

インテル® Fortran コンパイラー 11.0 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
2008.10.22

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 製品の内容

インテル® Fortran コンパイラー 11.0 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
- クラスター OpenMP* (個別にライセンスが必要)
- 各種ドキュメント

1.2 動作環境

1.2.1 プロセッサ用語

インテル® コンパイラーは、一般的なプロセッサとオペレーティング・システムを組み合わせた3つのプラットフォームをサポートしています。このセクションでは、本ドキュメント、インストール手順、およびサポートサイトでプラットフォームの記述に使用されている用語について説明します。

- **IA-32 アーキテクチャー:** 32 ビットのエペレーティング・システム ("Linux x86") を実行している、インテル® Pentium® II プロセッサと互換性のある 32 ビット・プロセッサ (インテル® Pentium® 4 プロセッサ、インテル® Xeon® プロセッサなど)、または同じ命令セットをサポートしている他社製のプロセッサがベースのシステムを指します。
- **インテル® 64 アーキテクチャー:** 64 ビット・アーキテクチャーに対応するように拡張された IA-32 アーキテクチャー・プロセッサ (インテル® Core™2 プロセッサ・ファミリーなど) をベースとし、64 ビット・オペレーティング・システム ("Linux x86_64") を実行するシステムを指します。32 ビットの Linux オペレーティング・システムを実行しているシステムは、IA-32 とみなされます。"Linux x86_64" オペレーティング・システムを実行する AMD* プロセッサ・ベースのシステムも、インテル® 64 対応アプリケーション用インテル® コンパイラーでサポートされています。

- **IA-64 アーキテクチャー:** 64 ビット・オペレーティング・システムを実行している、インテル® Itanium® プロセッサ・ベースのシステム。

1.2.2 ネイティブおよびクロスプラットフォーム開発

「ネイティブ」とは、アプリケーションを実行するプラットフォームと同じプラットフォームでアプリケーションをビルドする (例えば、IA-32 システムで実行するアプリケーションを IA-32 システムでビルドする) ことを指します。「クロスプラットフォーム」または「クロスコンパイル」とは、アプリケーションを実行するプラットフォームとは異なる種類のプラットフォームでアプリケーションをビルドする (例えば、IA-64 システムで実行するアプリケーションを IA-32 システムでビルドする) ことを指します。すべての組み合わせのクロスプラットフォーム開発がサポートされているわけではありません。また、組み合わせによっては、オプションのツールとライブラリーをインストールする必要があります。

サポートされているホスト (アプリケーションをビルドするシステム) とターゲット (アプリケーションを実行するシステム) の組み合わせを次に示します。

- ホスト: IA-32 システム、サポートターゲット: IA-32
- ホスト: インテル® 64 システム、サポートターゲット: IA-32 とインテル® 64
- ホスト: IA-64 システム、サポートターゲット: IA-64

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。

1.2.3 要件

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 アーキテクチャー・プロセッサ (例: インテル® Pentium® 4 プロセッサ)、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システム
- RAM 512MB (1GB 推奨)
- 1.5GB 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* 9
 - Red Hat* Enterprise Linux* 3、4、5
 - SUSE* LINUX Enterprise Server* 9、10
 - TurboLinux* 11
 - Ubuntu* 8.04
- 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® 64 アーキテクチャー・システムで開発する場合は、Linux コンポーネントの crt1.o を提供する glibc-devel.i386

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 対応システムまたは AMD 64 ビット・プロセッサをベースとしたシステム
- RAM 512MB (1GB 推奨)
- 1.5GB 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 9
 - Red Hat Enterprise Linux 3、4、5
 - SGI* ProPack* 5
 - SUSE LINUX Enterprise Server 9、10
 - TurboLinux 11
 - Ubuntu 8.04
- 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++
- 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

- インテル® Itanium® プロセッサ・ベースのシステム
- RAM 512MB (1GB 推奨)
- 1.5GB 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
 - TurboLinux 11
 - Ubuntu 8.04
- 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 5 (1.5.0)

インテル® クラスター OpenMP (オプション機能) を使用するためのその他の要件

- 必要な最小ハードウェア構成 (ノード単位):
 - インテル® 64 アーキテクチャー・システムまたは IA-64 アーキテクチャー・システム
- 推奨ハードウェア (ノード単位):
 - RAM 2GB
 - 10GB のディスク空き容量
- オペレーティング・システム:
 - Red Hat Enterprise Linux 3.0、4.0
 - SUSE Linux Enterprise Server 9.0、10.0
- POSIX* スレッド: NPTL

Infiniband* サポートの場合

- Open Fabrics Enterprise Distribution (OFED) 1.0 以降

OFED ソフトウェアは、<https://svn.openfabrics.org/svn/openib/gen2/branches/> (英語) からダウンロードできます。

推奨ソフトウェア

- インテル® トレース・アナライザー/コレクター
- インテル® スレッド・プロファイラー
- インテル® スレッド・チェッカー

クラスター要件

サポートされている通信ファブリック (Ethernet*、Gigabit Ethernet*、Infiniband*、または TCP/IP をサポートする任意のファブリック) のいずれか。

クラスター OpenMP プログラムの実行に関わるすべてのノードは、同じオペレーティング・システムと同じカーネルバージョンを実行している必要があります。また、実装されているファイルシステムやシステムパスに関してできる限り同一なものにしなければなりません。

ディスクには十分なスワップ領域が必要です。Linux プログラムで必要な通常のスワップ領域に加え、クラスター OpenMP は、共有可能な外部記憶装置用に個別に割り当てたディスク空き容量が必要です。共有可能な外部記憶装置はデフォルトで /tmp に割り当てられ、プログラムに割り当てられている共有可能なページの 2 倍の空き容量が必要です。kmp_cluster.ini ファイルにある --backing-store オプションを使用して、/tmp 以外のディレクトリーに共有外部記憶装置を割り当ててください。

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、[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。

1.2.4 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 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。

1.3.1 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。

- インテル® Fortran コンパイラー 11.0 プロフェッショナル・エディションでは、IA-32 およびインテル® 64 アーキテクチャー・システムに搭載の Ubuntu 8.04 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 8.04 を搭載しているインテル® 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は、本リリースでは正式にサポートされていませんが、同様の問題があ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 を搭載したインテル® 64 アーキテクチャー・システムで、インテル® Fortran コンパイラー 11.0 プロフェッショナル・エディションの 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](#) (英語) で評価版のシリアル番号を入手してください。

1.4 インストール先フォルダー

11.0 製品は、前のバージョンとは異なる構成でフォルダーにインストールされます。新しい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/11.0/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
 - compiler_f
 - idb
 - mkl
- 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 桁のアップデート番号です。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。

1.5 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。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。

1.6 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。

1.7 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/flin> (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。

2.1 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 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ーク間の最適化 (-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。

注: バージョン 11 では、IA-32 システムのデフォルトでのコード生成において、アプリケーションを実行するシステムでインテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) がサポートされていると仮定するように変更されました。詳細は、[下記を参照](#)してください。

2.2 新機能と変更された機能

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。

- Fortran 2003 の機能
 - 列挙子
 - 型拡張子 (ポリモアフィックではありません)
 - 割り当て可能なスカラー変数 (長さ無指定文字ではありません)
 - ALLOCATE と DEALLOCATE の ERRMSG キーワード
 - ALLOCATE の SOURCE= キーワード
 - MAX/MIN/MAXVAL/MINVAL/MAXLOC/MINLOC と CHARACTER 引数
 - IEEE_EXCEPTIONS、IEEE_ARITHMETIC、IEEE_FEATURES 組み込みモジュール
 - ASSOCIATE 構造
 - PROCEDURE 宣言
 - プロシージャーク・ポインター
 - ABSTRACT INTERFACE
 - PASS 属性と NOPASS 属性
 - 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
 - BLANK、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
- OpenMP* 3.0 のサポート
- UNROLL_AND_JAM 宣言子と NOUNROLL_AND_JAM 宣言子
- VECTOR NONTEMPORAL 宣言子で変数名の指定が可能

- VECTOR TEMPORAL 宣言子

2.3 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。

- -axSSE2
- -axSSE3
- -axSSSE3
- -axSSE4.1
- -axSSE4.2
- -diag-error-limit
- -diag-once
- -falign-stack
- -fast-transcendentals
- -finline
- -fma
- -fp-relaxed
- -fpie
- -fstack-protector
- -m32
- -m64
- -mia32
- -minstruction
- -mssse3
- -msse4.1
- -openmp-link
- -openmp-threadprivate
- -opt-block-factor
- -opt-jump-tables
- -opt-loadpair
- -opt-mod-versioning
- -opt-prefetch-initial-values
- -opt-prefetch-issue-excl-hint
- -opt-prefetch-next-iteration
- -opt-subscript-in-range
- -pie
- -prof-data-order
- -prof-func-order
- -prof-hotness-threshold
- -prof-src-dir
- -prof-src-root
- -prof-src-root-cwd
- -tcollect-filter
- -vec
- -xHost
- -xSSE2
- -xSSE3
- -xSSE3_ATOM
- -xSSSE3

- -xSSE4.1
- -xSSE4.2

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。

2.3.1 -xHost オプション

バージョン 11.0 から -xHost オプションが新しく追加されました。このオプションは、ソースをコンパイルするシステム上のプロセッサの種類に基づいて命令セットを自動的に選択します。動作は次のとおりです。

システムのプロセッサ	使用されるオプション
インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2) 以上の命令をサポートするインテル® プロセッサ	-xSSE4.2、-xSSE4.1、-xSSSE3、-xSSE3 または -xSSE2 をプロセッサに応じて使用
古いインテル® プロセッサ	-mia32
インテル以外のプロセッサ	-mSSE3、-mSSE2 または -mia32 を CPUID フラグに応じて使用

命令セットオプションを使用する場合、指定した命令セットがアプリケーションを実行するシステムでサポートされていることを確認してください。アプリケーションのコンパイルと実行に同じシステムを使用する場合は、-xHost オプションを使用することを推奨します。

2.4 その他の変更および注意

2.4.1 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 11.0 では、スクリプトは 1 つのバージョンしかなく、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/11.0/xxx/bin/ifortvars.sh argument
```

<install-dir> はインストール・ディレクトリ (デフォルトのインストール先は /opt/intel) で、xxx はアップデート番号です。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 (ldb) 環境も構築されます。

2.4.2 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

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 プロセッサや特定の AMD* プロセッサなど、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。

2.4.3 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 11.0 では、デフォルトで互換ライブラリーが OpenMP アプリケーションで使用されます。`-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。

2.4.4 プロシージャー・ポインターおよび PASS 属性

バージョン 11 では、Fortran 2003 のプロシージャー・ポインター機能をサポートします。プロシージャー・ポインターは、派生型のコンポーネントとして扱われます。PASS および NOPASS 属性は、囲まれた派生型が被呼び出しプロシージャーへの引数として渡されるかどうかを指定します。標準では、PASS 属性のプロシージャー・ポインターはポリモフィックである必要があります。バージョン 11.0 では、ポリモフィック・プロシージャーはまだサポートしていません。このため、PASS 属性のプロシージャー・ポインターを使用するプログラムを記述することは可能ですが、標準に準拠する方法で行うことはできません。

2.4.5 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。インストール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。

2.5 既知の問題

2.5.1 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 11.0 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 11.0 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。

2.6 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 (長さ無指定文字ではありません)
- ALLOCATE と DEALLOCATE の ERRMSG キーワード
- ALLOCATE の SOURCE= キーワード
- 型拡張子 (ポリモアフィックではありません)
- 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
- 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=
- 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、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
- COMMAND_ARGUMENT_COUNT 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

3 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) の新しいグラフィカル・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。

3.1 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) 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
```

3.2 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、「[コンパイラ環境の構築](#)」で説明されているコンパイラ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>`

3.3 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。

3.4 デバッガー機能

3.4.1 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。

3.4.2 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 (MMX®) レジスターウィンドウ
- OpenMP 情報ウィンドウ
- FORTRAN の向上
- 国際化サポート
- OpenMP 構成サポート
- クラスター OpenMP サポート

3.5 既知の問題

3.5.1 [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。

3.5.2 共有ライブラリーのデバッグ

実行時に共有ライブラリーをロードするアプリケーションをデバッグする際に、次のエラーが発生することがあります。

Error: could not start debuggee (エラー: debuggee を起動できません。)

Could not start process for <executable> (<executable> のプロセスを開始できません。)

No image loaded ... Recovering ... (イメージはロードされませんでした... 回復中...)

この問題は、LD_LIBRARY_PATH 環境変数を、共有ライブラリーが保存されているディレクトリに設定しても解決されないでしょう。また、このエラーメッセージは誤解を招く恐れもあります。このエラーメッセージは、debuggee は起動されますが、デバッガーが関連付けられた共有ライブラリーを見つけられないことを示しています。

3.5.3 list コマンド

GDB モードでは、引用符が付いていないファイル名は動作しません。ファイル名には引用符を付けてください。(例: `list "test.f90":10`)

3.5.4 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。

3.5.5 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。

3.5.6 並列領域のシリアル化

GUI の [Serialize Parallel Region (並列領域のシリアル化)] ツールバーアイコンが、正常に更新されません。このアイコンは、実際は切り替えボタンで、並列領域のシリアル化機能が有効な際に、アイコンのまわりにフレームを示すものです。フレームがない場合は、無効な状態です。ボタンは正しく表示されません。step/run-stop などの後、機能が有効でも、ボタンは常に無効モードになります。これは表示の問題であり、アイコンにマウスを置くと、次のステップまで状態を正常に表示します。

並列領域のシリアル化オプションは、[Parallel (並列)] メニューから表示された場合は正常に動作します。有効な場合はチェックマークが正常にセットされ、無効な場合はセットされません。

3.5.7 OpenMP ロック: "利用可能な情報がありません"

本リリースでは、OpenMP ランタイム・ライブラリーは Lock、Barrier、Taskwait オブジェクトの情報を提供することができないため、これらのウィンドウには常に "利用可能な情報がありません" と表示されます。ロックの情報は、コンソールウィンドウでコマンドラインから取得することができます。次のコマンドを入力します。

```
idb info lock <lock_id>
```

<lock_id> は、プログラムのロック名です。

3.5.8 オンラインヘルプのエラー “Web ブラウザーを開けません”

IDB からディスクのヘルプにアクセスしようとするすと、「[0] で Web ブラウザを開けません」という旨のエラーメッセージが、特定の Linux ディストリビューションで表示されます。これは、Mozilla* Web ブラウザーが見つからないことが原因で発生します。これを回避するには、Firefox* など、インストールされているブラウザへのシンボリック・リンクを作成してください。次に例を示します。

```
sudo ln -s /usr/bin/firefox /usr/bin/mozilla
```

または、sudo root 権限がない場合:

```
ln -s /usr/bin/firefox <user_dir>/mozilla
```

その後、<user_dir>/Mozilla を \$PATH に追加します。

4 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。

4.1 新機能と変更された機能

- BLAS のパフォーマンスの向上
 - 32 ビット
 - インテル® Xeon® プロセッサー 5300 番台で (Z,C)GEMM が 40-50% 向上
 - インテル® Xeon® プロセッサー 5400 番台で GEMM コードが 10% 向上
 - 64 ビット
 - インテル® Xeon® プロセッサー 5400 番台の 1 つのスレッドで DGEMM が 2.5-3% 向上
 - インテル® Core™ i7 プロセッサー・ファミリーで SGEMM が 50% 向上
 - インテル® Core™ i7 プロセッサー・ファミリーの 1 つのスレッドで CGEMM が 3% 向上
 - インテル® Core™ i7 プロセッサー・ファミリーの 1 つのスレッドで ZGEMM が 2-3% 向上
 - インテル® Core™ i7 プロセッサー・ファミリーで DTRSM の右辺のケースが 30% 向上
- 直接法スパースソルバー (DSS/PARDISO) の改良
 - アウトオブコア PARDISO のパフォーマンスが平均 35% 向上しました。
 - DSS/PARDISO に個別の前進/後退代入のサポートが追加されました。
 - DSS インターフェイスに反復改善をオフにする新しいパラメーターが追加されました。
 - PARDISO インターフェイスにスパース行列構造を確認する新しいパラメーターが追加されました。
- コールバック関数メカニズムから長い計算の状況を追跡する機能と計算を中断する機能が追加されました。mkl_progress という関数をユーザー・アプリケーションで定義して、MKL LAPACK ルーチンのサブセットから呼び出すことができます。詳細は、『リファレンス・マニュアル』の「LAPACK 補助ルーチンとユーティリティー・ルーチン」の章を参照してください。この機能をサポートしている LAPACK 関数を確認するには、各関数の説明を参照してください。

- 転置関数がインテル® MKL に追加されました。詳細は、『リファレンス・マニュアル』を参照してください。
- C++ の `std::complex` 型を MKL 固有の複素数型の代わりに使用できるようになりました。
- Boost uBLAS 行列-行列乗算ルーチンの実装により、インテル® MKL BLAS で DGEMM の高度に最適化されたバージョンを使用できるようになりました。詳細は、『ユーザズガイド』を参照してください。
- スパース BLAS の改良
 - すべてのデータ型 (単精度、複素数および倍精度複素数) のサポートが追加されました。
 - 圧縮スパース行形式で格納された 2 つのスパース行列の和と積を計算するルーチンが追加されました。
- ベクトル・マス・ライブラリー関数 `CdfNorm`、`CdfNormInv`、および `ErfcInv` が最適化され、パフォーマンスが向上しました。
- インテル® Core™ i7 プロセッサ・ファミリーにおけるパフォーマンスの向上
 - 次の VML 関数が 3-17% 向上: `Asin`、`Asinh`、`Acos`、`Acosh`、`Atan`、`Atan2`、`Atanh`、`Cbrt`、`CIS`、`Cos`、`Cosh`、`Conj`、`Div`、`Erflnv`、`Exp`、`Hypot`、`Inv`、`InvCbrt`、`InvSqrt`、`Ln`、`Log10`、`MulByConj`、`Sin`、`SinCos`、`Sinh`、`Sqrt`、`Tanh`。
 - 一様乱数生成が 7-67% 向上しました。
 - Wichmann-Hill、Sobol、および Niederreiter BRNG に基づく VSL 分布生成器が 3-10% 向上しました (64 ビットのみ)。
- 設定ファイルの機能が削除されました。インテル® MKL の動作を設定する代わりにの方法については、『ユーザズガイド』を参照してください。
- インテル® MKL の関数が MPI プログラムから呼び出されると、デフォルトでは (明示的に制御されない限り) 1 つのスレッドで動作します。
- VML 関数 (`CdfNorm`、`CdfNormInv` および `ErfcInv`) が追加されました。
- `DftiCopyDescriptor` 関数が追加されました。
- DSS/PARDISO の LP64 インターフェイスは、64 ビット・オペレーティング・システム上で内部配列に 64 ビット・アドレッシングを使用するようになりました。この変更により、ソルバーでより大きな方程式を解くことができるようになりました。
- インテル® MKL のデフォルトの OpenMP ランタイム・ライブラリーが `libguide` から `libiomp` に変更されました。詳細は、Documentation ディレクトリーにある『ユーザズガイド』を参照してください。
- インテル® Pentium® III プロセッサ用に最適化されたコードパスおよびこのプロセッサ固有のダイナミック・リンク・ライブラリーが削除されました。このプロセッサ上でインテル® MKL は引き続き使用できますが、デフォルトのコードパスが使用されるため、パフォーマンスは低下します。
- 区間線形ソルバー関数が削除されました。
- インテル® MPI 1.x のサポートが終了しました。
- ドキュメントの更新
 - VML 関数と VSL サービス関数で Eclipse IDE の `infopop` がサポートされました。`infopop` のサポートにより、カーソルを Eclipse Editor パネルの関数/ルーチン名に移動すると、関数の情報がポップアップ・ウィンドウで表示されます。この Eclipse 機能は、CDT 5.0 で実装されました。
 - FFTW ラッパーの説明が製品パッケージから削除され、『リファレンス・マニュアル』の付録 G に統合されました。
 - 新しい関数の説明が『リファレンス・マニュアル』に追加され、Boost uBLAS 行列-行列乗算のサポートが『ユーザズガイド』で説明されています。
 - ScaLAPACK をサポートする PBLAS (Parallel BLAS) の説明が『リファレンス・マニュアル』に追加されました。

- 『リファレンス・マニュアル』の VML および VSL 関数の説明に、Fortran 77 のサポートに関する情報が追加されました。

4.2 既知の制限事項

4.2.1 スパースソルバーと最適化ソルバーの制限事項

- スパースソルバーと最適化ソルバー・ライブラリー関数はスタティック形式でのみ提供されます。

4.2.2 FFT 関数の制限事項

- DFTI_TRANSPOSE モードは、デフォルトケースでのみ実装されます。
- DFTI_REAL_STORAGE モードにはデフォルト値のみ指定可能で、DftiSetValue 関数 (例えば、DFTI_REAL_STORAGE = DFTI_REAL_REAL) によって変更することはできません。
- ILP64 バージョンのインテル® MKL では、現在 1 つの次元の長さが $2^{31}-1$ を超える FFT をサポートしていません。 $2^{31}-1$ を超える 1D FFT、またはいずれかの次元が $2^{31}-1$ を超える多次元 FFT では、“DFTI_1D_LENGTH_EXCEEDS_INT32” エラーメッセージが返されます。この制限は、各次元の長さが $2^{31}-1$ を超えない限り、 $2^{31}-1$ 個を超える成分を持つ多次元 FFT には適用されないことに注意してください。
- クラスター FFT 関数の配列サイズでいくつかの制限があります。詳細は、『リファレンス・マニュアル』 (mklman.pdf) を参照してください。
- 動的にリンクされているアプリケーションでクラスター FFT 関数を使用する場合、インテル® MKL のスタティック・インターフェイス・ライブラリーもリンクする必要があります。
例: `-Wl,--start-group $MKL_LIB_PATH/libmkl_intel_lp64.a $MKL_LIB_PATH/libmkl_cdft_core.a -Wl,--end-group $MKL_LIB_PATH/libmkl_blacs_intelmpi20_lp64.a -L$MKL_LIB_PATH -lmkl_intel_thread -lmkl_core -liomp5 -lpthread`

4.2.3 LAPACK 関数の制限事項

- ILAENV 関数 (ローカル環境の問題依存パラメーターを選択するために LAPACK ルーチンから呼び出される) は、ユーザーのバージョンでは代用できません。
- CPU の周波数が一定でない場合、second() および dsecnd() 関数は正しくない結果を返すことがあります。

4.2.4 ベクトル・マス・ライブラリー (VML) 関数と ベクトル・スタティスティカル・ライブラリー (VSL) 関数の制限事項

- mkl_vml.fi を使用すると、TYPE ERROR_STRUCTURE 長に関する警告が生成されることがあります。

4.2.5 ScaLAPACK 関数の制限事項

- PjLAENV 関数はユーザーのバージョンでは代用できません。この関数は、ローカル環境の問題依存パラメーターを選択するために ScaLAPACK ルーチンから呼び出されます。
- ScaLAPACK ライブラリーは、スタティック形式でのみ利用可能です。

4.2.6 インテル® MKL の ILP64 バージョンの制限事項

- ILP64 バージョンのインテル® MKL には、ライブラリーのすべての機能は含まれていません。ILP64 バージョンに含まれる機能の一覧は、Documentation ディレクトリーにある『ユーザーズガイド』を参照してください。
- g77 で ILP64 ライブラリーを使用することはできません。

4.2.7 LAPACK の Fortran 95 インターフェイスの制限事項

- GNU gfortran で LAPACK の Fortran 95 インターフェイスをコンパイルする場合、?GEES、?GEESX、?GGES、?GGESX (? は S、D、C、または Z) プロシージャ引数を含むすべてのサブルーチンから、pure 属性を手動で削除する必要があります。

4.2.8 g77 コンパイラー・サポートの制限事項

- 一部のインテル® MKL 関数は、名前に下線が含まれています (例: mkl_dcsrsymv、mkl_cspblas_dcsrsymv)。これらの関数は g77 のデフォルトの命名規則をサポートしていません。回避策として、-fno-second-underscore コンパイルフラグを使用してください。
例: g77 -fno-second-underscore test.f

4.2.9 その他の制限事項

- MP LINPACK のハイブリッド・バージョンをビルドするときに DHPL_CALL_CBLAS オプションは使用できません。
- インテル® 64 対応システムでは、GNU Fortran コンパイラー (バージョン 3.2.3) でコンパイルされたユーザープログラムは、-fno-f2c GNU Fortran コンパイラー・フラグが使用されていない場合、単精度値を返すインテル® MKL の関数で不正な結果を返すことがあります。GNU Fortran コンパイラーは、デフォルトでリターンレジスターの最初の 8 バイトが REAL*4 である (倍精度値として表現される) と想定しますが、インテル® Fortran コンパイラーは、リターンレジスターの最初の 4 バイトが REAL*4 であると想定します。インテル® MKL の動作は、インテル® Fortran コンパイラーの動作と互換性があります。GNU Fortran コンパイラーの動作は、-fno-f2c フラグを使用することでインテル® Fortran コンパイラーと互換になるように変更されます。
- FFT および PDE サポート関数は Fortran 77 から呼び出すことはできません。これらのコンポーネントは、(構造などが) Fortran-90/95 インターフェイス固有であるため、Fortran 77 では使用できません。
- インテル® コンパイラーでインテル® MKL のサンプル・ソース・コードをコンパイルする場合は、-Od オプションを使用することを推奨します。現在のビルドスクリプトでは、このオプションは指定されません。また、これらのコンパイラーでは、デフォルトでベクトル化を行うように変更されました。
- VSL 関数はすべて、エラーステータスを返します。例えば、VSL API のデフォルトは、以前のバージョンのインテル® MKL ではサブルーチン形式でしたが、現在では関数形式です。つまり、Fortran のユーザーは、VSL ルーチンを関数として呼び出す必要があります。

関数の呼び出し例:

```
errstatus = vslnrggaussian(method, stream, n, r, a, sigma)
```

サブルーチンの呼び出し例:

```
call vslnrggaussian(method, stream, n, r, a, sigma)
```

ただし、インテル® MKL では、下位互換用にサブルーチン形式のインターフェイスも用意しています。サブルーチン形式のインターフェイスを使用するには、手動で (include ディレクトリーにある) mkl.fi ファイルの include 'mkl_vsl.fi' という行を include

'mkl_vsl_subroutine.fi' に変更し、mkl_vsl.fi ファイルの代わりに mkl_vsl_subroutine.fi ファイルを組み込みます。VSL の API 変更は、C/C++ ユーザーには影響しません。

4.3 メモリー割り当て

より高いパフォーマンスを得るため、インテル® MKL によって割り当てられたメモリーは解放されません。これは仕様で、インテル® MKL ルーチンがメモリーバッファを操作するのは 1 回 (割り当て) だけです。ツールによっては、これをメモリーリークとして報告することがあるため、注意してください。必要に応じて、メモリーを解放することができます。プログラムでインテル® MKL の MKL_FreeBuffers() 関数を使用するか、各呼び出しの後に MKL_DISABLE_FAST_MM 環境変数を設定します (詳細は、Documentation ディレクトリーにある『ユーザーズガイド』を参照してください)。しかし、これらの方法を使用してメモリーを解放しても、メモリーリークが報告されなくなるとは限りません。実際、ライブラリーを複数回呼び出す場合、各呼び出しごとに新しいメモリーの割り当てが必要になり、報告される数は増えることもあります。上記の方法で解放されなかったメモリーは、プログラムの終了時にシステムによって解放されます。この制限を回避するには、上記のようにメモリー管理を無効にします。

Red Hat Enterprise Linux 3.0 では、正しいサポート・ライブラリーが確実にリンクされるように、環境変数 LD_ASSUME_KERNEL を設定する必要があります。

例: 'export LD_ASSUME_KERNEL=2.4.1'

4.4 その他の注意

GMP コンポーネントはソルバー・ライブラリーにあります。インテル® 64 および IA-64 プラットフォームでは、これらのコンポーネントは LP64 インターフェイスのみをサポートします。

5 クラスター OpenMP*

5.1 概要

クラスター OpenMP は、Ethernet* などの通信ファブリックによって接続されたノードセット上の OpenMP プログラムの実行をサポートするシステムです。このようなノードには、OpenMP でデザインされている共有メモリー・ハードウェアがありません。そのため、クラスター OpenMP は、ハードウェアをソフトウェア・メカニズムによってシミュレーションします。クラスター OpenMP で使用されているソフトウェア・メカニズムは、通常、分散共有メモリーシステムと呼ばれます。

クラスター OpenMP は、インテル® コンパイラーの非サポートアドオン製品です。クラスター OpenMP 向けにコードをコンパイルするには、個別のライセンス (無償) を入手する必要があります。クラスター OpenMP の使用に関する詳細は、<http://whatif.intel.com> を参照してください。

5.2 製品の内容

クラスター OpenMP は、主に OpenMP サポートレイヤーと DVSM レイヤーの 2 つで構成されます。OpenMP サポートレイヤーは、OpenMP のセマンティクスを実装し、DVSM レイヤーを使用して、クラスター間で共有されているメモリーを管理します。DVSM レイヤーは独占的にライセンスを得ており、Willy Zwaenepoel 教授、Alan Cox 教授、その同僚と学生によりライス大学で開発された TreadMarks 分散共有メモリーシステムを大幅に変更し拡張したものです。

5.3 新機能

次の項目については、『クラスター OpenMP* ユーザーズガイド』のセクション 1.4 を参照してください。

- コード速度の短縮とスケーラビリティの向上
- コンパイラー・ディレクトリー・ツリーのトップレベルの cluster_omp ディレクトリー
- 新しいパフォーマンス解析ツール: segvprof.pl と dashboard
- 共有可能なセクション機能

5.4 既知の問題と制限

インテル® 64 アーキテクチャーでのみ、次の問題があります。

- 古いバージョンのクラスター OpenMP ランタイム・ライブラリー (libclusterguide.so) は、“libirc” ライブラリーから、新しいハードウェアとは互換性のないいくつかのシンボルを使用します。新しいバージョンの libclusterguide.so では libirc は使用しなくなりました。古いバージョンの libclusterguide.so では、実行時に次のようなエラーが表示されることがあります。

```
a.out: symbol lookup error: a.out: undefined symbol:
__intel_cpu_indicator (a.out: シンボル検索エラー: a.out: 未定義のシン
ボル: __intel_cpu_indicator)
```

このエラーが発生した場合は、アプリケーションを新しい libclusterguide.so ライブラリーで再リンクする必要があります。

すべてのアーキテクチャーで次の問題があります。

- C/C++ コンパイラー 9.1.045 以前のバージョンまたは Fortran コンパイラー 9.1.040 以前のバージョンでビルドされたクラスター OpenMP プログラムは、新しいバージョンのライブラリーで実行できるようにする前に、新しいバージョンのコンパイラーで再リンクまたは再コンパイルする必要があります。
- クラスター OpenMP は、“linuxthreads” として知られる Posix スレッドのバージョンをサポートしていません。“NPTL” として知られる Posix スレッドのバージョンをサポートしています。使用しているカーネルでサポートされている Posix スレッドのバージョンは、次の Linux コマンドで確認できます。

```
$ getconf GNU_LIBPTHREAD_VERSION
```

出力は、次のいずれかのようになります。

```
NPTL 0.60
```

または

```
linuxthreads-0.10
```

システムで linuxthreads がサポートされている場合は、システム管理者に問い合わせ、NPTL サポートを有効にしてください。

- /etc/security/limits.conf ファイルを持つすべてのシステムで、“memlock” の制限を少なくとも次の大きさに設定する必要があります。
 - soft memlock 4000000
 - hard memlock 4000000
- 一部の Linux ディストリビューション、特に SuSE 10 では、“randomize_va_space” として知られるカーネル・セキュリティ機能が有効になっています。この機能は、メモリー・マップ・コードの仮想メモリー・アドレスとデータをそれぞれのプロセスの起動時に変更します。クラスター OpenMP では、それぞれのプロセスが同じ仮想アドレスの共有データにマップされる必要があるため、この機能によりクラスター OpenMP で問題が生じます。/proc/sys/kernel/randomize_va_space ファイルを確認して、使用しているシステムがこの影響を受けるかどうかを特定することができます。“1” が含まれている場合は、システムでこの機能は有効です。クラスター OpenMP をこの環境で正しく動作させるには、ファイルに“0”を挿入して、randomize_va_space を無効にする必要があります。これを行うには、“root” でログインして、/etc/sysctl.conf ファイルに次の行を追加します。

```
kernel.randomize_va_space=0
```

その後、再起動します。

- クラスター OpenMP は、クラスター OpenMP プログラムの実行に関わるノードがすべて同じオペレーティング・システムと同じカーネルバージョンを実行する構成でのみテストされ、サポートされています。
- クラスター OpenMP は、入れ子された並列処理は 1 レベルのみサポートします。外側の並列領域がアクティブで内側の並列領域に到達すると、内側の領域は順次実行されます。
- クラスター OpenMP によってサポートされるスレッドの合計数の制限は、すべてのプロセスでおよそ 65,536 です。
- クラスター OpenMP でサポートされている共有メモリーの合計量の制限は、256GB です。
- クラスター OpenMP プログラムのビルド中に、OpenMP に静的にリンクされるライブラリーを使用している場合、静的にリンクされた OpenMP コードを含むライブラリーの前に、クラスター OpenMP ライブラリー (libclusterguide.so) がリンクされていることを確認しなければなりません。詳細は、『クラスター OpenMP* ユーザーズガイド』のセクション 10.2 を参照してください。
- クラスター OpenMP 並列領域内で入れ子されていない OpenMP 並列領域は、シングルノードで並列に実行されます。クラスター OpenMP 並列領域内で入れ子されている OpenMP 並列領域は、到達したそれぞれのスレッド上で実行されます。
- クラスター OpenMP プログラムは、共有オブジェクトライブラリーの一部としては動作しません。これは、すべてのノードで同じ仮想アドレスでロードされる各共有変数に依存するためです。
- C++ STL (標準テンプレート・ライブラリー) の一部のバージョンには、共有可能なアロケーターに関連した不具合があります。共有可能なアロケーターを STL コンテナとともに使用する方法については、『クラスター OpenMP* ユーザーズガイド』のセクション 10.7.1.4 と 10.7.1.5 を参照してください。

5.5 ドキュメントとサンプル

インストール先のコンパイラー・ディレクトリーに保存されています。次のディレクトリーを参照してください。

- <install-dir>/Documentation/cluster_omp/docs - クラスター OpenMP ドキュメント
- <install-dir>/Documentation/cluster_omp/examples - クラスター OpenMP 使用例
- <install-dir>/Documentation/cluster_omp/tools - クラスター OpenMP プログラムに役立つツール

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、商品性に関する保証、第三者の特許権、著作権、その他知的財産権の侵害への保証を含む) にも一切応じないものとします。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。そのようなエラッタは、インテルの保証範囲外です。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/Third-Party-Software>) 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、MMX、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。

© 2008 Intel Corporation. 無断での引用、転載を禁じます。