

# PGI (Accelerator) Visual Fortran<sup>®</sup> 2013

For Microsoft<sup>®</sup> Visual Studio

Windows<sup>®</sup>版 (Release 2013) - 入門ガイド -

2013年2月版 (Rev. 13.2-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/)



目	次
н	- 21

1	は	じめに	1
	1.1 1.2 1.3 1.4	本文書の概要. WINDOWS上での PVF ソフトウェアの実装. PVF コンパイラの利用方法. PVF コンパイラのコマンド・オプションについて.	1 1 1 2
2	PV	F コンパイラの起動 (Microsoft® Visual Studio 統合)	4
	<ol> <li>2.1</li> <li>2.2</li> <li>2.3</li> <li>2.4</li> <li>2.5</li> <li>2.6</li> <li>2.7</li> <li>2.8</li> <li>2.9</li> </ol>	VISUAL STUDIO 2005/2008/2010 の初回起動.         プロジェクトの作成方法.         PVFプロジェクト・テンプレート.         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.         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.         1         プログラムのコンパイルと実行(デバッグモード)         1         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)         1         プログラムの実行(入力データファイルのリダイレクト)         2         MPI プログラムのビルド	4     4     6     7     0     4     8     5     7
3	NV	IDIA GPU 用 PGI Accelerator と CUDA Fortran を使用する3	0
	3.1 3.2	THE PGI ACCELERATOR PROGRAMMING MODEL (OPENACC ディレクティブ使用)	0 3
4	PV	F コンパイラの起動 (コマンド・ライン) 3	6
	4.1 4.2 4.3	PVFコマンドプロンプトの起動       3         PVFコンパイラ・コマンドの使用       3         WINDOWS®上で使用する際の留意点       3	6 7 9
5	その	の他	0
	5.1 5.2	実行モジュールの再配布	0

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 © 2013 SofTek Systems, Inc. All rights reserved.

## 1 はじめに

#### 1.1 本文書の概要

本文書は、Microsoft<sup>®</sup> Visual Studio 2005/2008/2010 による統合開発環境(IDE)上 で使用する PGI<sup>®</sup> Visual Fortran 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<sup>®</sup>社のドキュメント等を参考にしてください。

#### 【ご注意】

Microsoft<sup>®</sup> 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 2010/2012 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<sup>®</sup> Visual Studio 2008/2010/2012 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 9¥PGI Visual Fortran (VS2008 用) C:¥Program Files¥Microsoft Visual Studio 10¥PGI Visual Fortran (VS2010 用) C:¥Program Files¥Microsoft Visual Studio 11¥PGI Visual Fortran (VS2012 用) 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 9¥PGI Visual Fortran C:¥Program Files (x86)¥Microsoft Visual Studio 10¥PGI Visual Fortran C:¥Program Files (x86)¥Microsoft Visual Studio 11¥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はなく、コマンド・ライン上でも 利用可能です。

1 11	
利用方法	内容
Visual Studio の統合開発環境	Visual Studio を起動して、IDE 上の GUI ベースで
<b>(IDE)</b> 上での使用	操作する。
	PVF Command Prompt(32bit) あるいは、PVF
コマンド・ライン上での使用	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。

PVF コンパイラの利用形態

#### 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)名とその使用方法は、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以下のページをご参照ください。

http://www.softek.co.jp/SPG/Pgi/comp-tips.html

例えば、以下の pgfortran 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。(一般的な 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が、PVF では、「-」を前に付けて指定します)

\$ 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)

PGI コンパイラのコンパイルオプションの詳細に関しては、以下の URL をご覧ください。

http://www.softek.co.jp/SPG/Pgi/comp-tips.html

PVFの Visual Studio (IDE)上での使用においても、同じ形態でコンパイラ・オプションがセットされております。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。

#### 【テクニック】

以下の画面は、コンパイラのオプション等を設定するプロパティの画面です。 Fortran あるいは Linker の中の個々のプロパティ・ウイザードでオプションを設定 するのが一般的ですが、<u>これが煩わしい場合</u>は、以下のように、「Command Line」 で、必要とする PGI のコマンド・オプションを直に指定することでも代替できます。 この場合は、その内容をプロパティ・ウイザードで設定する必要はありません。な お、Linker の「Command Line」にも同じオプションの指定が必要な場合もありま す。指定するコマンド・オプションは、PGI コンパイラの一般的なオプションの全 てが指定することが出来ます。もちろん、ウイザードで指定したオプションが重複 して指定したとしても問題はありません。



Visual Studio 上での PVF コンパイルオプションの任意設定

# 2 PVF コンパイラの起動(Microsoft<sup>®</sup> 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 2008/2010/2012 Standard/Professional Edition 統合環境上で使用する際には、「Visual Studio 2008/2010/2012」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 20xx」を選択して起動しても、同様な Visual Studio 2008/2010/2012 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 2008/2010/2012 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。

既定の環境設定の選択	<b>×</b>								
Visual Studio 2010 Professional									
アプリケーションを最初に使用する前に、Visual Basic または Visual C# など最も従事する開発作 業の種類を指定してください。この情報に基づいて、あらかじめ定義された設定のコレクションが ユーザーの開発作業に合わせた環境に適用されます。									
いつでも設定の別のコレクションを使用するように変更できます。[ツール] メニューの [設定のイ ンポートとエクスポート] を選択し、[すべての設定をリセット] を選択します。									
☑以前のパージョンから有効な設定を移行し、以下で選択した既定の設定と合わせて適用する(M)									
	24.00.								
以前の設定	a元9月:								
Visual Basic 開発設定	アプリケーションの以前のバージョンに近い開発環								
Visual C# 開発設定	境を提供します。複数のプログラミング言語で開発								
Visual C++ 開光設定 Visual C+ 明孫設定	する場合は、この設定コレクションを選択してくた								
Web 開発	CV1.								
Web 開発 (コードのみ)									
プロジェクト管理の設定									
全般的な開発設定									
Visual Studio の開始( <u>S)</u> Visual Studio の終了( <u>X</u> )									

#### 2.2 プロジェクトの作成方法

**PGI Visual Fortran**、あるいは Visual Studio 2008/2010/2012 を開始して、新しい「プロジェクト」を作成します。Visual Studio 2008/2010/2012 を起動後、「ファイル」->「プロジェクト」を選択します。

Visual Studio 2008/2010/2012 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 2008/2010/2012 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。

×	Microsoft Visual Studio				クイック起動 (Ctr	rl+Q) _ <b>□</b> ×
77	マイル(F) 編集(E) 表示(V) デ (ッグ(D)	チーム(M) SQL((	))	<u> ツール(エ) テスト(S) 分析(N)</u> ウィンド	Ⴢ(W) へルプ(ŀ	ł)
	新規作成(N)		18	プロジェクト(P)	Ctrl+Shift+N	
	閣<(0)			W-1-1/- (1) (1)	Shift+Alt+N	
	閉じる(C)		E	チーム プロジェクト(T)		
	ソリューションを閉じる(T)		ö	ファイル(F)	Ctrl+N	
10	選択されたファイルを上書き保存(S)	Ctrl+S		既存のコードからプロジェクトを作成(E)		
	選択したファイルに名前を付けて保存(A)		<b> </b> _			_
E.	すべてを保存(L)	Ctrl+Shift+S				
	テンプレートのエクスポート(E)					
	ソース管理(R)	F				
₽	ページ設定(U)					
-	印刷(P)	Ctrl+P				
	最近使ったプロジェクトとソリューション(J)	Þ				
×	終了(X)	Alt+F4				
	出力 出力元の表示(S):		•	<u> 등 등 등 등</u> (1)	- # ×	
	エラー一覧 出力					コード分析 ソリューション エ…

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)

fしいプロジェクト			? ×
▶ 最近使用したファイル .NE	T Framework 4.5 🔹 並べ替え基	準: 既定	• 🏢 🧮 インストール済み テン 🔎 •
▲ インストール済み	Console Application (64-bit)	PGI Visual Fortran	種類: PGI Visual Fortran
▲ テンプレート	Dvnamic Library (64-bit)	PGI Visual Fortran	A project for creating a PGI Visual Fortran console application
▷ Visual Basic			
Visual C++	Static Library (64-bit)	PGI Visual Fortran	
▷ Visual F#	Windows Application (64-bit)	PGI Visual Fortran	
SQL Server			
▲ PGI Visual Fortran	Empty Project (64-bit)	PGI Visual Fortran	
Win32	PVF用テンプ	レートの選択	ł.
▶ その他のノロンエクトの種類			
サンプル 32bit or 64bi	t		
▶ オンライン <b>のモード選択</b>	<b>ソリューショ</b>	ンとプロジ	ェクトの保存先フォルタ
名前( <u>N</u> ): Himeno2 (1			
場所( <u>L</u> ): D:¥PGI¥PVF¥		-	参照( <u>B</u> )
ソリューション名( <u>M</u> ): Himeno2 📿			✓ ソリューションのディレクトリを作成(D)
			□ ソース管理に追加( <u>U</u> )
			OK キャンセル

上図で①の部分は、Visual Studio 2008/2010/2012 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 2008/2010/2012 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」の構成で使用することが多いのですが、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 2008/2010/2012 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。

ウィンドウを使わないキャラクタベース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ータブル・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**  $(\dot{p} \prec \nu \ddot{r} \sigma )$ 

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 に移行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 2008/2010/2012 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 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。

新しいプロジェクト			··· ·		? <mark>×</mark>
▶ 最近使用したファイ)	L	.NET F	ramework 4.5 👻 並べ替え基準: 既知	ŧ.,	• 🎬 🧮 インストール済み テンプレー 👂 •
▲ インストール済み		C/A	Console Application (64-bit)	PGI Visual Fortran	種類: PGI Visual Fortran
▲ テンプレート ▷ Visual Basic		ġ_	Dynamic Library (64-bit)	PGI Visual Fortran	A project for creating a PGI Visual Fortran console application
▷ Visual C# ▷ Visual C++		Ē	Static Library (64-bit)	PGI Visual Fortran	<b>\</b>
▷ Visual F# SQL Server			Windows Application (64-bit)	PGI Visual Fortran	
<ul> <li>PGI Visual Fortr x64 Win32</li> <li>▶ その他のプロジェ サンプル</li> <li>▶ オンライン</li> </ul>	ran E クトの種類	2	Empty Project (64-bit)	PGI Visual Fortran	
名前( <u>N</u> ):	PVFProject1				
場所( <u>L</u> ): D:¥PGI¥tmp				参照(B)	
ソリューション名( <u>M</u> ):	PVFProject1				<ul> <li>✓ ソリューションのディレクトリを作成(D)</li> <li>□ ソース管理に追加(U)</li> </ul>
					OK キャンセル

#### ■ 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 と言うファイル名で 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。

<ul> <li>PVFProject1 - Microsoft Visual Studio</li> <li>ファイル(E) 編集(E) 表示(Y) プロジェクト(E) ビルド(E) デバッ・ ヘルプ(H)</li> </ul>	$\mathcal{T}(\underline{\mathbb{D}}) = \mathcal{F} - \mathcal{L}(\underline{\mathbb{M}})  \operatorname{SQL}(\underline{\mathbb{Q}})  \mathcal{Y} - \mathcal{H}(\underline{\mathbb{T}})$	クイック起動 (Ctrl+Q)  ク -  ロ  × テスト( <u>S</u> ) 分析( <u>N</u> ) ウィンドウ( <u>W</u> )
●・● 〒・■ ■ ■ P・C・ トアタッチ…・ Debug ・ ConsoleApp.f90 + × ConsoleApp.f90 + × ConsoleApp.f90 + × ConsoleApp.f90 Fortran Console Application Generated by PGI Visual Fortran(R) 2013/02/25 21:01:38 Program prog implicit none ! Variables ! Body end program prog	〕⊯ ৣৣৣৣৣৣৣৣৣৣৣৣৣঢ়ঀৣ৾৾ঀৠৠ D編集領域	
100 % ・ 《 出力 出力元の表示( <u>S</u> ): ・ エラーー覧 出力	(1) 등 1 등 1 등 1 등 1 등 1 등 1 등 1 등 1 등 1 등	□-ド分析 ソリューション エクスプローラー プロ/ディ ・ ↓ × ・ ■ 型↓

「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



© SofTek Systems, Inc

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。以下の図では、テンプレートとして、Free-Format Fortran source file (.f90) を選択します。

新しい項目の追加 - PVFProject1						
▲ インストール済み	並べ替え基準: 既定 🗸 🔡 🔚	インストール済み テンプレート の検乳 🔎 -				
PGI Visual Fortran	Free-Format Fortran source fits (.f90) PGI	Visual Fortran 種類: PGI Visual Fortran				
▶ オンライン	Fixed-Format Fortran source file (.f) PGI	A free-format Fortran source file Visual Fortran				
	Fortran include file (.fi) PGI	Visual Fortran				
	CUDA Fortran source file (.cuf) PGI	Visual Fortran				
指定する	3					
名前(N): SourceFile1.f	90					
		追加(A) キャンセル				

新しいソースファイルが作成されます。



#### 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 2008/2010/2012 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「FFTE2011」とし、プロジェクト名を「FFTE」と言う名前で定義します。



「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 P V F 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「Source Folder」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。

▲ 既存項目の追加 - FFTE									
	ディスク(D:) 🕨 PGI 🔸 ffte-5.0 🕨	<b>-</b> ↓	ffte-5.0の検索	Q					
🚺 ダウンロード 🔺	名前	更新日時	種類	サイズ					
📃 デスクトップ	📔 mpi	2013/02/25 20:50	ファイル フォル						
🗓 最近表示した場所	PVF	2013/02/26 11:15	ファイル フォル…						
	🐌 tests	2013/02/26 11:17	ファイル フォル…						
🍃 ライブラリ	F dzfft2d.f	2011/11/03 12:00	Fortran Source	4 KB					
□ ドキュメント	F dzfft3d.f	2011/11/03 12:00	Fortran Source	5 KB					
■ ピクチャ ■	F fft235.f	2011/11/03 12:00	Fortran Source	7 KB					
■ ドデオ	F kernel.f	2011/11/03 12:00	Fortran Source	10 KB					
N ST - STWD	F mfft235.f	2011/11/03 12:00	Fortran Source	8 KB					
	F vzfft1d.f	2011/11/03 12:00	Fortran Source	4 KB					
	F vzfft2d.f	2011/11/03 12:00	Fortran Source	2 KB					
🜏 ホームグループ 🌱	F vzfft3d.f	2011/11/03 12:00	Fortran Source	3 KB					
	F zdfft2d.f	2011/11/03 12:00	Fortran Source	4 KB					
👰 コンピューター	F zdfft3d.f	2011/11/03 12:00	Fortran Source	5 KB					
🚢 ローカル ディス	F zfft1d.f	2011/11/03 12:00	Fortran Source	4 KB					
- ローカル ディス	F zfft2d.f	2011/11/03 12:00	Fortran Source	3 KB					
	F zfft3d.f	2011/11/03 12:00	Fortran Source	4 KB					
ファイ	ファイル名(N): "zfft3d.f" "fft235.f" "kernel.f"								
			追加( <u>A)</u>	キャンセル					

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、ファイルの種別 を確認して Visual Studio 2008/2010/2012 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。さらに、include ファイルがある場合は、同じ操 作で、当該ファイルを登録します。(下図は、その様子を表しています)



正確に言うと、「既存」のソースファイルは、Visual Studio 2008/2010/2012 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 2008/2010/2012 の PVF プロジェクトに登録すると言う形 を取ります。したがって、Visual Studio 2008/2010/2012 (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。しかし、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 2008/2010/2012 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。以下の例のように新しいフォルダ (Newfolder1)が新設されますので、そのフォルダ名を設定します。

×	FFTE2011	Microsoft Visual S	Studio					クイック起動	(Ctrl+Q)	p -	. 🗆	x
ファイ	ファイル(E) 編集(E) 表示(Y) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) SQL(Q) ツール(I) テスト(S) 分析(N)											
ウイン	ッドウ( <u>₩</u> )	ヘルプ(圧)										
6	- 🖸 🛅	- 🖆 🗎 🥙 🤊	- C - 🕨 アタ	'ツ <del>ブ</del> … ▼ D	ebug 👻 🎜	_} ⊨ @ *	p %   ∎	위기적	Ŧ			
¢ z	fft3d.f	kernel.f	fft235.f ↔ 🗙			param.ł	1 1 × -	צ-בעע	ション エクス	プローラー ※	oo <del>▼</del> ₽	×
L – 7	0	FFTE: A FAST FO	DURIER TRANSFOR	I PACKAGE				006	5, •0 <sup>™</sup>	a > 🔉	]	
:57	č	(C) COPYRIGHT S	SOFTWARE, 2000-	2004, 2008-	-2011, ALL RI	GHTS RESERVE	D		ション エクス	フローラーの	検家 (C	<i>۹</i> - ۹
Ŕ	Č	DAISUKE TAP	(AHASHI						ユーション 'F	FTE2011'(1)	プロジェ	クト)
I.	C C	FACULTY OF	ENGINEERING, II OF TSUKUBA	VFORMATION	AND SYSTEMS				FTE 3 Include File			
Ĭ	č	1-1-1 TENNO	DAI, TSUKUBA,	BARAKI 30	5-8573, JAPAN	l			n param.	h		
4	č	C-MAIL: Ual	SUKE@CS.(SUKUD)	a.ac.jp				Resource Files				
-14	C	RADIX-2, 3, 4,	5 AND 8 FFT RO	JTINE				<ul> <li>Source Files</li> </ul>				
T. V.	Č,	EODTDANZZ SOUD						NewFolder1				
λζ	č	FORTRANTT SOURC	JE PROGRAM						F fft235.f			
	C C	WRITTEN BY DAIS	SUKE TAKAHASHI						F Kernel.r			
		SUBROUTINE FFT2 IMPLICIT REAL*8 COMPLEX*16 A(*)	235(A,B,W,N,IP) 3 (A-H,O-Z) 2 B(*) W(*)						P 2000			
1	.00 % - 4		,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,					コード分ね	所 ソリューシ	/ヨン エクスス	プローラ-	- 11
B	出力						• 4 ×	プロパティ	۲ ·····		∞ <del>-</del> - <b>7</b>	×
1	出力元の表示	₹ <u>(S</u> ): ピルド			• <u>6</u>	<u>≤</u> ≥ ≚	ab⊒	IPVFPrjF	ProjHierFolde	erProps		-
								un set	ş			
								Folder	Name	NewFolder1		
								Unique	Identifier			•
							-	FolderNa	ame			
4	T=87	<b>#5</b>					•	Folder Na	ame			
-	- 」 見							_		_	_	_
この項	目はプレビュ	ューをサポートしてい	いません			21		55 列	55 文	<del>?</del>	挿	۵. ۸

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 系の場合は、Makefile でこうした依存性を定義 しましたが、Visual Studio 2008/2010/2012 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 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 2008/2010/2012 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。

FFTE2011 - Microsoft Visual Studio	イック起動 (Ctrl+Q) 👂 – 🗖 🗙
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) SQL(Q) ソ	ソール(T) テスト(S) 分析(N)
ウィンドウ(W) ヘルプ(H)	
<ul> <li>O - O</li> <li>10 - C - </li> <li>10 - Debug - </li> <li>10 - C - </li> <li>10 - Debug - </li> <li>10 - C - </li> <li>10 - C - </li> <li>10 - Debug - </li> <li>10 - C - </li> <li>10 - C - </li> <li>10 - Debug - </li> <li>10 - C - </li> <li>10 - Debug - <!--</td--><td>×a ÷</td></li></ul>	×a ÷
transferred frequency of the speed of	ソリューション エクスプローラー 👓 🔹 🖡 🗙
C FFTE: A FAST FOURIER TRANSFORM PA 構成マネージャー	G C 🟠 'O - Z 🗇 🕨 🟳
C (C) COPYRIGHT SOFTWARE, 2000-2004, 2008-2011, ALL RIGHTS RESERVED	ソリューション エクスプローラー の検索 (C 👂 🗸
C BY C DAISUKE TAKAHASHI C FACULTY OF ENGINEERING, INFORMATION AND SYSTEMS UNIVERSITY OF TSUKUBA C UNIVERSITY OF TSUKUBA, IBARAKI 305-8573, JAPAN C E-MAIL: daisuke@cs.tsukuba.ac.jp	<ul> <li>□ ソリューション 'FFTE2011' (1プロジェクト)</li> <li>▲ FFTE</li> <li>▲ Include Files</li> <li>□ param.h</li> </ul>

#### ■ ソリューションのクリーン

Visual Studio 2008/2010/2012 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。

FFTE2011 - Microsoft Visual Studio		クイック起動 (Ctrl+Q) - C ×
ファイル(F) 編集(E) 表示(V) プロジェクト(P)	ビルド(B) デバッグ(D) チーム(M) SQL(Q)	ツール(T) テスト(S) 分析(N)
- S - S 12 - S - C - ▶ 開始		.tri+Snitt+B
triangle     triangle     triangle       tria	ソリューションのクリーン(C)     ソリューションでコード分析を実行(Y) A     FFTE のビルド(U)     FFTE のリビルド(E)     FFTE のクリーン(N)     プロジェクトのみ(J)     ずべての T4 テンプレートの変換(4)	Jt+F11 ン エクスプローラー ・ ↓ × 'o ・ さ 回 上 日 ン エクスプローラー の検索 (C P・ -ション 'FFTE2011'(1 プロジェクト) E nclude Files p param.h
C ZFFT3D SPEED TEST PROGRAM C FORTRAN77 SOURCE PROGRAM C WRITTEN BY DAISUKE TAKAHASHI IMPLICIT REAL+86 (A-H,O-Z) PARAMETER (NDA=16777216) COMPLEX+16 A(NDA)	バッチ ビルド(T) 構成マネージャー(0)	esource Files source Files tests F speed3d.f F fft235.f F kernel.f F zfft3d.f
100% • 1 出力	• ₽ >	<ul> <li>オリバー クリン エンバウロ ク</li> <li>オロバティ ・ ∓ ×</li> </ul>
出力元の表示(S): ビルド ビルド:0 正常終了、0 失敗、1 更新不 ( エラーー覧 出力	<ul> <li>√  ≦  ≦  ≦  ど  ね</li> <li>要、0 スキップ ······</li> </ul>	IPVFPrjProjHierFolderProps       ✓         IP       IP         FolderName       tests         UniqueIdentifier       ✓         FolderName       FolderName         Folder Name       Folder Name
ビルト正常終了		

#### ■ 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。

FFTE2011 - Microsoft Visual Studio	クイック起動	(Ctrl+0) ×
 ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B)	デバッグ(D) チーム(M) SQL(Q) ツール(T)	テスト(S) 分析(N)
ウィンドウ(W) ヘルプ(H)	ウィンドウ(W)	+
G - O 間 - 🏭 💾 🥍 🏷 - 🤆 - 🕨 開始 - Debug	グラフィックス	b -
⊈ zfft3d f kernel f fft235 f	▶ デバッグ開始(S)	F5
	▶ デバッグなしで開始(H)	Ctrl+F5
C FFTE: A FAST FOURIER TRANSFORM PACKAGE	'図 パフォーマンス分析の開始(A)	Alt+F2
C (C) COPYRIGHT SOFTWARE, 2000-2004, 2008- C BY	쪪 一時停止したパフォーマンス分析の開始(Y)	Ctrl+Alt+F2
C DAISUKE TAKAHASHI	e <sup>®</sup> プロセスにアタッチ(P)	
UNIVERSITY OF TSUKUBA	例外(X)	Ctrl+Alt+E
C E-MAIL: daisuke@cs.tsukuba.ac.jp	ら・ステップ イン(I)	F11
	ステップオーバー(0)	F10
C ZFFT3D SPEED TEST PROGRAM	ブレークポイントの設定/解除(G)	F9
C FORTRAN77 SOURCE PROGRAM		Ctrl , Chift , E0
C WRITTEN BY DAISUKE TAKAHASHI		
IMPLICIT REAL*8 (A-H,O-Z)	9へてのテータビントをクリア(A)	
COMPLEX*16 A (NDA)	データビントのインボート(ス)	
100 % 🔹 4		クスプローラー
出力		<b>▼</b> ‡ ×
出力元の表示(S): ビルド		
FFTE build succeeded.		ρ
Build log was saved at " <u>file://D:\PGI\fte=5.0\tests\PVF\F</u>	FTE2011¥FFTE¥x64¥Debus¥BuildLos.htm*	
=====================================		
	•	
エラー一覧出力		
準備完了	15 行 11 列	11 文字 挿入 🔬
C:¥WINDOWS¥system32¥CMD.EXE		- 🗆 ×
KatoTest		▲ I
NX_NY_NZ =		
100 100 100		
NY = 100 NY =	100 NZ = 100	TIME =
0.2241065526404147 403	7 9773927766156 MELO	
ENDEDANI STUD	.0//000//00/00 MILO	
FORMAN STOL 約4年末スには何わた。 お押レアノゼ・	*	
酸白するには凹がキーを押してくため	261	

#### ■ 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。

FFTE2011 - Microsoft Visual Studio	クイック起動 (Ctrl+Q) - C ×
ファイル(E) 編集(E) 表示(⊻) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) SQL(Q)	2) ツール(I) テスト( <u>S</u> ) 分析(ℕ)
ウィンドウ( $\underline{W}$ ) ヘルプ( $\underline{H}$ )	
💿 - O 🔞 - 🏭 🔐 ジ - ぐ - 🕨 開始 - Debug 🕞 🔎 🔒 🌾 ヤヤ 🖡 🐄	1 1 1 2
⊈ speed3d.f → × zfft3d.f kernel.f fft235.f	<ul> <li>ソリューション エクスプローラー ***** • # ×</li> </ul>
C WRITTEN BY DAISUKE TAKAHASHI	± ○ ○ ☆ 'o - ≠ ⓓ ≯ 📮
H C IMPLICIT REAL*8 (A-H,O-Z)	ソリューション エクスプローラー の検索 (C 👂・
PARAMETER (NDA=16777216) COMPLEX*16 A(NDA)	▶ ソリューション 'FFTE2011' (1 プロジェクト)
DIMENSION LNX(3),LNY(3),LNZ(3)	
T C WRITE(6.*) ' NX.NY.NZ ='	D param.h
READ(5,*) NX,NY,NZ	Resource Files
CALL FACTOR (NY, LNY)	Source Files
	F speed3d.f
CALL ZFFT3D (A,NX,NY,NZ,0)	₣ fft235.f
LOOP=1	F kernel.f
C !\$ 10 CONTINUE	·
	コード分析 ソリューション エクスプローラー
出力 🗸 🕂 🗙	x プロパティ <b>・ * *</b> x
出力元の表示( <u>S</u> ): ビルド 🔹 🖕 🛬	•
	-
エラー一覧 出力	
準備完了 31 行	1列 1文字 挿入

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ますと、「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。また、変数値の表示は、 ソースプログラム上の「変数」にカーソルを乗せると、その値が画面に表示されま す。



#### 2.7 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

最も簡単に行う方法は、ビルド・モードを「Debug」から「Release」に変更し てビルドすることです。(下図)



#### ■ 新しい最適化ビルド・モードの作成

以下の操作は、一般的にほとんど行うことはないですが、新しく個人用のビルド・ モードを新設する際に以下のような方法をとります。「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「<新規作成>」に変更します。

<b>以マネージャー</b>				? ×
アクティブ ソリューション構成( <u>C</u> ):	アクティブ ソ	リューシ	ョン プラットフ	フォーム( <u>P</u> ):
Release	▼ x64			-
Debug Release	クト構成をチェ	:ック)( <u>R</u> ):		
<新規作成>	プラットフォ	-4	ビルド	配置
<編集>	x64	-		
<新規作成>を選択				
				閉じる

新しいソリューション構成のウイザードが現れますので、この中に新しい構成の名 前を定義します。以下の例では、GPU コンパイル用の構成のために「Release-GPU」 としたものです。基本設定のコピー元はデフォルトの「Release」としています。

新しいソリューション構成 ?
名前( <u>N</u> ):
Release-GPU
設定のコピー元( <u>S</u> ):
Release 🗸
▼新しいプロジェクト構成を作成する(C)
OK キャンセル

次に、「プロジェクト」 -> 「…のプロパティ」を選択し実行します。



FFTE プロパティ ページ			? ×
構成( <u>C</u> ) アクティブ(Release-GP	IJ) → プラットフォーム( <u>P</u> ): (	アクティブ(x64) ・	構成マネージャー( <u>0</u> )…
▲ 構成プロパティ ▲	Display Startup Banner	No	•
構成を Release-	GPU ( Include Directorie		
HAN E Refease	ath	\$(IntDir)¥	
▲ Fortran	Object File Name	\$(IntDir)¥	
General	Debug Information Format	Disabled	
Optimization	Optimization	Maximize Speed (-fastsse)	
Preprocessing			
Code Generation			
Language			
Floating Point Opt			
External Procedur			
Libraries			
Target Processors	Display Startup Pappor		
Dispection	Display the compiler's startup b	apper (-)/)	
Diagnostics	bispidy the compiler's startup bi	anner (-v).	
Profiling			
Command Line			
⊳ Linker			
> Resources			
Build Events			
Custom Build Cton			
		ОК <b>‡</b> т>	·セル 適用( <u>A</u> )

プロジェクトの構成プロパティの画面となっていますので、その中の「Fortran」 と「Linker」のオプション項目の中で必要なものを指定します。ちなみに、上記で 定義した「Release-GPU」の基本設定のコピー元は、「Release」であるため、これ はデフォルトで「最適化オプションが設定済み」のモードのため、最適化オプショ ンはすでに適用済みと考えてください。その他の追加のオプションを任意で指定し ます。以下の図は、「Optimization」のオプション欄の例を示しています。

FFTE プロパティ ページ	CONTRACTOR OF STREET	# 16 (18 (18 (18))	? ×
構成( <u>C</u> ): アクティブ(Release-GPU)	→ プラットフォーム(P):	アクティブ(x64)	▼ 構成マネージャー( <u>0</u> )…
▲ 構成プロパティ 🔺	Optimization	Maximize Speed (-fastsse)	•
General	Global Optimizations	Default	
Debugging	Vectorization	Default	
▲ Fortran	Inlining	No	
General	Use Frame Pointer	No	
Optimization	Loop Unroll Count	Default	
Preprocessing	Auto-Parallelization	No	
Code Generation		Azvimum Snood	(-factore)
Language			(-lastsse)
Floating Point Opt	· · · · · · · · · · · · · · · · · · ·	こなつているはり	
External Procedur =			
Libraries			
Target Processors	Ontimization		
Target Accelerato	Colort option for overall code o	ntimization	
Diagnostics	select option for overall code o	purnización.	
Profiling			
Command Line			
Resources			
Build Events			
Custom Duild Cton			
		ОК	キャンセル 適用( <u>A</u> )

追加するオプションとして、GPU 対応の OpenACC オプション (Enable OpenACC Directives) を Yes とします。

Himeno2 プロパ	ペティ ページ	-	-		? ×	
構成( <u>C</u> ): Rel	lease	•	プラットフォーム( <u>P</u> ):	アクティブ(x64) ▼	構成マネージャー( <u>0</u> )	
▲ 構成プロ/	パティ 🔺	Fortr	an Dialect	Fortran 95		
Gener	ral	Treat	Backslash as Characte	Yes		
Debug	gging	Exter	nd Line Length	No		
▲ Fortra	an	Enab	le OpenMP Directives	Yes		
Ge	eneral	Enab	le OpenACC Directives	No	-	
Op	otimization	MPI		No	III	
Pre	eprocessing	Enab	le CUDA Fortran	NO		
6	de Generation			Yes (-acc)		
La						
FIO	bating Point Opt			Yes (-acc=strict)		
EXI	ternal Procedur			Voc ( acc-vop/strict	+)	
LID Tay	raat Processors			res (-acc=verystric	-)	
Tai	rget Accelerato	Enable	OpenACC Directives	I		
Dia	agnostics	When no	n-OpenACC accelerato	r directives are found: ignore (-acc).	. wa	
Pro	ofiling			5 ( ).		
Ru	intime					
Co	mmand Line					
▷ Linker	r 👘					
Resou	urces					
⊳ Build £	Events					
< Custor	m Duild Cton					
				ОК <b>+</b> т>	セル 適用(A)	

コンパイルオプション、リンクオプションを上記のような方法で設定する以外に、 さらに、「Fortran」->「Command Line」にて、その他の最適化オプションを直に指 定することも可能です。同様に、「Linker」-->「Command Line」にも指定が可能で す。

こうして、いくつかの追加オプションを定義することにより「Release-GPU」構成のビルド・モードが定義されます。

#### ■ 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

ソリューションエクスプローラ内の「プロジェクト名」をドラックします。これ は、このプロジェクトの「プロパティ」について変更するための初期操作となりま す。「プロジェクト」-「プロパティ」を選び、プロパティ画面を出します。

Himeno2 - Microsoft Visual Studio	クイック起動 (Ctrl+Q) 👂 🗕 🗖 🗙
ファイル(E) 編集(E) 表示(⊻) プロジェクト( <u>P</u> ) ビルド( <u>B</u> ) デバッグ( <u>D</u> ) チーム( <u>M</u> ) SQL( <u>Q</u> ) ツール	レ(I) テスト(S) 分析(N) ウィンドウ(W)
	c   % <sub>=</sub> े ⊑ ा   २ २   ∎ अ अ अ <sub>=</sub>
t himenoBMTxp.f90 → ×	ソリューション エクスプローラー 👓 👻 🕂 🗙
÷ ! ↑ integer :: nn H integer :: ic.icr.icm	
Y real(4)::flop,xmflops2,score,gosa プロジェクト名	
Itarget specifys the measuring period in sec     real(4), parameter :: ttarget=60.0	↓ Style
	Resource Files
call readparam	<ul> <li>Source Files</li> </ul>
I Initializing matrixes	F himenoBMTxp.f90
call initmt	
print *, "max=',imax, "jmax, "mjmax, "mkmax, "mkmax, "mkmax", "mkmax	
: call system_clock(ic.icr.icm) dt= 1.0/real(icr.8) print '(2x.a.el0.5)','Time measurement accuracy : ',dt !! Start measuring	
!	
<pre>print *,' Start rehearsal measurement process.' print *,' Measure the performance in 3 times.'</pre>	コード分析 ソリューション エクスプローラー
100 % - 4	プロパティ • 🕈 🗙
出力	Himeno2 IPVFPrjProjHierProps +
出力元の表示(S): ビルド 🚽 🖆 🖆	
	□ その他
	(Name) Himeno2
	ProiectPath D:¥PGI¥PVF¥Himeno
· · · · · · · · · · · · · · · · · · ·	(Name)
	Name of the project
1, 2, 2, 14/J	
この項目はブレビューをサポートしていません	

「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  $\mu$ ープ構造に対して、コンパイラが並列化を施します。

Himeno2 プロパティ ページ				
構成( <u>C</u> ): アクティブ(Release)	√ プラットフォーム(P):	アクティブ(x64) ・	構成マネージャー( <u>0</u> )…	
▲ 構成プロパティ ▲	Optimization	Maximize Speed (-fastsse)		
General	Global Optimizations	Default		
Debugging	Vectorization	Default		
⊿ Fortran	Inlining	No		
General	Use Frame Pointer	No		
Optimization	Loop Unroll Count	Default		
Preprocessing	Auto-Parallelization	Yes	•	
Code Generation				
Language				
Floating Point Opt				
External Procedur =				
Libraries				
Target Processors				
Target Accelerato	Auto-Parallelization			
Diagnostics	Enable auto-parallelization (-Mo	concur).		
Profiling				
Runtime				
Command Line				
▷ Linker				
▷ Resources	-Mconcur を記讨	*する		
Build Events	C 10/2			
Custom Duild Cton				
		OK =1	ァンセル 適用( <u>A</u> )	

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」-> 「Enable 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。

Himeno2 プロパティ ページ	the second		?
構成( <u>C</u> ): アクティブ(Release)	プラットフォーム(P):	アクティブ(x64) ・	構成マネージャー( <u>0</u> )
構成(C): アクティブ(Release) 構成プロパティ General Debugging Fortran General Optimization Preprocessing Code Generation Language Floating Point Opt External Procedur Libraries Target Processors Target Accelerato Diagnostics Profiling Runtime Command Line Linker Resources	<ul> <li>ブラットフォーム(P):</li> <li>Fortran Dialect Treat Backslash as Charact Extend Line Length</li> <li>Enable OpenMP Directives MPI Enable CUDA Fortran</li> <li>Enable OpenMP Directives Enable OpenMP Directives</li> </ul>	アクティブ(x64) Fortran 95 e Yes No Yes No None No Xtensions (-mp).	構成マネージャー( <u>0</u> )
Build Events     THE Puild Step			
		OK +7	ンセル 適用(A)

#### ■ 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

「Debug」あるいは「Release」のビルド・モードにおける「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。

**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 2008/2010/2012 を起動するようにしてください。

http://www.softek.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

#### 2.8 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 カ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 カファイルは、ここに置くだけでもよいです。

matmul プロパティ ページ	STATE AND A DESCRIPTION OF		2 ×
構成( <u>C</u> ): Release	→ プラットフォーム(P):	アクティブ(x64)	▼ 構成マネージャ( <u>0</u> )…
通応プロパティ General Debugging Fortran General Optimization Preprocessing Code Generation Language Floating Point Optio External Procedures Target Processors Target Accelerators Diagnostics Profiling Command Line Linker General	Application Command Application Arguments Environment Merge Environment MPI Debugging Working Directory " < C:¥tm を行う。絶 Application Arguments The command line arguments	< c:¥tmp¥fort.10 Yes Disabled \$(SolutionDir) の¥input_data" な対パス名で指定	と言う風な記述 しても良い。
Input Command Line Resources			
		ОК	キャンセル 適用( <u>A</u> )

#### ■ 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

くじみり。			
matmul プロパティ ページ	A CONTRACT OF A DATE OF A DATE	A R. D. R. B. BRELL, M. M.	? ×
構成(C): アクティブ(Debug)	・ プラットフォーム(P):	アクティブ(x64) ・	構成マネージャ(0)
構成プロパティ General Debugging Forban Linker Resources Build Events Custom Build Step	Application Command Application Arguments Environment Merge Environment Merge Environment Working Directory Working Directory Specify the application's worki	NCPUS=2 Yes Created C:Ymp	
		ОК <b>‡</b> т	ンセル 適用(A)

#### 2.9 MPI プログラムのビルド

#### ■ MPI プログラムのプロジェクトの作成

MPIプログラムのプロジェクトを作成します。

MPIHELLO - Microsoft Visual Studio	クイック起動 (Ctrl+Q)	ρ = 🗆 ×
ファイル(E) 編集(E) 表示(⊻) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) SQL(Q) ツー	ル(I) テスト( <u>S</u> ) 分析( <u>N</u>	) ウィンドウ( <u>W</u> )
ヘルプ(圧)		
○ - ○ 范 - 留 単 ヴ - ペ - ▶ 開始 - Debug - 月 <sub>→</sub> Ⅱ = ○ fo → G. G. C	<b>%</b> <sub>↓</sub> h fi \ \ \ \	■ 위 위 위 <sub>부</sub>
mpihello.f ≄ ×	<ul> <li>ソリューション エクスプロ</li> </ul>	□-∋- ∞∞∞ <b>-</b> ₽ ×
include 'mpif.h' H integer ierr, myid		
Call mpi_init(ierr)		
write(8,100) myid		HELLO (I DUSTOR
VI call mp_finalize(ierr)	📔 Include Files	
	Resource File	s
	F mpihello.f	
<b>ポ</b> シン		
κζ.		
	4	•
	コード分析 ソリューショ	ンエクスプローラー
	プロパティ	<b>▼</b> ₽ ×
エノ * + × 中力モの事子/C\ 」 ジーと と」 ※ 1 約5	<b>0=</b> . <b>0</b> . <b>0</b>	
	<u></u>	
エラー一覧 出力		
<b>準備完了 5</b> 行	18 列 18 文字	挿入 🦼

#### ■ プロジェクトのプロパティの設定

MPI プログラムのプロジェクトのプロパティに、MPI プログラムのコンパイル& リンクのための設定を行います。「プロジェクト」-「プロパティ」を開き、「Fortran」



- 「Language」を表示します。「MPI」の選択時に、「Microsoft MPI」を選択します。 このオプションの設定により、MPI プログラムのコンパイルとリンクが可能となり ます。具体的には、Microsoft MPI (MS-MPI)のインクルードファイルを取り込み、 MS-MPI ライブラリをリンクする設定を行います。

#### ■ MPI プログラムの実行

次に、並列実行のための設定を行います。プロジェクトのプロパティ画面の [General] - [Debugging] に、MPI 実行の環境に関して設定を行うパラメータがあり ます。以下の画面イメージを見て下さい。「MPI Debugging」という項目のプルダウ ンメニューで「Local」を選択して下さい。これは、このマシン上1台の中で MPI プ ロセスを起動して実行すると言う意味です。他に、「Cluster」というパラメータが ありますが、これは使用できません。

MPIHELLO プロパティ ページ	COLUMN TWO IS NOT	and, tant	? ×
構成(C): アクティブ(Debug)	▼ プラットフォーム(P):	アクティブ(x64) ▼	構成マネージャー(0)
▲構成プロパティ General Debugging ▲ Fortran General Optimization Preprocessing	Application Command Application Arguments Environment Merge Environment MPI Debugging Working Directory	Yes Local Disabled	
Code Generation Language Floating Point Opt External Procedur Libraries Target Processors		Local Cluster	
Target Accelerato Diagnostics Profiling Runtime Command Line	MPI Debugging Enable MPI debugging, either lo	ocally or on a cluster.	
Build Events     Guttem Build Sten		0K ±+7	ンセル 適用(A)

その後、「適用」ボタンを押すと、さらに詳しいパラメータ項目が追加表示されま す。「Number of Processes」と言う欄は、並列プロセス数を指定する項目です。こ こに、並列実行に必要なプロセス数を指定します。なお、システム性能を考慮する と、システムのプロセッサが有する最大コア数が、この値に指定できる最大数です。 これらの設定は、Visual Studio の「Release」モードで行う場合は同様な設定を行 う必要があります。

この設定後に、Visual Studio 上で実行を行えば、自動的に mpiexec 実行が行わ れます。

Windows 環境上での MS-MPI の使用法の詳細に関しては、以下の URL をご覧く ださい。

http://www.softek.co.jp/SPG/Pgi/TIPS/public/general/msmpi\_use.html

MPIHELLO プロパティ ページ	
構成(C): アクティブ(Release)	<ul> <li>         ・ プラットフォーム(P): アクティブ(x64)         ・         ・         ・</li></ul>
▲ 構成プロパティ General Debugging Fortran General Optimization Preprocessing Code Generation Language Floating Point Opt External Procedur Libraries Target Processors Target Accelerato Diagnostics Profiling Runtime Command Line Linker Resources Build Events Contane Build Chan	Application Command Application Arguments Environment Merge Environment Yes MPI Debuaging Loca Number of Processes 4 Working Directory \$(SolutionDir) Additional Arguments: mpie Location of mpiexec 4 並列で実行するとした場合の 設定 Number of Processes Specify the number of processes with which to run (mpiexec -n).
	OK         キャンセル         適用(A)

## 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 5.0 をサポ ートした PVF 13.2 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 (OpenACC 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。

#### OpenACC を Enable にする設定

「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。「Language」を選び、「Enable OpenACC Directives」を「Yes」にします(P21 参照)。これによって、ソースプログラム上の OpenACC のディレクティブを認識し、基本的にこの設定だけでも GPU 用のコード生成を行います。さらに、詳細な NVIDIA GPU のようなターゲット・アクセラレータの特性を指定したい場合は、以下の項目の設定も行うことが出来ます。

#### ■ 「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。

Himeno2 プロパティ ページ	AND DESCRIPTION OF THE OWNER.	? ×
構成( <u>C</u> ): アクティブ(Release)	<ul> <li></li></ul>	▼ 構成マネージャー( <u>0</u> )
<ul> <li>構成プロパティ General Debugging</li> <li>Fortran General Optimization Preprocessing Code Generation Language Floating Point Opt External Procedur Libraries</li> </ul>	Target NVIDIA Accelerator         Yes           Target Host         No	×
Target Processors Target Accelerato Diagnostics Profiling Runtime Command Line Linker Resources Build Events Custom Build Ston	arget NVIDIA Accelerator Select NVIDIA accelerator target (-ta=nvidia).	
	ОК	Fャンセル 適用( <u>A</u> )

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、<u>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</u>。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

Himeno2 プロパティ ページ	? ×
構成( <u>C</u> ): アクティブ(Release) → プラットフォーム( <u>P</u> ): アクティブ(x64) →	構成マネージャー( <u>0</u> )
<ul> <li>構成プロパティ General Debugging General Optimization Preprocessing Code Generation Language Floating Point Opt External Procedur Libraries Target Processors Target Accelerato Diagnostics Profiling Runtime Command Line Linker Resources Build Events Time Command Line Linker Resources Build Events Time Command Line Linker Resources Build Events Command Line Linker Resources Build Events Command Line Linker Resources Build Events Command Line Linker Command Line Linker Command Line Comman</li></ul>	
OK =1	·ンセル 適用( <u>A</u> )

#### ■ 「NVIDIA CUDA Toolkit」プロパティの設定

PVF にバンドルされている NVIDIA CUDA Toolkit のバージョンが表示されます。 複数の CUDA Toolkit のバージョンが現れた場合は、それを選択することができます。 なお、Toolkit とは別に、システムに実装されている NVIDIA のデバイス・ドライバ ーのバージョンを知りたい場合は、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 4.2 driver: CUDA Driver Version 4020 と表示 For a 5.0 driver: CUDA Driver Version 4000 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」は、各 PVF バージョンによって異なります。

#### ■ 「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, 2.0, 2.1, ... 3.5, ...)と言う番号が付けられています。例えば、2012 年末にリリースされた Kepler のハードウェアは、3.5 です。コンパイラはデフォルトで、自動的に 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

Himeno2 プロパティ ページ	second in a second in the		? ×
構成( <u>C</u> ): アクティブ(Release)	→ プラットフォーム(P):	アクティブ(x64) ・	構成マネージャー( <u>0</u> )…
<ul> <li>構成プロパティ         General         Debugging         Fortran         General         Optimization         Preprocessing         Code Generation         Language         Floating Point Opt         External Procedur         Libraries         Target Processors         Target Accelerato         Diagnostics         Profiling         Runtime         Command Line         Linker         &gt; Resources         Build Events         Time to the first second to the</li></ul>	NVIDIA: CUDA Toolkit NVIDIA: Compute Capability NVIDIA: CC 1.0 NVIDIA: CC 1.1 NVIDIA: CC 1.2 NVIDIA: CC 1.3 NVIDIA: CC 1.3 NVIDIA: CC 2.0 NVIDIA: CC 2.0 NVIDIA: Fermi NVIDIA: CC 3.0 NVIDIA: CC 3.5 NVIDIA: CC 3.5 NVIDIA: Kepler <b>NVIDIA: Compute Capability</b> Direct the compiler to automatic	Default Manual No No No No No No No Cally generate code compatible with a	
		OK キャンセ	ル 適用( <u>A</u> )

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力するには、 「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力すること ができます。

■ PGI Accelerator (OpenACC) 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。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、それを有効に設定するものです。

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。

PGI アクセラレータ用実行時の環境変数については、以下の URL にて説明をしてお りますので、ご参照ください。

GPU-IN-Module-OK プロパティ ページ				
構成(C): アクティブ(Debug)	<ul> <li>         ・ プラットフォーム(P): アクティブ(x64)         ・         ・         ・</li></ul>			
構成プロパティ General Debugging Fortran General Optimization Preprocessing Code Generation Language Floating Point Option External Procedures Target Processors Target Accelerators Diagnostics Profiling Command Line	Application Command         Application Arguments         Environment       ACC_NOTIFY=1         Merge Environment       Yes         MPI Debugging       Disabled         Working Directory       \$(SolutionDir)         Environment       Specify the environment for the application under debug, or variables t			
Resources Build Events Custom Build Step	OK キャンセル 適用(A)			

http://www.softek.co.jp/SPG/Pgi/TIPS/opt\_accel.html#ENV

#### 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 と言うファイル拡張 子を付けることにより認識されますが、適切なリンケージ処理を行うために、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。

GPU-IN-Module-OK - Microsoft Visual Studio	
ファイル(E) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) ツール(I) ウィンドウ(W)	へルプ( <u>H</u> )
🛐 • 🖼 • 💕 🛃 🗿 🐰 🗞 🛝 🖉 🕫 - 🖓 • 🖓 • 🖓 🖡 bebug 🔹 x64	🖄 🚆
[[國 點 點 ☆   建 準   国 월   □ 뒤 다 쥐 다 원 용 및 .	
test_cublasSgemm.F90 3device_val_in_module.cuf スタートページ マ 🗙	ソリューション エクス 👻 🖡 🗙
module cudamod use cudafor	
Contains	GPU-IN-Module-OK
<pre></pre>	📄 🛅 Include Files
implicit none	Resource Files
U integer, value :: n	Source Files
it = threadideWy	
$\frac{1}{2} \qquad \qquad$	
end subroutine test1	
program cuda_device	
use cudator use cudamod	
implicit none integer :: n=64	
шл <b>-</b> џ <b>&gt;</b>	
出力元の表示(S): ビルド ▼ ③ 4	
GPU-IN-Module-OK build succeeded.	
Build log was saved at "file://C:\PGI\GPGPU\TESTI\GPU-IN-Module-OK\GPU-IN-MO	
=====================================	
<u>د</u> ۱۱۱ ب	
3 エラー一覧 国 出力	۰ III ا
リビルドがすべて正常に終了しました。 11行 1列	1 文字 挿入

#### ■ 「CUDA Fortran」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。

Himeno2 🗆	プロパティ ページ			? ×
構成( <u>C</u> ):	アクティブ(Release)	√ プラットフォーム(P):	アクティブ(×64) ▼	構成マネージャー( <u>0</u> )…
▲ 構成:	プロパティ 🔺	Fortran Dialect	Fortran 95	
G	General	Treat Backslash as Characte	Yes	
D	ebugging	Extend Line Length	No	
⊿ F	ortran	Enable OpenMP Directives	Yes	
	General	Enable OpenACC Directives	No	
	Optimization	MPI	None	
	Preprocessing	Enable CUDA Fortran	Yes	•
	Code Generation			
	Language			
	Floating Point Opt			
	External Procedur =			
	Libraries			
	Target Processors			
	Target Accelerato	Enable CUDA Fortran	term (Mauda)	
	Diagnostics	Enable compliation of CUDA For	tran (-Mcuda).	
	Profiling			
	Runtime			
	Command Line			
	Inker			
	esources			
	Sustana Duild Chan			
•	4 III			
			ОК <b>+</b> †?	ンセル 適用( <u>A</u> )

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

Himeno2 プロパティ ページ		? <mark>×</mark>
構成( <u>C</u> ): アクティブ(Release)	プラットフォーム( <u>P</u> ): アクティブ(x64)	▼ 構成マネージャー(Q)…
<ul> <li>構成プロパティ General Debugging Fortran General Optimization Preprocessing Code Generation Language Floating Point Opt External Procedur Libraries Target Processors Target Accelerato Diagnostics Profiling Runtime Command Line Linker Resources Build Events Custom Puild Store III </li> </ul>	Enable CUDA Fortran       Yes         CUDA Fortran Register Limi          CUDA Fortran Use Fused ML Yes          CUDA Fortran Use Fast Mat No          CUDA Fortran Flush to Zero       Default         CUDA Fortran Toolkit       Default         CUDA Fortran Toolkit       Default         CUDA Fortran Compute Cap Automatic          CUDA Fortran Keep Binary       No         CUDA Fortran Keep Kernel S       No         CUDA Fortran Reep PTX       No         CUDA Fortran Generate RDI Default          Enable CUDA Fortran       Enable CUDA Fortran         Enable compilation of CUDA Fortran (-Mcuda).	
	ОК	キャンセル 適用( <u>A</u> )

## 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 for VS 20xx 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や文字色、背景色等の「プロパティ」を変更するには、ウ ィンドウ上部(以下の例では、青地部分)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。なお、プロパティを変更してその特性を保存したい場合 は、予め、「管理者権限」で「PVF for VS 20xx Cmd」を起動しなければなりません。

「PVF for VS 20xx Cmd」部分にカーソルを置いて右クリックすると、「管理者として実行」がありますので、これを選んでコマンドプロンプトを起動してください。

画 管理者: PGI Visual Fortran 2008 (64)	日 元のサイズに戻す(R)
C:¥Users¥kato>dir 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 647A-68B9 です	移動(M) サイズ変更(S) _ 最小化(N) □ 最大化(X)
C:¥Users¥kato のティレクトリ 2007/03/17 15:06 〈DIR〉 . 2007/03/17 15:06 〈DIR〉 . 2010/06/07 17:34 8,461 .bash」 2007/03/27 17:20 175 .cshrc	× 閉じる(C) 編集(E) ト 既定値(D) プロパティ(P)
2007/03/14         17:28         CDIR>         Contact           2009/08/19         18:21         CDIR>         Desktop           2008/07/26         20:14         CDIR>         Documen           2009/08/04         14:01         CDIR>         Downloa           2009/06/10         17:27         CDIR>         Favorit	s ts ds es
2007/03/14 17:28 (DIR) Links 2008/11/05 11:25 (DIR) Music 2008/09/26 09:43 (DIR) Picture 2007/03/14 17:28 (DIR) Saved G 2007/03/14 17:28 (DIR) Searche 2007/03/14 17:28 (DIR) Videos	s ames s
2007/03/14 17:20 くロレ Yideos 2 個のファイル 13 個のディレクトリ 13,780,3 C:¥Users¥kato>	8,636 バイト 53,024 バイトの空き領域 ▼

#### 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/Fortran2003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、PVF ソフトウェアでは用意しておりません。別途、<u>PGI</u> Workstation ソフトウェアをインストールしてください)



```
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。
Microsoft の DOS コマンドを使用します。
```

```
PGI Visual Fortran (64)
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:¥Users¥kato>cd C:¥ (ディレクトリを C:¥ トップへ)
C:¥>cd PGI (PGI と言うフォルダヘディレクトリ移動)
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>
              <DIR>
                         •
2007/05/28 13:25
                          . .
                    24,576 test.dwf
2007/05/28 13:25
2007/05/28 13:25
                   143,360 test.exe
2007/05/10 14:53
                     33 test.f
               1,458 test.obj
2007/05/23 10:40
          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
C:\PGI>test.exe (プログラムの実行)
hello!
```

(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。

■ 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のある環境変数は、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 物理コア数}

(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。実行形式ファイルは、デフォルトでは静的リンク形式 なため、その実行形式のファイルのみのファイルで済みます。しかし、DLL 形式の 実行モジュールを作成した場合は、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なけれ ばなりません。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¥redist (VS 2008 用) C:¥Program Files¥PGI¥ Microsoft Open Tools 10¥redist(VS 2010 用) C:¥Program Files¥PGI¥ Microsoft Open Tools 11¥redist(VS 2012 用)

#### 5.2 PVF ドキュメント

PVF コンパイラのドキュメントは、Visual Studio 2008/2010/2012 の「ヘルプ」内で、マニュアルとして PDF ファイルを参照できます。

<ul> <li>ヘルプス表(Y)</li> <li>Ctri+EL</li> <li>ヘルプス支(Y)</li> <li>Ctri+AL+FL</li> <li>ヘルプコンテンツの進起と期後(C)</li> <li>Ctri+AL+FL</li> <li>イレブなどの設定</li> <li>Microart Josef S (L)</li> <li>PGI Visal Fortran Linguage Reference</li> <li>サンプル(S)</li> <li>PGI Visal Fortran Reference</li> <li>TO(F7(R)</li> <li>PG</li></ul>	<b>X</b> 77	Himeno2 - Microsoft Visual Studio イル(F) 編集(E) 表示(V) プロジェクト(P)	ビルド(B)	デバッグ(D)	チーム(M)	SQL(Q)	א ע–ע	フイック起動 (C J <b>(T) テスト(</b> !	rrl+Q) 5) 分析(N)	ם - סקעדים (	• ×
<ul> <li>■ へいプロシテンジの想起と期頃(C) Ctri+Att+F1</li> <li>&gt; VJコーション エクスプローラー Im・・ キン</li> <li>&gt; MSDN フィーラム(M)</li> <li>&gt; MSDN フィーラム(M)</li> <li>&gt; MSDN フィーラム(M)</li> <li>&gt; MSDN フィーラム(M)</li> <li>&gt; PGI Visual Fortran User's Cude</li> <li>&gt; PGI Visual Fortran User's Cude<th>~</th><th>レブ(H) ヘルプの表示(V)</th><th>Ctrl+F1</th><th><b>8</b> . II</th><th>= 0 fo</th><th>⇒ <b>6.</b> (</th><th>a d</th><th><b>%</b> . 15 16</th><th>561</th><th>1 1 1 1</th><th>_</th></li></ul>	~	レブ(H) ヘルプの表示(V)	Ctrl+F1	<b>8</b> . II	= 0 fo	⇒ <b>6.</b> (	a d	<b>%</b> . 15 16	561	1 1 1 1	_
<ul> <li>○ MSDN フォーラム(M)</li> <li>○ PCI Visual Fortran Reference</li> <li>○ Source Files</li> <li< td=""><td>888</td><td>ヘルプ コンテンツの追加と削除(C) ヘルプ設定の設定</td><td>Ctrl+Alt+F</td><td>1</td><td></td><td></td><td></td><td>ソリューショ</td><td>ンエクスプロ</td><td></td><td>• 4 ×</td></li<></ul>	888	ヘルプ コンテンツの追加と削除(C) ヘルプ設定の設定	Ctrl+Alt+F	1				ソリューショ	ンエクスプロ		• 4 ×
<ul> <li>● PGL Vaual Fortran User's Guide</li> <li>● PGL Vaual Fortran National Reference</li> <li>● PGL Vaual Fortran National Reference</li> <li>● PGL Vaual Expanding Reference</li> <li>● Polo PGL</li></ul>	<i>♀</i> ⊡	MSDN フォーラム(M) パグの報告(B)					4	00 <b>ה</b> פּלּ-בעע	`○ - ₽ □ ン エクスプロ	1 <b>ト 回</b> コーラー の検索	(C ,P
<ul> <li>PGI Fortran Reference</li> <li>PGI Fortran Language Reference</li> <li>PGI Fortran Language Reference</li> <li>PGI Fortran Language Reference</li> <li>DXAFマーフィード(ックのオブション(F)</li> <li>BBSの空気(F)</li> <li>PSJ Fortran Language Reference</li> <li>*.nk tax</li>     &lt;</ul>	0	PGI Visual Fortran User's Guide							・ション 'Him	eno2' (1 プロ:	ジェクト
サンプル(5) カスタマーフィード(ックのオジョン(5) 品の意識(7) トラブルシューディング(7) ジークシュンドン(5) ジークシュンドン(5) シート分類、ソリューション エクスプローラー コート分類、ソリューション エクスプローラー	0 0	PGI Visual Fortran Reference PGI Fortran Language Reference						∎ II ■ R	clude Files esource Files	5	
カスタマーフィード/9 少のオブション(P) 製品の登録(P) トラブルシューダィング(R) ⑦ テクニカル サポート(T) DVD 長小パブのは文 オンラインのプライ/15-1に属する声明(0) Microsoft Yaual Studio (D)(-ジョン)/開設(A) ロート分類   ソリューション エクスプローラー プロパティ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		サンプル(S)						⊿ <b>≝</b> S	ource Files	Typ 600	
P クロコル サポート(T) DVD 原ヘレブの注文 オンライックライバクーに関する原明(0) Microsoft Visual Studie 00(1->3> /規制(A) サイロ・マーを受けていたいですのでないで、マン・マーマー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マ		カスタマー フィードバックのオプション(F) 製品の登録(P) トラブルシューティング(R)		',nknax x					TIMETODA	170130	
は、 オンラインのブライバシーに関する声明(0) Microsoft Vasual Studie のバージョン (開まる) ロード分類 (ソリューション エクスプローラー) プロパライ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	0	テクニカル サポート(T) DVD 晒ヘルプの注文		: '.dt							
Microsoft Visual Studie Oxf->3> 注意     コード分析 (ワーンジェングスプローラー)       フレージョン ジャンプローション ジャンプローラー)     フレージョン ジャンプローラー)       ブロ/マイ     ・ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		オンラインのプライバシーに関する声明(0)									
100% - 1 出力 出力 エラー目出力	_	Microsoft Visual Studio のパージョン情報(A)	III O UTINES				-	コード分析	ソリューショ	ンエクスプロ・	-=-
出力 (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)		100 % - 4					+	プロパティ		•••••••••••••••••••••••••••••••••••••••	• ¤ ×
(二) (1) (2) (2) (2) (2) (2) (2) (2) (2) (2) (2						• 1 •••	Ψ×				
< エラー一覧 出力		出力元の表示(S): ヒルト		• 1 12	5 5 <b>4</b>	540					
( エラー「夏 出力											
(エテーー関 出力											
エノー R 山の		4 									
		エノ- R 山川									