インテル® Visual Fortran Composer XE 2013 SP1 Windows* 版インストール・ガイドおよびリリースノート

資料番号: 321417-005JA
2013 年 10 月 10 日

目次

1 概要

このドキュメントでは、製品のインストール方法、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。

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。各コンポーネントの新機能の詳細は、各コンポーネントのリリースノートを参照してください。

1.1 製品アップデート

Update 1 – 2013 年 10 月

- インテル® Visual Fortran コンパイラーが 14.0.1 にアップデート
  - /assume:std_value の追加
  - /Q[a]xMIC-AVX512 コンパイラー・オプションの追加
  - /Qopt-gather-scatter-unroll=n コンパイラー・オプションの追加
- インテル® マス・カーネル・ライブラリーが 11.1 Update 1 にアップデート
- Microsoft* Windows 8.1* をサポート

1.2 インテル® Fortran Composer XE 2013 からの変更点

- インテル® Fortran コンパイラーがバージョン 14.0 にアップデート
  - インテル® Xeon Phi™ コプロセッサーを使用するアプリケーションの開発をサポート
  - 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インテル® Debugger Extension
- インテル® マス・カーネル・ライブラリーがバージョン 11.1 にアップデート
- オプションで必要なコンポーネントのみダウンロードするオンライン・インストーラーを提供
- 報告された問題の修正

1.3 製品の内容

インテル® Visual Fortran Composer XE 2013 SP1 Windows 版* には、次のコンポーネントが含まれています。

- インテル® Visual Fortran コンパイラー XE 14.0。IA-32 およびインテル® 64 アーキテクチャー・システムで動作するアプリケーションをビルドします。
- インテル® マス・カーネル・ライブラリー11.1
- 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インテル® Debugger Extension
- Microsoft* Visual Studio* で Fortran アプリケーションをデバッグするための Fortran Expression Evaluator (FEE)
- Microsoft* 開発環境への統合
- Microsoft* Visual Studio* 2010 Shell とライブラリー (評価版ライセンスでは提供されません)
- サンプルプログラム
- 各種ドキュメント

1.4 動作環境

アーキテクチャー名についての説明は、「Intel® Architecture Platform Terminology」 (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 IA-32 またはインテル® 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 (インテ

ル® Pentium® 4 プロセッサー以降、または互換性のあるインテル以外のプロセッサー)
    - 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 1GB (2GB 推奨)
- 2GB のディスク空き容量 (すべての機能およびすべてのアーキテクチャー)
- Microsoft* Windows* XP SP3、Microsoft* Windows* 7、Microsoft* Windows* 8、Microsoft* Windows* 8.1、Microsoft* Windows Server* 2012、Microsoft* Windows Server* 2008、Microsoft* Windows* HPC Server 2008 (エンベデッド・エディションはサポートされていません)
    - Microsoft* Windows Server* 2008 または Windows HPC Server 2008 では、Microsoft* Visual Studio* 2012、Microsoft* Visual Studio* 2010、Microsoft* Visual Studio* 2010 Shell、Microsoft* Visual Studio* 2008 SP1、または Visual Studio* 2008 SP1 アップデートが適用された Visual Studio* 2008 Shell が必要です。
    - Microsoft* Windows* 8 では、製品は「デスクトップ」環境にインストールされます。「新しい 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。
- 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* 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)
    - Microsoft* Visual Studio* 2010 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* 2008 Standard Edition 以降 (C++ コンポーネントと [x64 コンパイラおよびツール] コンポーネントがインストールされていること) [1]
    - Microsoft* Visual Studio* 2010 Shell (インテル® Fortran コンパイラーの特定のライセンスに付属) ベースのインテル® Visual Fortran 開発環境 [2]
    - Microsoft* Visual Studio* 2008 Shell (インテル® Fortran コンパイラー 11.0、11.1、およびインテル® Visual Fortran Composer XE 2011 Update5 までに付属) ベースのインテル® Visual Fortran 開発環境
- IA-32 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 Studio* Express 2012 for Windows Desktop
    - Microsoft* Visual C++* 2010 Express Edition [3]
    - Microsoft* Visual C++* 2008 Express Edition
- 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 Studio* Express 2012 for Windows Desktop
    - Microsoft* Windows* Software Development Kit for Windows* 8.1
    - Microsoft* Windows* Software Development Kit for Windows* 8
- Microsoft* Visual Studio* 2010 Shell には次の制限があります。
    - Windows* XP 64 ビットではサポートされていません。Windows* XP 64 ビットでは、以前のバージョンのインテル® Visual Fortran に付属の Microsoft* Visual Studio* 2008 Shell を使用できます。
- Windows* XP では、Microsoft* Visual Studio* 2010 Shell をインストールする前に Microsoft* .NET 4.0 Framework をインストールする必要があります。詳細は、このリリースノートの「インストール」セクションを参照してください。
- ドキュメントの参照用に Adobe* Reader* 7.0 以降

注

1. Microsoft* Visual Studio* 2008 Standard Edition では、[x64 コンパイラおよびツール] コンポーネントがデフォルトでインストールされます。Professional 以上のエディションでは、[カスタム] インストールが必要です。Microsoft* Visual Studio* 2010 では、すべてのエディションでこのコンポーネントがデフォルトでインストールされます。
2. Microsoft* Visual Studio* 2010 Shell ベースのインテル® Visual Fortran 開発環境は、インテル® Visual Fortran Composer XE のアカデミック・ライセンスと商用ライセンスに含まれています。評価版ライセンスには含まれていません。この開発環境は、Fortran アプリケーションの編集、ビルド、デバッグに必要なものがすべて揃っています。ただし、次のような、Visual Studio* 製品の一部の機能は含まれていません。
    - リソースエディター (代用としてサードパーティー・ツールの ResEdit* (http://www.resedit.net/ (英語)) を参照してください。)
    - Compaq* Visual Fortran プロジェクトの自動変換
    - Visual C++* や Visual Basic* などの Microsoft* の言語ツール
3. Microsoft* Visual C++* 2010 Express Edition とインテル® Visual Fortran Composer XE 2013 に付属の Microsoft* Visual Studio* 2010 Shell は共存できます。C++ と Fortran 開発環境はそれぞれ独立しています。
4. インテル® Visual Fortran コンパイラーは、デフォルトで、インテル® SSE2 命令対応のプロセッサーが必要な IA-32 アーキテクチャー・アプリケーションをビルドします。コンパイラー・オプションを使用して任意の IA-32 アーキテクチャー・プロセッサー上で動作するコードを生成できます。ただし、インテル® MKL を呼び出すアプリケーションではインテル® SSE2 命令に対応しているプロセッサーが必要です。
5. 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 Windows* 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。

1.4.1 Windows XP* のサポート終了予定

インテル® Visual Fortran Composer XE の将来のメジャーバージョンでは、Windows* XP はサポートされなくなる予定です。

1.4.2 Microsoft* Visual Studio* 2008 のサポート終了予定

インテル® Visual Fortran Composer XE の将来のメジャーバージョンでは、Microsoft* Visual Studio* 2008 はサポートされなくなる予定です。早めに Microsoft* Visual Studio* 2012 以上へ移行することを推奨します。

1.5 ドキュメント

製品ドキュメントは、「インストール先フォルダー」で示されているように、`Documentation` フォルダーに保存されています。

1.5.1 「インテル® Visual Fortran を使用した Windows* ベースのアプリケーションの作成とビルド」は Web から入手可能

以前のコンパイラー・ドキュメントに含まれていた「インテル® Visual Fortran を使用した Windows* ベースのアプリケーションの作成とビルド」は、Intel® Software Documentation Library Web サイトに移動しました。「Using Intel® Visual Fortran to Create and Build Windows-based Applications」 (PDF) を参照してください。

1.5.2 Visual Studio* 2008 のインテル® Composer XE ヘルプに最初にアクセスしたときの遅延

Microsoft* Visual Studio* 2008 にインストールされたヘルプ・ドキュメントに最初にアクセスしたとき、表示に時間がかかる場合があります。これは、Visual Studio* でヘルプを表示する前に、新しいヘルプをコレクションに統合して、コレクションの索引を再生成するためです。Visual Studio* にすでに統合されているヘルプとインストールするヘルプのサイズに応じて、新しくヘルプが表示されるまで数分以上かかる場合があります。

1.5.3 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。

1.6 最適化に関する注意事項

最適化に関する注意事項
インテル® コンパイラーは、互換マイクロプロセッサー向けには、インテル製マイクロプロセッサー向けと同等レベルの最適化が行われない可能性があります。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® SSE2)、インテル® ストリーミング SIMD 拡張命令 3 (インテル® SSE3)、ストリーミング SIMD 拡張命令 3 補足命令 (SSSE3) 命令セットに関連する最適化およびその他の最適化が含まれます。インテルでは、インテル製ではないマイクロプロセッサーに対して、最適化の提供、機能、効果を保証していません。本製品のマイクロプロセッサー固有の最適化は、インテル製マイクロプロセッサーでの使用を目的としています。インテル® マイクロアーキテクチャーに非固有の特定の最適化は、インテル製マイクロプロセッサー向けに予約されています。この注意事項の適用対象である特定の命令セットの詳細は、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。 改訂 #20110804

1.7 サンプル

製品コンポーネントのサンプルは、「インストール先フォルダー」の説明にある `Samples` フォルダーに用意されています。

1.8 日本語サポート

インテル® コンパイラーは、日本語と英語の両方を備えたインストーラーで日本語をサポートしています。エラーメッセージ、ビジュアル開発環境ダイアログ、ドキュメントの一部が英語のほかに日本語でも提供されています。エラーメッセージやダイアログの言語は、システムの言語設定に依存します。日本語版ドキュメントは、`Documentation` および `Samples` ディレクトリー以下の `ja_JP` サブディレクトリーにあります。

日本語サポートはすべての製品アップデートで提供されているわけではありません。

日本語版を英語のオペレーティング・システムで使用する場合や日本語のオペレーティング・システムで英語版を使用する場合は、「Changing Language Setting to see English on a Japanese OS Environment or Vice Versa on Windows」 (英語) の説明を参照してください。

1.9 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: 代理店がテクニカルサポートを提供している場合は、インテルではなく代理店にお問い合わせください。

2 インストール

2.1 インストール前の準備

2.1.1 インストールに必要なソフトウェア

製品に付属する Microsoft* Visual Studio* 2010 Shell をインストールする場合、インテル® Visual Fortran Composer XE 2013 SP1 をインストールする前に、追加で Microsoft* ソフトウェアのインストールが必要な場合があります。Microsoft* Visual Studio* 2010 Shell は、Windows* XP 64 ビットではサポートされていません。

Microsoft* Visual Studio* 2010 Shell のインストールには Microsoft* .NET 4.0 Framework が必要です。Microsoft* .NET 4.0 Framework は、次のリンクからインストーラーをダウンロードできます。

- .NET 4.0 Framework 32 ビットおよび 64 ビット

Windows* 8.1、Windows* 8、Windows* 7 または Windows Server* 2008 にインストールする場合は、Shell のインストール時に .NET Framework 4.0 が (システムにインストールされていない場合) 自動的にダウンロードされインストールされます。この処理に失敗すると、エラーメッセージが表示され、インテル® Visual Fortran Composer はインストールされません。Shell のインストールに失敗した場合は、上記のリンクから .NET 4.0 Framework をダウンロードしてやり直してください。

DVD または Visual Studio* 2010 Shell 付属製品のダウンロード版を使用してインテル® Visual Fortran Composer XE 2013 SP1 をインストールする場合、マシンに Visual Studio* 2010 がインストールされていないと、インストーラーは Visual Studio* 2010 Shell をインストールしようとします。Visual Studio* 2010 Shell をインストールしない場合は、「カスタム・インストール」を選択して、Visual Studio* 2010 Shell のチェックをオフにするか、"_novsshell.exe" (Visual Studio* 2010 Shell なし) のインストーラーをダウンロードして使用してください。

2.1.2 64 ビット・アプリケーション用の Visual Studio* の設定

Microsoft* Visual Studio* 2008 を使用し、64 ビット・アプリケーション (インテル® 64 アーキテクチャー向け) を開発する場合は、Visual Studio* の構成を変更して、64 ビット・サポートを追加します。

Visual Studio* 2008 Standard Edition または Visual Studio* 2008 Shell を使用する場合は、インテル® 64 対応アプリケーションのビルド用に構成を変更する必要はありません。その他のエディションの場合は、次の操作を行ってください。

1. [コントロール パネル] の [プログラムの追加と削除] から [Microsoft Visual Studio 2008] を選択し、[変更と削除] をクリックします。[Visual Studio メンテナンス モード] ウィンドウが表示されます。[次へ] をクリックします。
2. [機能の追加と削除] をクリックします。
3. [選択した機能をインストールします] で [言語ツール] の [Visual C++] を展開します。
4. [x64 コンパイラおよびツール] ボックスがオンになっていない場合は、オンにし、[更新] をクリックします。ボックスがオンの場合は、[キャンセル] をクリックします。

Microsoft* Visual Studio* 2010 または Visual Studio* 2012 を使用している場合、このステップは必要ありません。

2.2 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) のインストール

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® Visual Fortran Composer XE 2013 SP1 Windows* 版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。

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。

2.3 オンライン・インストーラー

デフォルトのダウンロード版インストール・パッケージが、サイズの小さいオンライン・インストーラーになりました。オンライン・インストーラーは、選択したパッケージを動的にダウンロードし、インストールします。このインストール・パッケージを利用するには、インターネット接続が必要です。また、インターネット・プロキシーを使用している場合は、プロキシーの設定が必要になることがあります。インターネットに接続されていないマシンにインストールする場合は、オンライン・パッケージではなくフル・パッケージを利用してください。

2.4 インストール

本製品のインストールには、有効なライセンスファイルまたはシリアル番号が必要です。本製品を評価する場合には、インストール時に [製品を評価する (シリアル番号不要)] オプションを選択してください。

DVD で製品を受け取った場合、製品 DVD を DVD ドライブに挿入します。自動でインストールが開始されます。自動で開始されない場合は、Windows* エクスプローラーで DVD ドライブのトップレベル・ディレクトリーを開き、setup.exe をダブルクリックします。

製品のダウンロード版を購入した場合は、ダウンロードしたファイル (.EXE) をダブルクリックして、インストールを開始します。利用可能なダウンロード・ファイルには各種あり、それぞれ異なるコンポーネントの組み合わせを提供していることに注意してください。ダウンロード・ページを注意深くお読みになり、適切なファイルを選択してください。

新しいバージョンをインストールする前に古いバージョンをアンインストールする必要はありません。新しいバージョンは古いバージョンと共存可能です。以前のバージョンの削除は、このバージョンをインストールする前でも後でも行うことができます。

インテル® ソフトウェア開発製品レジストレーション・センターでシリアル番号を登録すると、製品のアップデートや以前のバージョンを利用できます。

2.4.1 インストール後の再起動を推奨

インテル® Visual Fortran Composer XE をインストールすると、(インテル® MKL を除く) コンパイラーのランタイム DLL が含まれるフォルダー名が PATH 環境変数に追加されます。一部のシステムでは、PATH の長さが非常に長くなると (2048-4096 文字)、システムを再起動するまでコマンドラインが動作しなくなることがあります。インテル® Visual Fortran Composer XE をインストールした後は、システムを再起動することを推奨します。

2.4.2 クラスターでのインストール

インストールするマシンに Microsoft* Compute Cluster Pack のライセンスがあり、クラスターメンバーの場合、「フル・インストール」を選択すると、そのクラスターのアクセス可能なすべてのノードに製品がインストールされます。「カスタム・インストール」を選択すると、現在のノードのみにインストールするオプションを選択できます。

2.4.3 ライセンスサーバーの使用

「フローティング・ライセンス」を購入された場合は、「Licensing: Setting Up the Client for a Floating License」 (英語) を参照してください。この記事には、多様なシステムにインストールすることができる FLEXlm* ライセンス・マネージャーに関する情報も記述されています。

2.4.4 Microsoft* Visual Studio* 2010 用ドキュメントをインストールするための追加ステップ

Microsoft* Visual Studio* 2010 がインストールされているシステムにインテル® Visual Fortran Composer XE 2013 SP1 を初めてインストールするとき、Visual Studio* 2010 のドキュメントの「ローカルストア」を初期化するかどうか確認するメッセージが表示されます (初期化を行っていない場合)。「ヘルプ ライブラリ マネージャー」によってインテル® Visual Fortran Composer XE 2013 SP1 ヘルプ・ドキュメントが Visual Studio* 2010 内に登録されます。「ヘルプ ライブラリ マネージャー」のインストール・ウィザードの説明に従って、Visual Studio* 2010 用のインテル® Visual Fortran Composer XE 2013 SP1 ヘルプ・ドキュメントをインストールします。

このステップは 1 回のみ実行する必要があります。将来インテル® Visual Fortran Composer XE 2013 SP1 のアップデートをインストールするときに、「ヘルプ ライブラリ マネージャー」を使用してドキュメントを再登録する必要はありません。

詳細は、http://msdn.microsoft.com/en-us/library/dd264831.aspx を参照するか、microsoft.com で「ヘルプ ライブラリ マネージャー」を検索してください。

2.5 インテル® Software Manager

インテル® Software Manager は、製品アップデートの配信方法を簡素化し、現在インストールされているすべてのインテル® ソフトウェア製品のライセンス情報とステータスを表示します。

将来の製品設計の参考のため、製品使用状況に関する匿名情報をインテルに提供する、インテル® ソフトウェア向上プログラムに参加できます。このプログラムは、デフォルトで無効になっていますが、インストール中または後から有効にして参加できます。参加はいつでも取りやめることができます。詳細は、「Intel® Software Improvement Program」 (英語) を参照してください。

2.6 製品の変更、更新、削除

Windows* のコントロールパネルの [プログラムの追加と削除] / [プログラムと機能] でインストールまたは削除する製品コンポーネントを変更します。インストールした製品に応じて、以下のいずれかのエントリーが表示されます。

- インテル® Visual Fortran Composer XE 2013 SP1 Windows* 版
- インテル® Composer XE 2013 SP1 Windows* 版
- インテル® Parallel Studio XE 2013 SP1 Windows* 版

コンパイラーのインストールの一部として Microsoft* Visual Studio* 2010 Shell をインストールした場合、以下の追加エントリーが表示されます。

- Microsoft* Visual Studio* 2010 Shell (Integrated) - JPN
- インテル(R) Visual Fortran 用 Microsoft* Visual Studio* 2010 ファイル
- Microsoft Visual Studio 2010 Remote Debugger – JPN

製品を完全に削除する場合を除き、これらのエントリーは削除しないでください。

製品のアップデート・バージョンをインストールする際、古いバージョンを最初にアンインストールする必要はありません。アップデートを最初にインストールする場合、古いバージョンを置換するか、システムで古いバージョンと新しいバージョンの両方を使用するかを選択します。この選択は、将来のアップデートにも適用されます。Microsoft* Visual Studio* の [ツール] > [オプション] > [Intel Composer XE (インテル(R) Composer XE)] > [Visual Fortran (インテル(R) Visual Fortran)] > [Compiler (コンパイラー)] ダイアログから、使用するコンパイラーのバージョンを選択できます。バージョン 12.0 (インテル® Visual Fortran Composer XE 2011) よりも古いコンパイラーは、Visual Studio* で選択できません。インストールされているすべてのバージョンをコマンドラインから使用できます。

新しいバージョンのコンパイラーを削除した場合、以前のバージョンの Microsoft Visual Studio への統合を再インストールする必要があります。

2.7 サイレント・インストール/アンインストール

コンパイラーの自動インストール/アンインストールについては、「Intel® Compilers for Windows* Silent Installation Guide」 (英語) を参照してください。

2.8 インストール先フォルダー

インストール・フォルダーの構成を以下に示します。一部含まれていないフォルダーもあります。システム環境変数 `IFORT_COMPILER14` を使用して、マシンにインストールされている最新バージョンのインテル® Visual Fortran Composer XE 2013 SP1 を検出できます。

- `C:\Program Files\Intel\Composer XE 2013 SP1`
  - `bin`
    - `ia32`
    - `ia32_intel64`
    - `intel64`
    - `intel64_mic`
    - `sourcechecker`
  - `compiler`
    - `include`
      - `ia32`
      - `intel64`
      - `mic`
    - `lib`
      - `ia32`
      - `intel64`
      - `mic`
  - `debugger`
  - `Documentation`
  - `Help`

- mkl
  - benchmarks
  - bin
  - examples
  - include
  - interfaces
  - lib
  - tests
  - tools
- redist
- Samples
- setup_x_xxx

bin、include および lib 配下のフォルダーは次のとおりです。

- ia32: IA-32 上で動作するアプリケーションのビルドに使用するファイル
- intel64: インテル® 64 上で動作するアプリケーションのビルドに使用するファイル
- ia32_intel64: IA-32 上での実行用のコンパイラー。インテル®64 上で動作するアプリケーションをビルドします。

英語以外の Windows* システムにインストールする場合、Program Files フォルダー名が異なる場合があります。インテル® 64 アーキテクチャー・システムでは、フォルダー名は Program Files (X86) またはそれに相当する名前です。

デフォルトでは、アップデートによって既存のディレクトリーの内容が置換されます。最初のアップデートをインストールするときに、以前のインストールとは別に新しいアップデートをインストールして、システムに両方のファイルを残すオプションを選択できます。両方を残すオプションを選択した場合、古いアップデートのトップレベルのフォルダー名は Composer XE 2013 SP1.*nnn* (*nnn* はアップデート番号) に変更されます。

3 インテル® Visual Fortran コンパイラー

このセクションでは、インテル® Visual Fortran コンパイラーの変更点、新機能、および最新情報をまとめています。

3.1 互換性

一般に、インテル® Fortran コンパイラー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 14.0 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を使用してビルドされた 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 (/Qipo) オプションを使用してビルドされた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 12.0 よりも前のコンパイラーを使用してビルドされた 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を使用してインテル® 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を使用してコンパイルされた、派生型宣言の外部で ATTRIBUTES ALIGN 宣言子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。この問題が発生した場合、問題を通知するメッセージが表示されます。
- 派生型宣言の内部で ATTRIBUTES ALIGN 宣言子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。

3.1.1 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 (12.0)

以前のリリースでは、REAL(16) または COMPLEX(16) (REAL*16 または COMPLEX*32) 項目が値で渡されたとき、スタックアドレスは 4 バイトでアラインされていました。パフォーマンスを向上させるため、バージョン 12.0 (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ます。引数は 16 バイト境界でアラインされます。

この変更は、主にライブラリーが生成した REAL(16) 値の計算を行うライブラリー (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 13 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。

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。

3.1.2 インテル® OpenMP* ライブラリーのスタティック・ライブラリーの提供を終了

インテル® OpenMP* ライブラリーのスタティック・ライブラリー libiomp5mt.lib の提供が終了し、`/Qopenmp-link:static` コマンドライン・オプションがサポートされなくなりました。libiomp5mt.lib に対するすべての参照を、DLL インポート・ライブラリー libiomp5md.lib に変更してください。この変更に伴い、OpenMP* を使用するアプリケーションを、インテル® コンパイラーが存在しないシステムに配布する場合、インテル® コンパイラーの再配布可能コードのインストールが必要になることがあります。詳細は、「Redistributable Libraries for Intel® Visual Fortran Composer XE」 (英語) を参照してください。

3.1.3 Fortran Expression Evaluator

Fortran Expression Evaluator (FEE) は、インテル® Visual Fortran コンパイラーとともにインストールされる Microsoft* Visual Studio* のプラグインです。Fortran コードを処理できるように、Microsoft* Visual Studio* IDE の標準デバッガーを拡張します。その他は標準デバッガーと同じです。

3.2 新規および変更されたコンパイラー機能

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 2003 規格 (PDF (英語)) および Fortran 2008 規格 (PDF (英語)) を参照してください。

3.2.1 Fortran 2003 の機能

- ユーザー定義の派生型 I/O

3.2.2 OpenMP* 機能

OpenMP* 4.0 の次の宣言子、節、およびプロシージャーがコンパイラーでサポートされます。これらの機能の一部は、暫定仕様に基づきインテル® Fortran Composer XE 2013 Update 2 でサポートされました。また、以前サポートされていたいくつかのキーワード (DECLARE TARGET MIRROR, DECLARE TARGET LINKABLE, MAPTO, MAPFROM, SCRATCH) はサポートされなくなりました。さらに、一部の構文は以前の仕様から変更されています。

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。

SIMD 宣言子:

- OMP SIMD
- OMP DECLARE SIMD
- OMP DO SIMD
- OMP PARALLEL DO SIMD

コプロセッサー宣言子:

- OMP TARGET DATA
- OMP TARGET
- OMP TARGET UPDATE
- OMP DECLARE TARGET

その他の宣言子:

- OMP PARALLEL PROC_BIND
- OMP TASKGROUP

節:

- MAP

プロシージャー:

- OMP_GET_DEVICE_NUM
- OMP_GET_PROC_BIND
- OMP_SET_DEVICE_NUM

3.2.2.1 KMP_PLACE_THREADS 環境変数 (13.1.0)

この環境変数を使用すると、ユーザーは明示的なアフィニティー設定やプロセス・アフィニティー・マスクを記述する代わりに、OpenMP* アプリケーションで使用するコア数およびコアごとのスレッド数を簡単に指定することができます。

3.2.3 新しい宣言子と追加された宣言子

インテル® Composer XE 2013 SP1 では、次のコンパイラー宣言子が追加、変更されています。詳細は、ドキュメントを参照してください。

- [NO]FMA

3.2.4 その他の機能

これらの機能に関する詳細は、コンパイラー・ドキュメントを参照してください。

- ESTABLISHQQ ライブラリー・ルーチンは、Fortran ランタイム・ライブラリーがランタイムエラーを出力する直前にユーザールーチンを呼び出すように指定します。このルーチンは、モジュール IFPORT で宣言されています。
- `-[no-]wrap-margin` コマンドライン・オプションと FORT_FMT_NO_WRAP_MARGIN 環境変数は、リスト指定出力で前のレコードが 80 文字を超える場合、新しいレコードを開始するかどうかを制御します。
- 新しい事前定義済みシンボル __INTEL_COMPILER_UPDATE、__INTEL_OFFLOAD、__MIC__

- 新しい環境変数 FOR_FORCE_STACK_TRACE。1 に設定した場合、実行時に診断メッセージが出力されると、コンパイラーはトレースバックを提供します。FOR_FORCE_STACK_TRACE は、FOR_DISABLE_STACK_TRACE よりも優先されます。

3.2.5 Co-Array (13.0)

共有メモリー環境で Co-Array を使用するプログラムの実行に特別なプロシージャーは必要ありません。実行ファイルを実行するだけでかまいません。根本的な並列化の実装にはインテル® MPI が使用されます。コンパイラーをインストールすると、共有メモリーでの実行に必要なインテル® MPI ランタイム・ライブラリーが自動的にインストールされます。

`/coarray:distributed` オプションを使用するには、インテル® Cluster Studio のライセンスが必要です。Windows* 上で分散 Co-Array アプリケーションを実行する方法については、「Windows* 環境での分散 Co-Array アプリケーションのビルドと実行」を参照してください。

現在、インテル® MPI 以外の MPI 実装や OpenMP* を使用した Co-Array アプリケーションの使用はサポートされていません。

デフォルトでは、作成されるイメージの数は現在のシステムの実行ユニットの数と同じです。メインプログラムをコンパイルする ifort コマンドで `/Qcoarray-num-images:<n>` オプションを指定することで、この設定を変更することができます。また、環境変数 `FOR_COARRAY_NUM_IMAGES` でイメージ数を指定することもできます。

3.2.6 派生型のコンポーネントでの ATTRIBUTES ALIGN 宣言子の指定 (13.0.1)

コンパイラー 13.0.1 では、派生型の ALLOCATABLE または POINTER コンポーネントに ATTRIBUTES ALIGN 宣言子が指定されます。宣言子は派生型宣言内に配置しなければなりません。拡張型の場合、宣言子は親の型のコンポーネントを指定してはなりません。

この宣言子が指定されると、コンパイラーは明示的な ALLOCATE または (ALLOCATABLE コンポーネントに対する) Fortran 言語規則に従った暗黙の割り当てによりコンポーネントが割り当てられたときに指定されたアライメントを適用します。

派生型コンポーネントに ATTRIBUTES ALIGN 宣言子を含むモジュールはバージョン 13.0.1 よりも前のコンパイラーで使用できません。

3.2.7 ファイル・バッファリング動作の変更 (13.1)

インテル® Visual Fortran Composer XE 2013 (コンパイラー 13.0) 以前のバージョンでは、Fortran ランタイム・ライブラリーは、可変長の書式なしシーケンシャル・ファイルのレコードを読み取るときにすべての入力をバッファリングしていました。このデフォルトのバッファリングは、任意のサイズの可変長レコードをメモリーに保持できる大きな内部バッファーを割り当てることで行われます。非常に大きなレコードの場合、メモリーが過度に使用され、最悪の場合は利用可能なメモリーを使い果たす可能性があります。しかし、レコードを読み取るときのデフォルトのバッファリング動作を変更する方法は用意されていませんでした (レコードを書き込むときにレコードのバッファリングを要求または拒否することは可能でした)。

このデフォルトのバッファリング動作は、インテル® Visual Fortran Composer XE 2013 で変更され、これらのレコードはすべてデフォルトではバッファリング**されず**、ディスクからユーザープログラムの変数に直接読み込まれるようになりました。この変更はメモリーを確保する必要があるプログラムを支援するために行われたものですが、多くの小さなコンポーネントで構成されているレコードを読み取るときにパフォーマンスが低下する場合があります。実際、一部のユーザーから、パフォーマンスの低下が報告されました。

このため、インテル® Visual Fortran Composer XE 2013 Update 2 (コンパイラー 13.1) では、ユーザーがこれらの可変長書式なしレコードをバッファリングするかどうかを選択できる手

法が提供されました。デフォルトの動作は 13.0 と同じで、これらのレコードはデフォルトではバッファリングされません。13.1 でこの種の I/O を使用したときにパフォーマンスが低下する場合は、レコードの出力のバッファリングを有効にするかどうかを選択する場合と同じ方法で入力のバッファリングを有効にすることができます。

- ファイルの OPEN 文で BUFFERED="YES" を指定する
- 環境変数 FORT_BUFFERED に YES、TRUE、またはゼロ以外の整数値を指定する
- コンパイラーのコマンドラインで `-assume buffered_io` を指定する

これらの手法は、これまで、可変長書式なしシーケンシャル・ファイルの書き込みを行う場合にのみ適用されていたものです。これらの手法を使用すると、Fortran ランタイム・ライブラリーは、ファイルのレコードのサイズに関係なく、ファイルの入力レコードをすべてバッファリングします。

つまり、13.0 より前のデフォルトの動作に戻ることになります。

3.3 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。

3.3.1 インテル® Composer XE 2013 SP1 の新規および変更されたコンパイラー・オプション

- /assume:std_value (14.0.1)
- /Q[a]xMIC-AVX512 (14.0.1)
- /Qfma
- /Qimf-domain-exclusion
- /Qmic
- /Qoffload
- /Qoffload-attribute-target
- /Qoffload-option
- /Qopenmp-offload
- /Qopenmp-simd
- /Qopt-assume-safe-padding
- /Qopt-gather-scatter-unroll=n (14.0.1)
- /Qopt-prefetch-distance
- /Qopt-streaming-cache-evict
- /Qopt-threads-per-core
- /Qvecabi
- /QxATOM_SSE4.2
- /wrap-margin

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。

3.3.1.1 Fortran 2003 VALUE 属性に影響する新しいオプション

インテル® Fortran コンパイラーの Fortran 2003 VALUE 属性の実装は、BIND(C) 属性が指定されていないプロシージャーで使用される場合、Fortran 2003 標準規格の動作とは異なります。インテル® Fortran コンパイラーのデフォルトの動作では、Fortran 2003 VALUE 属性を DEC$ ATTRIBUTES VALUE 宣言子と同様に扱い、引数は「値渡し」されます。一方、Fortran 2003 標準規格では、引数の再定義可能なコピーが渡されます。また、この違いにより、OPTIONAL 属性と VALUE 属性を一緒に使用することができません。プロシージャーで BIND(C) 属性が指定されている場合は、インテル® Fortran コンパイラーの動作は標準規格と同じで、VALUE 属性は値渡しされます。

インテル® Fortran コンパイラー 14 では、Fortran 2003 標準規格と同じ動作になるように設定できますが、これは以前の実装を想定する既存のアプリケーションで問題が発生する可能性があるため、デフォルトでは有効になりません。標準規格と同じ動作にするには、/assume:std_value (Windows*) または -assume std_value (Linux* および OS X*) コンパイラー・オプションを追加します。このオプションはドキュメントに記載されていません。Windows* で Visual Studio* を使用する場合、このオプションはプロジェクト・プロパティーの [Fortran] > [コマンドライン] > [追加のオプション] で指定できます。/standard-semantics (Windows*) または -standard-semantics (Linux* および OS X*) が指定されると、/assume:std_value (または -assume std_value) も指定されます。

インテル® Fortran コンパイラーの将来のメジャーバージョンでは、VALUE 属性のデフォルトの動作が標準規格と同じになるように変更される可能性があります。

3.3.1.2 新しい /Q[a]xMIC-AVX512 コンパイラー・オプション (14.0.1)

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) 命令対応のインテル® プロセッサー向けに最適化します。このオプションを指定すると、インテル® プロセッサー向けのインテル® AVX-512 の基本命令、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 CORE-AVX2 で有効になる命令を生成します。

3.3.1.3 新しい -opt-gather-scatter-unroll=n コンパイラー・オプション (14.0.1)

このオプションを使用して、インテル® MIC アーキテクチャーの集約 (Gather) と分散 (Scatter) ループに対して別のループ・アンロール・シーケンスを指定し、集約 (Gather)/分散 (Scatter) 処理のパフォーマンスを向上できる可能性があります。このオプションは、インテル® MIC アーキテクチャーにのみ適用されます。

3.4 Visual Studio* 統合の変更点

3.4.1 新しいプロジェクトでは DLL ライブラリーがデフォルト

インテル® Visual Fortran Composer XE 2013 SP1 をインストールした後に Fortran プロジェクトを新規作成すると、DLL 形式のランタイム・ライブラリーを使用するようにプロジェクト・プロパティーが設定されます。これは、Microsoft* Visual C++* の動作と同じですが、インテル® Visual Fortran Composer XE の以前のバージョンの動作とは異なります。スタティック・ライブラリーを使用する場合は、プロジェクト・プロパティーの [Fortran] > [Libraries (ライブラリー)] > [Runtime Library (ランタイム・ライブラリー)] で変更します。OpenMP* ライブラリー libiomp5md.dll は DLL 形式でのみ提供され、アプリケーションで OpenMP* を使用する場合は、どちらの設定を選択してもこの DLL が使用されます。

3.4.2 並列ビルドオプション (13.1)

Visual Studio* ビルド環境に、マルチコアまたはマルチプロセッサー・システムで未解決の依存性がないソースを並列ビルドできる機能が追加されました。この機能を利用すると、大規模なプロジェクトのビルドに必要な時間を短縮できます。

この機能を有効にするには、プロジェクトのプロパティー・ページを開いて、[Fortran] > [General (全般)] > [Multi-processor Compilation (マルチプロセッサーのコンパイル)] で [Yes (はい)] を選択します。

3.5 既知の問題

3.5.1 日本語ファイル名に関するコマンドライン診断表示の問題

コンパイル診断で日本語が含まれているファイル名は、ネイティブのインテル® 64 対応アプリケーション用コンパイラーを使用して、Windows* コマンドでコンパイルした場合に正しく表示されません。Visual Studio* を使用する場合やインテル® 64 対応アプリケーション用

クロスコンパイラーまたは IA-32 対応アプリケーション用コンパイラーを使用する場合は、この問題は発生しません。

3.5.2 Microsoft* Visual Studio* 2012 のみの環境でデバッグに失敗する問題

Microsoft* Visual Studio* 2012 のみがインストールされている Microsoft* Windows* システムでは、Fortran アプリケーションのデバッグに失敗することがあります。ウォッチ (式の評価) や条件付きブレークポイントなどに失敗します。

インテル® Visual Fortran Composer XE 2013 (SP1) は、Fortran アプリケーションをデバッグできるようにするため、Fortran Expression Evaluator (FEE) と呼ばれるデバッガー拡張を提供しています。一部の FEE 機能には、Microsoft* Visual Studio* 2010 ライブラリーが必要です。

1 つの方法として、Microsoft* Visual Studio* 2012 に加えて、Microsoft* Visual Studio* 2010 をインストールすることができます。別の方法として、次の Web サイトから Microsoft* Visual C++* 2010 SP1 再頒布可能パッケージ (x86) をダウンロードしてインストールできます。http://www.microsoft.com/en-us/download/details.aspx?id=8328

3.6 Microsoft* Visual Studio* 2010/2012 に関する注意事項

Microsoft* Visual Studio* 2010 ではいくつかの変更があります。そのほとんどは、メインプログラムが C/C++ の言語が混在したアプリケーションのビルドに影響するものです。これらの変更は、Visual Studio* 2012 にも適用されます。

3.6.1 インテル® Fortran ランタイム・ライブラリーを参照するための Microsoft* Visual C++* の設定

以前のリリースでは、インテル® Fortran の LIB フォルダーを C/C++ プロジェクトで利用できるようにするために [ツール] > [オプション] > [プロジェクトおよびソリューション] > [Visual C++ ディレクトリ] で設定を行っていました。Visual Studio* 2010 では、この方法が変更されています。

1. Visual Studio* で C++ プロジェクトを含むソリューションを開き、[表示] > [プロパティ マネージャー] を選択します。[表示] メニューの直下に [プロパティ マネージャー] が見つからない場合は、[表示] > [その他のウィンドウ] の下にあります。[プロパティ マネージャー] ダイアログボックスが表示されます。これは、[プロパティ] ウィンドウや [プロパティ ページ] とは関係ありません。
2. プロパティー・ツリーの Debug | Win32 の横にある三角または + 記号をクリックしてこのフォルダーを展開します。
3. Microsoft.Cpp.Win32.user をダブルクリックします。
4. [VC++ ディレクトリ] を選択します。
5. [ライブラリ ディレクトリ] の右側のフィールドをクリックします。
6. ドロップダウンから <編集...> を選択します。
7. [新しい行] ボタンをクリックするか、Ctrl+Insert キーを押します。
8. 表示された新しいフィールドに、次のように入力します。

```
$(IFORT_COMPILER14)\compiler\lib\ia32
```

9. [OK] をクリックします。もう一度 [OK] をクリックして、[プロパティ ページ] も閉じます。
10. Visual Studio* のメニューから [ファイル] > [すべてを保存] を選択します。

インテル® 64 (x64) 構成でビルドする場合は、次の手順を実行してください。

1. [プロパティ マネージャー] を開いて、Debug | x64 フォルダーを展開します。
2. Microsoft.Cpp.x64.user をダブルクリックします。

3. [VC++ ディレクトリ] を選択します。
4. [ライブラリ ディレクトリ] の右側のフィールドをクリックします。
5. ドロップダウンから <編集...> を選択します。
6. [新しい行] ボタンをクリックするか、Ctrl+Insert キーを押します。
7. 表示された新しいフィールドに、次のように入力します。

```
$(IFORT_COMPILER14)\compiler\lib\intel64
```

8. [OK] をクリックします。もう一度 [OK] をクリックして、[プロパティ ページ] も閉じます。
9. Visual Studio* のメニューから [ファイル] > [すべてを保存] を選択します。

[ソリューション エクスプローラー] タブをクリックするか、Ctrl+Alt+L キーを押して [ソリューション エクスプローラー] を表示します。

Debug | x64 フォルダーに Microsoft.Cpp.x64.user プロパティー・ページが見つからない場合は、フォルダーを右クリックして [新しいプロジェクト プロパティ シートの追加] を選択します。そして、MsBuild 4.0 プロパティー・ページの場所を参照します。Windows* XP では、通常以下の場所にあります。

```
C:\Documents and Settings\<username>\Local Settings\Application Data
\Microsoft\MSBuild\v4.0
```

Windows* 7 および Windows* 8 では、通常以下の場所にあります。

```
C:\Users\<username>\AppData\Local\Microsoft\MSBuild\v4.0
```

これらのパスを表示するためには、隠しファイルと隠しフォルダーの表示を有効にする必要があります。

Microsoft.Cpp.x64.user.props を選択して [開く] をクリックします。後は、上記の手順に従ってください。

3.6.2 プロジェクトの依存関係の調整

以前のバージョンの Visual Studio* から依存関係が設定されているプロジェクトを変換する場合、既存のプロジェクトの依存関係は Visual Studio* 2010/2012 によって参照に変換されます。C/C++ プロジェクトで Fortran プロジェクトを参照している場合、C/C++ プロジェクトのビルドで MSB4075 エラーが発生することがあります。この問題を解決するには、次の操作を行います。

1. C/C++ プロジェクトを右クリックして、[参照] を選択します。
2. 参照リストに Fortran プロジェクトがある場合は、プロジェクトを選択してから [参照の削除] をクリックします。参照リストにあるすべての Fortran プロジェクトに対してこの操作を行います。[OK] をクリックします。
3. ほかの C/C++ プロジェクトでも上記の手順を実行します。

これにより、プロジェクトの依存関係が更新されます。

1. C/C++ プロジェクトを右クリックして、[プロジェクトの依存関係] を選択します。
2. このプロジェクトと依存関係のあるプロジェクトのチェックボックスをすべてオンにします。
3. [OK] をクリックします。
4. 依存関係のあるほかの C/C++ プロジェクトでも上記の手順を実行します。

以前のバージョンの Visual Studio* とは異なり、Visual Studio* 2010/2012 は依存関係のあるプロジェクトの出力ライブラリーを自動でリンクしません。そのため、親プロジェクトのプロパティー・ページで [Linker (リンカー)] > [Additional Directories (追加のライブラリー・ディレクトリー)] からこれらのライブラリーを明示的に追加する必要があります。必要に応じて、Visual Studio* のマクロである $(ConfigurationName) と $(PlatformName) を使用してパスを指定することができます。次に例を示します。

```
..\FLIB\$(ConfigurationName)\FLIB.lib
```

$(ConfigurationName) は Release または Debug に置換されます。同様に、$(PlatformName) は Win32* または x64 に置換されます。

3.6.3 Windows Server* 2012 で Visual Studio* 2012 のドキュメントを表示できない場合

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。

3.7 Fortran 2003 および Fortran 2008 機能の概要

インテル® Fortran コンパイラーは、Fortran 2003 の多くの機能をサポートしています。現在サポートしていない Fortran 2003 機能についても、今後サポートしていく予定です。現在のコンパイラーでは、以下の Fortran 2003 機能がサポートされています。

- Fortran 文字セットが次の 8 ビット ASCII 文字を含むように拡張: ~ \ [ ] ` ^ { } | # @
- 最大長 63 文字までの名前
- 最大 256 行の文
- 角括弧 [ ] を (/ /) の代わりに配列の区切り文字として使用可能
- コンポーネント名とデフォルト初期化を含む構造コンストラクター
- 型と文字列長仕様を含む配列コンストラクター
- 名前付き PARAMETER 定数は複素定数の一部
- 列挙子
- 割り当て可能な派生型のコンポーネント
- 割り当て可能なスカラー変数
- 無指定文字長エンティティー
- PRIVATE コンポーネントの PUBLIC 型と PUBLIC コンポーネントの PRIVATE 型
- ALLOCATE と DEALLOCATE の ERRMSG キーワード
- ALLOCATE の SOURCE= キーワード
- 型拡張子
- CLASS 宣言
- 多相型エンティティー
- 継承と関連付け
- 遅延バインディングと抽象型
- 型バインド・プロシージャー
- TYPE CONTAINS 宣言
- ABSTRACT 属性
- DEFERRED 属性
- NON_OVERRIDABLE 属性
- 型バインド・プロシージャーの GENERIC キーワード
- FINAL サブルーチン
- ユーザー定義の派生型 I/O
- ASYNCHRONOUS 属性および文
- BIND(C) 属性および文

- PROTECTED 属性および文
- VALUE 属性および文
- VOLATILE 属性および文
- ポインター・オブジェクトの INTENT 属性
- 多相オブジェクトのデフォルトの初期化
- 代入文の左辺と右辺の形状または長さが異なる場合に、左辺の割り当て可能な変数を再割り当て (無指定文字長でない場合、`/assume:realloc_lhs` オプションが必要)
- ポインター代入の境界の仕様と境界の再マップ
- ASSOCIATE 構造
- SELECT TYPE 構造
- すべての I/O 文で、次の数値は任意の種類で指定可能:UNIT=, IOSTAT=
- NAMELIST I/O が内部ファイルで許可
- NAMELIST グループのエンティティーの制限の緩和
- 書式付き入出力で IEEE 無限大と NaN の表現方法が変更
- FLUSH 文
- WAIT 文
- OPEN の ACCESS='STREAM' キーワード
- OPEN およびデータ転送文の ASYNCHRONOUS キーワード
- INQUIRE およびデータ転送文の ID キーワード
- データ転送文の POS キーワード
- INQUIRE の PENDING キーワード
- 次の OPEN 数値は任意の種類で指定可能:RECL=
- 次の READ および WRITE 数値は任意の種類で指定可能:REC=、SIZE=
- 次の INQUIRE 数値は任意の種類で指定可能:NEXTREC=、NUMBER=、RECL=、SIZE=
- 開始する新しい I/O が自身以外の内部ファイルを修正しない内部 I/O の場合、再帰 I/O を利用可能
- IEEE 無限大および非数は Fortran 2003 で指定されるフォーマット出力で表示
- BLANK、DECIMAL、DELIM、ENCODING、IOMSG、PAD、ROUND、SIGN、SIZE I/O キーワード
- DC、DP、RD、RC、RN、RP、RU、RZ 書式編集記述子
- I/O フォーマットで、繰り返し指定子が続く場合、P 編集記述子の後のカンマはオプション
- USE 内のユーザー定義演算子名の変更
- USE の INTRINSIC および NON_INTRINSIC キーワード
- IMPORT 文
- 割り当て可能なダミー引数
- 割り当て可能な関数結果
- PROCEDURE 宣言
- 外部プロシージャーを参照する場合、汎用インターフェイス・ブロックの MODULE PROCEDURE からキーワード MODULE を省略
- プロシージャー・ポインター
- ABSTRACT INTERFACE
- PASS 属性と NOPASS 属性
- SYSTEM_CLOCK 組込み関数の COUNT_RATE 引数が任意の種類の REAL で指定可能
- STOP 文の実行で IEEE 浮動小数点例外が発生すると警告を表示
- `/assume:noold_maxminloc` が指定された場合、ゼロサイズの配列の MAXLOC または MINLOC でゼロを返す
- 型問い合わせ組込み関数
- COMMAND_ARGUMENT_COUNT 組込み関数
- EXTENDS_TYPE_OF と SAME_TYPE_AS 組込み関数
- GET_COMMAND 組込み関数
- GET_COMMAND_ARGUMENT 組込み関数

- GET_ENVIRONMENT_VARIABLE 組込み関数
- IS_IOSTAT_END 組込み関数
- IS_IOSTAT_EOR 組込み関数
- MAX/MIN/MAXVAL/MINVAL/MAXLOC/MINLOC 組込み関数 (CHARACTER 引数)
- MOVE_ALLOC 組込み関数
- NEW_LINE 組込み関数
- SELECTED_CHAR_KIND 組込み関数
- 次の組込み関数においてオプションで KIND= 引数を指定可能:ACHAR、COUNT、IACHAR、ICHAR、INDEX、LBOUND、LEN、LEN、TRIM、MAXLOC、MINLOC、SCAN、SHAPE、SIZE、UBOUND、VERIFY
- ISO_C_BINDING 組込みモジュール
- IEEE_EXCEPTIONS、IEEE_ARITHMETIC、IEEE_FEATURES 組込みモジュール
- ISO_FORTRAN_ENV 組込みモジュール

このリリースではまだ実装されていないか、動作しない Fortran 2003 機能の一部を次にリストします。

- パラメーター化された派生型
- 初期化式での変形組込み関数 (MERGE や SPREAD など) の使用

インテル® Fortran コンパイラーは、Fortran 2008 規格のいくつかの機能もサポートしています。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はない場合、仮引数に INTENT (IN) 属性が含まれていなければなりません。

4 インテル® Xeon Phi™ コプロセッサーを使用するアプリケーションの開発

このセクションでは、インテル® Visual Fortran Composer XE 2013 SP1 Windows* 版を使用したインテル® Xeon Phi™ コプロセッサー向けの開発の変更点、新機能、および最新情報をまとめています。

4.1 概要

インテル® Visual Fortran Composer XE 2013 SP1 は、インテル® MIC アーキテクチャーのコプロセッサー (インテル® Xeon Phi™ 製品ファミリー) に作業をオフロードするアプリケーションの開発がサポートされました。インテル® Xeon Phi™ コプロセッサーが利用可能な場合、コードのこれらのセクションはコプロセッサーで実行されます。コプロセッサーが利用できない場合は、ホスト CPU で実行されます。インテル® Xeon Phi™ コプロセッサーでネイティブに実行するアプリケーションの開発もサポートされました。

本リリースノートでは、オフロード操作のターゲットについて、*コプロセッサー*と *ターゲット*という 2 つの用語を使用しています。

4.2 ドキュメント

最新のドキュメントとアップデートは、「Intel® Composer XE 2013 Documentation Updates for Intel® MIC Architecture」 (英語) を参照してください。

4.3 既知の問題と変更点

このセクションでは、製品ドキュメントの訂正または追加をまとめています。

4.3.1 インテル® コンパイラー 14.0 初期リリースでインテル® MIC アーキテクチャー・ベースの Windows* システム向けにビルドされた Fortran コードはインテル® コンパイラー 14.0 Update 1 でリビルドが必要

インテル® MIC アーキテクチャー・プラットフォーム (Linux* または Windows*) 向けにビルドしたプログラムを、別のプラットフォームに配布できるように、インテル® Visual Fortran コンパイラー 14.0 Update 1 で修正されました。この修正により、インテル® Visual Fortran コンパイラー 14.0 初期リリースでインテル® MIC アーキテクチャー・ベースの Windows* システム向けにビルドされたすべての Fortran コードは、インテル® Visual Fortran コンパイラー 14.0 Update 1 以降のライブラリーにオブジェクトを再リンクする場合、または I/O 処理でセグメンテーション違反を回避するためにオブジェクトでダイナミック・ライブラリーを使用している場合、インテル® Visual Fortran コンパイラー 14.0 Update 1 以降でリビルドする必要があります。

4.3.2 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 -offload=mandatory オプションまたは -offload=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、-offload=mandatory コンパイラー・オプションまたは -offload=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。

4.3.3 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* タグが追加されるようになりました。このタグは、offload 宣言子が見つかったときに、ターゲットのコンパイル診断にのみ追加されます。

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。

4.3.4 直接 (ネイティブ) モードにおける libiomp5.so 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) には、インテル® コンパイラーのライブラリー (<common_files>\Intel\Shared Libraries\redist\lib\mic 以下) は含まれません。<common_files> は、デフォルトでは C:\Program Files (x86)\Common Files (64 ビットの Windows* システムの場合) を示します。

このため、直接モード (例えば、コプロセッサー上) でアプリケーションを実行する場合は、アプリケーションで使用する共有オブジェクト・ライブラリーのコピーを (scp 経由で) 最初にアップロードする必要があります。例えば、アプリケーションを実行する前に OpenMP* ライブラリー (<common_files>\Intel\Shared Libraries\redist\intel64_mic/libiomp5.so) をコプロセッサー (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、…) にコピーします。

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。

一部のアプリケーションでは、別のライブラリーもアップロードする必要があります。

4.4 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インテル® Debugger Extension

このセクションでは、インテル® Debugger Extension の変更点、新機能、カスタマイズ、および既知の問題をまとめています。インテル® Debugger Extension は、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コードのみサポートします。

4.4.1 機能

- オフロード拡張を使用して、コプロセッサーのネイティブ・アプリケーションとホスト・アプリケーションの両方をサポート
- 同時に複数のコプロセッサー・カードをデバッグ (オフロード拡張を使用)

4.4.2 インテル® Debugger Extension の使用

インテル® Debugger Extension は Microsoft* Visual Studio* IDE のプラグインです。Microsoft* Visual Studio* IDE で定義されたプロジェクトのデバッグを可能にします。インテル® Xeon Phi™ コプロセッサー向けアプリケーションは、ロードして実行することも、アタッチすることもできます。

インテル® Xeon Phi™ コプロセッサーにオフロードされるアプリケーションをデバッグするには、Microsoft* Visual Studio* IDE を起動する前に環境変数を設定する必要があります。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) のバージョンに応じて、次の変数を設定します。

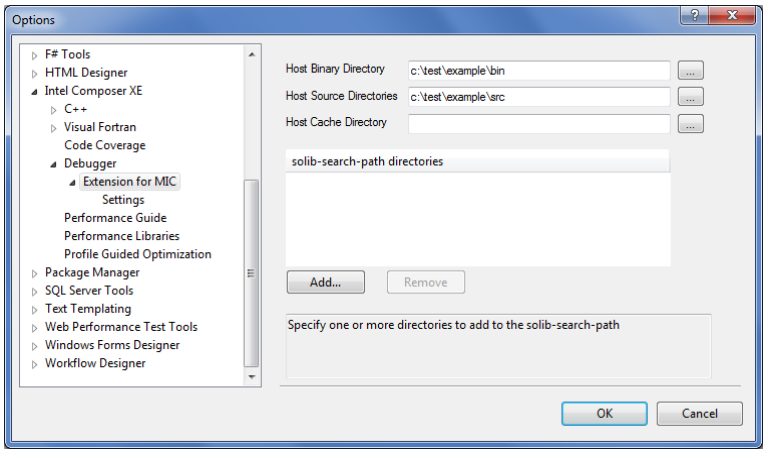
- インテル® MPSS 3.1:
  `AMPLXE_COI_DEBUG_SUPPORT=TRUE`
  `MYO_WATCHDOG_MONITOR=-1`
- インテル® MPSS 2.1:
  `COI_SEP_DISABLE=FALSE`
  `MYO_WATCHDOG_MONITOR=-1`

4.4.2.1 インテル® Xeon Phi™ コプロセッサー向けアプリケーションのロードと実行

これは、2 つのオフロードモデルのいずれかを使用しているアプリケーションでのみ行うことができます。インテル® Xeon Phi™ コプロセッサー向けプロジェクトを開き、[デバッグ] > [デバッグ開始] を選択してデバッグセッションを開始します。デバッグセッション中に表示されるコプロセッサーと情報 (スレッド、ブレークポイントなど) は、使用するオフロードモデルに依存します。つまり、オフロードモデルを使用していない場合、コプロセッサーでコードは実行されません。

4.4.2.2 インテル® Xeon Phi™ コプロセッサー上のプロセスにアタッチする

これは、インテル® Xeon Phi™ コプロセッサーでネイティブ実行されるアプリケーションでのみ行うことができます。デバッグするプロセスは、すでにコプロセッサーで実行されていなければなりません。
Microsoft* Visual Studio* IDE でプロセスバイナリーを指定します。[ツール] > [オプション...] を選択、次のように展開します。



[Host Binary Directory (ホストのバイナリー・ディレクトリー)] で、コプロセッサーでデバッグするバイナリー (のコピー) があるディレクトリーを指定します。ソースの検索に使用するルート・ディレクトリーは、[Host Source Directory (ホストのソース・ディレクトリー)] で指定できます。オプションで、コプロセッサー・アプリケーションの共有ライブラリーの検索パスを [solib-search-path directories (共有ライブラリーの検索パスのディレクトリー)] に追加できます。セマンティクスは GNU* GDB と同じです。

4.4.3 ドキュメント

インテル® デバッガーのドキュメントは、以下の場所にあります。

```
<install-dir>\Documentation\[en_US|ja_JP]\debugger\gdb\
        pdf\vsmigdb_config_guide.pdf
```

4.4.4 既知の問題

- オフロードデバッグは Microsoft* Visual Studio* 2012 でのみサポートされています。
- 同じ Visual Studio* セッションで、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プロセスに二度アタッチすると、[スレッド] ウィンドウにメインスレッド (ID=1) が表示されません。
  回避方法:
  メインスレッドに関する重要な情報は、[プロセス] ウィンドウで確認できます。
- [関数のブレークポイント] は動作しません。正常に動作しているように見えますが、実行を中断しません。
  回避方法:
  ソースファイルでブレークポイントを設定することができます。
- データ・ブレークポイントはサポートされていません。
- ブレークポイントのヒットカウント、条件、ヒット時の動作はサポートされていません。
- [逆アセンブル] ウィンドウでは、開始アドレスから 1024 バイトを超える範囲にスクロールすることはできません。
- [逆アセンブルアドレス] ウィンドウで式とアドレスはサポートされていません。
- インテル® MIC アーキテクチャー・アプリケーションの例外処理はサポートされていません。
- ブレークポイントの無効化は正しく動作しません。
  回避方法:
  ブレークポイントを削除して、再度設定します。
- アプリケーション実行中のブレークポイントの変更は正しく動作しません。変更されたかのように見えますが、変更が適用されません。
- インテル® MIC アーキテクチャーのネイティブ・アプリケーションの開始はサポートされていません。現在実行中のアプリケーションにアタッチすることはできます。
- 場合によっては、コプロセッサーで実行中のすべてのスレッドが [スレッド] ウィンドウに表示されないことがあります。
- Microsoft* Visual Studio* の [スレッド] ウィンドウには、スレッドの凍結、凍結解除、名前変更を行うコンテキスト・メニューがあります。これらのコンテキスト・メニューは、コプロセッサー上のスレッドでは正しく動作しません。
- オフロードコード領域のコード行でブレークポイントを設定すると、ホスト側とコプロセッサー側で有効になります。Microsoft* Visual Studio* の [ブレークポイント] ウィンドウでブレークポイントを展開すると、2 つのブレークポイントが表示されます。デバッグセッションを再起動すると、コプロセッサー側のブレークポイントは無効なブレークポイントとして残りますが、これは無視してください。
- 場合によっては、オフロードコードのデバッグ中、オフロード宣言子のソース位置で実行が停止することがありますが、実際の実行位置はライブラリー・ルーチン内です。その位置から実行を続行できます。
- インテル® 64 対応アプリケーションのみサポートされています。

5 インテル® マス・カーネル・ライブラリー

このセクションでは、インテル® マス・カーネル・ライブラリー (インテル® MKL) の変更点、新機能、および最新情報をまとめています。

5.1 インテル® MKL 11.1 Update 1 の新機能

- インテル® AVX-512 命令セットのサポート (特定の最適化のみ)
- BLAS:
  - インテル® アドバンスト・ベクトル・エクステンション (インテル® AVX) およびインテル® アドバンスト・ベクトル・エクステンション 2 (インテル® AVX2)

対応のすべての 64 ビットのインテル® プロセッサーにおいて、DSDOT のパフォーマンスが向上し、マルチスレッドをサポート
    - *TRSM で対角行列のデノーマル数の処理が向上
    - インテル® メニー・インテグレーテッド・コア (インテル® MIC) アーキテクチャーにおいて、小さな N と大きな M および K で SGEMM のパフォーマンスが向上
    - インテル® SSE4.2 以降対応のすべてのインテル® プロセッサーにおいて *HEMM の並列パフォーマンスが向上
    - インテル® SSSE3 以降対応のすべてのインテル® プロセッサーにおいて 64 ビットの *SYRK/*HERK の並列パフォーマンスが向上
    - インテル® SSE4.2 以降対応のすべてのインテル® プロセッサーにおいて 64 ビットの {D,S}SYRK のシリアル・パフォーマンスが向上
    - インテル® MIC アーキテクチャーにおいて DTRSM のパフォーマンスが向上
    - インテル® AVX 対応インテル® プロセッサー向けインテル® Optimized HPL Benchmark の runmultiscript 機能を拡張
    - インテル® MIC アーキテクチャーにおいてインテル® Optimized HPL Benchmark のパフォーマンスが向上
- LAPACK:
    - 並列 LAPACK 関数 (OR/UN)M(QR/RQ/QL/LQ) のメモリー使用率が減少
    - LAPACK 関数のスタックメモリー使用率が減少
    - 固有値のみ必要な場合、大きな次元で (S/D)SYRDB および (S/D)SYEV のパフォーマンスが向上
- ScaLAPACK:
    - デフォルトの NETLIB 複素数型と MKL 複素数型が混在できるように PBLAS ヘッダーを更新
- DFT: 複素数-複素数および実数-複素数の変換を最適化
- 転置: 縦長の行列と横長の行列で mkl_?omatcopy ルーチンのパフォーマンスが向上
- DFTI インターフェイスと FFTW ラッパーがスレッドセーフに対応
- 並列領域から MKL DFT を使用する場合、NUMBER_OF_USER_THREADS パラメーターは任意設定に変更

5.2 インテル® MKL 11.1 の新機能

- 条件付きの数値再現性:アライメントされていないデータで条件付き数値再現性 (CNR) モードをサポート
- ノードごとにプロセッサーの種類や搭載されているインテル® Xeon Phi™ コプロセッサーの数が異なるヘテロジニアス・クラスターで MP LINPACK をサポート
- Windows* において、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・ベースのインテル® Xeon Phi™ コプロセッサーでコンパイラーによるオフロード支援と自動オフロード・プログラミング・モデルをサポート
- 最新の AMD* システムにおいて CNR=AUTO モードのパフォーマンスが向上
- BLAS:
    - インテル® SSE4.2 以降対応のすべてのインテル® プロセッサーにおいて [S/D]GEMV のパフォーマンスが向上
    - 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) において [D/Z]GEMM および倍精度のレベル 3 BLAS 関数を最適化
    - インテル® アドバンスト・ベクトル・エクステンション (インテル® AVX) およびインテル® AVX2 において [Z/C]AXPY および [Z/C]DOT[U/C] を最適化
    - インテル® MIC アーキテクチャーにおいて DTRMM のシーケンシャル・バージョンを最適化
    - インテル® AVX2 において DAXPY をチューニング
- LAPACK:
    - 固有値のみ必要な場合、大きな次元で (S/D)SYRDB および (S/D)SYEV のパフォーマンスが向上

    - M,N<10 のような小さなサイズで xGESVD のパフォーマンスが向上。
- VSL:
    - 平均絶対偏差のサポートとサンプルの追加
    - alpha=1 の場合のワイブル乱数ジェネレーター (RNG) のパフォーマンスが向上
    - 外積および平均絶対偏差の行列において、次数 4 までのローデータおよび中央部の統計的総和をサポート
    - S. Joe および F. Y. Kuo により設計された、最大 21,201 次元まで発生できるソボル QRNG の使用法を示す VSL サンプルを追加
    - インテル® MIC アーキテクチャーにおいて SFMT19937 基本乱数ジェネレーター (BRNG) のパフォーマンスが向上
- DFT:
    - インテル® MIC アーキテクチャーにおいて倍精度の複素数-複素数変換のパフォーマンスが向上
    - インテル® AVX2 において複素数-複素数 DFT を最適化
    - インテル® Xeon® プロセッサー E5 v2 ファミリー (開発コード名 IvyTown) において 2 次元の複素数-複素数 DFT を最適化
    - インテル® Xeon® プロセッサー E5 ファミリー (インテル® AVX) およびインテル® AVX2 において GENE アプリケーション固有のワークロードでパフォーマンスが向上
    - DFTI 計算関数のドキュメントのデータレイアウトが向上
    - 大規模な実数-複素数 FFT のスケーリング
- データ・フィッティング:
    - インテル® Xeon® プロセッサーおよびインテル® MIC アーキテクチャーにおいて df?Interpolate1D および df?SearchCells1D 関数のパフォーマンスが向上
    - インテル® MIC アーキテクチャー、インテル® Xeon® プロセッサー X5570、インテル® Xeon® プロセッサー E5-2690 において、線形および 3 次 Hermite/Bessel/Akima スプライン用 df?construct1d 関数のパフォーマンスが向上
- 転置:
    - 正方行列でインプレース転置のパフォーマンスが向上
- インストール時間を短縮するためパッケージに含まれるインテル® MKL のサンプルとテストをアーカイブ
- リンクツールおよびリンク・ライン・アドバイザー:Windows* でインテル® MIC アーキテクチャーをサポート

5.3 注意事項

- インテル® MKL では、インストールするコンポーネントを選択できるようになりました。PGI* コンパイラー、Compaq* Visual Fortran コンパイラー、SP2DP インターフェイス、BLAS95 および LAPACK95 インターフェイス、クラスターサポート (ScaLAPACK および Cluster DFT)、インテル® MIC アーキテクチャーのサポートに必要なコンポーネントは、インストール時に明示的に選択しない限りインストールされません。
- インテル® MKL クラスター・コンポーネント (ScaLAPACK および Cluster DFT) では、アライメントされていない CNR は利用できません。
- BOOST/uBLAS および Java* でのインテル® MKL の使用例は、製品パッケージからは削除され、以下の記事 (英語) からダウンロードすることができます。
    - How to use Intel® MKL with Java*
    - How to use BOOST* uBLAS with Intel® MKL

5.4 既知の問題

既知の制限事項の詳細なリストは、インテル® デベロッパー・ゾーンにある「Intel® MKL Article List」 (英語) を参照してください。

5.5 権利の帰属

エンド・ユーザー・ソフトウェア使用許諾契約書 (End User License Agreement) で言及されているように、製品のドキュメントおよび Web サイトの両方で完全なインテル製品名の表示 (例えば、"インテル® マス・カーネル・ライブラリー") とインテル® MKL ホームページ (www.intel.com/software/products/mkl (英語)) へのリンク/URL の提供を正確に行うことが最低限必要です。

インテル® MKL の一部の基となった BLAS の原版は http://www.netlib.org/blas/index.html (英語) から、LAPACK の原版は http://www.netlib.org/lapack/index.html (英語) から入手できます。LAPACK の開発は、E. Anderson、Z. Bai、C. Bischof、S. Blackford、J. Demmel、J. Dongarra、J. Du Croz、A. Greenbaum、S. Hammarling、A. McKenney、D. Sorensen らによって行われました。LAPACK 用 FORTRAN 90/95 インターフェイスは、http://www.netlib.org/lapack95/index.html (英語) にある LAPACK95 パッケージと類似しています。すべてのインターフェイスは、純粋なプロシージャー用に提供されています。

インテル® MKL クラスター・エディションの一部の基となった ScaLAPACK の原版は http://www.netlib.org/scalapack/index.html (英語) から入手できます。ScaLAPACK の開発 は、L. S. Blackford、J. Choi、A. Cleary、E. D'Azevedo、J. Demmel、I. Dhillon、J. Dongarra、S. Hammarling、G. Henry、A. Petitet、K. Stanley、D. Walker、R. C. Whaley らによって行われました。

インテル® MKL の PARDISO は、バーゼル大学 (University of Basel) から無償で提供されている PARDISO 3.2(http://www.pardiso-project.org (英語)) と互換性があります。

本リリースのインテル® MKL の一部の FFT 関数は、カーネギーメロン大学からライセンスを受けて、SPIRAL ソフトウェア生成システム (http://www.spiral.net/ (英語)) によって生成されました。SPIRAL の開発は、Markus Püschel、José Moura、Jeremy Johnson、David Padua、Manuela Veloso、Bryan Singer、Jianxin Xiong、Franz Franchetti、Aca Gacic、Yevgen Voronenko、Kang Chen、Robert W. Johnson、Nick Rizzolo らによって行われました。

インテル® MKL Extended Eigensolver の機能は、Feast Eigenvalue Solver 2.0 (http://www.ecs.umass.edu/~polizzi/feast/) をベースにしています。

6 著作権と商標について

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を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、その欠陥や故障によって人身事故が発生するようなアプリケーションでの使用を想定した設計は行われていません。

インテル製品は、予告なく仕様や説明が変更される場合があります。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。

本書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。

本書で紹介されている注文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® Visual Fortran コンパイラーおよびインテル® マス・カーネル・ライブラリーは、インテルのエンド・ユーザー・ソフトウェア使用許諾契約書 (EULA) の下で提供されます。

GNU* プロジェクト・デバッガー (GDB) は、General GNU Public License GPL V3 の下で提供されます。

Intel、インテル、Intel ロゴ、Pentium、Xeon、Intel Xeon Phi 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。

* 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。