること) [3]
    - Microsoft® Visual Studio® Community 2015 以上 (「Visual C++ 2015 用の共通ツール」コンポーネントがインストールされていること) [3]
    - Microsoft® Visual Studio® 2013 Professional Edition 以上 (C++ コンポーネントがインストールされていること)
    - Microsoft® Visual Studio® Community 2013 以上 (C++ コンポーネントがインストールされていること)
    - Microsoft® Visual Studio® 2012 Professional Edition 以上 (C++ コンポーネントがインストールされていること)
- IA-32 [1] アーキテクチャー・アプリケーションのビルドに、コマンドライン・ツールのみを使用する場合は、次のいずれか:
    - Microsoft® Visual C++® Express 2015 for Windows® Desktop
    - Microsoft® Visual C++® Express 2013 for Windows® Desktop
    - Microsoft® Visual C++® Express 2012 for Windows® Desktop
- インテル® 64 対応アプリケーションのビルドに、コマンドライン・ツールのみを使用する場合は、次のいずれか:
    - Microsoft® Visual C++® Express 2015 for Windows® Desktop
    - Microsoft® Visual C++® Express 2013 for Windows® Desktop
    - Microsoft® Visual C++® Express 2012 for Windows® Desktop
    - Microsoft® Windows® Software Development Kit for Windows® 8
- ドキュメントの参照用に Adobe* Reader* 7.0 以降

注

1. アプリケーションは、上記の開発用と同じ Windows® バージョンで実行できます。また、Windows® XP よりも前の非エンベデッドの Microsoft® Windows® 32 ビット・バージョンでも実行できますが、インテルではこれらの互換性テストは行われていません。開発アプリケーションが、古いバージョンの Windows® にはない Win32 API ルーチンを使用している可能性があります。アプリケーションの互換性テストをご自身の責任で行ってください。アプリケーションを実行するには、特定のランタイム DLL をターゲットシステムにコピーしなければならないことがあります。
2. インテル® C++ コンパイラーは、Windows® 8 UI アプリの開発をサポートしていません。
   インテルでは、ユーザーの皆様のご意見を常に参考にしています。例えば、Windows® 8 UI アプリケーションにインテル® C++ コンパイラーまたはその他のインテル® ソフトウェア開発製品の機能を利用したい方は、インテル® プレミアサポート (https://premier.intel.com/ (英語)) からご意見をお送りください。
3. インテル® C++ コンパイラーを Microsoft® Visual Studio® 2015 で使用するには、Visual Studio® から「Visual C++ 2015 用の共通ツール」コンポーネントをインストールする必要があります。この記事 (英語) の説明を参照してください。

インテル® メニーコア・プラットフォーム・ソフトウェア・スタック (インテル® MPSS)

インテル® メニーコア・プラットフォーム・ソフトウェア・スタック (インテル® MPSS) は、インテル® C++ コンパイラーのインストール前またはインストール後にインストールできます。
最新バージョンのインテル® MPSS を使用することを推奨します。インテル® Parallel Studio XE for Windows® を登録すると、インテル® ソフトウェア開発製品レジストレーション・センター (http://registrationcenter.intel.com) から入手できます。

ユーザー空間およびカーネルドライバーのインストールに必要な手順については、インテル® MPSS のドキュメントを参照してください。

先頭へ戻る

インテル® C++ コンパイラーの使用方法

インテル® Parallel Studio XE 2017: コマンドラインおよび Microsoft® Visual Studio® からのインテル® C++ コンパイラーの使用方法についての情報は、「インテル® C++ コンパイラー 17.0 for Windows® 入門」(*<install-dir>*\documentation_2017\ja\compiler_c\ps2017\get_started_wc.htm) に含まれています。

先頭へ戻る

ドキュメント

製品ドキュメントは、*<install-dir>*\documentation_2017\ja\compiler_c\ps2017\get_started_wc.htm からリンクされています。

Microsoft® Visual Studio® のオンラインヘルプ形式

オンラインヘルプ形式はブラウザーベースです。Microsoft® Visual Studio® の [ヘルプ] メニューからインテルのドキュメントを参照する場合、または *F1* キー、ダイアログボックスにあるヘルプボタン、その他の GUI で状況依存ヘルプを参照する場合、デフォルトのブラウザーに対応するヘルプトピックが表示されます。デフォルトのブラウザーによっては、いくつかの小さな問題が発生することがあります。次のような既知の問題があります。

- [ヘルプ設定の設定] が [ブラウザーで起動] に設定されている場合、[ツール] > [オプション] > [F# ツール] または [ツール] > [オプション] > [Intellitrace] で *F1* キーを押すと、ブラウザーが 2 つ開きます。
- **Chrome™:** 検索またはキーワードからトピックを表示すると、目次が同期しません。[トピックを同期] も動作しません。
- **Firefox*:** 目次が表示されなくなることがあります。検索の大文字と小文字は区別されます。
- **Safari*:** Windows® の反応が遅くなります。

Windows Server® 2012 の Microsoft® Internet Explorer® 10 でドキュメントが表示されない問題
Windows Server® 2012 の Internet Explorer® 10 でヘルプまたはドキュメントを表示できない場合、Microsoft® Internet Explorer® のセキュリティー設定を変更すると表示されるようになります。[ツール] > [インターネット オプション] > [セキュリティ] を選択して、信頼済みサイトのリストに "about:internet" を追加します。オプションで、ドキュメントを参照した後に信頼済みサイトのリストから "about:internet" を削除できます。

Windows Server® 2012 で Visual Studio® 2012 のドキュメントが表示されない問題
Windows Server® 2012 で Visual Studio® 2012 のヘルプまたはドキュメントを表示できない場合、Microsoft® Internet Explorer® のセキュリティー設定を変更すると表示されるようになります。[ツール] > [インターネット オプション] > [セキュリティ] を選択して、[インターネット] ゾーンで [MIME スニッフィングを有効にする] および [アクティブ スクリプト] を有効にします。

日本語のサポート

日本語対応のインテル® コンパイラーをインストールした場合、オプションで日本語のサポートが提供されます。エラーメッセージ、仮想開発環境のダイアログ、一部のドキュメントが (英語に加えて) 日本語で提供されます。デフォルトでは、エラーメッセージとダイアログの言語はオペレーティング・システムの言語で表示されます。日本語ドキュメントは、ドキュメントの **ja** サブディレクトリーに含まれています。

日本語のサポートは、すべてのアップデートではなく、一部のアップデートで提供されます。

日本語オペレーティング・システムで英語のサポートを使用する (または英語オペレーティング・システムで日本語のサポートを使用する) 方法については、この記事 (英語) を参照してください。

先頭へ戻る

インテルが提供するデバッグ・ソリューション

- インテルが提供するデバッグ・ソリューションは GNU* GDB ベースです。詳細は、「インテル® Parallel Studio XE 2017 Composer Edition for C++ - デバッグ・ソリューション・リリースノート」(英語) を参照してください。

先頭へ戻る

サンプル

製品のサンプルは、「インテル® ソフトウェア製品のサンプルとチュートリアル」(英語) からダウンロードできます。

先頭へ戻る

テクニカルサポート

インストール時に製品の登録を行わなかった場合は、インテル® ソフトウェア開発製品レジストレーション・センター (http://registrationcenter.intel.com) で登録してください。登録を行うことで、サポートサービス期間中 (通常は 1 年間)、製品アップデートと新しいバージョンの入手を含む無償テクニカルサポートが提供されます。

テクニカルサポート、製品のアップデート、ユーザーフォーラム、FAQ、ヒント、およびその他のサポート情報は、http://www.intel.com/software/products/support/ (英語) を参照してください。
注: 販売代理店がこの製品のテクニカルサポートを提供している場合、インテルではなく販売代理店にお問い合わせください。

先頭へ戻る

新機能と変更された機能

このバージョンでは、次の機能が新たに追加または大幅に拡張されています。これらの機能に関する詳細は、ドキュメントを参照してください。

Windows® で long double 関数の使用をサポートする "*math.h*" を追加

インテル® C++ コンパイラー 17.0 for Windows® およびインテル® Visual Fortran コンパイラー 17.0 for Windows® に long double 関数をサポートする "*math.h*" ファイルが追加され、"*mathimf.h*" の代わりに "*math.h*" を使用して long double 関数を呼び出すことで Microsoft® Visual C++® の long double サポートとの互換性問題が発生しないようになりました。
詳細は、『インテル® C++ コンパイラー 17.0 デベロッパー・ガイドおよびリファレンス』を参照してください。

第 2 世代インテル® Xeon Phi™ プロセッサー・ファミリー向けに新しい CPU 名 "*mic_avx512*" を追加

インテル® アドバンスト・ベクトル・エクステンション 512 (インテル® AVX-512) 基本命令、競合検出命令、指数および逆数命令、プリフェッチ命令、および RDSEED および ADX (Multi-Precision Add-Carry Instruction Extensions) 命令を含むインテル® アドバンスト・ベクトル・エクステンション 2 (インテル® AVX2) 対応第 2 世代インテル® Xeon Phi™ プロセッサー・ファミリー向けに、新しい CPU 名 "*mic_avx512*" を追加しました。
新しい CPU 名の使用方法の詳細は、『インテル® C++ コンパイラー 17.0 デベロッパー・ガイドおよびリファレンス』を参照してください。

インテル® C++ コンパイラーのヘッダーファイルのリストをサブフォルダーに移動

コンパイラーのヘッダーファイルのリストを既存の include フォルダーのサブフォルダーに移動しました。インテル® C++ コンパイラーのヘッダーを使用するソースコードを変更する必要はありません。新しいサブフォルダーは、コンパイラー・ドライバーによりコンパイル中に自動的に検索されます。

SIMD Data Layout Templates (SDLT) に SIMD プログラムの集約 (Gather)/分散 (Scatter) を減らす N 次元配列のサポートを追加

- C++ AOS レイアウトでは、SIMD ループ/関数をベクトル化するときの集約 (Gather)/分散 (Scatter) の生成を最小限に抑えるように AOS->SOA 変換の注釈を付けるプログラマー用の言語拡張は存在しません。
- SDLT プリミティブ・テンプレート V2 は、n-D コンテナーをサポートしています。このコンテナーは、n-D AOS レイアウトを n-D SOA レイアウトに変換するプリミティブ/メソッドのセットをサポートする C++11 機能を使用して設計および実装されています。

OpenMP* 4.0 以降の新機能をサポート

- *#pragma omp for linear (list [: linear-step])* をサポート
    - *list* は *list* または *modifier(list)* のいずれか
- *linear* 節の *ref*、*val*、*uval modifier* をサポート
    - 例: *linear(ref(p)), linear(val(i):1), linear(uval(j):1)*
- *#pragma omp simd simdlen(n)* をサポート
- *#pragma omp ordered [simd]* をサポート
- 配列全体のリダクション: *int x[n]; #pragma omp simd reduction(+:x)*
- *processor* 節の拡張を *#pragma omp declare simd* に追加 (OpenMP* 4.5 の正式な機能ではありません)

新しいインテル® Xeon Phi™ プロセッサー/コプロセッサーへのオフロード機能

- OpenMP* 4.5 節の変更
    - 結合構造または複合構造の場合、if 節でディレクティブ名修飾子をサポート

        - *if([directive-name-modifier :] scalar-expression)* 構造が directive-name-modifier で指定された場合、if 節はその構造のセマンティクスにのみ適用されます。その他の場合、if 節を適用できるすべての構造に適用されます。
          例: *#pragma omp target parallel for if(target : do_offload_compute)*
    - *use_device_ptr(list)* 節を *#pragma omp target data* に実装
    - *is_device_ptr(list)* 節を *#pragma omp target* に実装
- 結合 *target* 構造のサポート
    - *#pragma omp target parallel*
    - *#pragma omp target parallel for*
    - *#pragma omp target simd*
    - *#pragma omp target parallel for simd*
- 新しいデバイスメモリー API のサポート
    - *void* omp_target_alloc()*
    - *void omp_target_free()*
    - *int omp_target_is_present()*

アノテーション付きソースリスト

- この機能は、コンパイラーによる最適化レポートをソースファイルに追加します。リスト形式はテキストまたは html のいずれかで指定します。リストを表示する場所は、呼び出し元、呼び出し先、または両方で指定できます。

コード・アライメント用の新しい属性、プラグマ、コンパイラー・オプション

- 新しい属性 *__attribute__((code_align(n)))* は、関数を 2 の累乗のバイト境界 *n* でアライメントします。
- 新しいプラグマ *#pragma code_align [(n)]* は、後続のループの先頭を 2 の累乗のバイト境界 *n* でアライメントします。
- 新しいコンパイラー・オプション */Qalign-loops[:n]* は、すべてのループを 2 の累乗のバイト境界 *n* でアライメントします。*/Qalign-loops-* (デフォルト) は、特別なループのアライメントを行いません。

C++14 の機能をサポート

インテル® C++ コンパイラー 17.0 は、*/Qstd=c++14* (Windows®) または *-std=c++14* (Linux*/OS X*) コンパイラー・オプションで以下の C++14 の機能をサポートします。

- 可変テンプレート (N3651)
- constexpr の緩和 (拡張) (N3652)
- サイズ指定された解放 (N3663)
- 以前の主要バージョンのコンパイラーとの比較を含む、サポートしている機能の最新リストは、「インテル® C++ コンパイラーでサポートされる C++14 の機能」を参照してください。

C11 の機能をサポート

インテル® C++ コンパイラーは、*/Qstd=c11* (Windows®) または *-std=c11* (Linux*/OS X*) コンパイラー・オプションで以下の C11 の機能をサポートします。

- _Atomic および __attribute((atomic)) キーワードを除くすべての C11 の機能
- 以前の主要バージョンのコンパイラーとの比較を含む、サポートしている機能の最新リストは、「インテル® C++ コンパイラーにおける C11 サポート」を参照してください。

新規および変更されたコンパイラー・オプション

コンパイラー・オプションの詳細は、『インテル® C++ コンパイラー 17.0 デベロッパー・ガイドおよびリファレンス』の「コンパイラー・オプション」セクションを参照してください。

- */fp: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ーで、一貫した再現性のある結果を有効にします。
- */guard:keyword* 制御フローの保護メカニズムを有効にします。
- */MP-force* /MP オプションが指定されたときに使用されるデフォルトのヒューリスティックを無効にします。これにより、スポーンされるプロセス数を制御することができます。
- */Qalign-loops[:n]* 2 の累乗のバイト境界でループをアライメントします。
- */Qopt-report-annotate* アノテーション付きソースリスト機能を有効にし、その形式を指定します。
- */Qopt-report-annotate-position* アノテーション付きソースリスト機能を有効にし、ループの最適化によりインライン展開が行われた場合に最適化メッセージを表示するアノテーション付きソースの位置を指定します。
- */Zo[-]* 最適化されたコードの拡張デバッグ情報の生成を有効/無効にします。

廃止予定のコンパイラー・オプションのリストは、『インテル® C++ コンパイラー 17.0 デベロッパー・ガイドおよびリファレンス』の「コンパイラー・オプション」セクションを参照してください。

先頭へ戻る

終了予定のサポート

Microsoft® Visual Studio® 2012 のサポートを終了予定

Microsoft® Visual Studio® 2012 のサポートは、将来のリリースで終了する予定です。

終了したサポート

第 3 世代インテル® Core™ プロセッサーの GFX オフロードサポートを終了

第 3 世代インテル® Core™ プロセッサーのプロセッサー・グラフィックスの GFX オフロードサポートは、インテル® C++ コンパイラー 17.0 で終了しました。

Microsoft® Visual Studio® 2010 のサポートを終了

Microsoft® Visual Studio® 2010 のサポートを終了しました。

IA-32 ホストへのインストールのサポートを終了

IA-32 ホストへのインストールのサポートを終了しました。32 ビット・ターゲット用コード生成のサポートは 64 ビット・ホストでサポートされます (*/Qm32* コンパイラー・オプションを使用)。

_GFX_enqueue を削除

_GFX_enqueue は削除されました。*_GFX_offload* に変更してください。

先頭へ戻る

既知の制限事項

Microsoft® Visual Studio® 2015 におけるインテル® Xeon Phi™ コプロセッサーをターゲットとするオフロードコードのデバッグ

/ZI オプションの処理に関する既知のコンパイラーの問題により、インテル® Xeon Phi™ コプロセッサーをターゲットとするオフロードコードのデバッグに必要なデバッグシンボル情報が生成されません。デバッグシンボル情報が不足していると、ブレークポイントで停止しない、特定のプログラム変数の詳細を確認できないなど、オフロードコードでデバッガーが正常に動作しません。このデバッグシンボル情報の不足により、その他の問題が発生する可能性もあります。
この問題は、コンパイラーのコマンドラインまたは Microsoft® Visual Studio® 2015 の設定 ([プロパティ] > [C/C++] > [General (全般)] > [Debug Information Format (デバッグ情報の書式)]) で /ZI オプションを使用した場合にのみ発生します。この問題を回避するには、この設定またはコンパイラーのコマンドラインで /Zi オプションを使用します。サポートしているほかのバージョンの Microsoft® Visual Studio® には影響はありません。

警告メッセージ "警告 #10373: '/ZI' は Linux オプションと一致しません"

/ZI を使用してインテル® Xeon Phi™ コプロセッサーをターゲットとするオフロードコードをコンパイルすると、上記の警告メッセージ "警告 #10373: '/ZI' は Linux オプションと一致しません" が出力されます。
オフロードコードのコンパイルで /ZI コンパイラー・オプションが正しく処理されず、コンパイル時に警告メッセージ "icl: 警告 #10373: '/ZI' は Linux オプションと一致しません" が出力されます。
この警告が出力された場合、オフロードコードのデバッグに影響があります。警告を出力しないようにするには、代わりに /Zi オプションを使用して、オフロードコードのデバッグに必要なデバッグシンボル情報を生成します。Microsoft® Visual Studio® 2015 の場合は、[プロパティ] > [C/C++] > [General (全般)] > [Debug Information Format (デバッグ情報の書式)] の設定を参照してください。

ポインターチェッカーにダイナミック・ランタイム・ライブラリーが必要

/Qcheck-pointers オプションを使用する場合は、ランタイム・ライブラリー *libchkp.dll* をリンクする必要があります。*/MD* のようなオプションを */Qcheck-pointers* とともに使用すると、設定に関係なくこのダイナミック・ライブラリーがリンクされることに注意してください。詳細は、
http://intel.ly/1jV0eWD (英語) を参照してください。

日本語版 Windows® にインストールすると、IDE からインテル® コンパイラーのヘルプ・ドキュメントを起動できない

インテル® Parallel Studio XE 2017 を日本語版 Windows® にインストールすると、Microsoft® Visual Studio® IDE からインテル® コンパイラーのヘルプ・ドキュメントを起動できないことがあります。この問題の詳細は、こちら (英語) を参照してください。

複数のペインを含むドキュメントが Visual Studio® 内のブラウザーで正しく表示されない

Visual Studio® 内のブラウザーには複数のペインを含むドキュメントが正しく表示されない制限があります (左のペインに目次が表示されますが、右のペインにコンテンツが表示されません)。
回避策: Visual Studio® の [ヘルプ] メニューから同じドキュメントにアクセスします。

異なるコンパイラーを使用して 256 ビット・ベクトル型引数をコンパイルするとランタイムにアクセス違反が発生するコードが生成される

2 つの異なるコンパイラー (Microsoft® Visual C++® 2013 コンパイラーおよびインテル® C++ コンパイラー15.0 以降) を使用してアプリケーションを作成した場合、アライメントされていないデータアクセスによる一般保護違反が発生することがあります。この問題は、256 ビット・ベクトル型引数が呼び出しの際に参照で渡され、呼び出し元を Visual C++® でビルドし、その引数をインテル® C++ コンパイラーでビルドした関数でアクセスすると発生します。

原因は、256 ビット・ベクトル型引数のアライメントが一致しないためです。

<code> にインテル® AVX 以降の新しいコード値 (CORE-AVX-I、CORE-AVX2、その他) を指定して */Qx<code>* コンパイラー・オプションを使用した場合は、アプリケーションのソースコードで *__mm256_stream_** (非テンポラルデータのロード/ストア組込み関数) が明示的に使用されない限り、アライメントされていないアクセス命令がこれらのインスタンスで実際に使用されるため、この問題は発生しません。

Visual Studio® の既知の問題

- バージョン管理システムでのインテル® C++ プロジェクトの使用

  Microsoft® Visual Studio® プロジェクトがバージョン管理システム (例: Microsoft® Visual SourceSafe や Microsoft® Visual Studio® Team Foundation Server など) で管理されている場合、プロジェクトでインテル® C++ プロジェクト・システムを使用するには追加のステップが必要です。このトピックについての詳細な記事は、http://intel.ly/plmnpo (英語) を参照してください。
- *MSVCP90D.dll* (またはその他の Microsoft® ランタイム DLL) が見つからない

  サンプル・プロジェクト (および Microsoft® Visual C++® プロジェクト) を実行するときに Microsoft® Visual Studio® のランタイム DLL が見つからない場合、ランタイムエラーが発生します。これは、マニフェスト・ファイルや SXS アセンブリーが見つからないことが原因です。この問題を解決するには、使用しているバージョンの Microsoft® Visual Studio® の redist フォルダー (デフォルトの場所は *c:\program files[(x86)]\Microsoft Visual Studio X.X\VC\redist*) に移動します。*amd64*、*x86*、*Debug_NonRedist* サブフォルダーで、必要なランタイムが含まれているフォルダーを探します (デバッグ・ライブラリーを探す場合は、ファイル名の最後が D のファイルが含まれているフォルダーを探します)。必要なランタイムが含まれているフォルダーが見つかったら、そのフォルダーの (.manifest ファイルを含む) すべての内容を、実行する .exe ファイルのあるフォルダーにコピーします。
- Microsoft Edge™ の状況依存 (F1) ヘルプ問題
- Microsoft Edge™ を Microsoft® Visual Studio® のデフォルトブラウザーとして設定している場合、特定の機能/関数で状況依存 (F1) ヘルプを呼び出すと、機能/関数の説明に関連するトピックの代わりに、対応するドキュメントのタイトルページが表示されます。正しい動作にするには、異なるデフォルトブラウザーを使用するように Microsoft® Visual Studio® の設定を変更してください。
- Visual Studio® 2015 Update 1 へのインテル® C++ コンパイラー IDE 統合には、出力ウィンドウのコードカバレッジ・レポートのリンクに正しいパスが含まれない問題があります。リンクには、パスの最初のスペースより前の部分のみ含まれています。
- 例: *"file:\\C:\Users\Documents\Visual Studio 2015\Projects\ConsoleApplication6\ConsoleApplication1\x64\Release\CCovLog.html"*
- この問題を回避するには、出力ウィンドウからリンクをコピーして、ファイルシステムでリンクを開いてください。

インテル® メニー・インテグレーテッド・コア (インテル® MIC) アーキテクチャーの既知の問題

- _Cilk_shared の制限
    - 仮想基本クラスで _Cilk_shared 属性を指定することはできません。
    - 複数の基本クラスから _Cilk_shared 属性が指定されたクラスを派生させることはできません (複数の継承は許可されません)。
    - _Cilk_shared 属性が指定されたクラスで仮想デストラクターを定義することはできません。
    - _Cilk_shared 属性が指定されたクラスを別の _Cilk_shared クラスの基本クラスとして使用する場合、そのサイズが 8 の倍数になるように (必要に応じて、仮のフィールドを追加して) プログラマーが調整する必要があります。
    - _Cilk_offload は、共有ライブラリー (DLL) を使うプログラムでは使用できません
- 共有ライブラリーに含まれるコードをオフロードする際に */Qoffload=mandatory* オプションまたは */Qoffload=optional* オプションを指定してメインプログラムのリンクが必要

  オフロードには初期化処理が必要ですが、これはメインプログラムでのみ行うことができます。つまり、共有ライブラリーに含まれるコードをオフロードする場合、初期化処理が行われるように、メインプログラムもリンクしなければなりません。メインコードやメインプログラムへスタティック・リンクされたコードにオフロード構造が含まれる場合、これは自動で行われます。そうでない場合、*/Qoffload=mandatory* コンパイラー・オプションまたは */Qoffload=optional* コンパイラー・オプションを指定して、メインプログラムをリンクする必要があります。
- コンパイル時の診断の **MIC** タグ

  ターゲット (インテル® MIC アーキテクチャー) とホスト CPU のコンパイルを区別できるようにコンパイラーの診断インフラストラクチャーが変更され、出力メッセージに **MIC** タグが追加されました。このタグは、インテル® MIC アーキテクチャー用のオフロード拡張を使用してコンパイルしたときに、ターゲットのコンパイル診断にのみ追加されます。

  下記の例で、サンプル・ソース・プログラムは、ホスト CPU とターゲット (インテル® MIC アーキテクチャー) のコンパイルの両方で同じ診断を行っています。ただし、プログラムによっては、2 つのコンパイルで異なる診断メッセージが出力されます。新しいタグが追加されたことで、CPU とターゲットのコンパイルを容易に区別できることが分かります。

  ```
  $ icl -c sample.c

  sample.c(1): 警告 #1079: *MIC* 関数 "main" の戻り型は "int" でなければなりません。
  void main()
  ^
  sample.c(5): 警告 #120: *MIC* 戻り値の型が関数の型と一致しません。
  return 0;
  ^
  sample.c(1): 警告 #1079: 関数 "main" の戻り型は "int" でなければなりません。
  void main()
  ^
  sample.c(5): 警告 #120: *MIC* 戻り値の型が関数の型と一致しません。
  return 0;
  ```
- ランタイム型情報 (RTTI) は未サポート

  仮想共有メモリー・プログラミングでは、ランタイム型情報 (RTTI) はサポートされていません。特に、*dynamic_cast<>* と *typeid()* の使用はサポートされていません。
- 直接 (ネイティブ) モードにおけるランタイム・ライブラリーのコプロセッサーへの転送

インテル® メニーコア・プラットフォーム・ソフトウェア・スタック (インテル® MPSS) に、 */lib* 以下のインテル® コンパイラーのランタイム・ライブラリー (例えば、OpenMP* ライブラリー *libiomp5.so*) が含まれなくなりました。

このため、直接モード (例えば、コプロセッサー・カード上) で OpenMP* アプリケーションを実行する場合は、アプリケーションを実行する前にインテル® MIC アーキテクチャー OpenMP* ライブラリー (*<install_dir>\compilers_and_libraries_2017\windows\lib\mic\libiomp5.so*) のコピーをカード (デバイス名の形式は micN。最初のカードは *mic0*、2 番目のカードは *mic1*、...) に (*scp* 経由で) アップロードする必要があります。

このライブラリーが利用できない場合、次のようなランタイムエラーが発生します。

/libexec/ld-elf.so.1: "sample" で要求された共有オブジェクト "libiomp5.so" が見つかりません。

libimf.so のような別のコンパイラー・ランタイムでも同様です。必要なライブラリーは、アプリケーションおよびビルド構成により異なります。

- オフロード領域からの *exit()* の呼び出し

  オフロード領域から *exit()* を呼び出すと、"オフロードエラー: デバイス 0 のプロセスがコード 0 で予想外に終了しました" のような診断メッセージが出力され、アプリケーションが終了します。

インテル® グラフィックス・テクノロジーへのオフロードの既知の問題

- gfx_linker: : error : command 'ld.exe' exited with non-zero exit code -107374170

  x64 プロジェクトでインテル® グラフィックス・テクノロジーへコードをオフロードすると、binutils に含まれている *ld.exe* でリンカーエラーが表示されることがあります。この問題を解決するには、64 ビット用の binutils bin ディレクトリーを *PATH* 環境変数に追加して、Microsoft® Visual Studio® を再起動してください。
- オフロードコードのホストバージョンが並列化されない

  コンパイラーは、#pragma offload 以下に並列ループのターゲットバージョンとホストバージョンの両方を生成します。ホストバージョンは、オフロードが実行できない場合 (通常は、ターゲットシステムにインテル® グラフィックス・テクノロジーが有効なユニットがない場合) に実行されます。並列ループは、オフロードの並列セマンティクスを含む cilk_for の並列構文または配列表記文を使用して指定する必要があります。ターゲットバージョンはターゲット実行の際に並列化されますが、現在、ホスト側のバックアップ・バージョンが並列化されない制限があります。cilk_for を使用したときにオフロード実行が行われないと、バックアップ・コード実行のパフォーマンスに大きく影響する場合があることに注意してください。配列表記文は現在ホスト側で並列コードを生成しないため、パフォーマンスに影響はありません。これは既知の問題で、将来のリリースで修正される予定です。
- インテル® グラフィックス・テクノロジーへのオフロードの既知の制限事項
  -
  - Windows® 7 では、オフロードが行われたときにディスプレイをロックできません。アクティブ・ディスプレイが必要です。
  - オフロードコードでは、次の機能を使用できません。
    - 例外処理
    - RTTI
    - *longjmp/setjmp*
    - VLA
    - 変数引数リスト
    - 仮想関数、関数ポインター、その他の間接呼び出しまたはジャンプ
    - 共有仮想メモリー

        - 配列や構造体のようなポインターを含むデータ構造
        - ポインターまたは参照型のグローバル変数
        - OpenMP*
        - *cilk_spawn* または *cilk_sync*
        - インテル® Cilk™ Plus のレデューサー
        - ANSI C ランタイム・ライブラリー呼び出し (SVML、*math.h*、*mathimf.h* 呼び出し、およびその他いくつかの例外あり)
    - 64 ビット浮動小数点演算および整数演算は非効率

インテル® Cilk™ Plus の既知の問題

- スチールが行われた後、対応する *_Cilk_sync* の前に SEH 例外がスローされると、Microsoft® C++ 構造化例外処理 (SEH) は失敗します。

ガイド付き自動並列化の既知の問題

- プログラム全体のプロシージャー間の最適化 (*/Qipo*) が有効な場合、単一ファイル、関数名、ソースコードの指定範囲に対してガイド付き自動並列化 (GAP) 解析は行われません。

後置する戻り型の外部定義の decltype 式でテンプレート依存関数の呼び出しが行われた場合のスプリアスエラー

- これはインテル® C++ コンパイラー 16.0 Update 2 における既知のリグレッションです。次に例を示します。

```
template <class T>
struct C {
  int then(int*,int*);
  template <class T2>
  auto then(T2 arg) -> decltype(this->then(&arg, &arg));
};
template <class T>
template <class T2>
auto C<T>::then(T2 arg) -> decltype(this->then(&arg, &arg)) { return 0; }
void foo() {
   C<int> f;
   f.then(99);
}
```

この問題を回避するには、定義を内部に移動するか、(戻り型を明示的に宣言して) 後置する戻り型を使用しないようにします。

c++14 の constexpr の緩和と Boost 問題

- -std=c++14 モードで Boost を使用していて constexpr 機能に関連すると思われるコンパイルエラーが表示された場合、boost_1_59_0/boost/config/compiler/gcc.hpp で BOOST_NO_CXX14_CONSTEXPR を定義してください。変更する行 (行 256 の前後) を次に示します。

変更前:

#if !defined(__cpp_constexpr) || (__cpp_constexpr < 201304)
 # define BOOST_NO_CXX14_CONSTEXPR
#endif

変更後:

//#if !defined(__cpp_constexpr) || (__cpp_constexpr < 201304)
 # define BOOST_NO_CXX14_CONSTEXPR
//#endif

先頭へ戻る

著作権と商標について

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ー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ー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ーに関して、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ー依存の最適化は、インテル® マイクロプロセッサー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ー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

本資料で紹介されている資料番号付きのドキュメントや、インテルのその他の資料を入手するには、1-800-548-4725 (アメリカ合衆国) までご連絡いただくか、http://www.intel.com/design/literature.htm (英語) を参照してください。

インテル・プロセッサー・ナンバーはパフォーマンスの指標ではありません。プロセッサー・ナンバーは同一プロセッサー・ファミリー内の製品の機能を区別します。異なるプロセッサー・ファミリー間の機能の区別には用いません。詳細については、http://www.intel.co.jp/jp/products/processor_number/ を参照してください。

インテル® C++ コンパイラーは、インテルのソフトウェア使用許諾契約書 (EULA) の下で提供されます。

詳細は、製品に含まれるライセンスを確認してください。

先頭へ戻る

コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。