



THE PORTLAND GROUP

PGI (Accelerator) Visual Fortran® 2011

For Microsoft® Visual Studio

Windows®版 (Release 2011)

– 入門ガイド –

PGI インストール関係の日本語ドキュメントは、以下の URL に全てアーカイブしてあります。オンラインでご覧になりたい場合は、以下の URL にアクセスしてください。

http://www.softek.co.jp/SPG/Pgi/inst_document.html

2011 年 1 月版 (Rev. 11.1-A)

株式会社 ソフテック HPC ソリューション部

(<http://www.softek.co.jp/SPG/>)

SofTek

目次

1	はじめに.....	1
1.1	本文書の概要.....	1
1.2	WINDOWS 上での PVF ソフトウェアの実装.....	1
1.3	PVF コンパイラの利用方法.....	1
1.4	PVF コンパイラのコマンド・オプションについて.....	2
2	PVF コンパイラの起動 (Microsoft® Visual Studio 統合).....	3
2.1	VISUAL STUDIO 2005/2008/2010 の初回起動.....	3
2.2	プロジェクトの作成方法.....	3
2.3	PVF プロジェクト・テンプレート.....	5
2.4	新規にプロジェクトを作成し、新規にプログラム開発するための手続き.....	6
2.5	既存のソースファイルを PVF プロジェクトに移行するための手続き.....	8
2.6	FORTRAN ソースファイルの属性(固定形式、自由形式).....	12
2.7	プログラムのコンパイルと実行 (デバッグモード).....	13
2.8	プログラムのコンパイルと実行 (最適化オプションの適用).....	17
2.9	プログラムの実行 (入力データファイルのリダイレクト).....	21
3	NVIDIA GPU 用 PGI Accelerator と CUDA Fortran を使用する.....	23
3.1	THE PGI ACCELERATOR PROGRAMMING MODEL (ディレクティブ使用).....	23
3.2	PGI CUDA FORTRAN のコンパイル.....	27
4	PVF コンパイラの起動 (コマンド・ライン).....	28
4.1	PVF コマンドプロンプトの起動.....	29
4.2	PVF コンパイラ・コマンドの使用.....	30
4.3	WINDOWS®上で使用する際の留意点.....	32
5	その他.....	32
5.1	実行モジュールの再配布.....	32
5.2	ヘルプ.....	32

本資料の全ての情報は、現状のまま提供されます。株式会社ソフテックは、本資料に記述あるいは表現されている情報及びその中に非明示的に記載されていると解釈されうる情報に対して一切の保証をいたしません。また、本資料に含まれる情報の誤りや、それによって生じるいかなるトラブルに対しても一切の責任と補償義務を負いません。また、本資料に掲載されている内容は、予告なく変更されることがあります。本資料で使用されている社名、製品名などは、一般に各社の商標または登録商標です。

株式会社ソフテック
〒154-0004 東京都世田谷区太子堂 1-12-39
<http://www.softek.co.jp>

Copyright © 2011 SofTek Systems, Inc.
All rights reserved.

1 はじめに

1.1 本文書の概要

本文書は、Microsoft® Visual Studio 2005/2008/2010 による統合開発環境 (IDE) 上で使用する PGI® Visual Fortran 2005/2008/2010 コンパイラ (以下、「PVF」と言う。) の一般的な使用方法を簡単に纏めた入門ガイドです。Visual Studio の細かな操作方法に関しては、Microsoft®社のドキュメント等を参考にしてください。

(ご注意)

Microsoft® Visual Studio 2005 にプラグイン可能な PGI Visual Fortran は、PVF 10.9 以前のバージョンのソフトウェアとなります。PVF 11.0 (2010 年 12 月リリース) 以降は、Visual Studio 2005 上で使用できませんのでご了承下さい。無償の Visual Studio 2010 shell 等をインストールしてお使い下さい。

1.2 Windows 上での PVF ソフトウェアの実装

Microsoft® Visual Studio 2005/2008/2010 が実装されているシステム上で PVF ソフトウェアのインストールを行うと、以下のディレクトリ・パス上にソフトウェアが実装されます。PVF のコンポーネントのデフォルトのインストール・パスは、以下の形態となります。PVF ソフトウェアを構成するコンポーネントは以下の二つに大別されます。

Visual Studio に統合するための PVF モジュール (PVF IDE 部と言う)

PGI コンパイラ本体のコンポーネント

【Win32 システム上】

C:\Program Files\Microsoft Visual Studio 10\PGI Visual Fortran (PVF IDE 部)

C:\Program Files\PGI (32 ビット PGI コンパイラ本体)

【Win64 システム上】

C:\Program Files (x86)\Microsoft Visual Studio 10\PGI Visual Fortran

C:\Program Files\PGI (64 ビット PGI コンパイラ本体)

C:\Program Files (x86)\PGI (32 ビット PGI コンパイラ本体)

1.3 PVF コンパイラの利用方法

PVF コンパイラを使用する形態は、以下の二つの方法があります。PVF は、統合開発環境 (IDE) 上でコンパイラの利用できるだけでなく、コマンド・ライン上でも利用可能です。

PVF コンパイラの利用形態

利用方法	内容
Visual Studio の統合開発環境 (IDE) 上での使用	Visual Studio を起動して、IDE 上の GUI ベースで操作する。
コマンド・ライン上での使用	PVF Command Prompt (32bit) あるいは、PVF Command Prompt (64bit) のウィンドウを開き、コマンドベースでコンパイラを操作する。

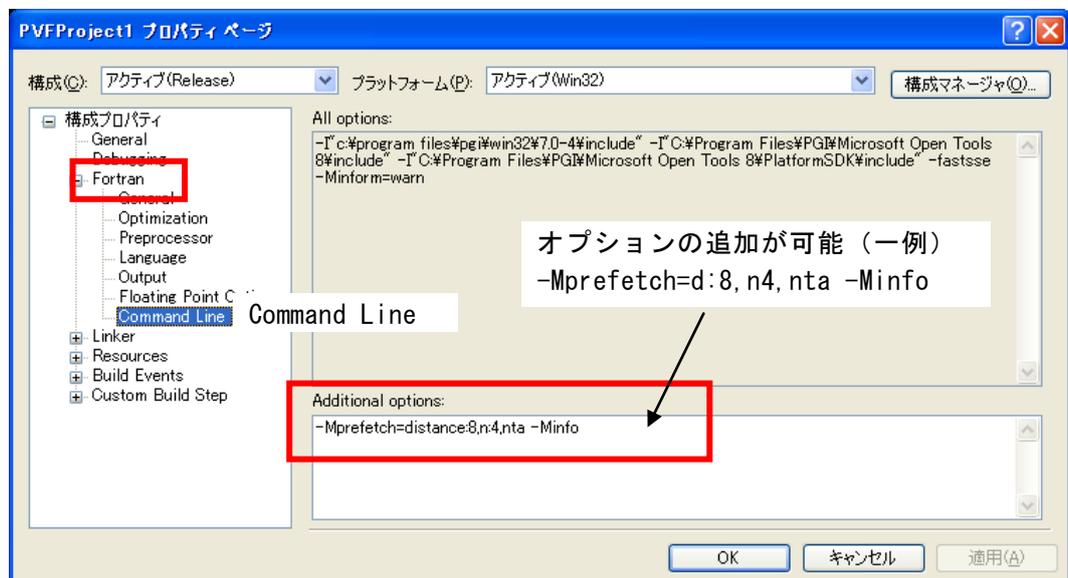
1.4 PVF コンパイラのコマンド・オプションについて

PGI コンパイラには、本 PVF コンパイラ製品だけではなく、Linux/Windows/Apple Mac OS X 等の OS 下においてコマンド・ライン上で操作する「PGI Workstation & Server 製品」があります。これらの製品の種別を問わず、PGI コンパイラで使用するコンパイラ・オプション（スイッチ）名とその使用法は、一部、OS に依存したものを除き、同じものとお考えください。PGI コンパイラのオプションの説明に関しては、弊社ホームページ上のコンテンツ、あるいは、ダウンロードサイトで提供しております「PGI コンパイラ使用ガイド」（PDF ファイル）をご覧ください。

例えば、以下の pgfortran (pgf95/pgf90) コマンドの例は、全ての PGI 製品のコマンド・ライン上で、全く同じ形で使用できます。PGI コマンド列は、Linux 流のコマンド列コンベンション（慣用的な使用法）を踏襲しているため、Windows 上においても、コマンド・オプションは、「-」で始まる形態となります。（Microsoft Windows 上でのコマンド・オプションは、その始めに「¥」（バックスラッシュ）を付けて記述するのが一般的です）

```
$ pgfortran -fastsse -Minfo test.f
（「-」で始まるものは、コンパイラ・オプションを意味します）
```

PVF の Visual Studio (IDE) 上での使用においても、同じオプション形態でコンパイラ・オプションをセットすることができます。また、以下の図のように任意のコンパイラ・オプションを IDE 上の「プロジェクトのプロパティ」でセットすることが可能です。「コマンド・ライン」プロパティに設定したオプションは、PVF でデフォルト設定されているオプションに追加される形で、コンパイラに認識されます。



Visual Studio 上での PVF コンパイルオプションの任意設定

(ご参考)

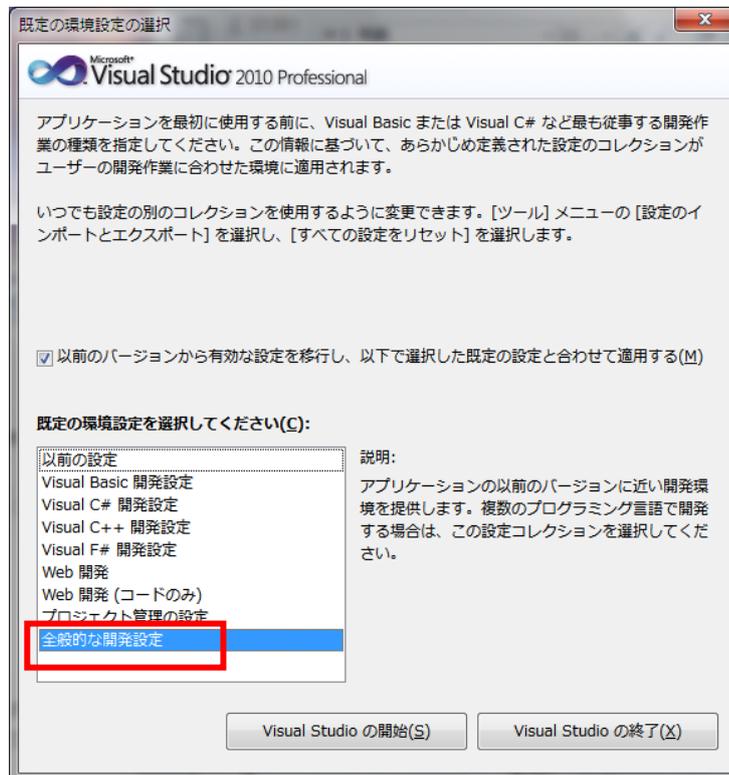
[「PGI コンパイラ使用ガイド」\(PDF ファイル\)](#)

2 PVF コンパイラの起動（Microsoft® Visual Studio 統合）

2.1 Visual Studio 2005/2008/2010 の初回起動

PVF コンパイラを Visual Studio 2005/2008/2010 Standard/Professional Edition 統合環境上で使用する際には、「Visual Studio 2005/2008/2010」を起動する必要があります。あるいは、Windows の「スタート」->「すべてのプログラム(P)」->「PGI Visual Fortran」->「PGI Visual Fortran 20xx」を選択して起動しても、同様な Visual Studio 2005/2008/2010 の画面が現れます。（Visual Studio 2008/2010 Shell integrated mode を使用する際は、以下の画面は現れません）

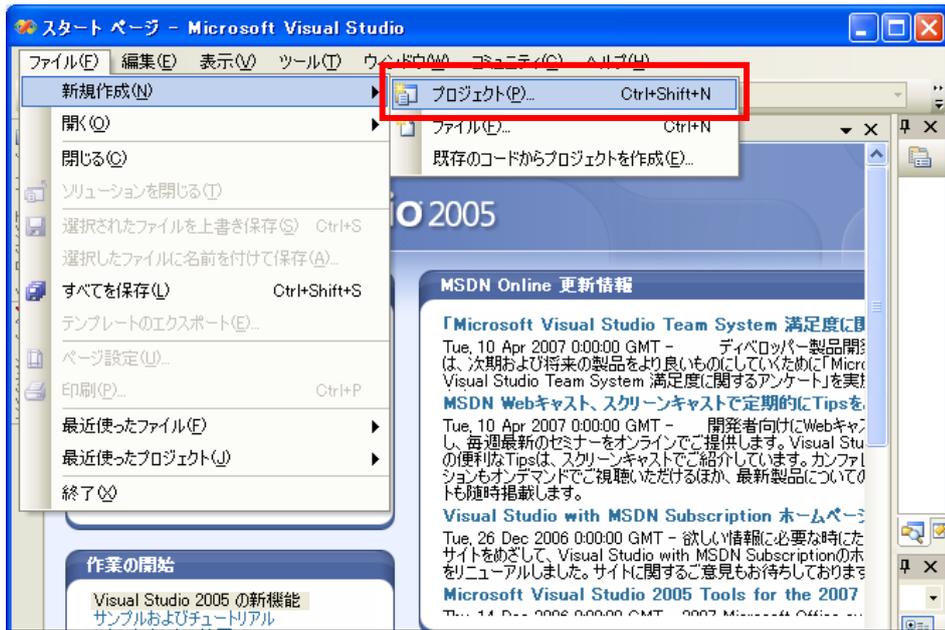
Visual Studio 2005/2008/2010 Standard Edition を初めて起動した際、以下のような画面が表示されます。Visual Studio の開発作業環境のデフォルトを指定するものですが、これは、「全般的な開発設定」を選んで Visual Studio の開始を行ってください。



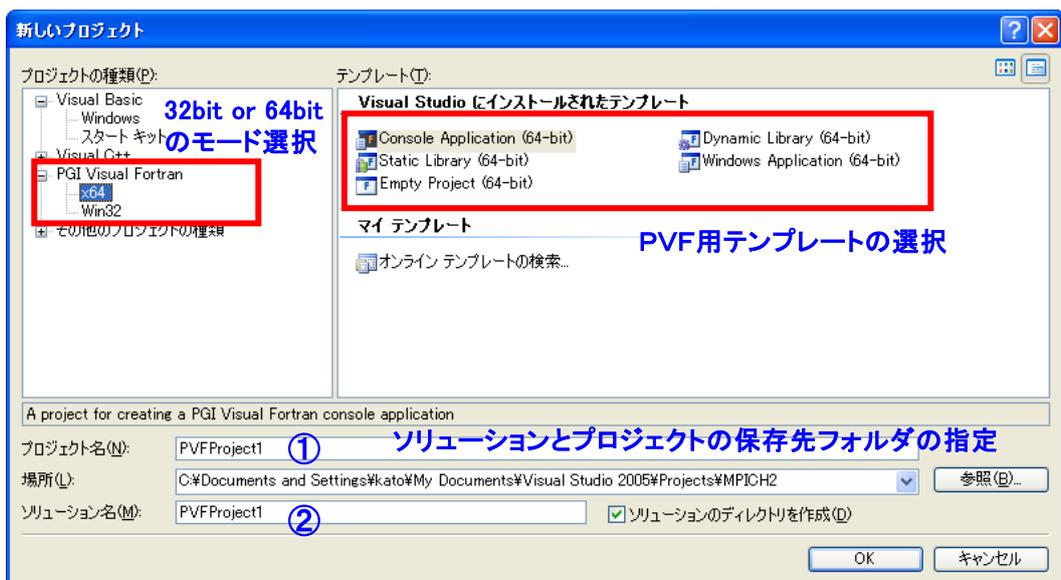
2.2 プロジェクトの作成方法

PGI Visual Fortran、あるいは Visual Studio 2005/2008/2010 を開始して、新しい「プロジェクト」を作成します。Visual Studio 2005/2008/2010 を起動後、「ファイル」->「プロジェクト」を選択します。

Visual Studio 2005/2008/2010 上で新規にプログラムを開発・作成する場合も、既存のプログラムを Visual Studio 2005/2008/2010 のプロジェクトの中に移行する場合も、この「プロジェクト」作成メニューを使用します。



プロジェクトの新規作成を選択しますと、以下の画面が現れます。画面左側に示される「PGI Visual Fortran」プロジェクトをクリック選択しますと、右側に PVF 関連の新規「テンプレート」が表示されます。なお、64 ビット Windows の場合は、「PGI Visual Fortran」プロジェクトは、32 ビットモード (Win32) と 64 ビットモード (x64) の二種類が表示されます。32 ビットアプリケーションの作成を行う際は、Win32 のテンプレートを使用し、64 ビットアプリケーション作成の場合は、x64 テンプレートを使用します。以下の画面は、64 ビット Windows x64 上での表示例を示したものです。(32 ビット Windows の場合は、32 ビットモード (Win32) モードのみ表示されます)

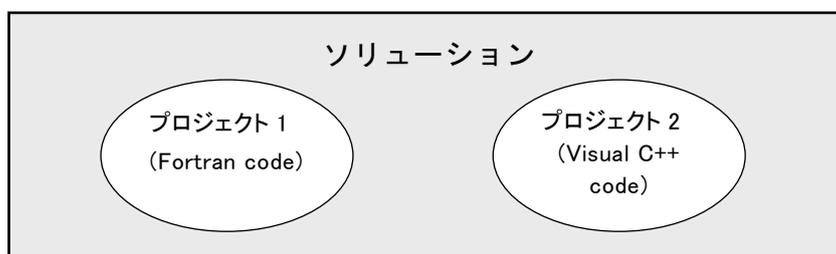


上図で①の部分は、Visual Studio 2005/2008/2010 上の「プロジェクト名」を指定するものです。②は、ソリューション名を指定します。初めてソリューションを作成する時は、「ソリューションのディレクトリを作成」にチェックを入れます。「場

所」で示される部分は、このプロジェクト関連で作成されるファイルを保存する場所を指定します。デフォルトは、Windows 上の「マイドキュメント」配下に作成されます。この場所を変えたい場合、あるいはプロジェクト名を変更したい場合は、そのパス名、フォルダ名を指定してください。なお、「場所」のデフォルト値は、「ツール」->「オプション」->「プロジェクトおよびソリューション」->「全般」で変更できます。

すでに作成されている「ソリューション」の中に、新たな「プロジェクト」を追加作成したい場合は、「場所」と②の欄の「ソリューション」名を指定し、「ソリューションのディレクトリを作成」は行いません。

以下の図は、Visual Studio 2005/2008/2010 における「ソリューション」と「プロジェクト」の関係を表したものです。一般的には、一つの「ソリューション」の中に一つの「プロジェクト」の構成で使用する人が多いのですが、例えば、PVF Fortran コードと Visual C++コードから成る混合プログラムの場合は、明確に言語種別でプロジェクトを分けて構成しなければなりません。



2.3 PVFプロジェクト・テンプレート

Visual Studio 2005/2008/2010 上での PVF は、以下に示すプロジェクト・タイプのテンプレートを用意しています。

■ Console Application (コンソールアプリケーション)

ウィンドウを使わないテキストの入出力を伴うアプリケーションを作成するためのプロジェクト。一般的には、これが多用されます。

■ Dynamic Library (ダイナミックライブラリ)

DLL モジュールを作成するためのプロジェクト。DLL とは、プログラムが必要とされる時に、オンデマンドでローディングされるエグゼキュタブル・ファイルです。

■ Static Library (スタティックライブラリ)

実行モジュールを生成する際に、リンクすることが可能な一つもしくは複数のオブジェクトから成るアーカイブ・ファイルを作成するためのプロジェクト。

■ Windows Application (ウインドウズアプリケーション)

ウィンドウ、ダイアログ・ボックス、メニュー等のコンポーネントを使用する GUI を備えたアプリケーションを作成するためのプロジェクト。このようなアプリケーションのプログラム・エントリ・ポイントの名前は、WinMain となります。

■ Empty Project (空のプロジェクト)

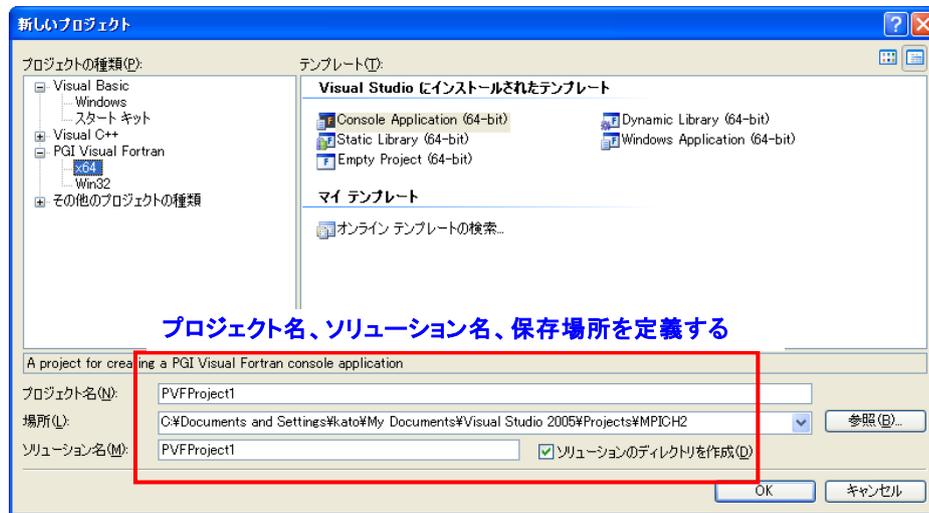
既存のアプリケーション (ソースコード等) を PVF に以降させる際に使用するスケルトン・プロジェクトです。これも、よく使用します。

2.4 新規にプロジェクトを作成し、新規にプログラム開発するための手続き

Visual Studio 2005/2008/2010 上で新規にプロジェクトを作成して、その配下で新たにプログラムのコーディング並びに開発作業を行うための準備について説明します。一般には、すでに所有している「プログラム・ソース・ファイル」を使用し、これを PVF プロジェクトに統合して開発を続行する方法がとられますが、これについては次項 2.5 項で説明します。

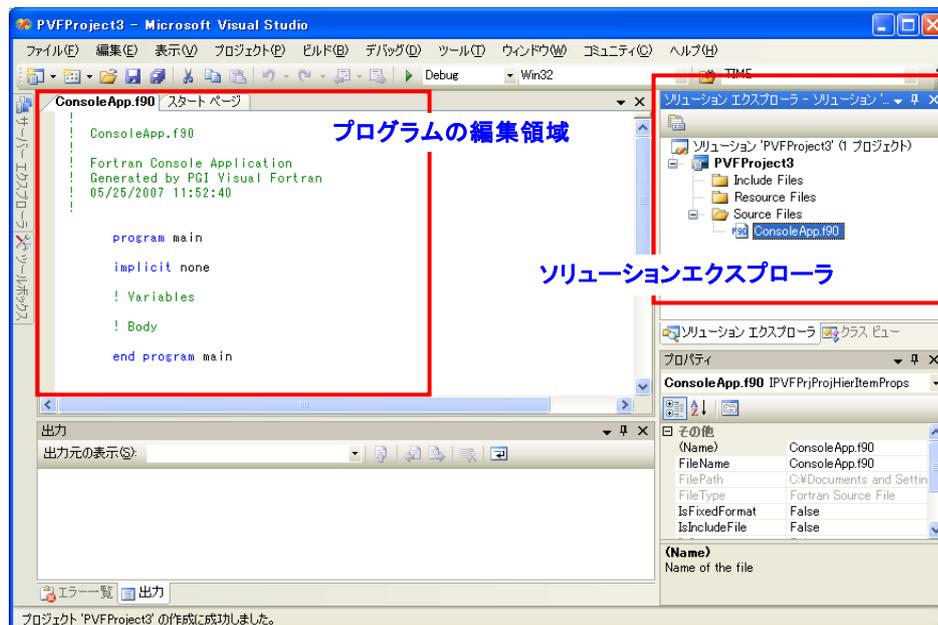
■ プロジェクトの新規作成

「ファイル」->「プロジェクト」を開き、「PGI Visual Fortran」のテンプレートの中の「Console Application」を選択します。なお、64 ビット Windows x64 の場合は、64 ビットアプリケーションあるいは、32 ビットアプリケーションの作成のどちらかを選択した上で、行ってください。



■ PVFプロジェクト画面表示例

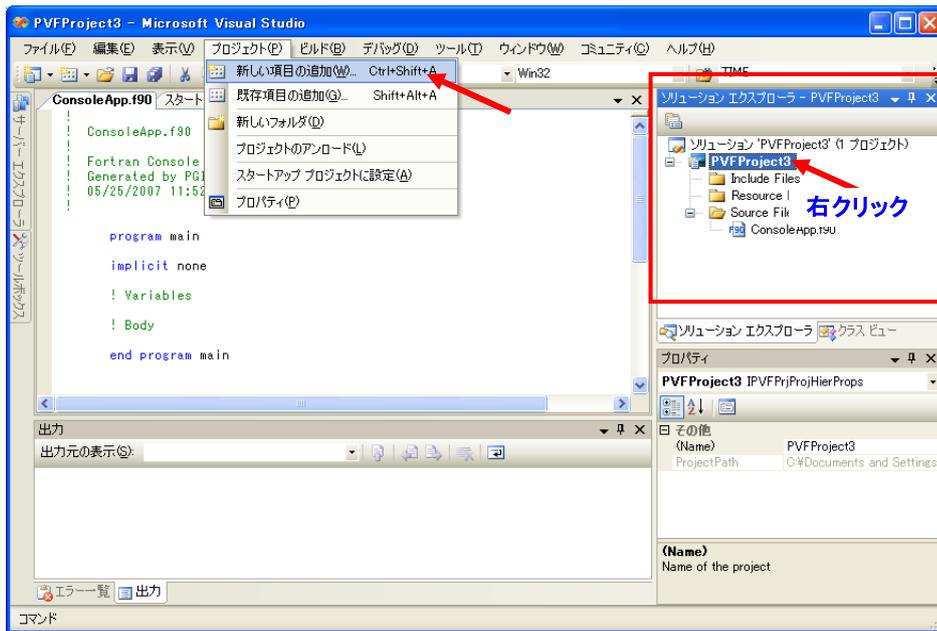
プロジェクトを新規作成した場合、ConsoleApp.f90 というファイル名で Fortran のスケルトン・コードが作成されます。この中でプログラムを開発します。



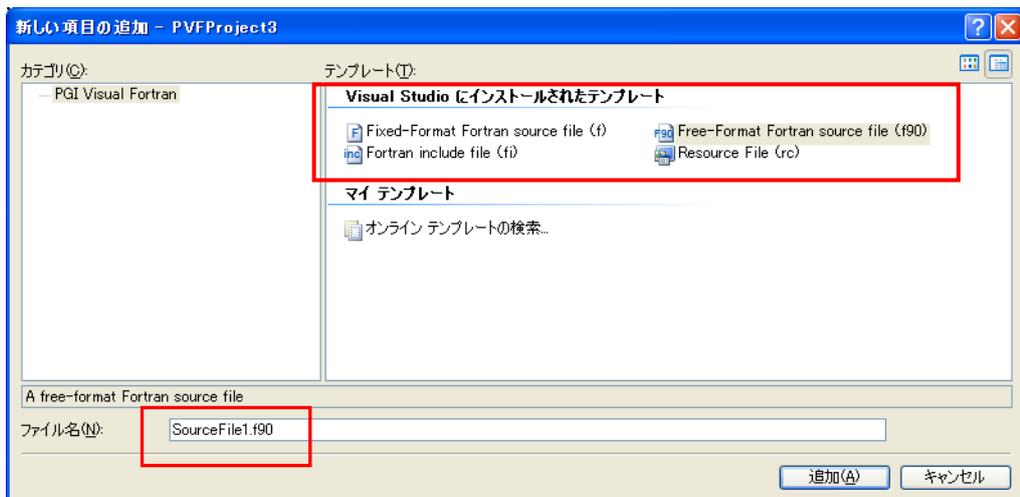
「ソリューションエクスプローラ」は、IDE 管理の下にある「ソリューション」、「プロジェクト」、その配下の各ファイル・フォルダを管理するためのものです。この中でソースファイル等が管理できます。

■ 新しいソースファイルを追加

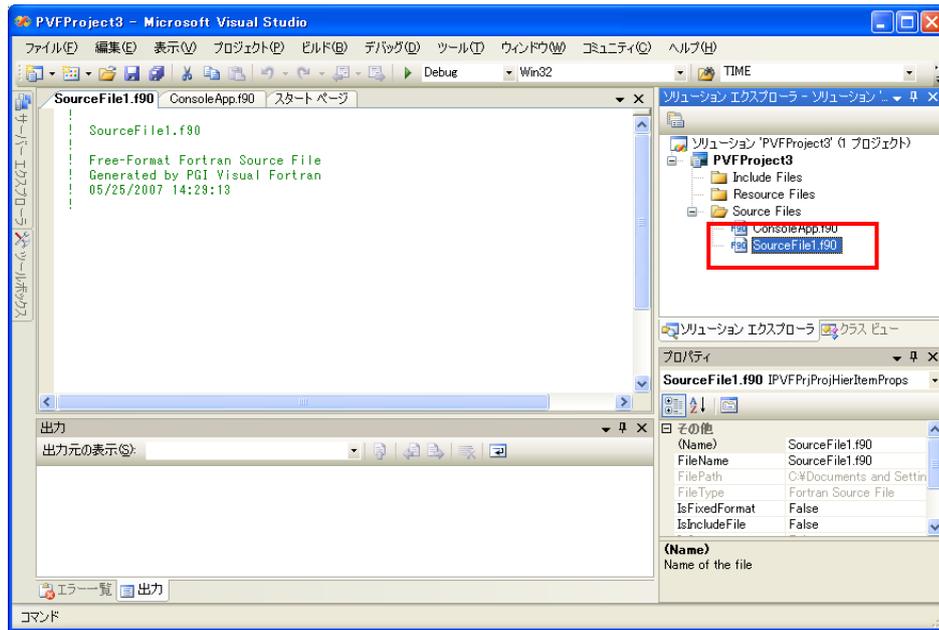
「プロジェクト」内に新しいソースファイルを追加したい場合は、「プロジェクト」->「新しい項目の追加」を選択します。あるいは、「ソリューションエクスプローラ」内で、プロジェクト名の文字列部分を右クリックし、「追加」->「新しい項目」を選択します。



Fortran のソースファイルのテンプレートを選択し、ファイル名を記述し、「追加」ボタンをクリックします。



これによって新しいファイル (SourceFile1.f90) が作成されます。

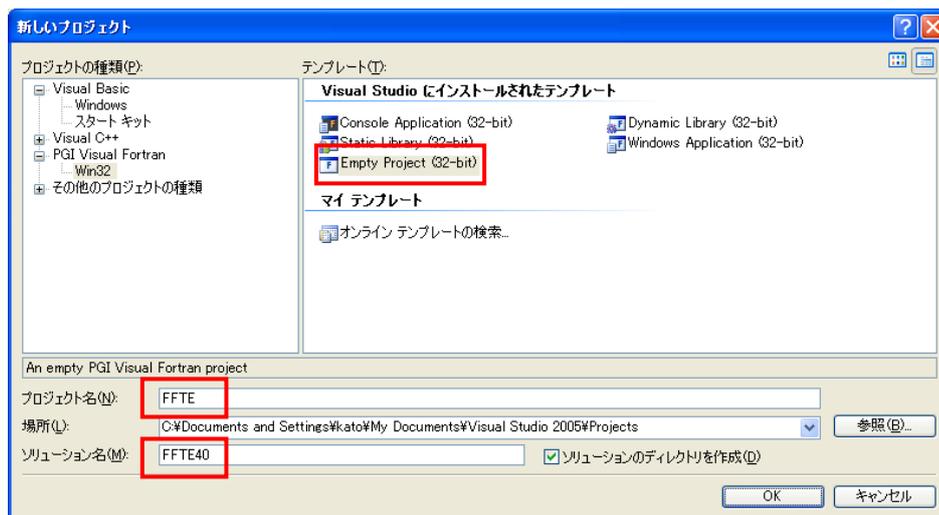


2.5 既存のソースファイルを PVF プロジェクトに移行するための手続き

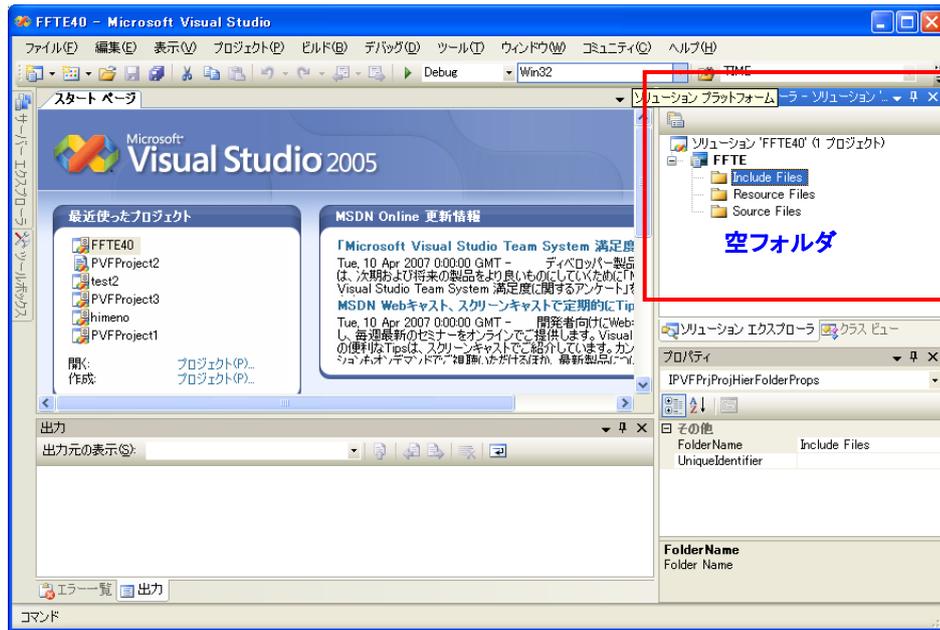
現在、Windows 上のフォルダに既存のプログラムファイルを有しており、これを Visual Studio 2005/2008/2010 の PVF プロジェクトに移行する方法を説明します。

■ プロジェクトの新規作成

「ファイル」->「プロジェクト」を開き、「PGI Visual Fortran」のテンプレートの中の「Empty Project」を選択します。さらに、ここでの例として、ソリューション名を「FFTE40」とし、プロジェクト名を「FFTE」という名前前で定義します。

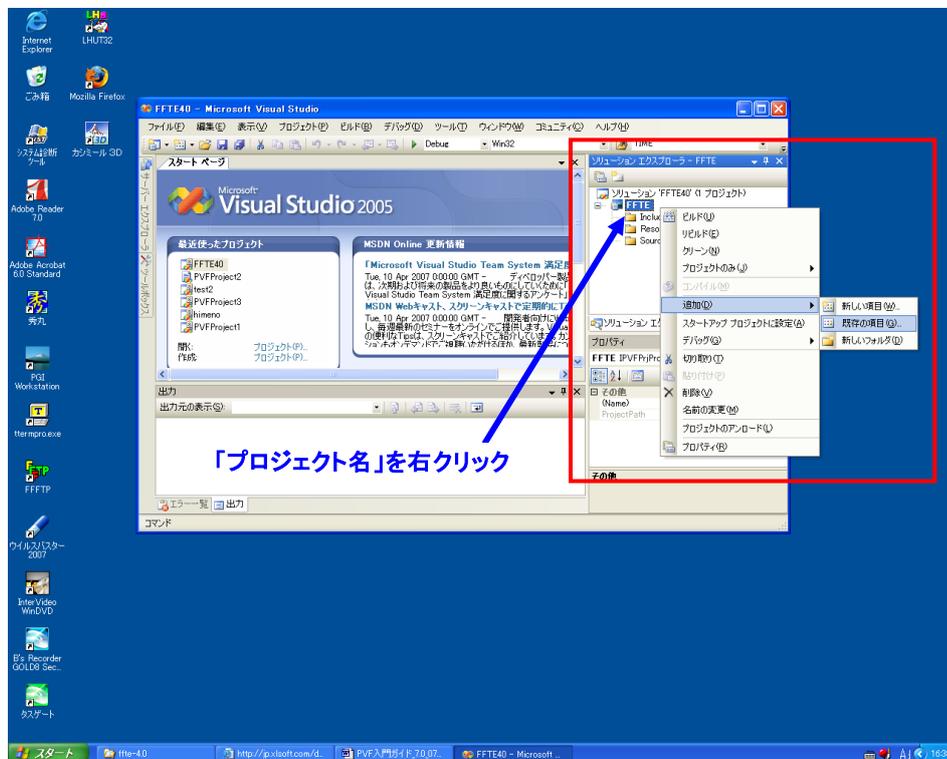


「Empty Project」を作成すると「ソリューションエクスプローラ」の中に、空のフォルダが作成されます。



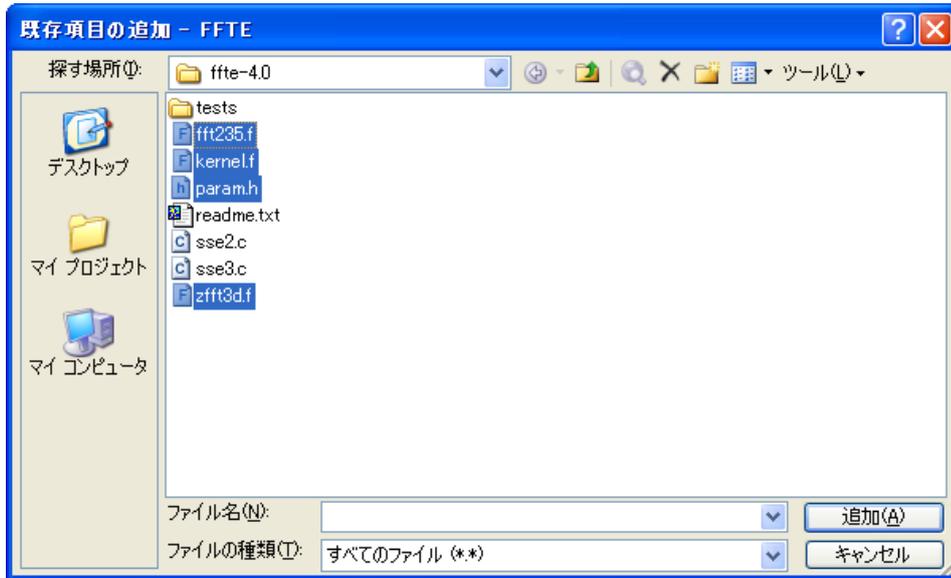
■ 既存のソースファイルをPVF環境へ移行

さて、これより、この IDE 環境の中に、既存のソースファイルを一つの「PVF プロジェクト」としてリンクします（移行します）。この実現方法にはいくつかの方法がありますが、ここでは、「ソリューションエクスプローラ」の中から操作する方法を説明します。「ソリューションエクスプローラ」の中の「プロジェクト名」を右クリックして現れるメニューの「追加」を選択し、「既存の項目(G)」を選びます。

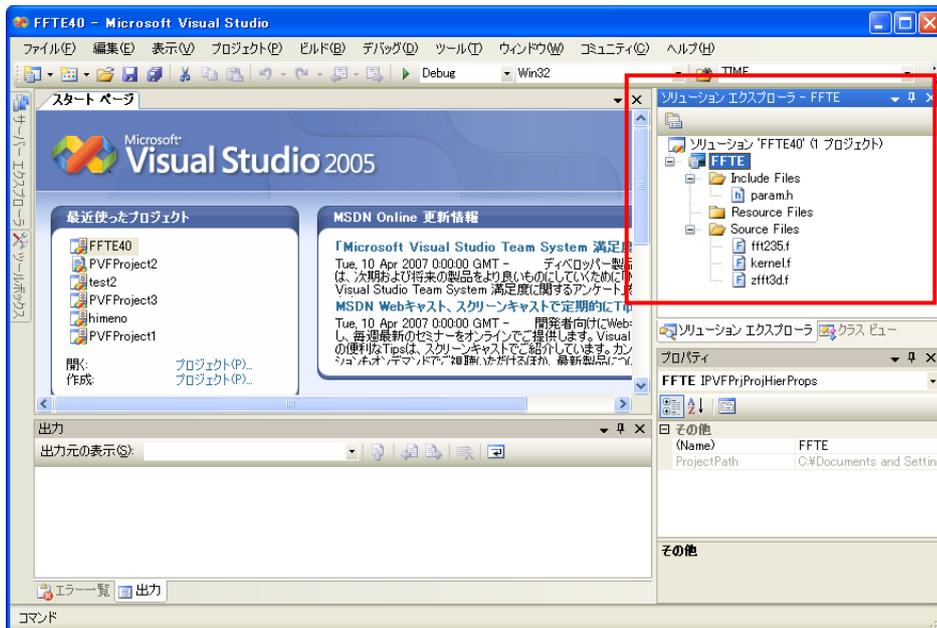


次に、「追加」する既存のファイルを指定するための画面が現れます。Shift キーを押しながら、「FFTE」プロジェクトに組み込みたいソースファイルを

選択し、「追加」ボタンを押します。



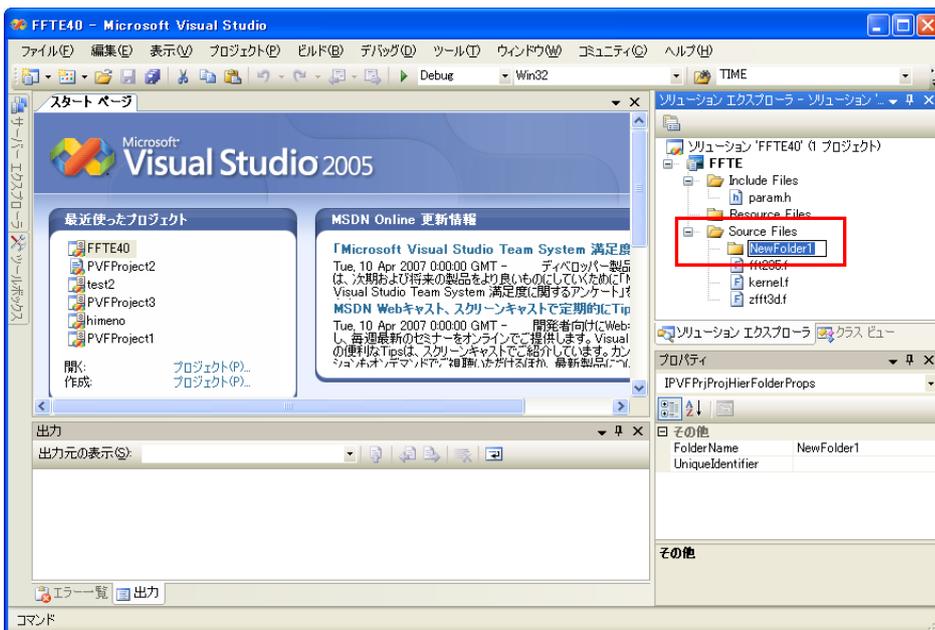
ソースファイルとヘッダーファイル（インクルードファイル）は、ファイルの種類別を確認して Visual Studio 2005/2008/2010 プロジェクト内の所定のファイル・フォルダの中に自動的に登録されます。（以下の図は、その様子を表しています）



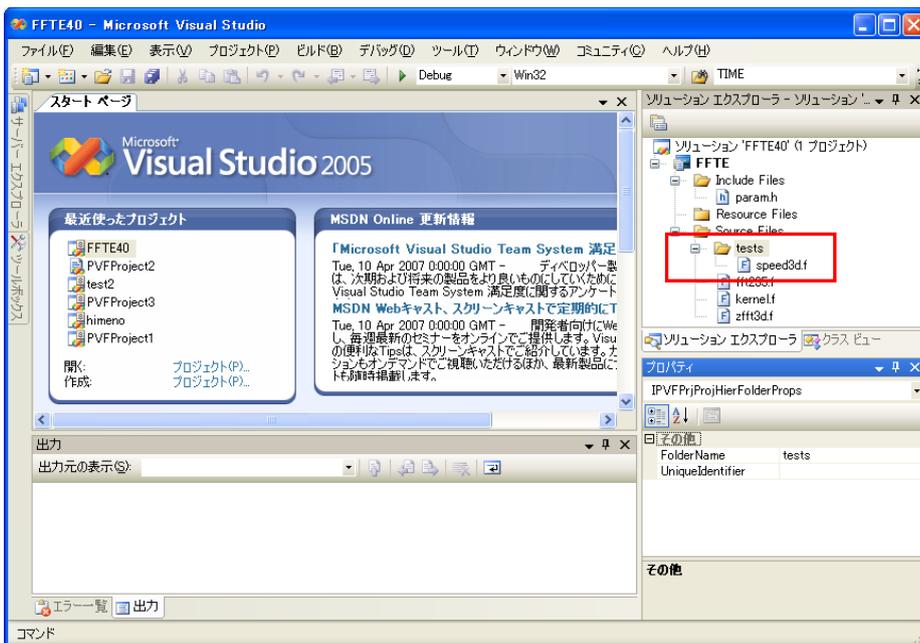
正確に言えば、既存のソースファイルは、Visual Studio 2005/2008/2010 の当該プロジェクト用のフォルダ内にコピーされるのではなく、既存のファイル・フォルダの位置を Visual Studio 2005/2008/2010 の PVF プロジェクトに登録するということになります。したがって、Visual Studio 2005/2008/2010 (PVF) 上で編集・変更されたソースファイル等は既存のフォルダ内のファイルが編集され、また、ファイルの削除を IDE 上で行った場合も、実際の既存フォルダ内のファイルが削除されますのでご注意ください。ただし、既存のソースファイル以外のコンパイル時の中間オブジェクトファイルや生成される実行モジュールは、Visual Studio 2005/2008/2010 でプロジェクトを登録した「場所」に保持されます。

■ PVFプロジェクト内にフォルダを新設

「ソリューションエクスプローラ」の中の「プロジェクト」配下には、予め用意されている三つのフォルダが存在します。その中の一つである「Source Files」フォルダ内に、(PVF に移行した) ソースファイルが登録されております。この「Source Files」フォルダ内にさらに新規のフォルダを新設し、その中にも既存のソースファイルを移行するための作業を行ってみます。「Source Files」フォルダの文字列を右クリックして現れるメニューの「追加」->「新しいフォルダ(D)」を選びます。以下のように新しいフォルダが新設されますので、そのフォルダ名を設定します。



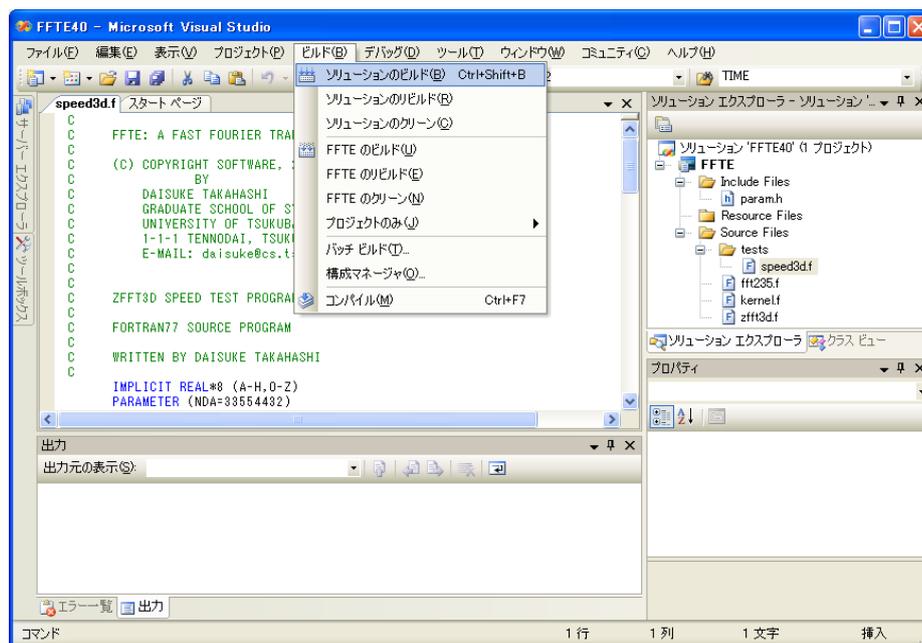
ここでは、新しいフォルダを「tests」と定義し、前と同様に既存のソースファイルをこの「tests」フォルダの中へ移行・登録します。「tests」フォルダ上で右クリックして、「追加」->「既存の項目(G)」を選び、既存のソースファイル（以下の例では speed3d.f）の場所を指定して登録します。この例では、この speed3d.f が Fortran のメイン・プログラムになります。



■ ルーチン間の依存性の解析について

プログラムが複数のルーチン、Fortran モジュール、インクルードファイルから構成され、さらにその関係に依存性がある場合は、そのコンパイルする順序が的確でなければなりません。Linux の GNU 系では、これを Makefile で依存性を定義しましたが、Visual Studio 2005/2008/2010 では、そのような Makefile を作成する必要がありません。初回の「ソリューション・ビルド」において、ルーチン間の依存性の存在を検証し内部的にコンパイルする順序の情報を保持します。

既存のソースファイルを PVF プロジェクトに移行した場合は、以下のように、「ビルド」 → 「ソリューションのビルド」を初回に行ってください。これによって、ルーチン間の依存性解析を行い、その情報を保持します。

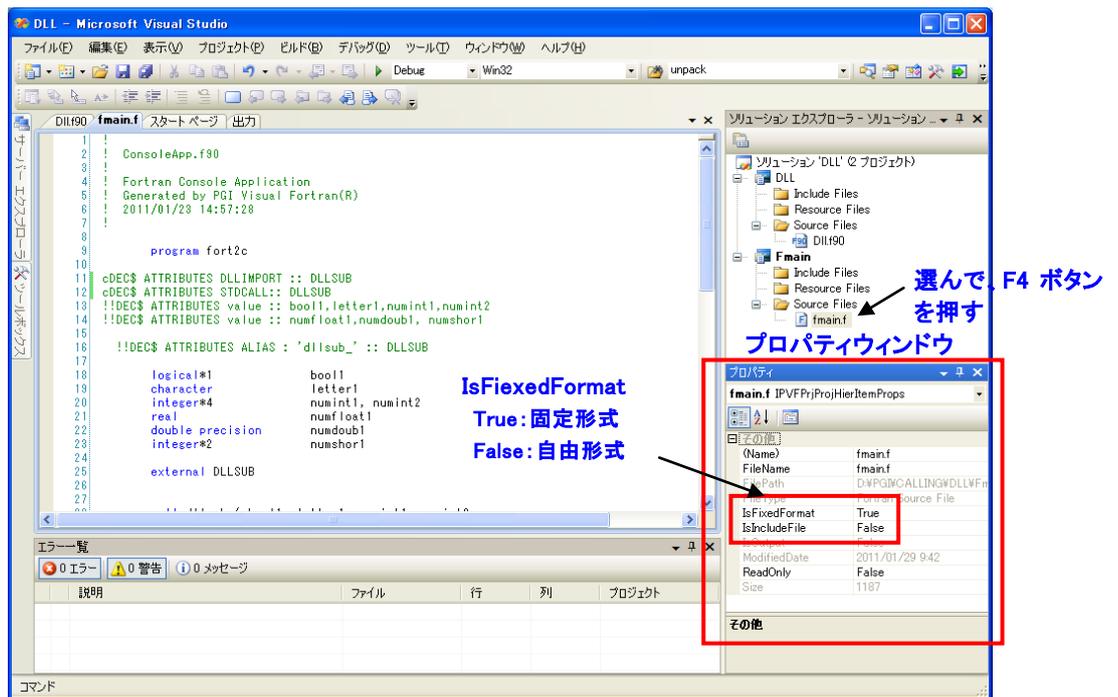


2.6 Fortran ソースファイルの属性(固定形式、自由形式)

PVF は、FORTRAN 77 構文で採用されている 1 行に 1 から 72 カラム内で構文を記述する「固定形式 (fixed)」と Fortran 90 以降に採用された「自由形式 (free)」の構文記述のどちらでも使用できます。PGI の Fortran ファイル名に関するコンベンション (慣習) についての詳細は、以下の PGI 使用ガイドの 1.4 項 (P4) をご覧ください。

http://www.softtek.co.jp/SPG/Pgi/public/doc/PGI_softtek_ug.pdf

PVF は、ファイル名のサフィックスを .f や .for としている場合は、「固定形式のソースファイル」として、また、.f90 や .f95 のサフィックスを有する場合は、「自由形式のソースファイル」として認識します。各ソースファイルの属性に関して調べる場合は、「ソリューションエクスプローラ」のウィンドウ内の当該ファイルを選び (ドラッグ) してから、F4 ファンクションキーを押します。(あるいは、「表示」メニュー内の「プロパティウィンドウ (F4)」を選択します。以下のような画面が現れます。これが、ファイルの属性情報のウィンドウです。



この中で、以下のプロパティで、固定形式か自由形式かを選択し、形式を変更することができます。

IsFixedFormat :True は、固定形式、Fales は、自由形式

また、以下のプロパティは、次の意味を持ちます。

IsIncludeFile :True は、インクルードファイルであることを指示し、コンパイルは行わないファイルと認識します。Fales は、ビルドにおいて使用する Fortran/Resource ファイルであることを指示します。このフラグを使って、あえて、コンパイル(ビルド)に使用しないファイルとして指示する場合にも使うことができます。

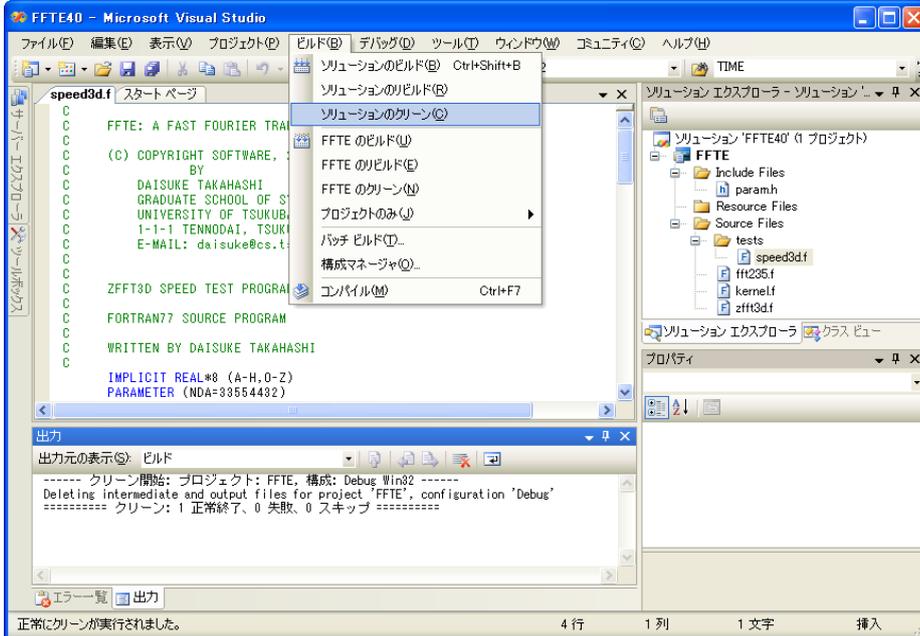
2.7 プログラムのコンパイルと実行(デバッグモード)

PVF 上でプログラムをコンパイルする方法を説明します。Visual Studio 2005/2008/2010 でのデフォルトのプロジェクト構成は、「デバッグ構成」のため、最適化レベルが0でシンボリック情報が含まれた形で実行モジュールがビルドされます。(注意：デバッグモードの実行バイナリは、性能が遅いです。)



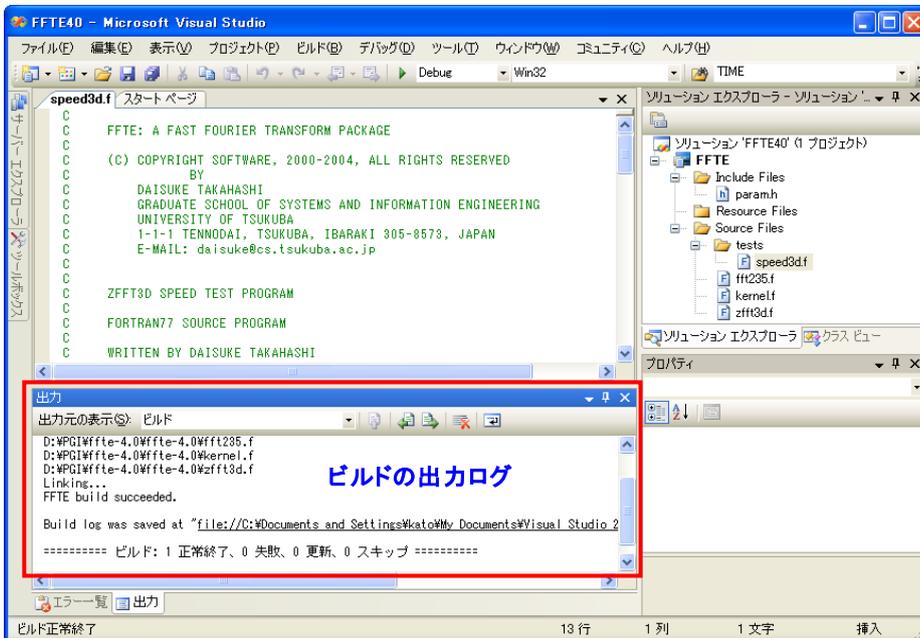
■ ソリューションのクリーン

Visual Studio 2005/2008/2010 でソリューションをビルドする前に、クリーンアップします。「ビルド」 → 「ソリューションのクリーン」を実行します。



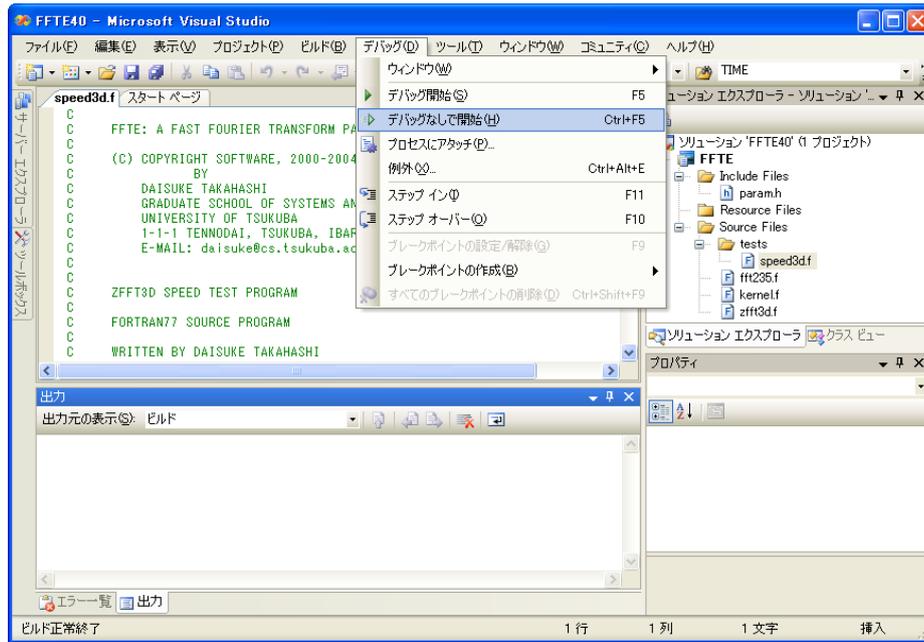
■ ソリューションのビルド

次に、「ビルド」 → 「ソリューションのビルド」を実行します。ビルドの出力ログが表示されます。「FFTE build succeeded」と表示されますとビルドが成功したことを意味します。



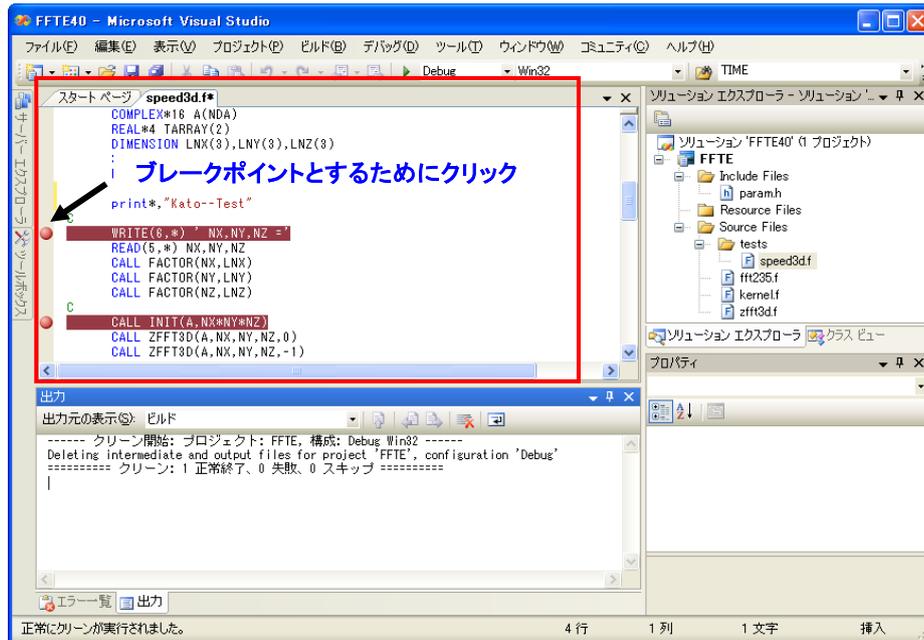
■ プログラムの実行(デバッグなし)

ビルドされた実行モジュールを実行します。「デバッグ」 → 「デバッグなしで開始」を実行するとプログラムの実行が開始されます。

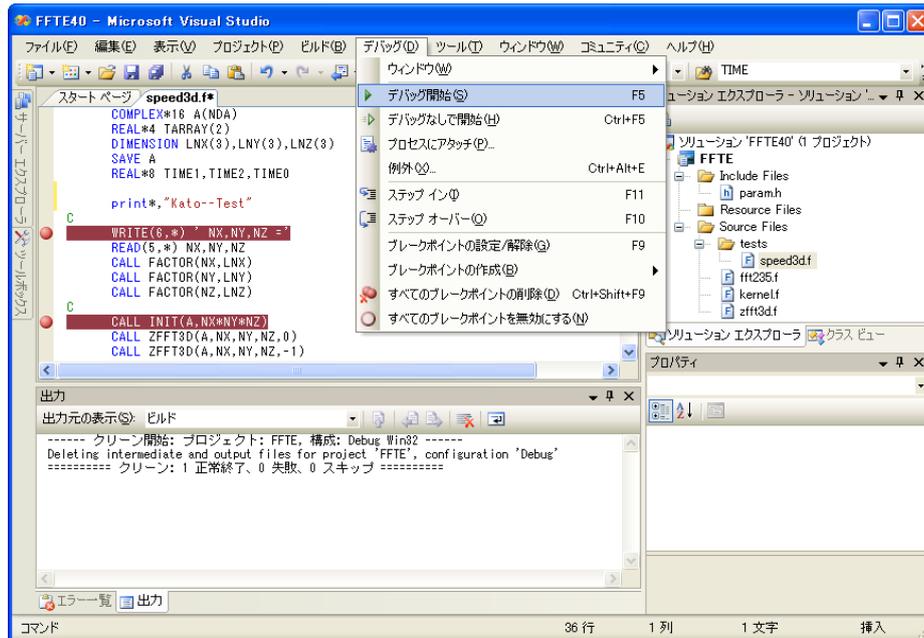


■ プログラムの実行(デバッグあり)

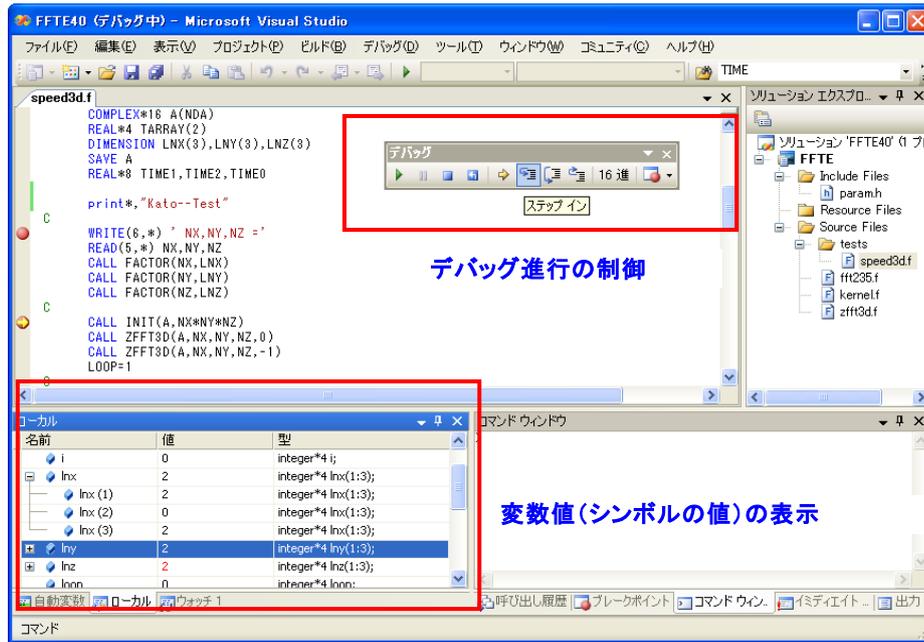
ソースレベルでデバッグを行いたい場合は、予め、プログラムの表示ウィンドウ内で、「ブレークポイント」を設定しておきます。「ブレークポイント」の設定は、対象となるソースラインの一番左端をクリックすることで設定できます。デバッグ付で実行した場合、このブレークポイントで実行が停止します。



次に、「デバッグ」 -> 「デバッグ開始」を実行するとプログラムのデバッグ実行が開始されます。そして、最初の「ブレークポイント」で実行が停止します。



デバッグが開始されると、「ブレークポイント」での変数値、配列値の確認を行い、ステップ実行等の操作で問題となる部分の検証を行います。

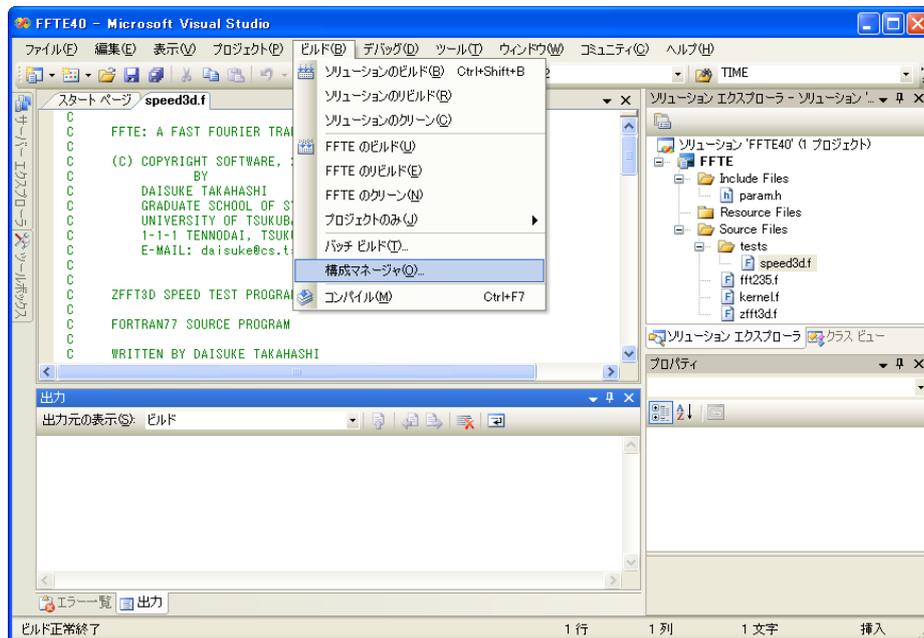


2.8 プログラムのコンパイルと実行(最適化オプションの適用)

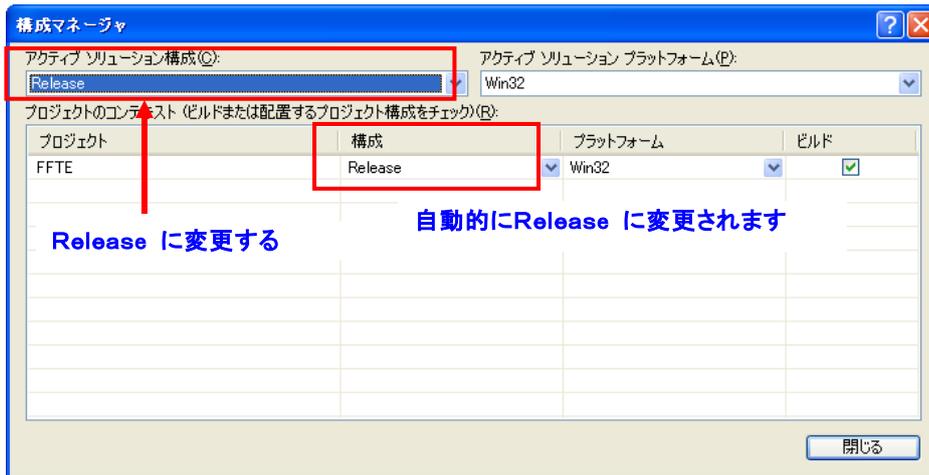
PVF コンパイラによる最適化オプションを適用してビルドする方法を説明します。

■ ソリューションの最適化ビルド

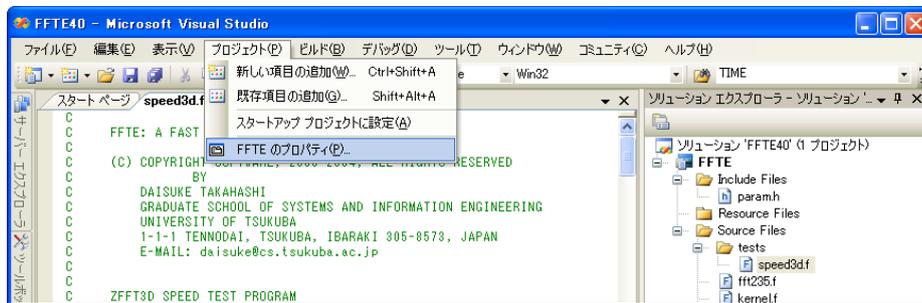
「ビルド」 -> 「構成マネージャ」を実行します。



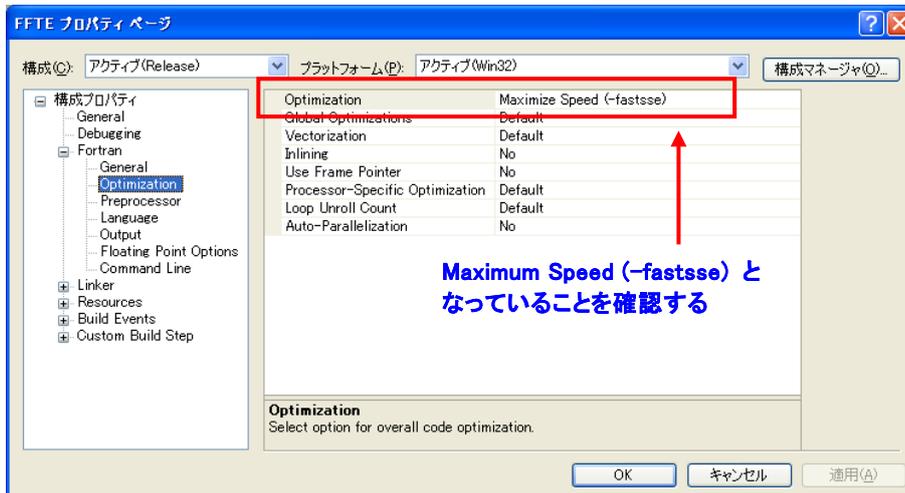
構成マネージャの画面が出ましたら、「アクティブソリューション構成」のメニューを「Debug」から「Release」に変更します。



次に、「プロジェクト」 -> 「…のプロパティ」を選択し実行します。



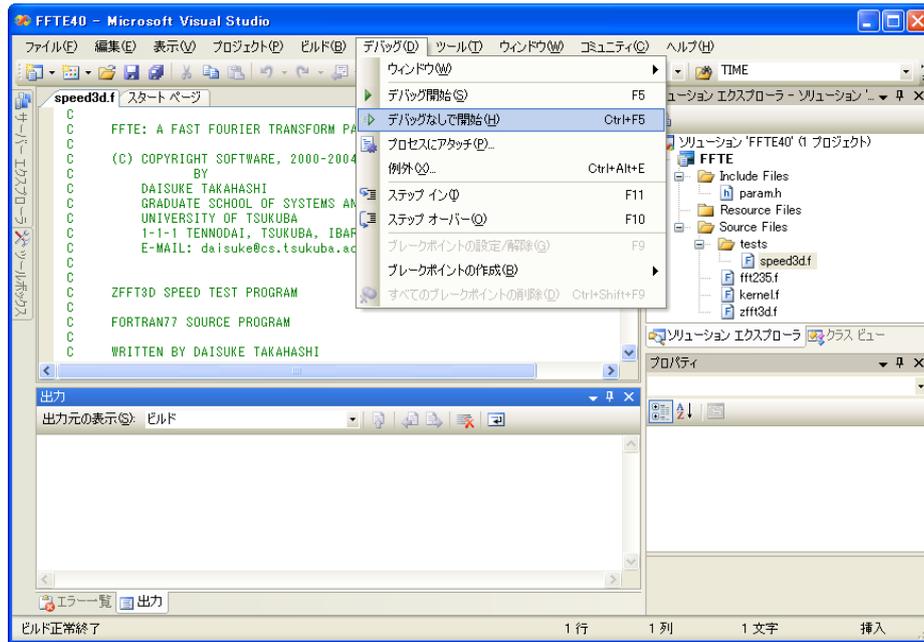
プロジェクトのプロパティ画面が現れますので、その中の「Fortran」->「Optimization」を選択し、この設定が[Maximum Speed (-fastsse)]となっていることを確認（あるいは設定）してください。さらに、「Fortran」->「Command Line」にて、その他の最適化オプションを定義することも可能です。



PVF のプログラムの最適化レベルのデフォルトは-02 です。上記の-fastsse をセットすると、最良な性能最適化のデフォルト値がセットされます。

■ プログラムの実行(最適化オプションあり)

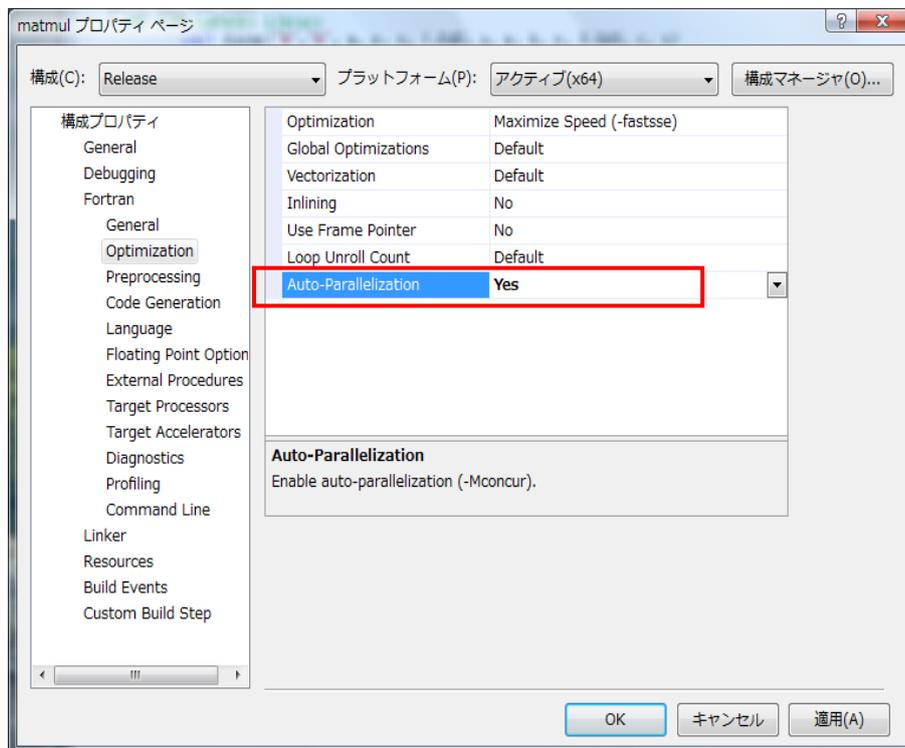
ビルドされた実行モジュールを実行します。「デバッグ」->「デバッグなしで開始」を実行すると最適化されたプログラムの実行が開始されます。



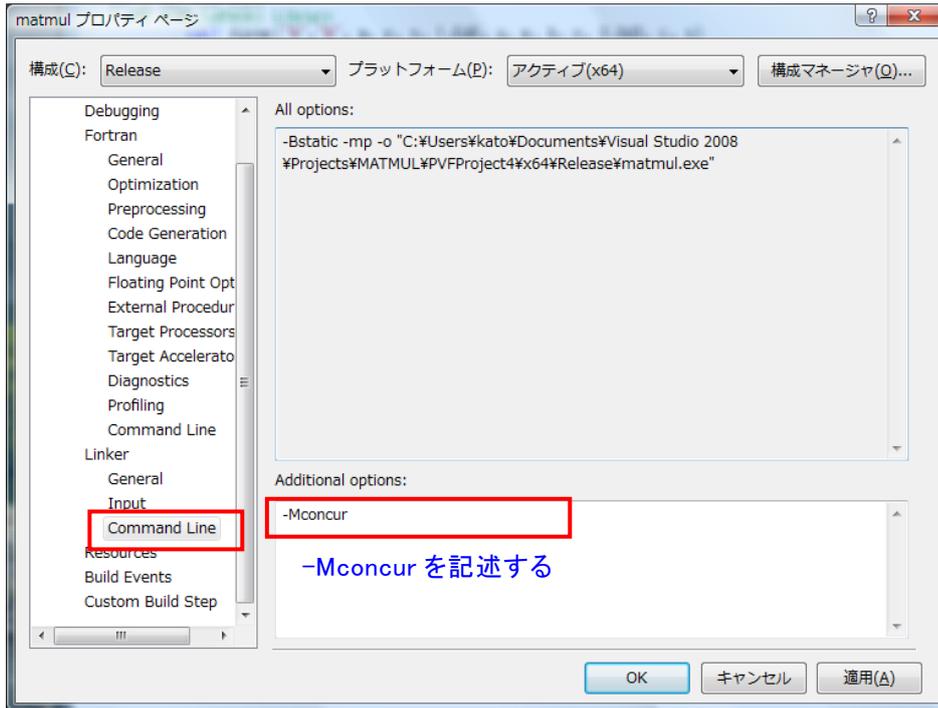
■ 並列化最適化オプションについて

① 自動並列化オプション

「プロジェクトのプロパティ」画面で、「Fortran」->「Optimization」->「Auto-Parallelization」の設定を[Yes]とします。これによって、並列依存性のないループ構造に対して、コンパイラが並列化を施します。

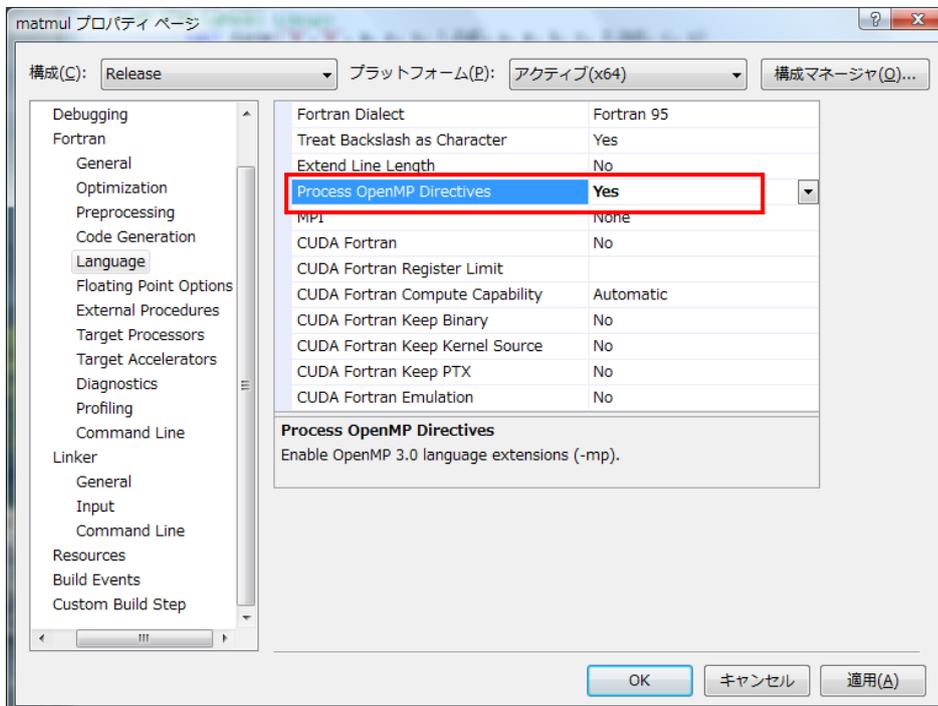


次に、「Linker」->「Command Line」の Additional options: に -Mconcur を入力します。



② OpenMP 並列化オプション

「プロジェクトのプロパティ」画面で、「Fortran」→「Language」→「Process OpenMP Directives」の設定を [Yes] とします。これによって、コンパイラは OpenMP ディレクティブを解釈し、並列化コードを生成します。



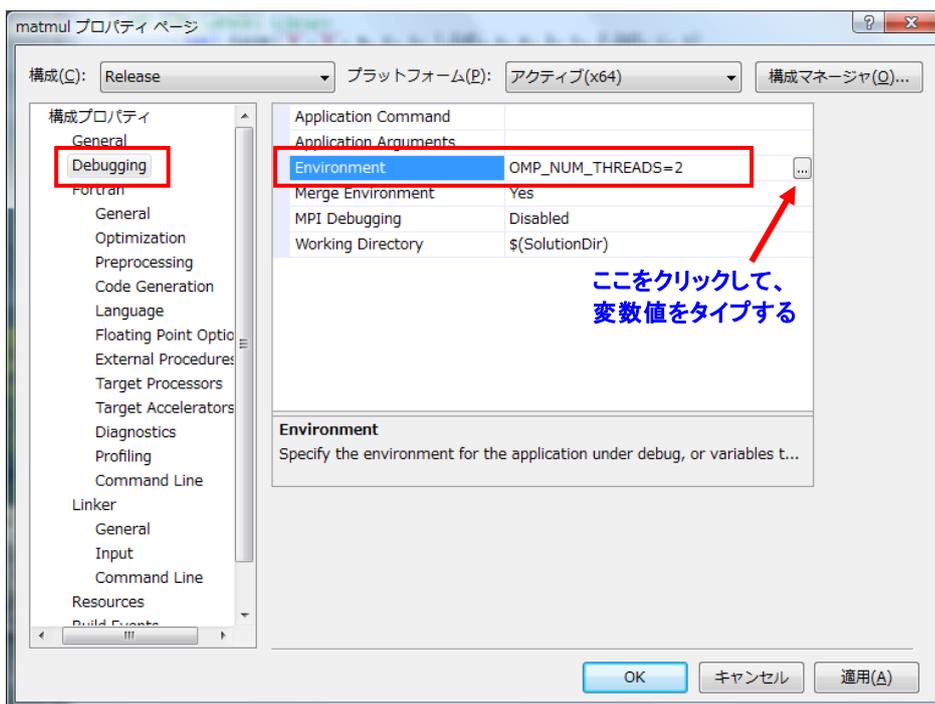
■ 自動並列、OpenMP並列実行時の並列スレッド数の環境変数の設定

「プロジェクトのプロパティ」画面で、「Debugging」->「Environment」の設定ボタンをクリックして、以下の環境変数をセットします。これを事前にセットした後、プログラムを実行してください。この変数を設定しなければ並列実行しません。この設定は、「デバッグモードの実行」だけでなく、「Release モードの実行」においても有効となります。

OMP_NUM_THREADS=<並列 CPU コア数> (例：OMP_NUM_THREADS=2)

あるいは、

NCPUS=<並列 CPU コア数> (例：NCPUS=2)



あるいは、別の方法として、Windows®システム上の「環境変数」を設定する方法があります。Windows®の「環境変数」の設定方法は、以下の URL をご参照ください。なお、設定する変数は、上記で示した OMP_NUM_THREADS あるいは、NCPUS となります。この変数を反映させるために、本変数設定後、Visual Studio 2005/2008/2010 を起動するようにしてください。

<http://www.softtek.co.jp/SPG/Pgi/win64/win64use.html>

PVF コマンドプロンプト（下記、4 章を参照）上で、各種の環境変数を指定する場合は、DOS コマンドの set コマンドを使用します。以下は一例です。コマンドで指定した場合は、そのコマンド画面上のみに有効です。

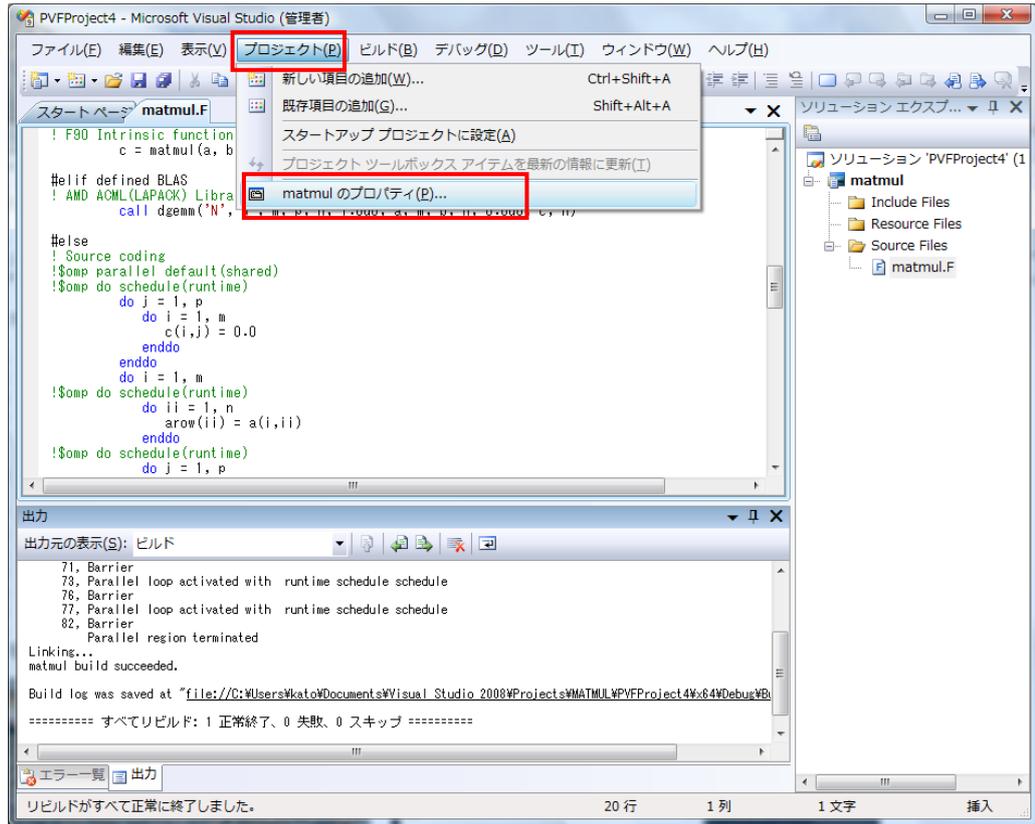
```
$ set OMP_NUM_THREADS=2
```

2.9 プログラムの実行(入力データファイルのリダイレクト)

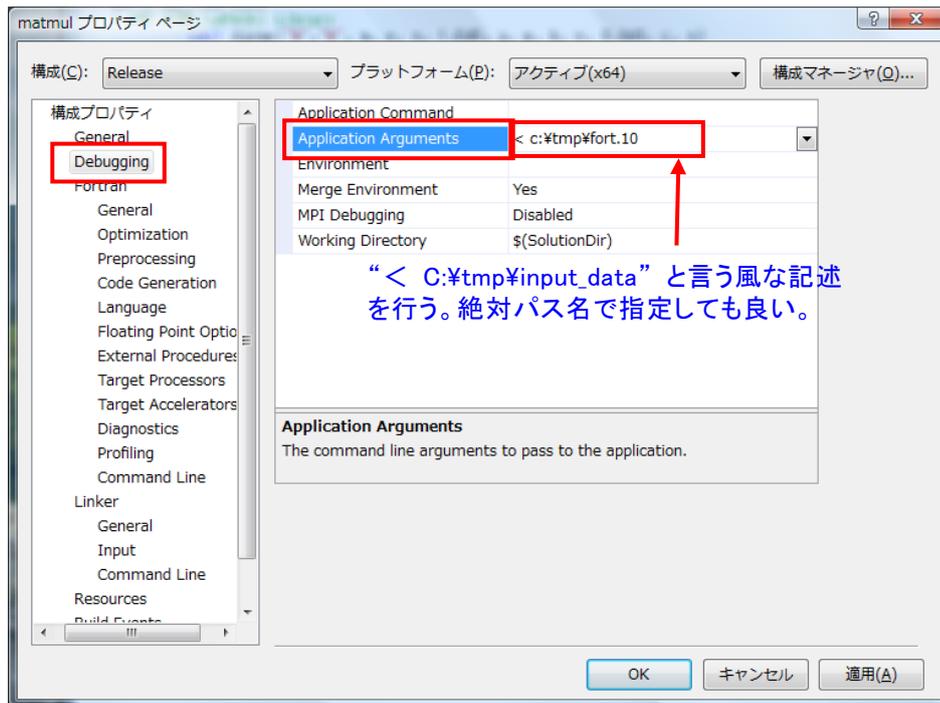
実行時に標準入力ファイルを指定して実行する方法を説明します。いわゆる、入力データを実行モジュールにリダイレクトする方法です。

■ 標準入力ファイルを指定する

「プロジェクト」->「(プロジェクト名) プロパティ」を選択し、プロパティページを開きます。

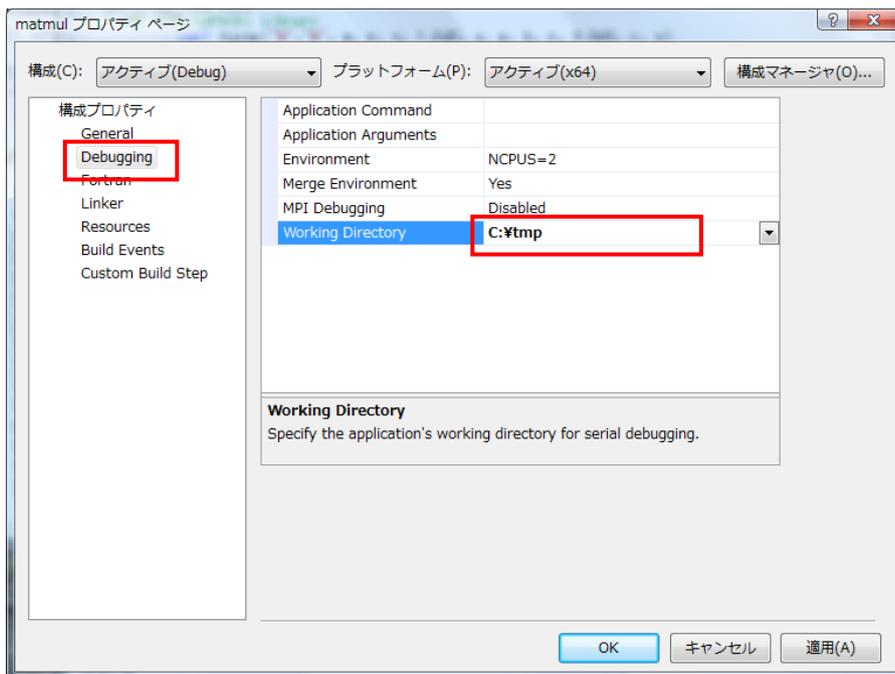


次に、「Debugging」->「Application Arguments」の欄に、“< ファイルパス名”と表記する。“<”マークは、リダイレクトを意味し、これに続けてファイル名を記します。一方、アプリケーション実行のデフォルトのワーキング・ディレクトリは、Visual Studio の「プロジェクトファイル」を含むディレクトリであるので、標準入力ファイルは、ここに置くだけでもよいです。



■ ワーキング・ディレクトリ(実行作業場所)の変更

「プロジェクト」 -> 「(プロジェクト名) プロパティ」を選択し、プロパティページを開きます。「Debugging」-> 「Working Directory」の欄に、実行時のワーキング・ディレクトリのパス名を指定します。デフォルトのワーキング・ディレクトリは、Visual Studio 上の使用プロジェクトの「プロジェクトファイル (*.pvfproj)」が置かれているフォルダとなります。このデフォルトのワーキング・ディレクトリを明示的に変更する際に指定します。この変更を行うと、このディレクトリ・フォルダの中に入力データ等を置くことができます。



3 NVIDIA GPU 用 PGI Accelerator と CUDA Fortran を使用する

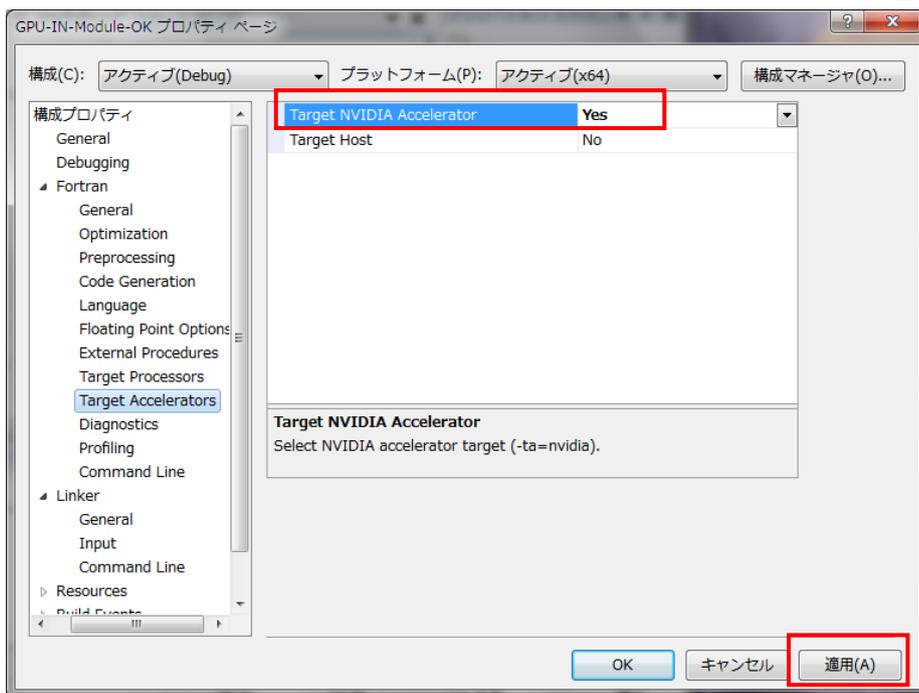
本章は、PGI アクセラレータ (Accelerator) コンパイラ・ライセンス (x64+GPU) の場合に使用できる機能を説明します。以下の画面イメージは、CUDA 3.0 をサポートした PVF 10.4 リビジョン以降の画面を使用しています。

3.1 The PGI Accelerator Programming Model (ディレクティブ使用)

PGI アクセラレータ・プログラミングモデルは、ユーザがプログラム上にディレクティブ (指示行) を挿入して GPU 並列計算ブロックを指示するモデルであり、プログラムの当該ループブロックを GPU 上の kernel プログラムとして実行することができるようにコンパイラが翻訳することができます。予め Accelerator 用ディレクティブを挿入してあるプログラムをコンパイルするための設定を説明します。

■ 「Target Accelerators」プロパティの設定

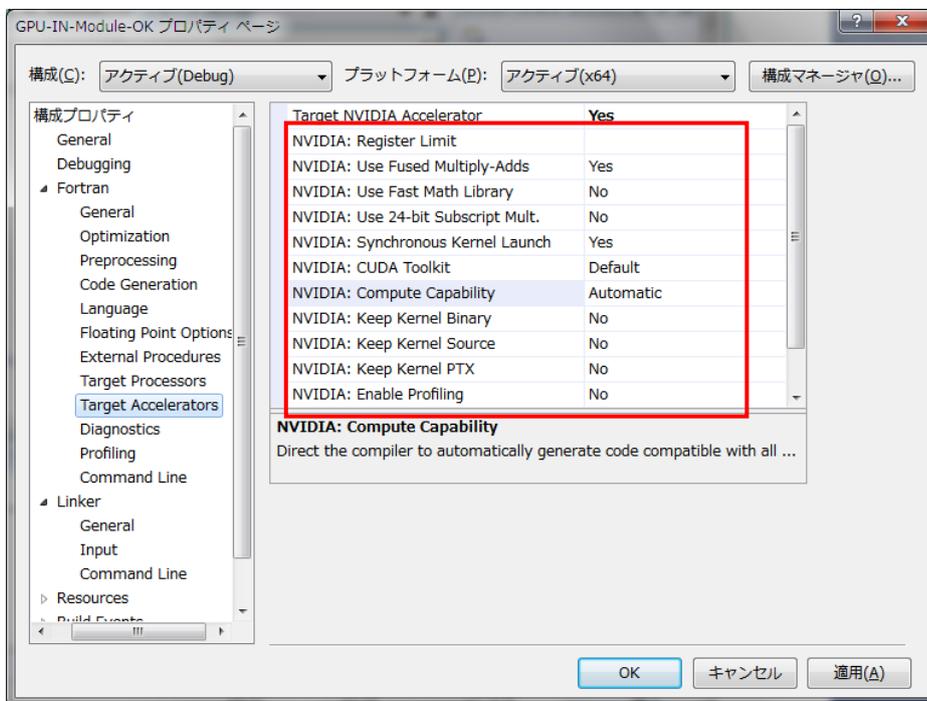
「プロジェクト」 -> 「(プロジェクト名) プロパティ」を選択し、プロパティページを開きます。「Target Accelerators」を選び、「Targeting NVIDIA Accelerator」を「Yes」にします。これによって、Accelerator 用のディレクティブを認識し、GPU 用のコード生成を行います。



上記の通り、「Yes」にした後、細かな Accelerator 用のコンパイルオプションを表示させるには、下部の「適用」ボタンをクリックすると、下図のような詳細なオプション・スイッチが現れます。なお、PGI Accelerator 用のコンパイルオプションの詳細は、以下の URL にて説明していますので、ご覧下さい。

(PGI アクセラレータ・コンパイル用のオプション)

http://www.softek.co.jp/SPG/Pgi/TIPS/opt_accel.html ([HTML リンク](#))



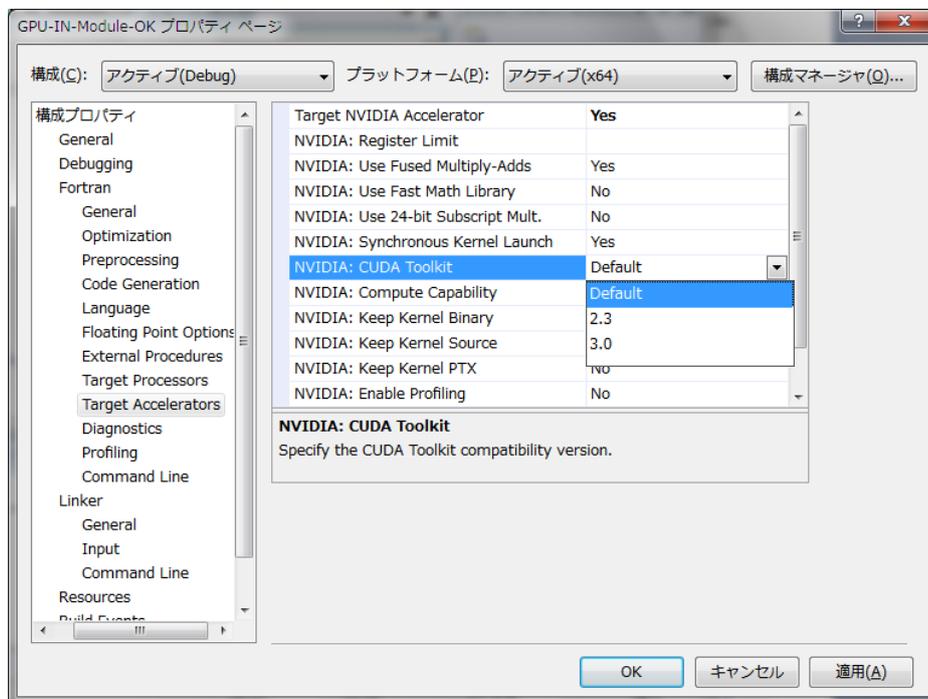
■ 「NVIDIA CUDA Toolkit」プロパティの設定

PVF 2010 の 10.4 以降では、CUDA Toolkit 3.x もサポートしましたので、それ以前の 2.3 も含め、二つの CUDA Toolkit のバージョンをサポートするようになりました

た。また、同様にシステムに実装されている NVIDIA のデバイス・ドライバーも 2.3 と 3. x のいずれかをサポートできるように構成されております。デバイス・ドライバーは、PVF Command Shell のコマンドプロンプト上で pgaccelinfo コマンドを実行するとドライバーのバージョンが表示されます。

For a 2.3 driver: CUDA Driver Version 2030 と表示

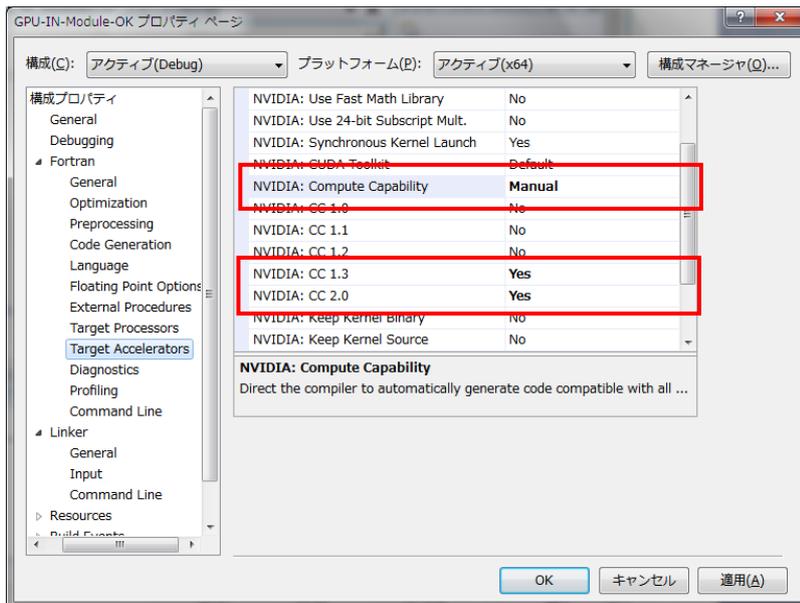
For a 3.1 driver: CUDA Driver Version 3100 と表示 (一例)



CUDA Toolkit のバージョンを明示的に指定して、その実行バイナリを作成する場合は、上記の画面で示すとおり「NVIDIA CUDA Toolkit」のプロパティでそのバージョンを指定します。「Default」を指定した場合は、2.3 バージョンがデフォルトとなります。

■ 「NVIDIA Compute Capability」プロパティの設定

NVIDIA の GPU には、そのハードウェア特性を識別するために、「NVIDIA Compute Capability」(1.0, 1.1, 1.2, 1.3 and 2.0) と言う番号が付けられています。例えば、十分な倍精度性能の対応がなされた Fermi のハードウェアは、2.0 以上です。コンパイラはデフォルトで、自動的に、使用している CUDA Toolkit でサポートされる全ての Compute Capability 用のコードを生成しますが、明示的に Compute Capability を (複数) 指定してバイナリを生成することもできます。



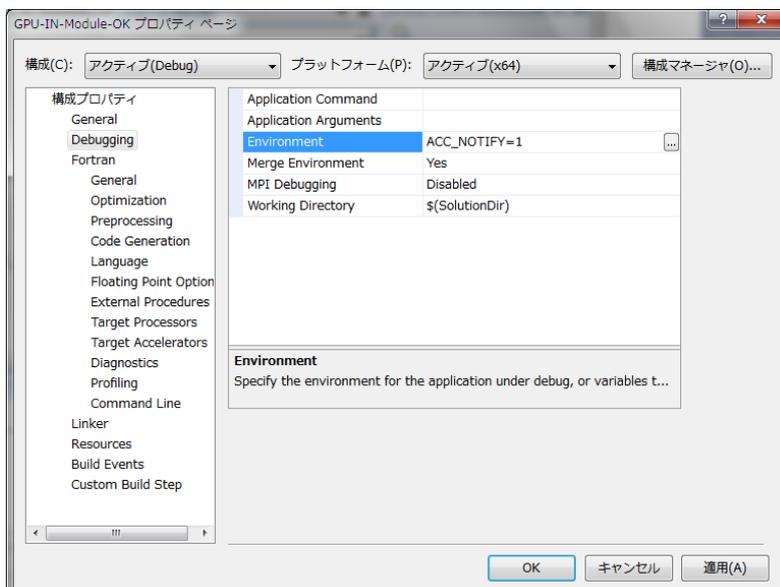
■ PGI Accelerator用のコンパイルメッセージ出力のプロパティの設定

PGI アクセラレータ用にコンパイルする際の詳細コンパイル情報を出すには、「Fortran」->「Diagnostic」->「Accelerator Information」を「Yes」とすることにより、コンパイル時、「出力ウィンドウ」内に詳細なコンパイル情報を出すことができます。

■ PGI Accelerator用の環境変数の設定

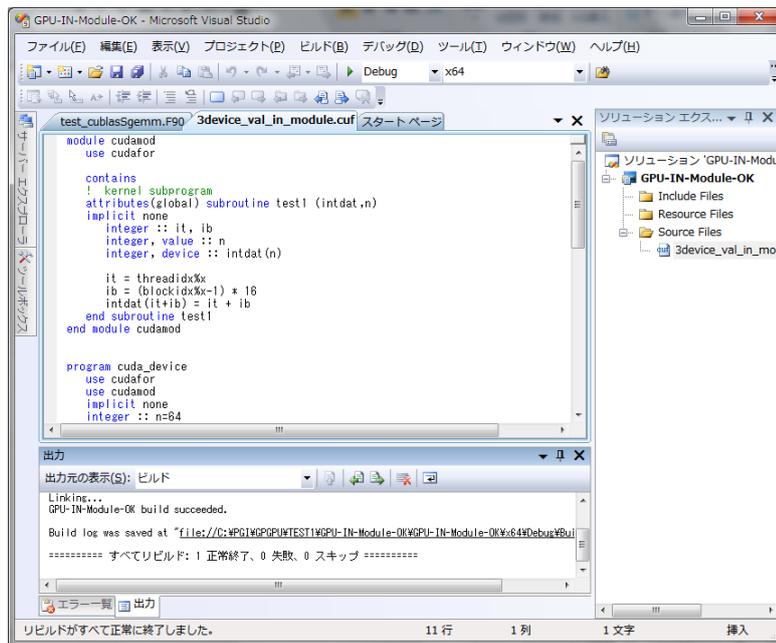
PGI アクセラレータ用の実行モジュールを実行する際の環境変数の設定は、以下のように行います。「Debugging」->「Environment」の入力欄に、環境変数を入力します。これは、Visual Studio の「デバッグモード」での実行だけでなく、「Releaseモード」のバイナリの実行においても、有効に機能します。以下の例は、**ACC_NOTIFY** と言う、GPU 内の kernel 実行が行われる毎にとの動作特性を出力するための機能ですが、それを有効に設定したものです。

また、複数の GPU ボードが実装されているシステムでは、**ACC_DEVICE_NUM** と言う環境変数でその論理番号を設定することにより、実行に使用する GPU を指定することができます。



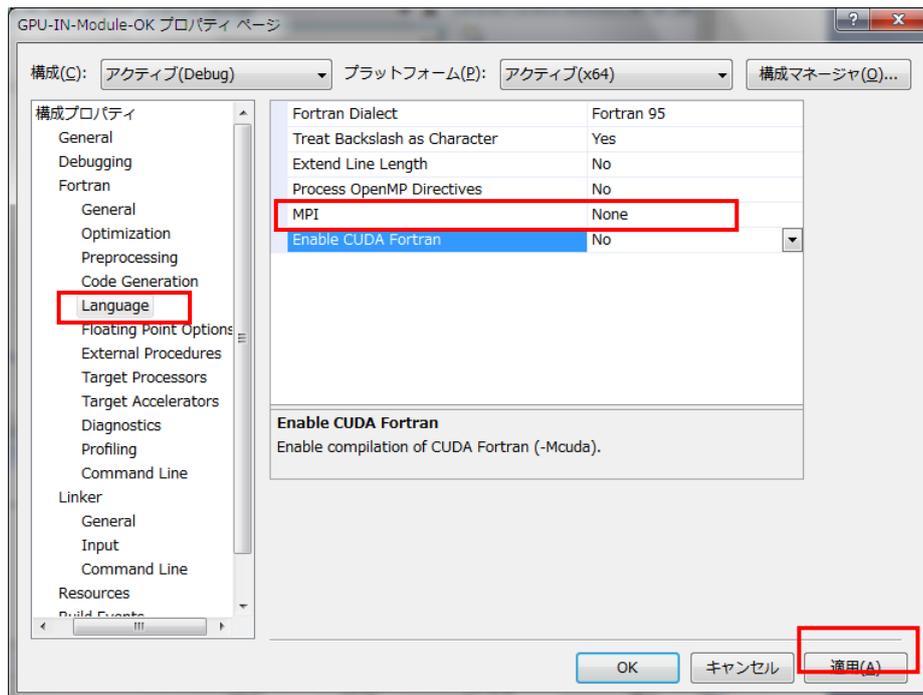
3.2 PGI CUDA Fortran のコンパイル

PGI CUDA Fortran は、NVIDIA CUDA C と同じ考え方による GPU の処理を抽象化できる Fortran 構文を実装し、NVIDIA GPU 用に直接プログラミングすることが可能な Fortran コンパイラです。PGI Visual Fortran は、CUDA Fortran をフルサポートし、CUDA Fortran コンパイル用のプロパティを設定することにより、コンパイルが可能となります。なお、CUDA Fortran のファイルは、.cuf というファイル拡張子を付けることにより認識されますが、適切なリンク処理を行うために、以下に説明する CUDA Fortran 用のプロパティを必ず設定して下さい。なお、PVF editor 上では、CUDA Fortran キーワード・構文の色識別ができます。



■ 「Target Accelerators」プロパティの設定

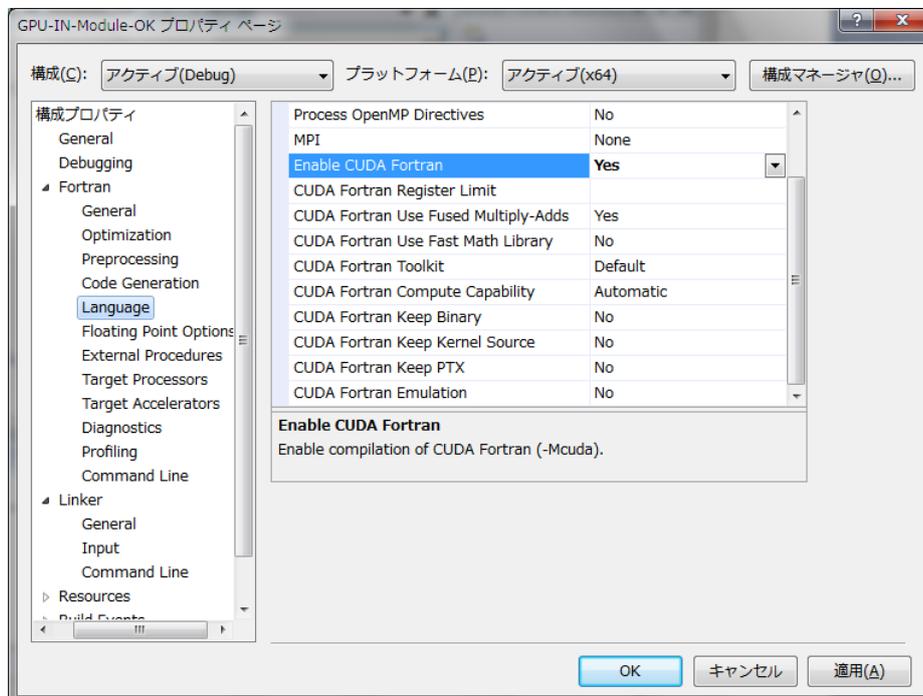
「プロジェクト」 → 「(プロジェクト名) プロパティ」を選択し、プロパティページを開きます。「Fortran」を選び、「Language」の中の「Enable CUDA Fortran」を「Yes」にします。これにより、CUDA Fortran 構文を認識してコンパイルを行うことができます。



なお、「Yes」にした後、細かな CUDA Fortran 用のコンパイルオプションを表示させるには、一旦、下部の「適用」ボタンをクリックすると、下図のような詳細なオプション・スイッチが現れます。

(PGI CUDA Fortran 用のオプションの説明 URL)

http://www.softek.co.jp/SPG/Pgi/TIPS/opt_cudaF.html ([HTML リンク](#))

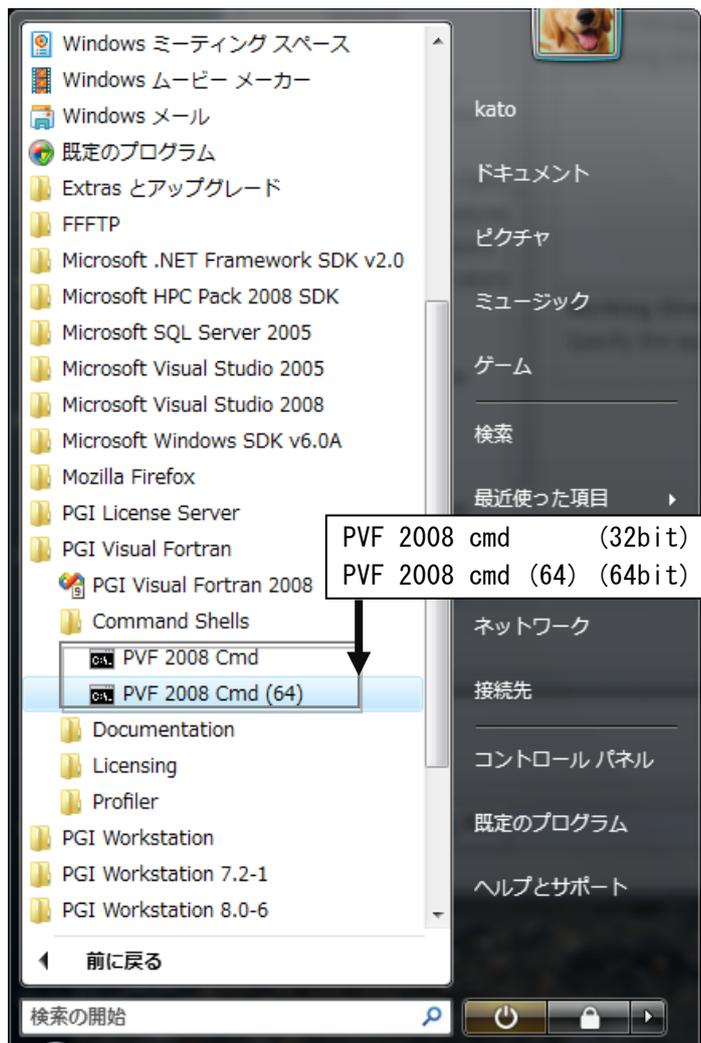


4 PVF コンパイラの起動 (コマンド・ライン)

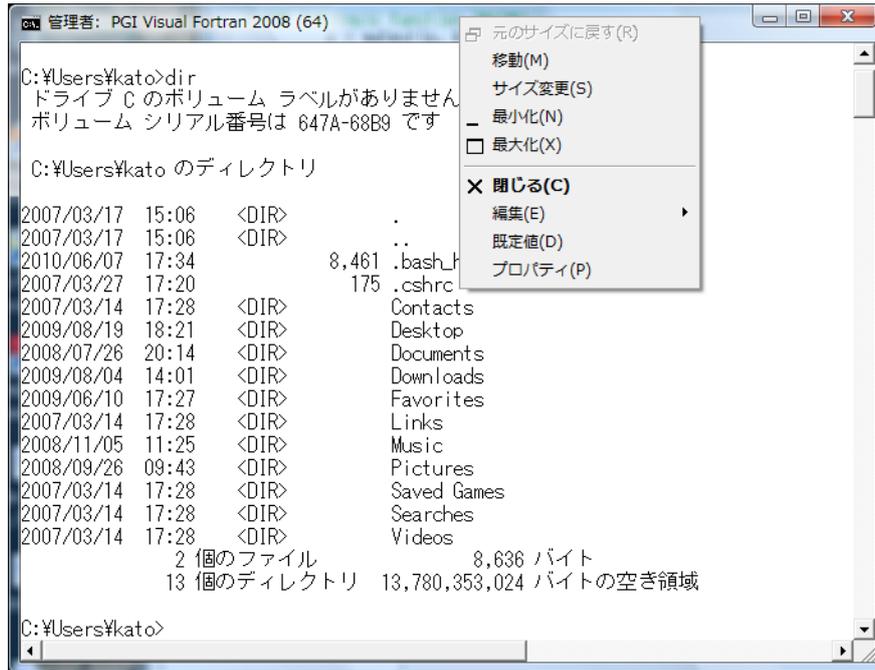
4.1 PVFコマンドプロンプトの起動

PVF Command Prompt(32bit) あるいは、PVF Command Prompt(64bit)のウィンドウを開き、コマンドベースでコンパイラを操作することができます。32ビット Windows 上では、PVF Command Prompt(32bit)のみ使用することができます。PVF コマンドプロンプトは、以下の方法で起動できます。

「スタート」メニューをクリック後、「すべてのプログラム」->「PGI Visual Fortran」->「Command shells」->「PVF 2008 cmd」(32ビット用、あるいは64ビット用)を選択すると、コマンドプロンプト画面(ウィンドウ)が現れます。

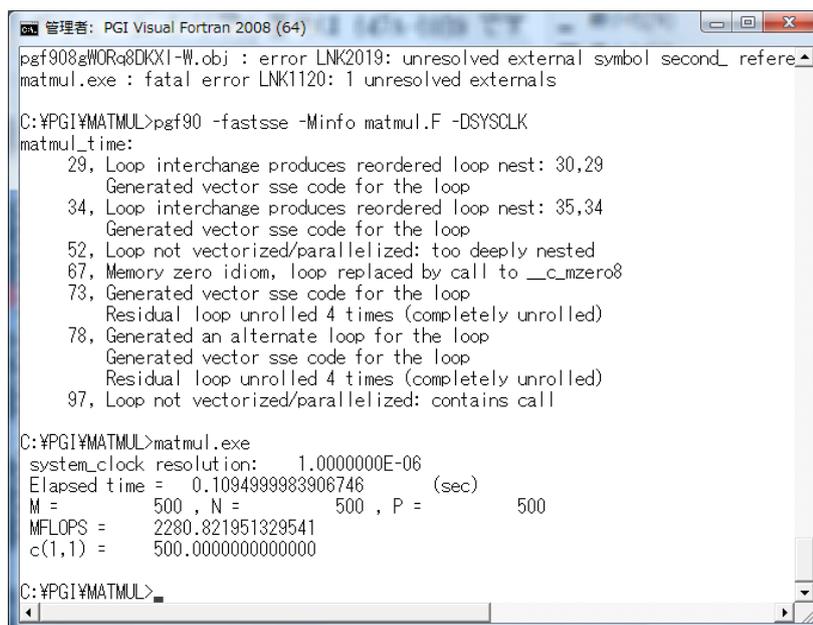


このウィンドウのサイズ等の「プロパティ」を変更するには、ウィンドウ上部(青地)にカーソルを置き、右クリックで下記のようなプルダウンメニューが現れますので、この中の「プロパティ」で、カスタマイズ・変更してください。



4.2 PVFコンパイラ・コマンドの使用

PVF コンパイラの操作は、このコマンドプロンプト画面内のコマンド・ライン上でテキストベースにより行います。コンパイラ・コマンドは、FORTRAN77 構文とその方言 (IBM/DEC) のみを対象にした pgf77、並びに FORTRAN77/Fortran90/Fortran95 の構文を全て解釈可能な pgfortran (pgf90、pgf95 も同じもの) コマンドがあります。コマンドの使用方法に関しては、「PGI Workstation & Server 製品」と同じであり、この詳細に関しましては、弊社ホームページ上のコンテンツ、あるいは、ダウンロードサイトで提供しております「PGI コンパイラ使用ガイド」(PDF ファイル) をご覧ください。なお、コマンドプロンプト画面内での Windows のコマンド体系は、DOS コマンドとなります。(「PGI Workstation & Server 製品」でインタフェースとして提供している Linux の bash 環境と等価なものは用意しておりません)



一般に、コマンドプロンプト内でのコマンド使用の例を以下に記します。

```

PGI Visual Fortran 9.0-1 (32-bit)
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\kato>cd C:\ (ディレクトリを C:\ トップへ)

C:\>cd PGI (PGI というフォルダへディレクトリ移動)

C:\PGI>dir (フォルダ内のファイルリストを表示)
ドライブ C のボリューム ラベルがありません。
ボリューム シリアル番号は 2863-1941 です

C:\PGI のディレクトリ

2007/05/28 13:25 <DIR> .
2007/05/28 13:25 <DIR> ..
2007/05/10 14:53 33 test.f
                1 個のファイル 33 バイト
                2 個のディレクトリ 45,879,316,480 バイトの空き領域

C:\PGI>pgf90 -fastsse -Minfo test.f (pgf90 コマンドを使用してコンパイル)

C:\PGI>dir (コンパイル後のフォルダ内のファイルリストを表示)
ドライブ C のボリューム ラベルがありません。
ボリューム シリアル番号は 2863-1941 です

C:\PGI のディレクトリ

2007/05/28 13:25 <DIR> .
2007/05/28 13:25 <DIR> ..
2007/05/28 13:25 24,576 test.dwf
2007/05/28 13:25 143,360 test.exe
2007/05/10 14:53 33 test.f
2007/05/23 10:40 1,458 test.obj
                4 個のファイル 169,427 バイト
                2 個のディレクトリ 45,879,316,480 バイトの空き領域

(注意) コンパイル&リンク後に生成されるファイルは、*.exe ファイルと言う名称の
実行モジュールだけでなく、*.obj(中間オブジェクトファイル)、*.dwf (シンボル情報
ファイル) が生成されます。なお、*.dwf ファイルは、コンパイラが一時的に使用する
ファイルですので、無視するかあるいは後で削除しても構いません。

C:\PGI>test.exe (プログラムの実行)
hello!

```

■ 自動並列、OpenMP並列実行時の並列スレッド数の環境変数の設定

コマンドプロンプト上で使用する場合、実行時に使用する様々な環境変数のセットの方法を説明します。Windows のコマンドプロンプト上での環境変数の設定は、「set」コマンドで行います。これは、一般的な Windows 上でのルールと同じですので、PGI コンパイラのランタイム時に指定する必要がある環境変数は、set コマンドでコマンドプロンプト画面を立ち上げる度に指定してください。これを事前にセットした後、プログラムを実行してください。

```
$ set OMP_NUM_THREADS=<並列 CPU コア数> (例 : set OMP_NUM_THREADS=2)
あるいは、
$ set NCPUS=<並列 CPU コア数> (例 : set NCPUS=2)
```

4.3 Windows®上で使用する際の留意点

Windows 上で PVF コンパイラをコマンドベースで使用する際の留意点は、以下の URL に補足説明をしております。基本的には Windows®のコマンド環境のルールをそのまま提供して結構です。

<http://www.softek.co.jp/SPG/Pgi/win64/win64use.html>

5 その他

5.1 実行モジュールの再配布

PVF コンパイラで生成された実行モジュールは、他の同種の Windows システムへ配布することができます。その際は、実行モジュールの他に、PGI 社が提供しているランタイム・ライブラリである DLL (ダイナミック・リンク・ライブラリ) ファイルも併せて配布していただく必要があります。この再配布可能な DLL ファイル群は、以下のディレクトリ配下にありますので、適時使用してください。基本的に、配布した実行モジュールと同じフォルダ内に必要な DLL が存在していれば、実行モジュールは動作します。

- 64 ビット Windows 上
 - C:\Program Files\PGI\win64\{リリース番号}\REDIST (64bit モジュール用)
 - C:\Program Files (x86)\PGI\win32\{リリース番号}\REDIST (32bit モジュール用)

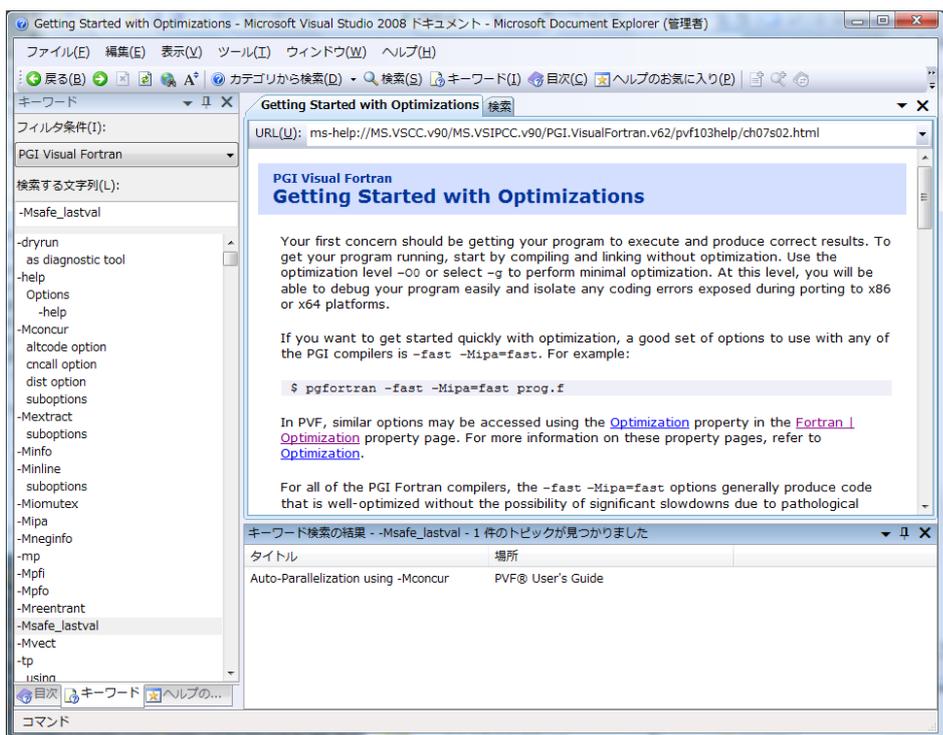
- 32 ビット Windows 上
 - C:\Program Files\PGI\win32\{リリース番号}\REDIST (32bit モジュール用)

同様に、Microsoft Open Tools の再配布可能 DLL ファイルは、以下に存在します。

- Microsoft Open Tools 用の DLL
 - C:\Program Files\PGI\Microsoft Open Tools 9(10)\redist

5.2 ヘルプ

PVF コンパイラのドキュメントは、Visual Studio 2005/2008/2010 の「PGI Documentation」->「PGI Help」を選択して現れる「Microsoft Document Explore」内で参照できます。フィルタ条件を「PGI Visual Fortran」にしてから検索して下さい。



以上