インテル® Fortran コンパイラー・クラシック 2021.1 およびベータ版インテル® Fortran コンパイラー 2021.1 リリースノート

2020 年 12 月 15 日

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

インテル® oneAPI HPC ツールキットには、Fortran 2018 標準を完全に実装したインテル® Fortran コンパイラー・クラシック (ifort ドライバー) が含まれています。

インテル® oneAPI HPC ツールキットには、ベータ版インテル® Fortran コンパイラー (ifx ドライバー) も含まれています。ベータ版インテル® Fortran コンパイラー (ifx) は、インテル® Fortran コンパイラー・クラシック (ifort) のフロントエンドとランタイム・ライブラリー (LLVM バックエンド・テクノロジーを使用) をベースとする新しいコンパイラーです。現時点で、ifx は Fortran 95 言語の機能と、ほとんどの OpenMP* 4.5 ディレクティブおよびオフロード機能をサポートしています。ifx はバイナリー (.o/.obj) およびモジュール (.mod) ファイル互換です。ifort で生成されたバイナリーおよびライブラリーは、ifx で構築されたバイナリーおよびライブラリーとリンクでき、一方のコンパイラーで生成された .mod ファイルは、もう一方のコンパイラーで使用できます (64 ビット・ターゲットのみ)。どちらのコンパイラーも ifort ランタイム・ライブラリーを使用します。ifx は、ifort がサポートしていない GPU オフロードを試すことに興味があるユーザー向けに、ベータ版としてリリースされています。ほかの Fortran ユーザーは、引き続き ifort を使用してください。

ベータ版インテル® Fortran コンパイラー (「マテリアル」) は、Intel End User License Agreement for Developer Tools (Version October 2020) (インテルの開発者向けソフトウェア使用許諾契約書 (バージョン 2020 年 10 月)) (「EULA」) で定義されている Free User (無料ユーザー) タイプでライセンスされます。「マテリアル」は、EULA のセクション 4.2 に記載されている「プレリリース・マテリアル)」です。

EULA のセクション 4.2 に記載されている「プレリリース・マテリアル」に関する次の制限は、本書により**免除**されます。

> You may not (i) modify or incorporate the Pre-Release Materials into Your Product; (ii) continue to use the Pre-Release Materials once a commercial version is released; or (iii) disclose to any third party any benchmarks, performance results, or other information relating to the Pre-Release Materials.

これらの「プレリリース・マテリアル」を使用することで、お客様は次の条項に同意したものとします。(a) 「プレリリース・マテリアル」はベータ品質レベルでのみ検証されており、エラーが含まれていたり、頻繁なアップデートが必要になる場合があります。(b) 現在のパフォーマンスは、最終的な製品バージョンのパフォーマンスを表していない場合があります。(c) これらの「プレリリース・マテリアル」の使用に起因する問題については、お客様が一切の責任を負います。

リリースの入手方法

このサイト (英語) から手順に従ってインテル® oneAPI HPC ツールキットをダウンロードし、インストール手順に従ってツールキットをインストールします。すべての機能を使用するには、最初にインテル® oneAPI ベース・ツールキットをインストールしてください。

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。

インテル® Fortran コンパイラー・クラシックおよびベータ版インテル® Fortran コンパイラー 2021.1 の新機能 (インテル® Fortran コンパイラー 19.1 からの変更)

- Fortran 2018 の新機能は ifort でのみ利用できます。ifort は Fortran 2018 標準を完全に実装しています。
- 新しいコンパイラー・オプション
- 報告された問題を修正

システム要件

インテル® Fortran コンパイラーのシステム要件 (英語) を参照してください。

インテル® Fortran コンパイラーのインストール方法

インストール手順は、ドキュメントの一部として、インテル® ソフトウェア開発製品に同梱されています。最新のインテル® oneAPI HPC ツールキットのインストール・ガイドは、オンラインでも利用できます。インストール・ガイド (英語) を参照してください。

インテル® Fortran コンパイラーの使用方法

インテル® Fortran コンパイラーの使用方法は、以下を参照してください。

- インテル® oneAPI ツールキット入門 (Linux* 版) (英語)
- インテル® oneAPI ツールキット入門 (Windows* 版) (英語)

ドキュメント

製品のドキュメントはオンラインで提供されています。

- インテル® oneAPI ツールキット入門 (Linux* 版) (英語)
- インテル® oneAPI ツールキット入門 (Windows* 版) (英語)
- デベロッパー・ガイドおよびリファレンス (英語)

テクニカルサポート

サポートが必要な場合は、インテル® Fortran コンパイラー・フォーラム (英語) にアクセスしてください。
商用サポートを利用可能な場合は、サポートチケットを作成 (英語) してください。

新規および変更されたコンパイラー機能

ifort

既知の問題

Linux* および macOS* で `--version` コンパイラー・オプションを使用すると、正しくないバージョン文字列 `ifort (IFORT) 2021.1 Beta 20201112` が表示されます。このコンパイラーはベータ版ではないため、「Beta」は正しくありません。`-V` コンパイラー・オプションを使用すると、正しいバージョン文字列 `Intel(R) Fortran Intel(R) 64 Compiler Classic for applications running on Intel(R) 64, Version 2021.1 Build 20201112_000000` が表示されます。`--version` コンパイラー・オプションで表示される文字列は将来のアップデートで修正される予定です。この ifort コンパイラー・オプションは Windows* では利用できません。

Fortran 2018 の新機能

これらの機能は ifx コンパイラー・ドライバーを使用した場合は利用できません。

- Co-Array のチームが実装されました。
    - 派生型 TEAM_TYPE が ISO_FORTRAN_ENV 組込みモジュールに追加されました。
    - CHANGE TEAM および END TEAM 文が実装されました。
    - FORM TEAM 文が実装されました。
    - SYNC TEAM 文が実装されました。
    - TEAM_NUMBER 組込み関数が実装されました。
    - GET_TEAM 組込み関数が実装されました。
    - オプションの TEAM 引数を STOPPED_IMAGES、NUM_IMAGES、および IMAGE_STATUS 組込み関数に追加しました。
    - IMAGE_INDEX および NUM_IMAGES の新しい形式にオプションの TEAM および TEAM_NUMBER 引数が実装されました。
    - TEAM 引数を含む THIS_IMAGE の新しい形式が実装されました。
    - オプションの TEAM または TEAM_NUMBER 指定子がイメージセレクターで利用できるようになりました。
- 変数定義コンテキストで未割り当ての Co-Array の暗黙的な割り当てが禁止されました。
- EXTENDS_TYPE_OF および SAME_TYPE_AS 組込み関数の非多相ポインター引数で割り当てステータスを定義する必要がなくなりました。

- ファイナライズ中の割り当て解除が明確になり、Fortran 2018 で定義されたセマンティクスを使用して発生するようになりました。
- STOP および ERROR_STOP 文の出力は条件付きで抑止されます。
- STOP および ERROR STOP コードを非定数式にできるようになりました。
- LOCK_TYPE 型の名前付き定数は許可されなくなりました。
- ALL、ANY、IALL、IANY、IPARITY、MAXLOC、MAXVAL、MINLOC、MINVAL、NORM2、PARITY、PRODUCT、SUM、および THIS_IMAGE 組込み関数の DIM 引数は現在の OPTIONAL 仮引数になります。
- PURE プロシージャーの VALUE 仮引数は変数定義コンテキストに表示されます。
- オプションの ERRMSG 引数を GET_COMMAND_ARGUMENT、GET_ENVIRONMENT_VARIABLE、および GET_COMMAND 組込みプロシージャーに追加しました。
- OUT_OF_RANGE 組込み関数が実装されました。
- RANDOM_INIT 組込みサブルーチンが実装されました。
- 定義された代入および操作プロシージャーの仮引数が VALUE 属性を含む場合、INTENT(IN) 属性を含む必要はありません。
- PURE プロシージャーの仮引数が VALUE 属性を含む場合、変数定義コンテキストに表示されます。
- オブジェクトの定数プロパティーはオブジェクトの初期化で使用できます。
- 多相構造コンストラクター・コンポーネントは対応する構造コンポーネントと同じ動的な型を持つ必要がなくなりました。
- RANDOM_INIT 組込みサブルーチンが実装されました。
- current team (現在のチーム)、ancestor team (先祖チーム)、parent team (親チーム)、および established coarrays (確立された Co-Array) などの Fortran 2018 の新しい用語がデベロッパー・ガイドの用語集に追加されました。
- PROTECTED 属性を含む非ポインター変数は、データターゲットまたは初期データターゲットとして表示されなくなります。
- VOLATILE 変数は PURE プロシージャーおよび文の関数で禁止されました。
- 暗黙の DO ループ変数 TYPE の型と種別を配列コンストラクターとデータ文の暗黙の DO ループで指定できるようになりました。
- `-assume ieee_compares` または `-standard-semantics` コンパイラー・オプションが指定された場合、浮動小数点の比較は `IEEE comareSignaling<relation>` 操作を実行します。
- デフォルトでは、プロシージャーは再帰向けにコンパイルされていません。この動作が必要なユーザーは、`-standard-semantics` コンパイラー・オプションまたは `-assume recursion` オプションを使用する必要があります。
- REDUCE 関数が実装されました。

ifx

既知の問題

Microsoft* Visual Studio* で、ベータ版インテル® Fortran コンパイラーの名前がインテル® Fortran コンパイラーとして誤って表示されます。これはベータ版の LLVM ベースの Fortran コンパイラーです。プロダクション環境では使用しないでください。ベータ版インテル® Fortran コンパイラーとその使用方法は、このリリースノートの説明を参照してください。

言語と OpenMP*

Fortran 77

- 完全に実装されました。

Fortran 90/95

- 完全に実装されました。コードが Fortran 90/95 に含まれていない Fortran 標準を使用しているかどうか判断するには、コマンドラインに `-stand f95` (Linux*) または `/stand:f95` (Windows*) 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、Fortran 90 または Fortran 95 標準ではない言語要素に警告メッセージを出力します。

Fortran 2003

- F2003 で導入された `iso_c_binding` 機能のほとんど (F2018 で導入されたものを除く) はサポートされています。

OpenMP*

- OpenMP* 4.5 は、ユーザー定義関数を除いてサポートされています。

参考情報

Fortran 2003、Fortran 2008、OpenMP* 4.5 および OpenMP* 5.0 機能の実装の詳細は、この記事 (英語) を参照してください。

新規および変更されたコンパイラー・オプション

ifort

これらの機能は ifort コンパイラー・ドライバーを使用した場合のみ利用できます。

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。次の新しいコンパイラー・オプションが追加されました。

Linux*/macOS*

- `-assume [no]failed-images`
- `-assume [no]ieee_compares`
- `-check [no]teams`

Windows*

- `/assume:[no]failed-images`
- `/assume:[no]ieee_compares`
- `/check:[no]teams`

assume [no]failed-images

STAT= 指定子なしのイメージ制御文、MOVE_ALLOC の呼び出し、または STAT 引数なしのアトミック・サブルーチンや集合サブルーチンを実行するとき、イメージのチームで失敗したイメージをチェックするようにランタイムシステムに指示します。デフォルトは `nofailed_images` です。

assume [no]ieee_compares

Fortran 2018 標準で要求されているように、浮動小数点比較に `IEEE compareSignaling<relation>` 操作を生成するようにコンパイラーに指示します。デフォルトは `ieee_compares` です。

check [no]teams

Co-Array チーム機能の非標準の使用を検出するようにランタイムシステムに指示します。例えば、現在のチームまたは現在のチームの先祖を記述しない NUM_IMAGES 組込み関数での TEAM 変数の使用、初期のチームを示すイメージセレクターでの `TEAM_NUMBER=-1` の使用などです。デフォルトは `noteams` です。

終了予定のコンパイラー・オプションの一覧は、ドキュメントの「コンパイラー・オプション」セクションを参照してください。新しく終了予定になったコンパイラー・オプションは、「終了予定のサポート」セクションを参照してください。

ifx

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。次の新しいコンパイラー・オプションが追加されました。

assume [no]ieee_compares

- **Linux***
  - `-assume [no]ieee_compares`
- **Windows***
  - `/assume:[no]ieee_compares`

Fortran 2018 標準で要求されているように、浮動小数点比較に `IEEE compareSignaling<relation>` 操作を生成するようにコンパイラーに指示します。デフォルトでは、ifx は NaN が浮動小数点比較のオペランド

にならないと仮定し、NaN のチェックを生成しません。浮動小数点比較での ifort の動作を取得するには、ifx ユーザーは `assume nan_compares` を指定する必要があります。

fp-model、fp コンパイラー・オプションの違い

- **Linux***
  - `-fp-model fast=1` または `-fp-model fast=2`
- **Windows***
  - `/fp:fast=1` または `/fp:fast=2`

`fp-model|fp fast=1` または `2` オプションを使用した場合の浮動小数点比較の動作は ifort と ifx で異なります。ifx で `fp-model|fp fast` オプションを使用したときの ifort の動作を取得するには、ifx のコマンドラインで `assume nan_compares` を指定する必要があります。

終了予定のサポート

- ディレクティブ `SIMD (!dir$ SIMD)` は古い機能 (非推奨) で、将来のリリースで削除される予定です。ユーザーは、ソースコードで、このディレクティブを OpenMP* SIMD ディレクティブ `!$omp simd` および関連する節に置換する必要があります。`-qopenmp-simd` または `-qopenmp` コンパイラー・オプションを追加することを忘れないでください。`-O2` を使用してコンパイルすると、`-qopenmp-simd` が暗黙的に指定されます。
- `-stand f15` (Linux*)、`/stand:f15` (Windows*) コンパイラー・オプションは古いオプション (非推奨) で、将来のリリースで削除される予定です。`-stand f18` (Linux*)、`/stand:f18` (Windows*) を使用してください。
- macOS* 11 Big Sur では IPO のサポートは動作しません。将来のリリースで削除される予定です。
- ifort の 32 ビット・ターゲットのサポートは古い機能 (非推奨) で、将来のリリースで削除される予定です。
- Linux* の `-mkl` コンパイラー・オプションは古いオプション (非推奨) で、将来のリリースで削除される予定です。この予定は、ifort と ifx の両方に適用されます。将来のリリースでは、`-qmkl` に置換されます。このコンパイラー・オプションは、インテル® oneAPI マス・カーネル・ライブラリー (インテル® oneMKL) の必要なライブラリーにリンクするようにコンパイラーに指示します。

終了したサポート

- 32 ビット Co-Array のサポートは終了しました。
- `-qnextgen` (Linux*)、`/qnextgen` (Windows*) コンパイラー・オプションは削除されました。`ifort -qnextgen` または `ifort /qnextgen` の代わりに `ifx` を使用してください。

既知の問題

ifort

--version コンパイラー・オプションを使用すると正しくないバージョン文字列が表示される

Linux* および macOS* で `--version` コンパイラー・オプションを使用すると、正しくないバージョン文字列 `ifort (IFORT) 2021.1 Beta 20201112` が表示されます。このコンパイラーはベータ版ではないため、「Beta」は正しくありません。`-V` コンパイラー・オプションを使用すると、正しいバージョン文字列 `Intel(R) Fortran Intel(R) 64 Compiler Classic for applications running on Intel(R) 64, Version 2021.1 Build 20201112_000000` が表示されます。`--version` コンパイラー・オプションで表示される文字列は将来のアップデートで修正される予定です。この ifort コンパイラー・オプションは Windows* では利用できません。

Ubuntu* 18.04.3 でコンパイラーを実行するとエラーになる

```
ifort: error #10105: ld: core dumped
```

Ubuntu* 18.04 LTS の最新ビルドは 18.04.4 です。以前のビルド 18.04.3 には、インテル® Fortran コンパイラー・クラシックを実行するとエラーになる問題があります。18.04.2 および 18.04.1 などの以前のビルドでは、この問題は発生しません。

次のコマンドを実行して、ビルド番号を確認できます。

```
cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.4 LTS"
```

DISTRIB_DESCRIPTION に注意してください。このシステムのビルドは、最新の 18.04.4 LTS です。

DISTRIB_DESCRIPTION が 18.04.3 と表示された場合、ビルド 18.04.3 に含まれる glibc の問題により、インテル® コンパイラーを実行するとエラーになります。

次のエラーが表示されます。

```
ifort hello.f90
ifort: error #10105: ld: core dumped
ifort: warning #10102: unknown signal(-320620368)
ifort: error #10106: Fatal error in ld, terminated by unknown
```

最新のビルド 18.04.4 にアップグレードすると、修正された glibc を含む重要なアップデートがインストールされ、この問題は発生しなくなります。

```
sudo apt-get update
sudo apt update
```

再配布ライブラリー

Fortran コンパイラー (ifort) の `fredist.txt` ファイルは 2021.1 (Gold) リリースではオンラインでのみ利用可能です。将来のリリースではコンパイラー・パッケージに含まれる予定です。

macOS* 11 での IPO

macOS* 11 で、インテル® oneAPI HPC ツールキットで利用可能なインテル® oneAPI Fortran コンパイラー・クラシックの IPO (`-ipo`) を使用すると、リンク時にエラーが発生します。macOS* 11 ではコンパイラー・オプションから `-ipo` を削除してください。

このリンクエラーは Linux* または Windows* では発生しません。また、現在の macOS* 10.x でも発生しません。新しい macOS* 11 Big Sur でのみ発生します。

beta10 では macOS* 11 Big Sur での IPO のサポートは動作しません。将来のリリースで削除される予定です。

Fortran Co-Array アプリケーションが FAIL IMAGE の実行後にハングする

FAIL IMAGE 文を使用してイメージを失敗させる場合は、文または操作で許可している場合にイメージが失敗する可能性のあるすべての Co-Array 操作で STAT= 指定子または STAT 引数を使用するか、`assume failed_images` または `standard-semantics` コンパイラー・オプションを指定します。STAT= 指定子や STAT 引数を使用しない場合、またはいずれかのコンパイラー・オプションを指定しない場合、これらの操作は失敗したイメージをチェックしません。イメージからの応答を待って失敗したイメージを調整しようとした場合、応答がないために、アプリケーションがハングします。

次の例は、STAT= 指定子を使用してアプリケーションのハングを防ぐ方法を示しています。

```
SUBROUTINE FAIL_AND_SYNC ( THIS_ONE )
   INTEGER THIS_ONE
   INTEGER MY_STAT
   IF (THIS_IMAGE() .EQ. THIS_ONE) THEN
       FAIL IMAGE
   END IF
   SYNC ALL (STAT=MY_STAT)       ! Would hang without STAT=
END SUBROUTINE FAIL_AND_SYNC
```

クロスコンパイル (Microsoft* Windows* を実行している 64 ビット・マシンでの 32 ビット・モードでのコンパイル)

- Microsoft* Windows* で実行している
- クロスコンパイル (64 ビット・マシンでの 32 ビット・モードでのコンパイル) を行っている
- Microsoft* Visual Studio* 2019 を使用している
- コードで例外、特に浮動小数点例外を処理している
- プログラムで浮動小数点例外を取得すると、SEGV 例外が発生する

上記に該当する場合、プログラムは意図したとおりに例外を処理していません。

この問題を解決するには、リンクコマンドに `legacy_x86_flt_exceptions.lib` を追加します。

Red Hat* 8 での Co-Array 機能の使用

Co-Array 機能を使用するアプリケーションを Red Hat* 8.0 以降で実行するには、インテル® MPI ライブラリー 2019 Update 7 以降がインストールされている必要があります。2019 Update 7 より前のバージョンがインストールされている場合、アプリケーションは正常に完了せず、障害に関する情報はほとんど提供されません。

ifx

これらの説明は ifx コンパイラー・ドライバーを使用した場合に適用されます。

Microsoft* Visual Studio* と ifx

Microsoft* Visual Studio* で、ベータ版インテル® Fortran コンパイラーの名前がインテル® Fortran コンパイラーとして誤って表示されます。これはベータ版の LLVM ベースの Fortran コンパイラーです。プロダクション環境では使用しないでください。ベータ版インテル® Fortran コンパイラーとその使用方法は、このリリースノートの説明を参照してください。

再配布ライブラリー

Fortran コンパイラー (ifx) の `fredist.txt` ファイルは 2021.1 (Gold) リリースではオンラインでのみ利用可能です。将来のリリースではコンパイラー・パッケージに含まれる予定です。

GPU ドライバーのハング問題

ネイティブ環境で実行時間の長い GPU 計算ワークロードを含むアプリケーションがある場合、ワークロードが終了するのを防ぐためハングチェック・タイムアウト期間を無効にする必要があります。詳細は、使用している OS のインストール・ガイド (英語) を参照してください。

言語と OpenMP*

OpenMP*

- PARALLEL DO または TEAMS との組み合わせを含む OMP TARGET はサポートされています。型に割付けコンポーネントが含まれる場合、派生型の変数を TARGET 領域に渡すときに既知の制限があります。

その他の既知の問題

- `!DIR$` ディレクティブはフロントエンドで認識されますが、ifx はそれらを実装しません。
- 遅延形状配列 (つまり、ポインター、割付けおよび形状引き継ぎ配列) のデバッグのサポートは追加されましたが、全体的なデバッグのサポートは引き続き制限されています。デフォルトでは、オフロードコンパイル中にデバッグ情報は生成されません。オフロードされたプログラムをデバッグするため `-g -debug` を使用してコンパイルすると、問題が発生することがあります。

スタティック・ライブラリーとターゲットオフロードの問題

バグにより、Linux* 上の ifx コンパイラーは、ターゲット・オフロード・コード (GPU または FPGA 向けのオフロードコードなど) を含むライブラリーで `-l` オプションを使用したライブラリー・アーカイブのリンクをサポートしていません。問題と回避策の説明は、この記事 (英語) を参照してください。この回避策は、インテルのほかの LLVM ベースのコンパイラー (icx、icpx および dpcpp) にも適用できます。

Linux* でリンク時の最適化 (LTO) を使用した場合のエラー

リンク時の最適化 (LTO) を使用した場合にエラーになることがあります。正常にリンクするには、OS に適したバージョンの binutils を使用してください (RHEL 7 を除く)。RHEL 7 では LTO はサポートされていません。

OS	ステータス
OpenSUSE* 15.2	binutils 2.35 で修正
SLES 15 SP2	報告済み、https://bugzilla.suse.com/show_bug.cgi?id=1177442 (英語)
RHEL 7	クローズ/修正されません
RHEL 8	binutils-2.30-82.el8 で修正
Fedora* 32	binutils-2.34-6.fc32 で修正
Ubuntu* 20.04 LTS	報告済み、 https://bugs.launchpad.net/ubuntu/+source/binutils/+bug/1902760 (英語)

設定スクリプトのメッセージ:「C から Fortran ライブラリーへのリンクに失敗」

症状

ifx を使用してコンパイルすると、GNU* Autconf で生成された `./configure` スクリプトで次のようなエラーメッセージが表示される。

```
    checking for Fortran 77 libraries of ifx...  -loopopt=0 -L/lib/../lib64 -
L/lib/../lib64/ -L/usr/lib/../lib64 -L/usr/lib/../lib64/ -L/lib64 -L/lib/ -
L/usr/lib64 -L/usr/lib -lifport -lifcoremt -limf -lsvml -lm -lipgo -lirc -
lpthread -lirc_s -ldl
    configure: WARNING: FLIBS does not work
    checking for ifx flag to add single underscore to external names... none
    checking for dummy main to link with Fortran 77 libraries... unknown
    configure: error: in `/path/to/build/dir':
    configure: error: linking to Fortran libraries from C fails
    See `config.log' for more details
    make: *** [build/config.status] Error 1
```

`config.log` ファイルを調べると、次のエラーにより `./configure` が終了していました。

```
    ld: cannot find -loopopt=0
```

問題

Fortran コードと C/C++ コードのリンクに必要なライブラリーを決定しようとすると、GNU* Autoconf 2.69 以前は、`-mllvm -loopopt=0` 詳細コンパイラー出力オプションの `-loopopt=0` をリンカーオプションとし

て誤って解釈し、リンカーに渡される FLIBS 変数に `-loopopt=0` を追加します。リンカーは存在しないライブラリーを探し、テストは失敗します。

ソースコードをダウンロードするユーザー向けのソリューション

このエラーの影響を受けるパッケージのメンテナーに問題を通知し、このリリースノートの情報を伝えてください。プロジェクトは `configure` スクリプトの処理方法が異なりますが、いくつかの一般的なルールが適用されます。パッケージメンテナーから連絡が届く前に問題を解決する必要があるユーザーは、`configure` スクリプトを自分で更新することも可能です。更新の難しさは、パッケージの配布方法によって異なります。

更新するには、まず、下記のパッケージメンテナー向けソリューションで説明されているように、GNU* Autoconf 2.70 以降をインストールします。

更新された Autoconf をユーザーのパスにインストールした後、次のコマンドを実行して、`configure.ac` ファイルと `configure` スクリプトを配布するパッケージを更新します。

```
autoreconf -if
```

これで、`configure` スクリプトが FLIBS で上記のエラーなしで完了するはずです。

プロジェクトのメンテナーが、tar または zip アーカイブのソース・ディストリビューションから `configure.ac` ファイルを削除するのは珍しいことではありません。その場合、ユーザーは通常、コード・リポジトリーからプロジェクトをダウンロードして、プロジェクトの指示に従ってビルドする必要があります。`configure` スクリプトのバージョンがバージョン・コントロールにコミットされ、自動的に再生成されないことがあります。その場合、上記のように `autoreconf -if` を実行するか、`configure` スクリプトを削除することで、ほとんどの場合はスクリプトが再生成されます。プロジェクトのドキュメントに `configure` スクリプトを再構築する方法が記載されている場合は、その方法に従ってください。

パッケージメンテナー向けソリューション

ユーザーにエラーが表示されないようにするには、GNU* Autoconf をバージョン 2.70 以降に更新し、プロジェクト `configure` スクリプトを再生成します。GNU* Autoconf 2.70 は 2020 年 12 月にリリースされました。必要な修正は、最新の GNU* Autoconf マスターブランチおよび GNU* Autoconf 2.69c ベータリリースにも含まれています。ソースコードは、`git clone http://git.sv.gnu.org/r/autoconf.git` で入手できます。GNU* Autoconf のドキュメントは、GNU* Autoconf プロジェクト・ページ (英語) から入手できます。

Fortran 2008 および Fortran 2018 機能の概要

インテル® Fortran コンパイラー・クラシック (ifort) は、すべての Fortran 2008 標準とすべての Fortran 2018 標準をサポートします。現在のバージョンでサポートされている新しい Fortran 2018 の機能は、「Fortran 2018 の新機能」セクションを参照してください。

必要に応じて、Fortran 2008 標準 (PDF、英語) および Fortran 2018 標準 (PDF、英語) を参照してください。

法務上の注意書き

インテルのテクノロジーを使用するには、対応したハードウェア、ソフトウェア、またはサービスの有効化が必要となる場合があります。

絶対的なセキュリティーを提供できる製品またはコンポーネントはありません。

実際の費用と結果は異なる場合があります。

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。

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

インテルは、明示されているか否かにかかわらず、いかなる保証もいたしません。ここにいう保証には、商品適格性、特定目的への適合性、および非侵害性の黙示の保証、ならびに履行の過程、取引の過程、または取引での使用から生じるあらゆる保証を含みますが、これらに限定されるわけではありません。

製品とパフォーマンス情報

[1] 性能は、使用法、構成、およびその他の要因によって異なります。
詳細については、www.Intel.com/PerformanceIndex (英語) を参照してください。