インテル® Fortran コンパイラー 17.0 Update 4 for Linux* 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。

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」をお読みください。
本リリースのインテル® Fortran 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

このセクションでは製品アップデートにおける重要な変更内容を説明します。

Update 4 (インテル® Fortran コンパイラー 17.0.4)

- 日本語版を更新
- 報告された問題を修正

Update 3 (インテル® Fortran コンパイラー 17.0.3)

- 報告された問題を修正

Update 2 (インテル® Fortran コンパイラー 17.0.2)

- DIR$ VECTOR [NO] MASK_READWRITE
- 報告された問題を修正

先頭へ戻る

Update 1 (インテル® Fortran コンパイラー 17.0.1)

- 日本語版を含む最初のアップデート
- OpenMP* 監視スレッド
- 報告された問題を修正

先頭へ戻る

インテル® Fortran コンパイラー 16.0 以降 (インテル® Fortran コンパイラー 17.0.0 での変更)

- インテル® Fortran コンパイラーが 17.0.0 にアップデート
- OpenMP* 4.5 のディレクティブ
- コンパイラーによる最適化レポートをソースリストに追加する新しいオプション
- 関数のコード・アライメントを要求する ATTRIBUTES code_align(n) を追加
- 割付け配列に対する組込み代入のデフォルト動作の変更
- 論理/数値演算が混在する場合の動作を変更
- OpenMP* 4.5 によるローカルスカラー変数のデフォルトのオフロード動作の変更
- 新規および変更されたコンパイラー・オプション
- Fortran 2008 の機能をサポート
- 報告された問題を修正

先頭へ戻る

動作環境

アーキテクチャー名についての説明は、「インテル® アーキテクチャー・プラットフォームの用語」 (英語) を参照してください。

- 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・ベースのプロセッサーを搭載したコンピューター (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)
    - 64 ビット・アプリケーションおよびインテル® Xeon Phi™ コプロセッサーに作業をオフロードするアプリケーションの開発は、64 ビット・バージョンの OS でのみサポートしています。32 ビット・アプリケーションの開発も、64 ビット・バージョンの OS でのみサポートしています。
    - 64 ビット・バージョンの OS で 32 ビット・アプリケーションを開発する場合は、Linux* ディストリビューションからオプションのライブラリー・コンポーネント (ia32-libs、lib32gcc1、lib32stdc++6、libc6-dev-i386、gcc-multilib、g++-multilib) をインストールする必要があります。
- 機能を最大限に活用できるよう、マルチコアまたはマルチプロセッサー・システムの使用を推奨します。
- 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® Xeon Phi™ コプロセッサー向けの開発/テスト
    - インテル® Xeon Phi™ プロセッサー (開発コード名 Knights Ferry、開発コード名 Knights Corner、開発コード名 Knights Landing)
    - 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)
- 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を開発する場合は、次の Linux* ディストリビューションのいずれか (本リストは、インテル社により動作確認が行われた

ディストリビューションのリストです。その他のディストリビューションでも動作する可能性はありますが、推奨しません。ご質問は、テクニカルサポートまでお問い合わせください。)
    - Debian* 7.0、8.0
    - Fedora* 24、25
    - Red Hat* Enterprise Linux* 6、7
    - SUSE* Linux* Enterprise Server 11 (SP3、SP4)、12
    - Ubuntu* 14.04 LTS、15.10、16.04 LTS
    - インテル® Cluster Ready
- Linux* 開発ツール・コンポーネント (gcc、g++ および関連ツールを含む)。(本リストは、インテル社により動作確認が行われたコンポーネント・バージョンのリストです。その他のバージョンでも動作する可能性はありますが、推奨しません。ご質問は、テクニカルサポートまでお問い合わせください。)
    - gcc 4.3-6
    - binutils 2.20-2.26
- -traceback オプションを使用するには、libunwind.so が必要です。一部の Linux* ディストリビューションでは、別途入手して、インストールする必要があります。

注

- インテル® コンパイラーは、さまざまな Linux* ディストリビューションと gcc バージョンで動作確認されています。一部の Linux* ディストリビューションには、動作確認されたヘッダーファイルとは異なるバージョンのものが含まれており、問題を引き起こすことがあります。使用する glibc のバージョンは、gcc のバージョンと同じでなければなりません。最良の結果を得るため、上記のディストリビューションで提供されている gcc バージョンのみを使用してください。
- 非常に大きなソースファイル (数千行以上) を -O3、-ipo および -qopenmp などの高度な最適化オプションを使用してコンパイルする場合は、多量の RAM が必要になります。
- 一部の最適化オプションには、アプリケーションを実行するプロセッサーの種類に関する制限があります。詳細は、オプションの説明を参照してください。

先頭へ戻る

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® Xeon Phi™ コプロセッサー向けのアプリケーションを開発する場合、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® Fortran 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 Linux* 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® Fortran コンパイラーの使用方法

インテル® Fortran コンパイラーの使用方法についての情報は、「インテル® Fortran コンパイラー for Linux* 入門」 (<install-dir>/documentation_2017/ja/ps2017/getstart_comp_lf.htm) に含まれています。

先頭へ戻る

ドキュメント

製品ドキュメントへのリンクは、<install-dir>/documentation_2017/ja/ps2017/getstart_comp_lf.htm にあります。

デベロッパー・ガイドおよびリファレンス、新機能とリリースノート、インストール・ガイド

すべてのツール・コンポーネントのデベロッパー・ガイドおよびリファレンス、新機能とリリースノート、インストール・ガイドは、Intel® Parallel Studio XE Support > Documentation (英語) から入手できます。

先頭へ戻る

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、こちらの記事 (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 2017 Composer Edition for Fortran - 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

再配布可能なライブラリー

詳細は、「インテル® Parallel Studio XE の再配布ライブラリー」 (英語) を参照してください。

先頭へ戻る

テクニカルサポート

インテル® ソフトウェア開発製品レジストレーション・センターでライセンスを登録してください。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

互換性

一般に、インテル® Fortran コンパイラー for Linux* の以前のバージョン (8.0 以降) でコンパイルされたオブジェクト・コードおよびモジュールは、バージョン 17 でもそのまま使用できます。ただし、次の例外があります。

- バージョン 12.0 よりも前のコンパイラーでビルドされた CLASS キーワードを使用して多相変数を宣言しているソースは再コンパイルする必要があります。
- マルチファイルのプロシージャー間の最適化 (-ipo) オプションを使用してビルドされたオブジェクトは、最新のバージョンで再コンパイルする必要があります。
- バージョン 12.0 よりも前のコンパイラーでビルドされた REAL(16)、REAL*16、COMPLEX(16)、COMPLEX*32 データ型を使用しているオブジェクトは再コンパイルする必要があります。
- バージョン 10.0 よりも前のコンパイラーでインテル® 64 アーキテクチャー用にビルドされたモジュール変数を含むオブジェクトは再コンパイルする必要があります。Fortran 以外のソースからこれらの変数を参照する場合、不正な先頭の下線を削除するように外部名を変更する必要があります。
- バージョン 11.0 よりも前のコンパイラーでコンパイルされた、派生型宣言の外部で ATTRIBUTES ALIGN ディレクティブを指定したモジュールは再コンパイルする必要があります。この問題を検出すると、コンパイラーはメッセージを表示します。
- 派生型宣言の内部で ATTRIBUTES ALIGN ディレクティブを指定したモジュールは 13.0.1 以前のコンパイラーでは使用できません。
- Fortran 2008 のサブモジュール機能を実装するため、バイナリー .mod ファイルの内部フォーマットが大幅に変更されました。このため、バージョン 16.0 以降の Fortran コンパイラーで作成されたモジュールファイルは、バージョン 15.0 以前の Fortran コンパイラーで使用することはできません。

先頭へ戻る

REAL(16) および COMPLEX(16) データ型のスタック・アライメントの変更

バージョン 12.0 よりも古いコンパイラーでは、REAL(16) または COMPLEX(16) (REAL*16 または COMPLEX*32) 項目が値で渡される場合、スタックアドレスは 4 バイトでアラインされます。パフォーマンスを向上するため、バージョン 12 以降のコンパイラーは、これらの項目を 16 バイトでアラインし、引数が 16 バイト境界でアラインされていると仮定します。これは、gcc とも互換性があります。

この変更は、主にコンパイラーにより生成される REAL(16) 値の計算を行うライブラリー・ルーチン (組込み関数を含む) の呼び出しに影響します。以前のバージョンでコンパイルしたコードをバージョン 12 のライブラリーとリンクする場合、またはアプリケーションをインテルのランタイム・ライブラリーの共有バージョンにリンクする場合、正しくない結果が返される可能性があります。

バージョン 12.0 よりも古いコンパイラーでコンパイルされている場合、この問題を回避するには、REAL(16) および COMPLEX(16) データ型を使用しているすべての Fortran ソースを再コンパイルしてください。

先頭へ戻る

新機能と変更された機能

一部の言語機能に関する説明はコンパイラーのドキュメントにはまだ含まれていません。必要に応じて、Fortran 2008 Standard (PDF、英語) および Proposed draft Fortran 2015 Standard を参照してください。

Fortran 2008 の機能

- 組込み型の型宣言
- ポインターの初期化
- 暗黙形状配列
- EXIT 文の構文名の拡張
- 内部プロシージャーの BIND(C) のサポート

割付け配列に対する組込み代入のデフォルト動作の変更 (17.0)

以前のリリースでは、コンパイラーはデフォルトで、割付け配列に対する組込み代入では、代入される配列は値と同じ形状に割付け済みであると仮定していました。形状が一致しない場合に Fortran 2003 の自動 (再) 割付けを行うには、-assume realloc_lhs コンパイラー・オプション (-standard-semantics に含まれる) を指定する必要がありました。

コンパイラー 17.0 では、デフォルトの動作が Fortran 2003 標準と一致するように変更され、必要であれば割付け配列は組込み代入で自動的に (再) 割付けされるようになりました。この変更はパフォーマンスに多少影響します。古い動作に戻す場合は、-assume norealloc_lhs または新しい -nostandard-realloc-lhs コンパイラー・オプションを指定します。

論理/数値演算が混在する場合の動作の変更 (17.0)

Fortran 標準規格は、論理データ型と数値 (integer/real/complex) データ型が混在する代入および演算を禁止しています。標準規格の拡張として、インテル® Fortran コンパイラーではこの混在を許可していますが、これまで動作規則の文書化が不十分で、コンテキストに依存して (一貫しない) 実装が行われていました。

バージョン 17.0 では、論理/数値演算が混在する場合のコンパイラーの実装が一貫するように変更されました。場合によっては、新しい動作が以前のバージョンの実装と異なるため、以前の動作が正しいと仮定しているプログラムでは結果が変わる可能性があります。

バージョン 17.0 の動作は次のとおりです。

- 論理値が数値コンテキストで使用された場合、"fpscomp logicals" コンパイラー・オプションの設定に応じて、true 値は整数 -1 または +1 に変換されます。false 値は整数 0 に変換されます。
- 数値が論理コンテキストで使用された場合、最初に整数に変換されます (必要な場合)。そして、"fpscomp logicals" コンパイラー・オプションの設定に応じて、整数値をどのように扱うかが決まります。"nologicals" が有効な場合、奇数値は true、偶数値は false として扱われます。"logicals" が有効な場合、非ゼロ値は true、ゼロ値は false として扱われます。
- 数値が論理変数に代入された場合、"fpscomp logicals" の設定に応じて値が .TRUE. または .FALSE. に変換され、新しい論理値が代入されます。以前のバージョンでは、バイナリー値が変換されずに直接コピーされることがありました。
- 論理値が数値変数に代入された場合、上記のように、最初に整数に変換された後、組込み代入の通常の規則が適用されます。

インテル® Fortran コンパイラーのデフォルトは "fpscomp nologicals" ですが、"standard-semantics" オプションを指定すると "fpscomp logicals" が設定されることに注意してください。

プログラムがこの拡張による影響を受けるかどうか確認するには、標準警告を有効にして (/stand) プログラムをビルドします。古い動作に戻す場合は、-assume old_logical_assign を指定します。

OpenMP* 4.5 によるローカルスカラー変数のデフォルトのオフロード動作の変更

OpenMP* 4.5 の DEFAULTMAP (TOFROM:SCALAR) 節のサポートにより、ローカルスカラー変数のデフォルトの動作が変わりました。以前のリリースでは、ローカルスカラー変数はデフォルトでオフロードされました。17.0 では、ローカルスカラー変数はマップされず、代わりに暗黙的に FIRSTPRIVATE 属性が指定されます。17.0 でローカルスカラー変数をオフロードするには、DEFAULTMAP (TOFROM:SCALAR) 節を使用する必要があります。詳細は、『インテル® Fortran コンパイラー・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。

OpenMP* 監視スレッド

バージョン 17.0.1 では、OpenMP* 監視スレッド (ヘルパースレッド) が生成されません。

OpenMP* 機能

OpenMP* 4.0 (英語) および OpenMP* 4.5 (英語) の次のディレクティブ、節、プロシージャーがコンパイラーでサポートされました。

詳細は、コンパイラー・ドキュメントまたは上記の OpenMP* 仕様へのリンクを参照してください。

OpenMP* 4.5 のディレクティブ:

- TARGET ENTER DATA
- TARGET EXIT DATA
- TASKLOOP

節:

- OMP TARGET および OMP TARGET UPDATE ディレクティブの DEPEND
- OMP TARGET および OMP TARGET UPDATE ディレクティブの NOWAIT
- OMP SIMD ディレクティブの SIMDLEN
- OMP ORDERED ディレクティブの SIMD
- OMP DECLARE SIMD (proc-name) ディレクティブの PROCESSOR(cpuid)
- OMP TARGET ディレクティブの DEFAULTMAP (TOFROM:SCALAR)

processor 節を OMP DECLARE SIMD に追加

インテル® Fortran コンパイラー 17.0 には、プログラマーが OpenMP* SIMD で YMM/ZMM レジスターを使用できる、OMP DECLARE SIMD の拡張が含まれています。PROCESSOR(cpuid) 節は、指定したプロセッサー向けのルーチンのベクトルバージョンを生成するようにコンパイラーに指示します。詳細は、『インテル® Fortran コンパイラー・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。

!$OMP DO SCHEDULE 節の SIMD 修飾子と NONMONOTONIC 修飾子

インテル® Fortran コンパイラー 17.0 には、DO ループの反復をチームのスレッド間でどのように分割するかについて、ユーザー制御を強化する OMP DO SCHEDULE 節の新しい SIMD 修飾子と NONMONOTONIC 修飾子の拡張が含まれています。詳細は、『インテル® Fortran コンパイラー・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。

OpenMP* 4.5 で定義されている taskloop および do across ループのサポート

インテル® Fortran コンパイラー 17.0 では、for/do ループを並列化する新しいループ構造がサポートされました。"taskloop" は cilk_for ループに似ていて、インテルのタスク実行モデルの下で動的な分割統治ループ分割を可能にします。"doacross" は、ループ伝播の依存があるループの並列化を可能にします。

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-logical-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: *!$omp target parallel for if(target : do_offload_compute)*
    - *use_device_ptr(list)* 節を *!$omp target data* に実装
    - *is_device_ptr(list)* 節を *!$omp target* に実装
- 結合 *target* 構造のサポート
    - *!$omp target parallel*
    - *!$omp target parallel for*
    - *!$omp target simd*
    - *!$omp target parallel for simd*

omp declare simd linear 節の新しい修飾子

omp declare simd ディレクティブの **linear** 節を新しい修飾子で拡張

linear (*linear-list [* : *linear-step]* **)**
linear-list は次のいずれかです。
list

modifier **(***list***)**
modifier は次のいずれかです。
ref
val
uval

- すべての *list* 項目は各 SIMD レーンで同時に呼び出される関数の仮引数でなければなりません。
- *modifier* を指定しない場合や **val** または **uval** 修飾子を指定した場合、各レーンの各 *list* 項目の値は、関数に入るときの *list* 項目の値 + レーンの論理番号 × *linear-step* に相当します。
- **uval** 修飾子を指定した場合、各呼び出しは各 SIMD レーンと同じメモリー位置を使用します。このメモリー位置は論理的な最終レーンの最後の値で更新されます。
- **ref** 修飾子を指定した場合、各レーンの各 *list* 項目のメモリー位置は、レーンの論理番号 × *linear-step* でインデックスされた関数に入るときのメモリー位置の配列に相当します。

新しいディレクティブと追加されたディレクティブ

ATTRIBUTES code_align(n)

コンパイラー 17.0 では、関数のコード・アライメントを要求する ATTRIBUTES code_align(n) ディレクティブを指定することができます。'n' は最小アライメント境界 (バイト単位) で、1 から 4096 の範囲の 2 の累乗の値でなければなりません (例: 1、2、4、8、16、32、64、128 など)。n = 1 はアライメントを行いません。n は必ず指定する必要があります。

PROCESSOR(cpuid) 節の拡張

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 *cpuid* キーワード "mic_avx512" を追加しました。新しい *cpuid* キーワードの使用方法は、『インテル® Fortran 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。

DIR$ VECTOR [NO] MASK_READWRITE

ベクトライザーによるメモリー・スペキュレーションを有効/無効にする DIR$ VECTOR [NO] MASK_READWRITE 節をサポートします。

MASK_READWRITE | NOMASK READWRITE

条件文内でのマスク付きロード/ストア操作の生成を有効/無効にします。

MASK_READWRITE 節は、メモリーのスペキュレーションを無効にし、条件文内でマスク付きロード/ストア操作を生成するようにコンパイラーに指示します。

NOMASK_READWRITE 節は、メモリーのスペキュレーションを有効にし、条件文内でマスクなしロード/ストア操作を生成するようにコンパイラーに指示します。

先頭へ戻る

新規および変更されたコンパイラー・オプション

詳細は、コンパイラーのドキュメントを参照してください。

- -f[no-]align-loops[=n]
- -qopt-report-annotate [= text | html]
- -qopt-report-annotate-position= [caller | callee | both]
- -fp-model consistent
- -assume [no]old_logical_assign
- -[no]standard-realloc-lhs

廃止予定のコンパイラー・オプションのリストは、ドキュメントのコンパイラー・オプションのセクションを参照してください。

新しい -f[no-]align-loops[=n] コンパイラー・オプション

この新しいオプションは、2 の累乗のバイト境界でループをアライメントします。

新しい -qopt-report-annotate [= text | html] コンパイラー・オプション

このオプションは、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 (text または html) を指定します。デフォルト形式は text です。

新しい -qopt-report-annotate-position= [caller | callee | both] コンパイラー・オプション

このオプションは、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。値は、"caller"、"callee" または "both" です。デフォルト値は caller です。

新しい -fp-model consistent コンパイラー・オプション

このオプションを指定すると、ドライバーはより一貫した浮動小数点結果を提供するオプションの組み合わせをセットします。

新しい -assume [no]old_logical_assign コンパイラー・オプション

このオプションは、論理値を数値変数へ代入する代入文と、数値を論理変数へ代入する代入文の動作を指定します。「論理/数値演算が混在する場合の動作の変更」も参照してください。

新しい -[no]standard-realloc-lhs コンパイラー・オプション

このオプションは、/assume:[no]realloc-lhs の代替形式です。「割付け配列に対する組込み代入のデフォルト動作の変更」も参照してください。

先頭へ戻る

終了予定のサポート

先頭へ戻る

終了したサポート

Red Hat* Enterprise Linux* 5 のサポートを終了

このオペレーティング・システム・バージョンのサポートを終了しました。新しいバージョンのオペレーティング・システムに移行してください。

32 ビット・ホストへのインストールのサポートを終了

32 ビット・ホストへのインストールのサポートは、このリリースで終了しました。32 ビット・ターゲット用コードの生成は 64 ビット・ホストでサポートされます (-m32 コンパイラー・オプションを使用)。

先頭へ戻る

既知の問題

パラメーター化された派生型で文字長引数の特定の使用法が完全に実装されていない

パラメーター化された派生型 (PDT) では、文字長引数の次の使用法は完全に実装されていません。

- 文字長引数を含む PDT 引数定数
- %RE と %IM は未実装

OpenMP* 4.5 の OMP THREADPRIVATE と共通ブロック名の特定の使用法が診断されない

OpenMP* 4.5 の規則では、共通ブロック名を指定する THREADPRIVATE ディレクティブが 1 つのプログラムユニットにある場合、同じ名前を指定する COMMON 文を含むすべてのプログラムユニットで、最後の該当する COMMON 文の後に THREADPRIVATE ディレクティブがなければなりません。インテル® Fortran コンパイラーでは、この使用法が適切に診断されません。

例えば、次のプログラムは OpenMP* 4.5 仕様に準拠していませんが、ifort は OMP THREADPRIVATE 文に続く 2 つの COMMON 文に対してエラーメッセージを出力しません。

```
PROGRAM ex1
    COMMON /common_blk1/x
    !$OMP THREADPRIVATE(/common_blk1/)
    COMMON /common_blk1/y
    COMMON /common_blk1/z
END PROGRAM
```

先頭へ戻る

Fortran 2008 および Fortran 2015 機能の概要

インテル® Fortran コンパイラーは、Fortran 2008 標準規格の多くの機能と Proposed draft Fortran 2015 標準規格の一部の機能をサポートします。その他の機能は将来のリリースでサポートされる予定です。現在のコンパイラーでは、以下の Fortran 2008 機能がサポートされています。

- 配列の最大次元数が 31 次元に (Fortran 2008 では 15 次元)
- Co-Array
- CODIMENSION 属性
- SYNC ALL 文
- SYNC IMAGES 文
- SYNC MEMORY 文
- CRITICAL および END CRITICAL 文
- LOCK および UNLOCK 文
- ERROR STOP 文
- ALLOCATE および DEALLOCATE で Co-Array を指定
- 組込みプロシージャー: ATOMIC_DEFINE、ATOMIC_REF、IMAGE_INDEX、LCOBOUND、NUM_IMAGES、THIS_IMAGE、UCOBOUND
- CONTIGUOUS 属性
- ALLOCATE の MOLD キーワード
- DO CONCURRENT
- OPEN の NEWUNIT キーワード
- G0 および G0.d フォーマット編集記述子
- 無制限のフォーマット項目繰り返しカウント指定子
- CONTAINS セクションは空にすることも可能
- 組込みプロシージャー: BESSEL_J0、BESSEL_J1、BESSEL_JN、BESSEL_YN、BGE、BGT、BLE, BLT、DSHIFTL、DSHIFTR、ERF、ERFC、ERFC_SCALED、GAMMA、HYPOT、IALL、IANY、IPARITY、IS_CONTIGUOUS、LEADZ、LOG_GAMMA、MASKL、MASKR、MERGE_BITS、NORM2、PARITY、POPCNT、POPPAR、SHIFTA、SHIFTL、SHIFTR、STORAGE_SIZE、TRAILZ
- 組込みモジュール ISO_FORTRAN_ENV の追加: ATOMIC_INT_KIND、ATOMIC_LOGICAL_KIND、CHARACTER_KINDS、INTEGER_KINDS、INT8、INT16、INT32、INT64、LOCK_TYPE、LOGICAL_KINDS、REAL_KINDS、REAL32、REAL64、REAL128、STAT_LOCKED、STAT_LOCKED_OTHER_IMAGE、STAT_UNLOCKED
- ALLOCATABLE または POINTER 属性を持たない OPTIONAL 仮引数は、対応する実引数に ALLOCATABLE 属性があるのに割り当てられない場合、POINTER 属性があるのに関連付けが解除されている場合、または NULL 組込み関数への参照の場合、無視されます。
- 仮引数がプロシージャー・ポインターの場合、そのポインターの有効な参照先か、または組込み関数 NULL への参照である実引数に関連付けられます。実引数がポインターでない場合、仮引数に INTENT (IN) 属性が含まれていなければなりません。
- BLOCK 構造
- EXECUTE_COMMAND_LINE 組込みサブルーチン
- サブモジュール
- IMPURE
- 組込み型の型宣言
- ポインターの初期化
- 暗黙形状配列
- EXIT 文の構文名の拡張
- 内部プロシージャーの BIND(C) のサポート

現在のバージョンでは、次の Proposed draft Fortran 2015 の機能がサポートされています。

- 「Technical Specification 29113 Further Interoperability with C」のすべての機能。次の機能を含みます。
    - 型引き継ぎ (TYPE(*))
    - ランク引き継ぎ (DIMENSION(..))
    - 互換性のある仮引数の制約の緩和
    - Fortran で使用される C コード操作「C 記述子」を定義する C インクルード・ファイル ISO_Fortran_binding.H

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

本資料に掲載されている情報は、インテル製品の概要説明を目的としたものです。本資料は、明示されているか否かにかかわらず、また禁反言によるとよらずにかかわらず、いかなる知的財産権のライセンスも許諾するものではありません。製品に付属の売買契約書『Intel's Terms and Conditions of Sale』に規定されている場合を除き、インテルはいかなる責任を負うものではなく、またインテル製品の販売や使用に関する明示または黙示の保証 (特定目的への適合性、商品適格性、あらゆる特許権、著作権、その他知的財産権の非侵害性への保証を含む) に関してもいかなる責任も負いません。インテルによる書面での合意がない限り、インテル製品は、インテル製品の欠陥や故障によって人身事故が発生するような用途向けに使用することを前提としたものではありません。

インテル製品は、予告なく仕様や説明が変更されることがあります。機能または命令の一覧で「留保」または「未定義」と記されているものがありますが、その「機能が存在しない」あるいは「性質が留保付である」という状態を設計の前提にしないでください。これらの項目は、インテルが将来のために留保しているものです。インテルが将来これらの項目を定義したことにより、衝突が生じたり互換性が失われたりしても、インテルは一切責任を負いません。この情報は予告なく変更されることがあります。この情報だけに基づいて設計を最終的なものとしないでください。

本資料で説明されている製品には、エラッタと呼ばれる設計上の不具合が含まれている可能性があり、公表されている仕様とは異なる動作をする場合があります。現在確認済みのエラッタについては、インテルまでお問い合わせください。

最新の仕様をご希望の場合や製品をご注文の場合は、お近くのインテルの営業所または販売代理店にお問い合わせください。

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® Fortran 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。