

インテル® Fortran コンパイラー 11.1 Linux* 版 プロフェッショナル・エディション インストール・ガイドおよびリリースノート

資料番号: 321415-002JA

2010年3月24日

目次

1	概要	4
1.1	変更履歴	4
1.2	製品の内容	5
1.3	動作環境	5
1.3.1	Red Hat Enterprise Linux 3、SUSE LINUX Enterprise Server 9 のサポート 終了予定	7
1.3.2	クラスター OpenMP ライブラリーのコマンドライン・オプションの廃止 予定	7
1.4	ドキュメント	7
1.5	日本語サポート	8
1.6	テクニカルサポート	8
2	インストール	8
2.1	サイレントインストール	8
2.2	既知のインストールの問題	9
2.3	インストール先フォルダー	9
2.4	削除/アンインストール	10
3	インテル® Fortran コンパイラー	11
3.1	互換性	11
3.1.1	型バインド・プロシージャの誤った派生型レイアウト	11
3.1.2	BIND(C) プロシージャの不適切な隠し引数の削除	11
3.1.3	-openmp-threadprivate compat の実装に関する変更	12
3.2	新機能と変更された機能	12
3.2.1	Fortran 2003 の機能	12
3.2.2	その他の変更	13
3.3	新規および変更されたコンパイラー・オプション	13
3.3.1	-O0 オプションの -mp の除外	13
3.3.2	-warn interface は -gen-interface を含意	13

3.4	その他の変更および注意	13
3.4.1	最適化レポートがデフォルトで無効に設定	13
3.4.2	コンパイラ環境の構築.....	14
3.4.3	デフォルトの命令セットがインテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) を必要とするものに変更.....	14
3.4.4	I/O を制御する新しい環境変数	14
3.4.5	OpenMP ライブラリーのデフォルトが "compat" に変更	15
3.4.6	OpenMP ライブラリーのデフォルトがダイナミック・リンクに変更	15
3.4.7	サンプリング・ベースのプロファイルに基づく最適化機能の削除.....	15
3.5	既知の問題.....	15
3.5.1	KMP_AFFINITY のデフォルト動作が変更	15
3.5.2	古いバージョンの ld による致命的なエラー	15
3.5.3	空の派生型の制限付きサポート	16
3.5.4	RedHat Enterprise Linux 5.4 の gdb ブレークポイントの設定問題	17
3.6	Fortran 2003 機能の概要	17
4	インテル® デバッガー (IDB).....	19
4.1	Java ランタイム環境の設定.....	19
4.2	デバッガーの起動.....	19
4.3	その他のドキュメント	20
4.4	デバッガー機能.....	20
4.4.1	IDB の主な機能.....	20
4.4.2	新機能と変更された機能.....	20
4.5	既知の問題.....	20
4.5.1	データ共有検出の問題.....	20
4.5.2	[Signals (シグナル)] ダイアログが動作しない	20
4.5.3	再入可能呼び出しにおけるブレークの未サポート	21
4.5.4	GUI のサイズ調整	21
4.5.5	プロセスの終了	21
4.5.6	10 進浮動小数点の未サポート.....	21
4.5.7	\$cdir ディレクトリー、\$cwd ディレクトリー.....	21
4.5.8	info stack の使用	21
4.5.9	\$stepg0 のデフォルト値の変更.....	21
4.5.10	一部の Linux システムでの SIGTRAP エラー	22
4.5.11	MPI プロセスのデバッグには idb GUI は使用できません	22

5	インテル® マス・カーネル・ライブラリー	22
5.1	本バージョンでの変更.....	22
5.1.1	インテル® マス・カーネル・ライブラリー 10.2 Update 5	22
5.1.2	インテル® マス・カーネル・ライブラリー 10.2 Update 4	23
5.1.3	インテル® マス・カーネル・ライブラリー 10.2 Update 3	23
5.1.4	インテル® マス・カーネル・ライブラリー 10.2 Update 2	24
5.2	既知の問題.....	26
5.3	注意事項.....	26
5.4	権利の帰属.....	26
6	著作権と商標について.....	27

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。

1.1 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。報告されている問題の修正リストは、[インテル® コンパイラー 11.1 プロフェッショナル・エディション 修正リスト](#) (英語)、[インテル® マス・カーネル・ライブラリー 10.2 修正リスト](#) (英語) を参照してください。

Update 6

- [インテル® マス・カーネル・ライブラリー](#)が 10.2 Update 5 に更新
- 報告されている問題の修正

Update 5 (11.1.069)

- [インテル® マス・カーネル・ライブラリー](#)が 10.2 Update 4 に更新
- [クラスター OpenMP* ライブラリーの廃止予定](#)に関する注意事項
- 「互換性」セクションに [OpenMP threadprivate 変数](#)に関する情報が追加されました。
- [RedHat* Enterprise Linux* 5.4 の gdb ブレークポイントの問題](#)に関する情報が追加されました。
- 報告されている問題の修正

Update 4 (11.1.064)

- [インテル® マス・カーネル・ライブラリー](#)が 10.2 Update 3 に更新
- 報告されている問題の修正

Update 3 (11.1.059)

- 報告されている問題の修正

Update 2 (11.1.056)

- Ubuntu* 9.04 のサポートの追加
- [Fedora* 10 システムでの非 RPM インストール](#)に関する注意事項の追加
- [隠し引数は BIND\(C\) ルーチンで使用されなくなりました](#)。これにより、ソース変更が必要な場合があります。
- [新しいコンパイラー・オプション](#) `-mkl` と `-xAVX` の記載の追加
- [-warn interface の動作変更](#)に関する注意事項の追加
- 報告されている問題の修正

Update 1 (11.1.046)

- 型バインド・プロシージャーを含む派生型を宣言または使用しているソースの[再コンパイルの必要性](#)
- [-O0 の動作変更](#)に関する注意事項の追加
- [FORT_BLOCKSIZE 環境変数と FORT_BUFFERCOUNT 環境変数](#)の記載
- 報告されている問題の修正

1.2 製品の内容

インテル® Fortran コンパイラー 11.1 Linux 版プロフェッショナル・エディションには、次のコンポーネントが含まれています。

- インテル® Fortran コンパイラー。Linux オペレーティング・システムを実行する IA-32、インテル® 64、および IA-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® デバッガー
- IA-64 対応アプリケーション開発用インテル® アセンブラー
- インテル® マス・カーネル・ライブラリー 10.2 Update 5
- 各種ドキュメント

1.3 動作環境

アーキテクチャー名についての説明は、<http://software.intel.com/en-us/articles/intel-architecture-platform-terminology/> (英語) を参照してください。

IA-32 対応アプリケーション開発に必要な環境

- 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 IA-32 またはインテル® 64 アーキテクチャー・プロセッサをベースとするコンピューター (インテル® Pentium® 4 プロセッサ以降、または互換性のあるインテル以外のプロセッサ)
 - ホストと異なるターゲットの開発を行う場合、Linux ディストリビューションから別のライブラリー・コンポーネントのインストールが必要になることがあります。
 - 機能を最大限に活用できるように、マルチコアまたはマルチプロセッサ・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 (すべての機能をインストールする場合)
- 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、[インテル® テクニカルサポート](#)までお問い合わせください。)
 - Asianux* 3.0
 - Debian* 4.0
 - Fedora 10
 - Red Hat* Enterprise Linux* 3、4、5
 - SUSE LINUX Enterprise Server* 9, 10, 11
 - TurboLinux* 11
 - Ubuntu 9.04
- 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)
- libstdc++.so.5 を提供する Linux コンポーネント compat-libstdc++
- -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。
- インテル® 64 アーキテクチャー・システムで開発を行う場合、一部の Linux ディストリビューションでは、次のいずれかまたは複数の Linux コンポーネントを追加でインストールしなければならない場合があります: ia32-libs、lib32gcc1、lib32stdc++6、libc6-dev-i386、gcc-multilib。

インテル® 64 対応アプリケーションの開発に必要な環境

- インテル® 64 アーキテクチャー・プロセッサをベースとするコンピューター (インテル® Pentium® 4 プロセッサ以降、または互換性のあるインテル以外のプロセッサ)
 - 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・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 (すべての機能をインストールする場合)
- 仮想メモリのページングファイル用に 100MB のディスク空き容量。インストールされている Linux のディストリビューションで推奨される最小容量以上の仮想メモリーを使用していることを確認してください。
- 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、[インテル® テクニカルサポート](#)までお問い合わせください。)
 - Asianux 3.0
 - Debian 4.0
 - Fedora 10
 - Red Hat Enterprise Linux 3、4、5
 - SUSE LINUX Enterprise Server 9、10、11
 - TurboLinux 11
 - Ubuntu 9.04
- 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)
- libstdc++.so.5 を提供する Linux コンポーネント compat-libstdc++
- -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。
- 32 ビット・ライブラリーを含む Linux コンポーネント (ia32-libs と呼ばれる)

IA-64 対応アプリケーション開発に必要な環境

- IA-64 アーキテクチャー・プロセッサ (インテル® Itanium® プロセッサ) をベースとするシステム
- RAM 1GB (2GB 推奨)
- 2GB 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)
- 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、[インテル® テクニカルサポート](#)までお問い合わせください。)
 - Asianux 3.0
 - Debian 4.0
 - Red Hat Enterprise Linux 3、4、5
 - SUSE LINUX Enterprise Server 9、10、11
 - TurboLinux 11
 - Ubuntu 9.04
- 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)
- libstdc++.so.5 を提供する Linux コンポーネント compat-libstdc++

インテル® デバッガーのグラフィカル・ユーザー・インターフェイスを使用するためのその他の要件

- IA-32 アーキテクチャー・システムまたはインテル® 64 アーキテクチャー・システム
- Java® ランタイム環境 (JRE) 5.0 (1.5) または 6.0 (1.6)
 - IA-32 アーキテクチャー・システムでは 32 ビット版の JRE、インテル® 64 アーキテクチャー・システムでは 64 ビット版の JRE を使用する必要があります。

説明

- 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、動作確認に使用し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いて、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 -openmp 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。
- 上記のリストにはすべてのプロセッサ・モデル名は含まれていません。リストされているプロセッサと同じ命令セットを正しくサポートしているプロセッサ・モデルでも動作します。特定のプロセッサ・モデルについては、[インテル® テクニカルサポート](#)にお問い合わせください。
- 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。

1.3.1 Red Hat Enterprise Linux 3、SUSE LINUX Enterprise Server 9 のサポート終了予定

インテル® Fortran コンパイラーの将来のメジャーリリースでは、Red Hat Enterprise Linux 3 と SUSE LINUX Enterprise Server 9 はサポートされなくなる予定です。これらのオペレーティング・システムを使用している場合は、インテルでは新しいバージョンへの移行を推奨しています。

1.3.2 クラスタ OpenMP ライブラリーのコマンドライン・オプションの廃止予定

非公式なオプションとして whatif.intel.com から利用可能なクラスタ OpenMP 機能は、コンパイラー製品のライブラリーとコマンドライン・サポートに依存しています。これらのライブラリーとコマンドライン・サポートは、コンパイラーの次のメジャーリリースで廃止されるため、クラスタ OpenMP 機能は使用できなくなります。

1.4 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 日本語サポート

インテル® コンパイラーは、日本語ユーザー向けのサポートを提供しています。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。

日本語の言語サポートを英語のオペレーティング・システムで使用する場合は、日本語のオペレーティング・システムで英語の言語サポートを使用する場合は、<http://software.intel.com/en-us/articles/changing-language-setting-to-see-english-on-a-japanese-os-environment-or-vice-versa-on-linux/> (英語) の説明を参照してください。

1.6 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: 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

初めて製品をインストールする場合は、インストール中にシリアル番号の入力が求められますので、あらかじめご用意ください。製品のインストールと使用には、有効なライセンスが必要です。

DVD 版を購入した場合は、DVD をドライブに挿入し、DVD のトップレベル・ディレクトリーにディレクトリーを変更 (cd) して、次のコマンドでインストールを開始します。

```
./install.sh
```

ダウンロード版を購入した場合は、次のコマンドを使用して、書き込み可能な任意のディレクトリーに展開します。

```
tar -xzvf name-of-downloaded-file
```

その後、展開したファイルを含むディレクトリーに移動 (cd) し、次のコマンドでインストールを開始します。

```
./install.sh
```

手順に従ってインストールを完了します。

2.1 サイレントインストール

自動インストール、「サイレント」インストール機能についての詳細は、<http://software.intel.com/en-us/articles/intel-compilers-for-linux-version-111-silent-installation-guide/> (英語) を参照してください。

2.2 既知のインストールの問題

- Linux ディストリビューションの Security-Enhanced Linux (SELinux) 機能を有効にしている場合は、インテル® Fortran コンパイラーをインストールする前に SELINUX モードを permissive に変更する必要があります。詳細は、Linux ディストリビューションのドキュメントを参照してください。インストールが完了したら、SELINUX モードを元の値に戻してください。
- 一部の Linux バージョンでは、自動マウントデバイスに“実行”許可がなく、インストール・スクリプトを直接 DVD から実行すると、次のようなエラーメッセージが表示されることがあります。

```
bash: ./install.sh: /bin/bash: bad interpreter: Permission denied
```

このエラーが表示された場合は、次の例のように実行許可を含めて DVD を再マウントします。

```
mount /media/<dvd_label> -o remount,exec
```

その後、再度インストールを行ってください。

- Fedora 10 の一部のバージョンには、問題のある rpm ユーティリティーが含まれており、インテル® コンパイラーが正常にインストールできません。そのため、Fedora 10 システムでは、コンパイラーは非 RPM インストールを強制的に行います。
- バージョン 11.1 では、IA-32 およびインテル® 64 アーキテクチャー・システム上で Ubuntu 9.04 をサポートしています。ただし、ソフトウェアのライセンス規約上、Ubuntu 9.04 を実行しているインテル® 64 アーキテクチャー・システム上で、IA-32 コンポーネントを評価する際に、評価ライセンス機能を使用することはできません。Ubuntu の以前のバージョン (本リリースでは未サポート) でも同様の問題が発生する可能性があります。これは、評価ライセンス機能を使用する場合のみの問題です。シリアル番号、ライセンスファイル、フローティング・ライセンス、その他のライセンス・マネージャー操作、およびオフラインでのアクティベーション操作 (シリアル番号を使用) には、影響はありません。Ubuntu を実行しているインテル® 64 アーキテクチャー・システムで、バージョン 11.1 の IA-32 コンポーネントの評価が必要な場合は、インテル® ソフトウェア評価センター (<http://www.intel.com/cd/software/products/asmo-na/eng/download/eval/> (英語)) で評価版のシリアル番号を入手してください。

2.3 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。

- <install-dir>/Compiler/11.1/xxx/
 - bin
 - ia32
 - intel64
 - ia64
 - include
 - ia32
 - intel64
 - ia64

- lib
 - ia32
 - intel64
 - ia64
- idb
 - gui
 - ia32
 - ia64
 - intel64
 - lib
 - third_party
- mkl
 - benchmarks
 - examples
 - include
 - interfaces
 - lib
 - tests
 - tools
- Documentation
- man
- Samples

<install-dir> はインストール・ディレクトリー (デフォルトのインストール先は /opt/intel) で、xxx は 3 桁のビルド番号です。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
- ia64: IA-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

インテル® C++ コンパイラーとインテル® Fortran コンパイラーの両方がインストールされている場合、所定のバージョンのフォルダーが共有されます。

2.4 削除/アンインストール

製品の削除 (アンインストール) は、製品をインストールしたユーザー (root または非 root ユーザー) で実行してください。インストールに sudo を使用した場合は、アンインストールの際にも使用する必要があります。パフォーマンス・ライブラリー・コンポーネントを残してコンパイラーのみを削除することはできません。

1. 端末を開いて、<install-dir> 以外のフォルダーに移動 (cd) します。
2. コマンド <install-dir>/bin/ia32/uninstall_cprof.sh を入力します (必要に応じて ia32 を intel64 または ia64 に変更してください)。
3. 画面の指示に従ってオプションを選択します。
4. 別のコンポーネントを削除するには、ステップ 2 と 3 を繰り返します。

同じバージョンのインテル® C++ コンパイラーをインストールしている場合は、C++ コンパイラーもリストに表示されます。

3 インテル® Fortran コンパイラー

このセクションでは、インテル® 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 Linux 版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 11.1 でもそのまま使用できます。ただし、次の例外があります。

- マルチファイルのプロシージャ間の最適化 (-ipo) オプションを使用してビルドされたオブジェクトは、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、REAL(16) または REAL*16 データ型を使用す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 または IA-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、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。

3.1.1 型バインド・プロシージャの誤った派生型レイアウト

初期のバージョンの 11.1 コンパイラーでは未使用の空間が型バインド・プロシージャを含む派生型に誤って追加されます。この問題は 11.1 Update 1 で修正されています。初期の 11.1 コンパイラーでコンパイルされたそのような型のオブジェクトを宣言または使用するすべてのソースは、バージョン 11.1 Update 1 以降を使用して再コンパイルする必要があります。

3.1.2 BIND(C) プロシージャの不適切な隠し引数の削除

インテル® Fortran コンパイラーの古いバージョンでは、BIND(C) 属性でプロシージャの隠し引数が不適切に渡されたり、使用されていました。例えば、Fortran ルーチンに CHARACTER 引数があり、BIND(C) が指定されている場合、コンパイラーはこの 2 つの隠し引数が関数のリターンアドレスと長さ用に渡されたものとみなし、さらに各文字引数の別の隠し引数とみなしていました。このような隠し引数は必要であり、非互換 (BIND(C) 属性を持たない) の Fortran プロシージャでは正しいため、Fortran 規格ではこのような隠し引数の互換プロシージャにおける使用を禁止しています。Fortran および C の引数リストは 1:1 で対応している必要があります。

この問題が発生する多くのケースでは、当該プロシージャは互換プロシージャの標準規則に合っていません。例えば、引数が CHARACTER 型の場合、長さは 1 でなければなりません。そのような文字の 1 つの配列が引数である場合があります。どのような配列でも互換プロシージャの関数の戻り値としては許可されていません。しかし、標準準拠の互換プロシージャを作成して、コンパイラーが隠し引数を渡したり、想定することが可能でした。

11.1 Update 2 では、隠し引数の受け渡しや想定はしないように修正されています。隠し引数を渡す必要のあるような C コードを記述している場合は、記述を変更し、再コンパイルする必要があります。実引数として渡された結果変数で関数をサブルーチンに変換しなければならない場合もあります。ご不便をおかけして申し訳ありませんが、この修正は正当性と移植性における重要なものです。ご了承ください。

3.1.3 -openmp-threadprivate compat の実装に関する変更

-openmp-threadprivate compat オプションを使用してコンパイルしたすべての Fortran ソースは、11.1 Update 5 以降で再コンパイルする必要があります。このオプションは、OpenMP threadprivate 変数の gcc* 実装と互換性のあるコードを生成しますが、以前のバージョンの Fortran ではリンク時にエラーとなったり、ランタイム時に正しく動作しない問題がありました。

新しいバージョンでは、threadprivate を作成する Fortran COMMON またはモジュール変数で初期化するか (COMMON の BLOCK DATA サブプログラムにある DATA またはモジュール変数の初期値)、あるいは対応する変数が __thread または __declspec(thread) 属性を持つ threadprivate な struct として宣言された C コードを使用してプログラムをリンクする必要があります。

3.2 新機能と変更された機能

いくつかの言語機能についての説明がコンパイラー・ドキュメントにまだ含まれていない可能性があります。必要に応じて、Fortran 2003 規格 (http://j3-fortran.org/doc/2003_Committee_Draft/04-007.pdf (英語)) を参照してください。

3.2.1 Fortran 2003 の機能

- オブジェクト指向の機能
 - CLASS 宣言
 - SELECT TYPE 構造
 - EXTENDS_TYPE_OF と SAME_TYPE_AS 組み込み関数
 - 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
 - 型問い合わせ組み込み関数
- 型バインド・プロシージャ
 - TYPE CONTAINS 宣言
 - ABSTRACT 属性
 - DEFERRED 属性
 - NON_OVERRIDABLE 属性
 - 注: GENERIC 属性、FINAL プロシージャ、型バインド操作はこのリリースではサポートされていません。
- 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
- 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
- SYSTEM_CLOCK 組み込み関数の 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 でゼロを返します。Fortran 95 では値はプロセッサ依存で、インテル® Fortran は 1 を返していました。-assume noold_maxminloc が指定されるとパフォーマンスが低下します。

3.2.2 その他の変更

- 文字列長チェックが有効 (`-check bounds`) で文字オブジェクトが引数として渡されると、渡された長さの最小値と呼び出されたプロシージャで宣言された長さが上限として使用されます。
- リストで対応する変数が LOGICAL ではない場合、リスト指定またはネームリスト指定の入力で、LOGICAL 定数形式の入力値項目 (例: T または .F) は許可されなくなりました。同様に、I/O リスト変数が LOGICAL 型の場合、対応する入力値は、LOGICAL 定数の形式でなければなりません。新しい `-assume old_logical_ldio` オプションを使用して古い動作に戻すことができます。
- 浮動小数点例外動作のコンパイルごとの制御 (`-fpe-all`)

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。

- `-assume [no]ieee_fpe_flags`
- `-assume [no]old_logical_ldio`
- `-assume [no]old_maxminloc`
- `-diag-enable sc-include`
- `-diag-enable sc-parallel`
- `-fpe-all`
- `-mkl[=lib]`
- `-xAVX`

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 -O0 オプションの -mp の除外

バージョン 11.1 では、最適化を無効にする `-O0` オプションは、浮動小数点精度を最大にする `-mp` を含意しなくなりました。`-mp` スイッチは廃止予定です。そのため、浮動小数点精度の影響を受けやすいアプリケーションには、明示的に `-fp-model` オプションを指定することを推奨します。

3.3.2 -warn interface は -gen-interface を含意

バージョン 11.1 で `-warn interface` を指定すると、エラーチェック目的でインターフェイスを生成し使用できます。`-warn interface` が有効な場合は、`-gen-interface` を指定する必要はなくなりました。

3.4 その他の変更および注意

3.4.1 最適化レポートがデフォルトで無効に設定

バージョン 11.1 以降、コンパイラーは、ベクトル化、自動並列化、OpenMP スレッド化ループに関する最適化レポートメッセージをデフォルトで表示しないようになりました。これらのメッセージを表示するには、`-diag-enable vec`、`-diag-enable par`、`-diag-enable openmp` を指定するか、`-vec-report`、`-par-report`、`-openmp-report` を使用する必要があります。

また、バージョン 11.1 以降、最適化レポートメッセージは `stdout` ではなく、`stderr` に送られます。

3.4.2 コンパイラー環境の構築

コマンドライン・ビルド環境の設定に使用されていた `ifortvars.sh` (`ifortvars.csh`) スクリプトが変更されました。以前のバージョンでは、`fc` または `fce` のいずれかのルート・ディレクトリーを選択することによってターゲット・プラットフォームが選択されました。バージョン 11.x では、スクリプトは 1 つのみで、引数を指定してターゲット・プラットフォームを選択します。

コマンドの形式は以下のとおりです。

```
source <install-dir>/Compiler/11.1/xxx/bin/ifortvars.sh argument
```

<install-dir> はインストール・ディレクトリー (デフォルトのインストール先は `/opt/intel`) で、xxx はリビジョン番号です。argument は、`ia32`、`intel64`、`ia64` のいずれかです (「[インストール先フォルダー](#)」を参照)。コンパイラー環境を構築すると、インテル® デバッガー (`idb`) 環境も構築されます。

3.4.3 デフォルトの命令セットがインテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) を必要とするものに変更

バージョン 11.0 以降、IA-32 アーキテクチャー向けのコンパイルでは、`-msse2` (旧: `-xw`) がデフォルトになりました。`-msse2` でビルドされたプログラムは、インテル® Pentium® 4 プロセッサーや特定のインテル以外のプロセッサーなど、インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) をサポートするプロセッサー上で実行する必要があります。互換性を保証するランタイムチェックは行われません。プログラムがサポートされていないプロセッサーで実行されている場合は、無効な命令フォルトが発生する場合があります。これにより、インテル® SSE 命令が x87 命令の代わりに使用され、高い精度ではなく、宣言された精度で計算が行われることがあるため、浮動小数点結果が変更される可能性があることに注意してください。

すべてのインテル® 64 アーキテクチャー・プロセッサーでインテル® SSE2 がサポートされています。

汎用 IA-32 の以前のデフォルトを使用する場合は、`-mia32` を指定してください。

3.4.4 I/O を制御する新しい環境変数

バージョン 11.1 では、アプリケーションの実行時に I/O 動作に影響する 2 つの環境変数が追加されています。

`FORT_BLOCKSIZE` は、OPEN 文で `BLOCKSIZE=` が省略されたときに使用されるデフォルトの `BLOCKSIZE` 値を指定できます。有効なサイズは 0 から 2147467264 までの値です。サイズは最も近い 512 バイト境界に丸められます。デフォルトの `BLOCKSIZE` 値は 128KB です。

`FORT_BUFFERCOUNT` は、OPEN 文で `BUFFERCOUNT=` が省略されたときに使用されるデフォルトの `BUFFERCOUNT` 値を指定できます。有効な値は 0 から 127 までの値です。0 が指定された場合、デフォルト値の 1 が使用されます。

3.4.5 OpenMP ライブラリーのデフォルトが "compat" に変更

バージョン 10.1 では、新しい OpenMP ライブラリー・セットが追加され、アプリケーションは、インテル® コンパイラーと gcc* コンパイラーの両方からの OpenMP コードを使用することが可能でした。この "互換" ライブラリーは古い "レガシー" ライブラリーよりも高いパフォーマンスを提供します。バージョン 11.x では、互換ライブラリーが OpenMP アプリケーションのデフォルト・ライブラリーとして使用されるようになりました。

-openmp-lib compat と等価です。古いライブラリーを使用する場合は、
-openmp-lib legacy を指定してください。

"レガシー" ライブラリーは、インテル® コンパイラーの将来のリリースからは削除される予定です。

3.4.6 OpenMP ライブラリーのデフォルトがダイナミック・リンクに変更

バージョン 11.0 より、デフォルトで OpenMP アプリケーションはダイナミック OpenMP ライブラリーにリンクされます。OpenMP ライブラリーのスタティック・リンクを指定するには、-openmp-link static を指定します。

3.4.7 サンプリング・ベースのプロファイルに基づく最適化機能の削除

ハードウェア・サンプリング・ベースのプロファイルに基づく最適化機能は提供されなくなりました。この変更に伴い、-prof-gen-sampling と -ssp の 2 つのコンパイラー・オプション、および profrun と pronto_tool の 2 つの実行ファイルが削除されました。インストルメント形式のプロファイルに基づく最適化機能は従来どおり利用できます。

3.5 既知の問題

3.5.1 KMP_AFFINITY のデフォルト動作が変更

スレッド・アフィニティー型の KMP_AFFINITY 環境変数のデフォルトは none (KMP_AFFINITY=none) です。KMP_AFFINITY=none の動作は、10.1.015 以降で変更されており、すべての 11.x コンパイラーでは、初期化スレッドによりマシン上の全スレッドの「フルマスク」が作成され、起動時に各スレッドはこのマスクにバインドします。この変更により、その他のプラットフォームのアフィニティー・メカニズム (SGI Altix マシンの dplace() など) に影響する可能性があることが判明しました。この問題を解決するため、新しいアフィニティー型の disabled がコンパイラー 10.1.018 とすべての 11.x コンパイラー (KMP_AFFINITY=disabled) で導入されています。KMP_AFFINITY=disabled を設定すると、OpenMP ランタイム・ライブラリーによるアフィニティー関連のシステムコールが回避されます。

3.5.2 古いバージョンの ld による致命的なエラー

状況により、アプリケーションをバージョン 11.x コンパイラーにリンクすると次のような内部エラーが発生し、アプリケーションが失敗することがあります。

```
ld: BFD 2.15.92.0.2 20040927 内部エラー。bfd_get_reloc_size の  
../bfd/reloc.c 行 444 で終了します。  
ld: この問題を報告してください。
```

この問題を解決するには、最新バージョンの binutils をインストールしてください。推奨する最小バージョンは、2.17.50 です。

3.5.3 空の派生型の制限付きサポート

Fortran 2003 では、派生型をデータ・コンポーネントなしで宣言する機能が追加されています。インテル® コンパイラーの現在のリリースでは、このサポートには制限があります。将来のリリースでこの制限は解除される予定です。制限事項は次のとおりです。

- 派生型のオブジェクトが宣言される際、型には少なくとも1つのデータ・コンポーネントがなければなりません。空の型の拡張がサポートされています。次に例を示します。

```
type t
end type
```

```
type, extends (t) :: t1
end type
```

```
type, extends (t1) :: t2
  integer i
end type
```

```
type, extends (t2) :: t3
end type
```

```
type (t) :: rec1 ! Not supported, type t is empty
type (t1) :: rec2 ! Not supported, type t1 is empty
type (t2) :: rec3 ! Supported, type t2 is not empty
type (t3) :: rec4 ! Supported, type t3 is not empty
```

例外として、クラスオブジェクトの空の型での宣言はサポートされています。次に例を示します。

```
class(t1) :: rec5
```

サポートされていない空の型が見つかったと、次のメッセージが表示されます。

```
Declaring an object with no data component fields is not yet
supported (データ・コンポーネント・フィールドのないオブジェクトの宣言はサ
ポートされていません)
```

- 空の型のコンポーネントの参照はサポートされていません。例えば、上記の宣言を仮定します。

```
call sub(rec4%t3, rec4%t1, rec3%t)
print *, rec3%t1, rec4%t
call sub2(rec3%t2, rec4%t2)
```

rec4%t3、rec4%t1、rec4%t、rec3%t1、rec3%t への参照はサポートされていません。rec3%t2 と rec4%t2 の参照はサポートされます。サポートされていない参照が見つかったと、次のメッセージが表示されます。

```
Accessing an empty type is not yet supported (空の型へのアクセスは
サポートされていません)
```

- 空の型の型コンストラクターはサポートされていません。上記の宣言を例にとると、型コンストラクター `t()` はサポートされていません。サポートされていないコンストラクターが見つかったと、次のメッセージが表示されます。

A type constructor for an empty type is not yet supported (空の型の型コンストラクターはサポートされていません)

3.5.4 RedHat Enterprise Linux 5.4 の gdb ブレークポイントの設定問題

RedHat Enterprise Linux 5.4 で提供されている gdb バージョンには既知の問題があり、言語が Fortran に設定されている場合にブレークポイントを設定できないことがあります。この問題は、Bugzilla No. 559291 として RedHat に報告済みです。この問題は、一時的に言語設定を C に変更してブレークポイントの設定を行ってから、言語設定を Fortran に戻すことで回避できます。

3.6 Fortran 2003 機能の概要

インテル® Fortran コンパイラーは、最新の Fortran 規格である、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
- 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
- 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
- TYPE CONTAINS 宣言
- ABSTRACT 属性
- DEFERRED 属性
- NON_OVERRIDABLE 属性
- 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
- 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 ("assume realloc_lhs" オプションが必要)
- 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
- プロシージャ・ポインター
- ABSTRACT INTERFACE
- PASS 属性と NOPASS 属性
- SYSTEM_CLOCK 組み込み関数の COUNT_RATE 引数が任意の種類で 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 でゼロを返します。
- 型問い合わせ組み込み関数
- COMMAND_ARGUMENT_COUNT 組み込み関数
- EXTENDS_TYPE_OF と SAME_TYPE_AS 組み込み関数
- GET_COMMAND 組み込み関数
- GET_COMMAND_ARGUMENT 組み込み関数
- GET_ENVIRONMENT_VARIABLE 組み込み関数
- IS_IOSTAT_END 組み込み関数
- IS_IOSTAT_EOR 組み込み関数
- MAX/MIN/MAXVAL/MINVAL/MAXLOC/MINLOC 組み込み関数 (CHARACTER 引数)
- MOVE_ALLOC 組み込み関数
- NEW_LINE 組み込み関数
- SELECTED_CHAR_KIND 組み込み関数
- 次の組み込み関数はオプションの KIND= 引数を使用: ACHAR, COUNT, IACHAR, ICHAR, INDEX, LBOUND, LEN, LEN_TRIM, MAXLOC, MINLOC, SCAN, SHAPE, SIZE, UBOUND, VERIFY
- ISO_C_BINDING 組み込みモジュール
- IEEE_EXCEPTIONS、IEEE_ARITHMETIC、IEEE_FEATURES 組み込みモジュール
- ISO_FORTRAN_ENV 組み込みモジュール

サポートされていない Fortran 2003 機能には次の項目が含まれます。

- 型バインド・プロシージャの型バインド操作と GENERIC バインド
- ユーザー定義の派生型 I/O
- パラメーター化された派生型

4 インテル® デバッガー (IDB)

次の注意事項は、IA-32 アーキテクチャー・システムおよびインテル® 64 アーキテクチャー・システムで実行するインテル® デバッガー (IDB) のグラフィカル・ユーザー・インターフェイス (GUI) についてです。このバージョンでは、`idb` コマンドは GUI を起動します。コマンドライン・インターフェイスを起動するには、`idbc` を使用します。

IA-64 アーキテクチャー・システムでは、GUI は利用できません。`idb` コマンドはコマンドライン・インターフェイスを起動します。

4.1 Java ランタイム環境の設定

インテル® IDB デバッガーのグラフィカル環境は、Java アプリケーションで構築されており、実行には Java ランタイム環境 (JRE) が必要です。デバッガーは、5.0 (1.5 と呼ばれます) または 6.0 JRE をサポートしています。

配布元の手順に従って JRE をインストールします。

最後に、JRE のパスを設定する必要があります。

```
export PATH=<path_to_JRE_bin_dir>:$PATH
```

4.2 デバッガーの起動

デバッガーを起動するには、まず始めに、「[コンパイラー環境の構築](#)」で説明されているコンパイラー環境が構築されていることを確認してください。その後、次のコマンドを使用します。

```
idb
```

または

```
idbc
```

(必要に応じて)

GUI が開始され、コンソールウィンドウが表示されたら、デバッグセッションを開始できます。

注: デバッグする実行ファイルが、デバッグ情報付きでビルドされ、実行可能ファイルであることを確認してください。必要に応じて、アクセス権を変更します。

例:

```
chmod +x <application_bin_file>
```

4.3 その他のドキュメント

インテル® コンパイラー/インテル® デバッガー・オンライン・ヘルプは、デバッガーのグラフィカル・ユーザー・インターフェイスの [Help (ヘルプ)] > [Help Contents (ヘルプ目次)] で表示できます。

[Help (ヘルプ)] ボタンが表示されているデバッガーのダイアログから状況依存ヘルプにもアクセスできます。

4.4 デバッガー機能

4.4.1 IDB の主な機能

デバッガーは、インテル® IDB デバッガーのコマンドライン・バージョンのすべての機能をサポートしています。デバッガー機能は、デバッガー GUI または GUI コマンドラインから呼び出すことができます。グラフィカル環境を使用する場合は、既知の制限を参照してください。

4.4.2 新機能と変更された機能

- IA-32 およびインテル® 64 アーキテクチャー用のデバッガー GUI
- 並列実行デバッグサポート
- セッションコンセプト
- ビットフィールド・エディター
- SIMD レジスターウィンドウ
- OpenMP サポート
 - タスク、バリア、タスクウェイト、ロック、チームおよびタスク・スパーン・ツリーの情報ウィンドウ
 - データ共有イベントと再入可能な呼び出しの検出
 - 再コンパイル不要のシリアルコードのデバッグ
- 国際化サポート

4.5 既知の問題

4.5.1 データ共有検出の問題

- [Stop on Event (イベントで停止)] アイコンが無効な場合、または [Parallel (並列)] > [Stop on Event (イベントで停止)] メニュー項目がオフの場合、データ共有イベントは [Data Sharing Events (データ共有イベント)] ウィンドウでは収集されません。デバッガーを停止し、[Data Sharing Events (データ共有イベント)] ウィンドウを開いた場合、最後のイベントのみが表示されます。
- [Data Sharing Events (データ共有イベント)] ウィンドウを閉じ、再度開くと、最近のイベントと重複する新しい解析実行ノードが表示されます。
- データ共有検出中に [Data Sharing Events (データ共有イベント)] ウィンドウを閉じると、検出後、ウィンドウを再度開いたときに最後のイベントのみが表示されます。

4.5.2 [Signals (シグナル)] ダイアログが動作しない

GUI ダイアログの [Debug (デバッグ)] > [Signal Handling (シグナル処理)]、またはショートカット・キーの Ctrl+S でアクセス可能な [Signals (シグナル)] ダイアログが正しく動作しないことがあります。シグナル・コマンドライン・コマンドを代わりに使用する場合は、インテル® デバッガー (IDB) マニュアルを参照してください。

4.5.3 再入可能呼び出しにおけるブレークの未サポート

このバージョンのデバッガーでは、再入可能呼び出しでブレーク機能はサポートされていません。[再入可能呼び出しでブレーク...] メニューアイコンをクリックするか、メニューから [Parallel (並列)] > [Break on Re-entrant Call... (再入可能呼び出しでブレーク...)] を選択して、条件を設定することはできますが、再入可能呼び出しが発生してもデバッガーは結果を表示しません。

4.5.4 GUI のサイズ調整

デバッガーの GUI ウィンドウのサイズが小さくなり、一部のウィンドウが表示されていないことがあります。ウィンドウを拡大すると、隠れているウィンドウが表示されます。

4.5.5 プロセスの終了

デバッガーの実行中は、[Debug (デバッグ)] メニューの [Kill Focused Process (フォーカスがあるプロセスの終了)] コマンドは動作しません。最初にデバッガーを停止してから、プロセスを終了してください。

4.5.6 10 進浮動小数点の未サポート

インテル® デバッガーでは、一部の C++ コンパイラーでサポートされている 10 進浮動小数点データ型はサポートされていません。そのため、デバッガーではそのような変数は文字配列のように表示されます。

4.5.7 \$cdir ディレクトリー、\$cwd ディレクトリー

\$cdir はコンパイル・ディレクトリーです (記録されている場合)。\$cdir は、ディレクトリーが設定されている場合にサポートされます。シンボルとしてサポートされるわけではありません。

\$cwd は現在の作業ディレクトリーです。セマンティクスもシンボルもサポートされていません。

\$cwd と ! の違いは、\$cwd はデバッグセッション中に変更された現在の作業ディレクトリーを追跡する点です。! は、ソースパスへのエントリーが追加されると直ちに現在のディレクトリーに展開されます。

4.5.8 info stack の使用

デバッガーコマンド `info stack` は、以下のオプションの構文では現在、負のフレームカウントをサポートしていません。

```
info stack [num]
```

フレームカウント `num` が正の場合、最内 `num` フレームを出力します。カウントが負またはゼロの場合、(最外 `num` フレームを出力するのではなく) フレームを出力しません。

4.5.9 \$stepg0 のデフォルト値の変更

デバッガー変数 `$stepg0` のデフォルト値が 0 に変更されました。値 "0" の設定では、"step" コマンドを使用する場合、デバッガーはデバッグ情報なしでコードにステップオーバーします。以前のデバッガーバージョンと互換性を保つようするには、次のようにデバッガー変数を 1 に設定します。

```
(idb) set $stepg0 = 1
```

4.5.10 一部の Linux システムでの SIGTRAP エラー

特殊な Linux カーネルを使用する一部のケースでは、デバッガーがブレークポイントで停止した後、ユーザーがデバッグを続行すると SIGTRAP エラーが発生することがあります。この問題を回避するには、SIGTRAP シグナルを次のようにコマンドラインで定義します。

```
(idb) handle SIGTRAP nopass noprint nostop
SIGTRAP is used by the debugger.
SIGTRAP      No      No      No      Trace/breakpoint trap
(idb)
```

4.5.11 MPI プロセスのデバッグには idb GUI は使用できません

MPI プロセスのデバッグに idb GUI を使用することはできません。コマンドライン・インターフェイス (idbc) を使用してください。

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® Fortran コンパイラー・プロフェッショナル・エディションに同梱されているインテル® 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 本バージョンでの変更

本バージョンおよび以前のバージョンの変更についての詳細は、<http://software.intel.com/en-us/articles/new-in-intel-mkl-10-2/> (英語) を参照してください。

5.1.1 インテル® マス・カーネル・ライブラリー 10.2 Update 5

新機能

- LAPACK 3.2.1 アップデート (主に LAPACK 3.2 に関する修正) に対応

パフォーマンスの向上

- FFT
 - インテル® 64 アーキテクチャー上で 3 次元以上の複素数 FFT のパフォーマンスが向上
- VSL
 - 45nm インテル® Core™2 Duo プロセッサー以降の 64 ビット・ライブラリーで、MT19937 と MT2203 基本乱数ジェネレーター (BRNG) のパフォーマンスが向上

ユーザービリティとインターフェイスの向上

- uBLAS の例で Boost 1.41.0 のサポートを追加
- 対角優位ソルバー関数 (?DTSVB、?DTTRFB、?DTTRSB) に Fortran 95 インターフェイスを追加
- インプレース多次元クラスター FFT のメモリー消費を大幅に削減

5.1.2 インテル® マス・カーネル・ライブラリー 10.2 Update 4

新機能

- 単精度複素数の絶対値を求める SCABS1 関数を追加
- 部分的なピボット演算を使用する一般的なソルバー (?GTSV) よりも 2 倍高速な対角優位の三重対角方程式用の ?DTSVB ソルバーを追加
- 対角優位の三重対角方程式用の因数分解ルーチン (?DTTRFB) と 前方/後方代入ルーチン (?DTTRSB) を追加

パフォーマンスの向上

- FFT
 - 8 または 13 の倍数の変換のパフォーマンスが向上
 - ベクトル長が 2 のべき乗でない 1D 複素数クラスター FFT を最適化
- VSL
 - デシメーションを必要とする畳み込み/相関演算のパフォーマンスが大幅に向上 (再リンクが必要。「[既知の問題](#)」を参照)

5.1.3 インテル® マス・カーネル・ライブラリー 10.2 Update 3

パフォーマンスの向上

- BLAS
 - インテル® Core™ i7 プロセッサおよびインテル® Xeon® プロセッサ 5300 番台、5400 番台、5500 番台で次の BLAS レベル 1、2 関数の 32 ビット OS バージョンがスレッド化: (D,S,C,Z)COPY、(D,S,C,Z)SWAP、(D,S,C,Z)AXPY、(S,C)ROT、(S,C)DOT、CDOTC、(D,S,C,Z)GEMV、(D,S,C,Z)TRMV、(S,C)SYMV、(S,C)SYR、(S,C)SYR2
 - インテル® Xeon® プロセッサ 5300、5400、5500 で次の BLAS レベル 1 関数の 32 ビットおよび 64 ビット OS バージョンが向上: ZAXPY、ZSCAL、ZDOT(U,C)、(D,S)ROT
 - インテル® Xeon® プロセッサ 5300 で列より非常に多い行を持つ行列の DGEMM スレッド化効率が向上
- LAPACK
 - 次の LAPACK 関数のスケラビリティが向上: ?POTRF、?GEBRD、?SYTRD、?HETRD、?STEDC 分割統治固有ソルバー
- FFT
 - 下層のカーネルが更新され、FFT において幅広くパフォーマンスが向上
 - 1 つの関数呼び出しで少数の変換が計算される場合に 3D FFT のスレッド化が向上
 - スレッド化を小規模の多次元変換に拡張
- VML
 - インテル® Xeon® プロセッサ 5500 番台の VML 関数がさらに最適化: v(s,d)Asin、v(s,d)Acos、v(s,d)Ln、v(s,d)Log10、vsLog1p、v[s/d]Hypot
- VSL
 - viRngPoisson および viRngPoissonV 乱数ジェネレーターのパフォーマンスが向上

ユーザービリティとインターフェイスの向上

- uBLAS、Java、FFTW3、LAPACK95、BLAS95 のサンプルプログラムの向上
- examples ディレクトリーとリファレンス・マニュアルのソースコードにある同一のサンプルは、リファレンス・マニュアルでは削除
- クラスター FFT の新しい 64 ビット整数 (ILP64) fftw_mpi インターフェイス

5.1.4 インテル® マス・カーネル・ライブラリー 10.2 Update 2

新機能

- LAPACK 3.2
 - 238 個の新しい LAPACK 関数
 - 超精密反復法の改良
 - ハウスホルダー QR 因数分解の非負対角
 - 低プロファイル行列でのハイパフォーマンス QR とハウスホルダー反射
 - 高速で正確な新しいヤコビ法 SVD
 - 矩形フル圧縮形式のルーチン
 - ピボットコレスキー
 - 混合精度反復法の改良 (コレスキー)
 - より安定した DQDS アルゴリズム
- DZGEMM 拡張 BLAS 関数の実装 (<http://www.netlib.org/blas/blast-forum/> の説明を参照)。リファレンス・マニュアルの BLAS セクションの *gemm 関数ファミリーの説明を参照してください。
- PARDISO で実数、複素数、単精度データをサポート

ユーザービリティ/インターフェイスの向上

- スパース行列形式変換ルーチン:
 - CSR (3-配列バリエーション) ↔ CSC (3-配列バリエーション)
 - CSR (3-配列バリエーション) ↔ 対角形式
 - CSR (3-配列バリエーション) ↔ スカイライン
- Fortran95 BLAS と LAPACK のコンパイル・モジュール・ファイル (.mod) を追加
 - モジュールは、インテル® Fortran コンパイラーで事前にビルドされており、インクルード・ディレクトリーにあります (フルパス情報については、インテル® MKL ユーザーズ・ガイドを参照してください)。
 - ほかのコンパイラー用のソースも提供されています。
 - インターフェイスについてのドキュメントは、インテル® MKL ユーザーズ・ガイドを参照してください。
- FFTW3 インターフェイスを直接メイン・ライブラリーに統合
 - デフォルトのインテル® Fortran コンパイラー規則と名前修飾で互換性のないコンパイラーでラッパーを作成するためのソースコードも提供されています。
 - 詳細は、リファレンス・マニュアルの付録 G を参照してください。
- DFTI_DESCRIPTOR_HANDLE を型の名前に追加。ユーザープログラムで型として参照できます。
- ユーザーデータへのアクセスを可能にするために最適化ソルバードメインのヤコビ行列計算ルーチンにパラメーターを追加 (詳細は、リファレンス・マニュアルの djacobix 関数の説明を参照してください)
- 64 ビット・アーキテクチャーでインテル® MKL の単精度 BLAS 関数 (頭文字 "s" または "c" の関数) から 64 ビット浮動小数点精度関数へのインターフェイス・マッピング呼び出しを追加 (詳細は、インテル® MKL ユーザーズ・ガイドの「sp2dp」を参照してください)
- 互換ライブラリー (「ダミーライブラリー」) を削除

パフォーマンスの向上

- インテル® 64 アーキテクチャー用にさらにスレッド化された BLAS レベル 1、2 関数
 - レベル 1 関数 (ベクトル-ベクトル): (CS,ZD,S,D)ROT、(C,Z,S,D)COPY、(C,Z,S,D)SWAP
 - キャッシュのデータ位置に応じて、4 コアのインテル® Core™ i7 プロセッサ上でバージョン 10.1 Update 1 に対してパフォーマンスが最大 1.7 ~ 4.7 倍向上
 - キャッシュのデータ位置に応じて、24 コアのインテル® Xeon® プロセッサ 7400 番台システム上でバージョン 10.1 Update 1 に対してパフォーマンスが最大 14 ~ 130 倍向上
 - レベル 2 関数 (行列-ベクトル): (C,Z,S,D)TRMV、(S,D)SYMV、(S,D)SYR、(S,D)SYR2
 - キャッシュのデータ位置に応じて、4 コアのインテル® Core™ i7 プロセッサ上でバージョン 10.1 Update 1 に対してパフォーマンスが最大 1.9 ~ 2.9 倍向上
 - キャッシュのデータ位置に応じて、24 コアのインテル® Xeon® プロセッサ 7400 番台システム上でバージョン 10.1 Update 1 に対してパフォーマンスが最大 16 ~ 40 倍向上
- インテル® Core™ i7 プロセッサ、インテル® Xeon® プロセッサ (5300 番台、5400 番台、7400 番台) で、DSYRK の 32 ビット逐次バージョンに導入された再帰アルゴリズムのパフォーマンスが最大 20% 向上。
- インテル® Xeon® 7460 プロセッサで、大規模な問題の LU 因数分解 (DGETRF) がバージョン 10.1 Update 1 に対して 25% 向上。また小規模な問題でも劇的に向上。
- BLAS *TBMV/*TBSV 関数でレベル 1 BLAS 関数を使用。インテル® Core™ i7 プロセッサ上で最大 3%、インテル® Core™2 プロセッサ 5300 番台と 5400 番台で最大 10% のパフォーマンスが向上。
- DGEMM パフォーマンスを強化するスレッド化アルゴリズムの向上
 - 8 スレッドで最大 7% の向上、3、5、7 スレッドで最大 50% の向上 (インテル® Core™ i7 プロセッサ)
 - 3 スレッドで最大 50% の向上 (インテル® Xeon® プロセッサ 7400 番台)
- 非素数サイズのスレッド化 1D 複素数-複素数 FFT
- 3D 複素数-複素数変換の新しいアルゴリズムにより 1 スレッドまたは 2 スレッドで小さな問題サイズ (最大 64x64x64) についてより優れたパフォーマンスを提供
- 対称正定行列の演算時におけるアウトオブコア (OOC) PARDISO のハイレベルな並列化実装
- すべての行列の型でインコアとアウトオブコアの両方で PARDISO のメモリー使用量が減少
- 実対称行列、複素エルミート行列、複素対称行列に対し PARDISO OOC で使用されるメモリーがインテル® MKL 10.1 で使用されていたメモリーの半分以下まで減少
- PARDISO/DSS における順序付けの並列化とシンボリック因子分解
- インテル® Core® i7 プロセッサとインテル® Core™2 プロセッサで次の VML 関数において最大 2 倍のパフォーマンスの向上 (平均で 30% の向上): v(s,d)Round、v(s,d)Inv、v(s,d)Div、v(s,d)Sqrt、v(s,d)Exp、v(s,d)Ln、v(s,d)Atan、v(s,d)Atan2
- インテル® Advanced Vector Extension (インテル® AVX) で次の関数の最適化バージョンが利用可能
 - BLAS: DGEMM
 - FFT
 - VML: exp、log、pow
 - 上記の関数にアクセスする `mkl_enable_instructions()` 関数に関する重要な情報については、インテル® MKL ユーザーズ・ガイドを参照してください。

5.2 既知の問題

本リリースにおける既知の制限事項の詳細なリストは、<http://software.intel.com/en-us/articles/intel-math-kernel-library-support-resources/> (英語) を参照してください。

5.3 注意事項

インテル® MKL の将来のバージョンでは以下の変更が予定されています。「[テクニカルサポート](#)」を参照してください。

- ファイル名に `solver` を含むライブラリーの内容をコア・ライブラリーに移動する予定です。これらの `solver` ライブラリーはその後削除される予定です。

5.4 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。本リリースのインテル® MKL の一部の FFT 関数は、ヒューストン大学からライセンスを受けて、UHFFT ソフトウェア生成システム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。

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ます。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、インテルの Web サイトを参照してください。

本製品の一部は、オープンソースのライブラリーを使用してビルドされています。これらのライブラリーのライセンス規約に従い、インテルでは本製品のユーザーがライブラリーを利用できるようにしています。ライブラリーは、インテル® ソフトウェア開発製品のナレッジベース記事 (<http://software.intel.com/en-us/articles/open-source-downloads/>) からダウンロードが可能です。これらのライブラリーは、本製品の使用には必須ではないことに注意してください。

Intel、インテル、Intel ロゴ、Intel Core、Itanium、Pentium、Xeon は、アメリカ合衆国および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。

© 2010 Intel Corporation. 無断での引用、転載を禁じます。