

# PGI<sup>®</sup> Visual Fortranコンパイラ Release 7.1

- 入門ガイド -

2007年11月版 (Rev. 7.1-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

1	はじめに	1
0	<ul> <li>1.1 本文書の概要</li> <li>1.2 WINDOWS上でのPVFソフトウェアの実装</li> <li>1.3 PVFコンパイラの利用方法</li> <li>1.4 PVFコンパイラのコマンド・オプションについて</li> </ul>	1
2	PVFユンバインの起動(Microsoft® Visual Studio 2005 統合)	3
	2.1 VISUAL STUDIO 2005 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 プログラムのコンパイルと実行(デバッグモード)	12
	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)	16
3	PVFコンパイラの起動 (コマンド・ライン)	20
	3.1 PVFコマンドプロンプトの起動	20
	3.2 PVFコンパイラ・コマンドの使用	21
	3.3 WINDOWS®上で使用する際の留意点	23
4	その他	24
	4.1 実行モジュールの再配布	24
	1 2  AVT	24
		<u>ы</u> т

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 © 2007, 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 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 2005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。

## 1.2 Windows 上での PVF ソフトウェアの実装

Microsoft<sup>®</sup> Visual Studio 2005 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 2005 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 8¥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 8¥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。

利用方法	内容
Visual Studio 2005 の統合開	Visual Studio 2005を起動して、IDE上の GUI べ
発環境(IDE)上での使用	ースで操作する。
	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 PGF95 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)

```
$ pgf95 <u>-fastsse</u> <u>-Minfo</u> 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 2005 (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。



Visual Studio 2005 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 2005 統合)

# 2.1 Visual Studio 2005 の初回起動

PVF コンパイラを Visual Studio 2005 統合環境上で使用する際には、「Visual Studio 2005」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」を選択 して起動しても、同様な Visual Studio 2005 の画面が現れます。

Visual Studio 2005 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	X
Visual Studio 2005	
Visual Studio を最初に使用する前に、Visual Basi 定してください。この情報に基づいて、あらかじめ定義さ に適用されます。	c または Visual C# など最も従事する開発作業の種類を指 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
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 ト] を表示して、[すべての設定をリセット] を選択しま	できます。[ツール] メニューから [設定のインボートとエクスボー す。
既定の環境設定を選択してください( <u>C</u> ):	
Visual Basic 開発設定 全般的な開発設定	説明: 設定のコレクションの1つを一覧から選択してください。

# 2.2 プロジェクトの作成方法

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

Visual Studio 2005 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 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)

新しいプロジェクト			? 🗙
プロジェクトの種類(P): ● Visual Basic → Windows スタート キット ● PGI Visual Fortra ● Win32 ● その把のフロシェク	32bit or 64bit のモード選択 an	テンプレート①: Visual Studio にインストールされたテンプレート Console Application (64-bit) Static Library (64-bit) Empty Project (64-bit) マイ テンプレート PVF用テンプレートの選択	
A project for creating	a PGI Visual Fortran o	onsole application ハリューションとプロジェクトの保存失フォルダの指定	,
プロジェクト名(N):	PVFProject1	ノウエー ノヨンビンロンエンドの保住ルンオルンの相反	
場所( <u>L</u> ):	C:¥Documents and Se	ttings¥kato¥My Documents¥Visual Studio 2005¥Projects¥MPICH2 🛛 🛛 💽 参照	(( <u>B</u> )
ソリューション名( <u>M</u> ):	PVFProject1	✓ ソリューションのディレクトリを作成(型)	
		OK	セル

上図で①の部分は、Visual Studio 2005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 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 2005 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 (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 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。

新しいプロジェクト				? 🔀
プロジェクトの種類(P): ・ Visual Basic ・ Windows クタートキット ・ Visual C++ ・ PGI Visual Fortra Win32 ・ その他のプロジェク	n ∽の種業類	テンプレート①: Visual Studio (こインストールされた: ■ Console Application (64-bit) ■ Static Library (64-bit) ■ Empty Project (64-bit) マイ テンプレート ■オンライン テンプレートの検索	テンプレート フ Dynamic Library (64-bit) Dwindows Application (64-bit)	
A project for creating	a PGI Visual Fortran c	onsole application		
プロジェクト名(N):	PVFProject1			
場所(L): C#Documents and Settings¥kato¥My Documents¥Visual Studio 2005¥Projects¥MPICH2 🛛 💽 📑				
ソリューション名(M):	PVFProject1		ソリューションのディレクトリを作成(D)	
			ОК	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。

🏶 PVFProject3 – 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① ウィン	ドウ(W) コミュニティ( <u>C</u> )	ヘルプ(円)	
i 🛐 - 🔛 - 💕 🛃 🥔 👗 🗈 🛍 🖤 - 🔍 - 💷 - 🖳 🕨	Debug 🗸 W	in32	TIME	
Console App. f90 スタート ページ		<b>-</b> ×	ソリューション エクスプロ	ome – Vlamese 🖢 🚽 🗴
ConsoleApp.f80 プログラ. Fortran Console Application Generated by PGI Visual Fortran 05/25/2007 11:52:40 Program main implicit none	ムの編集命	頁域 ▲		VFProject3'(1 プロジェクト) ct3 Files Files Files nsoleApp.190
と 教 - Variables		<u></u>		
l Body			<₽> - - - > - > - -</td <td><b>パブローラ</b></td>	<b>パブローラ</b>
end program main			プロパティ	<b>↓</b> ↓ ×
		~	ConsoleApp.f90 I	PVFPrjProjHierItemProps 🔹
			8≣ 2↓ 🗈	
出力		<b>↓</b> ₽ ×	□ その他	<u>^</u>
出力元の表示(2):			(Name) FileName FilePath FileType IsFixedFormat IsIncludeFile	ConsoleApp.f90 ConsoleApp.f90 C*Documents and Settin Fortran Source File False
3.15覧 □出力			(Name) Name of the file	
プロジェクト 'PVFProject3' の作成に成功しました。				

「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。

🌮 PVFProject3 – Microsoft Visual Studio	
ファイル(E) 編集(E) 表示(M) プロジェクト(P) ビルド(B) デバッグ(D) ツール(T) ウィンドウ(M) コミュニティ(	<ol> <li>ヘルプ(H)</li> </ol>
🛅 • 🔠 • 💕 🛃 🥥 🐰 🛅 新しい項目の追加(W) Ctrl+Shift+A 🚽 🔹 Win32	
👔 ConsoleApp.f90 スタート 😐 既存項目の追加(@) Shift+Alt+A 🗸 🗸 🗸	ソリューション エクスプローラ - PVFProject3 👻 🕂 🗙
(す Consoleáns f30 📸 新しいフォルダ(D)	
プロジェクトのアンロード(1)	レリューション 'PVFProject3' (1 プロジェクト)
H : Fortran Console ! Generated by PGI スタートアッププロジェクトに設定( <u>A</u> )	- Include Files
G 05/25/2007 11:52 回 プロパティ(P)	Resource 5000
	GonsoleApp.t9U
program main	
같 implicit none ~ ~	
경 ! Variables	
! Body	🔩 シリューション エクスプローラ 🚳 クラス ビュー
end program main	ਹੋਰ//ਰਿਮ 👻 🕂 🗙
8	PVFProject3 IPVFPrjProjHierProps
出力 🗸 🗘 🤟	く 日 その他 (Name ) D) (E During to D
出力元の表示(S):	ProjectPath C#Documents and Settings#
	Name of the project
25-一覧 🖪 出力	=
אלדב	

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。

新しい項目の追加	🛛 – PVFProject3		?×				
カテゴリ( <u>C</u> ):		テンプレート(①):					
PGI Visual Fo	ortran	Visual Studio にインストールされたテンプレート					
		Fixed-Format Fortran source file (f) Inter-Format Fortran source file (f90) Inter-Format Fortra					
マイ テンプレート							
□□ オンライン テンプレートの検索							
A free-format Fortran source file							
ファイル名(N):	SourceFile1.f90						
	L						
		<u>追加(A)</u> キャン	rtn				

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

-	eve	Proje	et3 - k	lierosof	t Visual Stud	lin							ſ	
_	- 4.11	/s) :	(日本)(F)	ホテムの	-*::-:	Sil K(P)	≓15m8(D)	21-11 (T)	ウムようのの		-(0)	<ul> <li></li></ul>		
2	21 70	(E) 7	編集に	表示しい	JUVIUNU	ENTRE	771900	9-110	9721-90	181_7	10/	AUDA		
1	]-	•		<b>3</b> 8		C - (1)	- 🖳 🕨	Debug	<ul> <li>▼ Win32</li> </ul>			- MIL		• •
	Ś	ource	File1.f90	Conso	leApp.f90 スタ	ートページ				-	х	ソリューション エクスプ	ローラーンリューション、	. 🗸 🖣 🗙
4		s	ourceFi	le1.f90							^			_
2			-									😡 עבע-געע 🔽	VFProject3' (1 プロジ:	፤/2Ի)
5		I F	ree-For	mat For d by PC	tran Source H I Misual Fort	File						😑 📑 PVFProje	ect3	
EX.		1 0	5/25/20	07 14:2	9:13	L ran						- include	re Files	
-		1										🖃 🗁 Source	Files	
V												- 🔤 Ca	nsoleApp.190	
3 B												F90 501	urceFile1.f90	
1-														
ませい														
۲Ľ														
												ערד לפער בויע 🔊	<b>470−7</b>   <b>2</b> ≰95∧ C3	
												プロパティ		<b>→</b> ╄ ×
											~	SourceFile1.f90	(PVFPrjProjHierItemP	rops 🔻
	<									>		8≣ 2↓ 🖻		_
	出力	)								🗕 🗘	х	□ その他		^
	出ナ	り元のず	表示( <u>S</u> ):			-			a)			(Name)	SourceFile1.f90	
												FileName	C#Documents an	d Settin
												FileType	Fortran Source Fi	le
												IsFixedFormat	False	
												IsIncludeFile	False	×
												(Name)		
			-									Name of the file		
	<b>1</b>	[5	"覧 📃 出	カ										
37	シド													

# 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」と言う名前で定義します。

プロジェクトの種類(P):       テンプレート(T):       (***)         ・ Visual Basic       ・ Visual Studio (こインストールされたテンプレート         ・ Windows       スタートキット       (***)         ・ Visual C++       ・ Console Application (32-bit)       ・ Dynamic Library (32-bit)         ・ Visual C++       ・ Console Application (32-bit)       ・ Dynamic Library (32-bit)         ・ Visual C++       ・ Console Application (32-bit)       ・ Dynamic Library (32-bit)         ・ Visual C++       ・ Console Application (32-bit)       ・ Dynamic Library (32-bit)         ・ Windows Application (32-bit)       ・ Empty Project (32-bit)       ・ Dynamic Library (32-bit)         ・ Windows Application (32-bit)       ・ Empty Project (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ C+ Fortan       ・ Dynamic Library (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ Empty Project (32-bit)       ・ Dynamic Library (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ Empty Project (32-bit)       ・ Dynamic Library (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ ・       ・ Dynamic Library (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ ・       ・ ・       ・ Dynamic Library (32-bit)         ・ ・       ・ ・       ・ Dynamic Library (32-bit)       ・ Dynamic Library (32-bit)         ・ ・       ・ ・ <th>新しいプロジェクト</th> <th></th> <th>? 🔀</th>	新しいプロジェクト		? 🔀
An empty PGI Visual Fortran project プロジェクト名(い) FFTE 場所①: C×Documents and Settings¥kato¥My Documents¥Visual Studio 2005¥Projects ◆照④	サレベブロシェクト フロジェクトの種類(2): Usual Basic → Visual Basic → 2,3 - トキット ■ PGI Visual C++ ■ PGI Visual Fortran → Win32 ■ その他のプロジェクトの種類	テンプレート①: Visual Studio にインストールされたテンプレート Console Application (32-bit) Empty Project (32-bit) マイ テンプレート ③オンライン テンプレートの検索	
ソリューション名(M): FFTE40	An empty PGI Visual Fortran pro プロジェクト名(い): FFTE 場所(い): C×Docume ソリューション名(M): FFTE40	ect nts and Settings¥kato¥My Documents¥Visual Studio 2005¥Projects マレンリューションのディレクトリを作成の	参照(B)

「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。

![](_page_10_Picture_4.jpeg)

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	W - FFTE		?×
探す場所①:	🛅 ffte-4.0		
र्ग २०७३ २१ २०७३ २१ २०७३७ २१ २०७३-७	tests fft235.f kernel.f paramh readme.txt sse2.c sse3.c zfft3d.f		
	ファイル名(N):	🕑 🚺 追加	I( <u>A</u> )
	ファイルの種類(工):	すべてのファイル (*.*)	ren 🛛

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 2005 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)

🏶 FFTE40 – Microsoft Visual Studio	
ファイル(E) 編集(E) 表示(M) プロジェクト(E) ビルド(E) デバッグ(D) ツール(E) ウィンドウ(M) コミュニティ(C)	
Xタート ベージ     Xタート ベージ     Xタート ベージ     Visual Studio 2005     最近使ったプロジェクト     MSDN Online 更新情報     IMicrosoft Visual Studio Team System 満足)     FFTE40     PVFFroject2     Visual Studio Team System 満足)     Tue, 10 Apr 2007 00000 GMT - ディベロッパー製	ソリューション IDスプローラ - FFTE ● ● > >
は、次期点と対視本の製品をおり良いものにしていなかのに PVFProject3 は、次期点と対視本の製品をおり良いものにしていなかのに PVFProject3 は、かり、 PVFProject1 間が、 「た成: プロジェクト(P) 作成: プロジェクト(P)	
出力	日 その他 (Name) FFTE
	ProjectPath C¥Documents and Setting
3 エラー一覧 3 出力	<b>その他</b>

正確に言えば、既存のソースファイルは、Visual Studio 2005 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 の PVF プロジェクトに登録すると言うことになります。したが って、Visual Studio 2005 (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 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 イルを移行すること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が新設されますので、そのフォルダ名を設定します。

🏕 FFTE40 – 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(T) ウィンドウ(W) コミュニティ(C	) ヘルプ(円)
🛅 🕶 🚟 🖌 💕 🛃 👗 🖺 隆 🔊 🗉 🔍 - 🖓 - 💷 🖳 🕨 Debug 💽 🔹 Win32	🛛 🥐 TIME 🔹 😜
🙀 /スタート ๙ーシ) 🔹 🗸	ソリューション エクスプローラー ソリューション 🖢 🚽 🗙
Microsoft Visual Studio 2005 参近使ったフロジェクト アVFProject2 PVFProject3 Phineno PVFProject1 N: アVFProject1 N: アVFProject1 N: アンドアProject2 PVFProject3 PVFProjec4 PVFProjec5 PVFPVFProjec5 PVFPVFPVFP	
出力 - 中 ×	日 その他
出力元の表示(2):	FolderName NewFolder1 UnigueIdentifier
<ul> <li>(9) 丁与報目 四 地力</li> </ul>	₹01₽
אלאב	

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。

🗱 FFTE40 – 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(T) ウィンドウ(W) コミュニティ(C	シ ヘルプ(田)
🛅 🕶 📷 🕶 🍘 📓 👗 🖦 🛍 🥙 - 🔍 - 🚚 - 🖳 🕨 Debug 🛛 - Win32	- 🦄 TIME - 🙄
👔 🕹 🕹 🗸 🕹 🗸 🕹	Уリューション エクスプローラー ソリューション 🖕 🗸 🗙
Microsoft         Microsoft	WJユーション 'FFTE40' (1 プロジェクト)     FFTE         ・
1755: 70910F(P)	IPVFPrjProjHierFolderProps
	8≣ 2↓ □
出力 <b>▼</b> <sup>1</sup> ×	日 <u>その他</u> FolderName tests
出力元の表示(S):	UniqueIdentifier
	7-0 <b>0</b>
15覧 ■出力	
אלאב	

#### ■ 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 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。

![](_page_13_Figure_4.jpeg)

# 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 2005 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。

<b>%</b> I	FTE40	- Micro	soft Visı	ual Studio										
77	イル(E)	編集( <u>E</u> )	表示(⊻)	プロジェクト( <u>P</u> )	ビルド( <u>B</u> )	デバッグ	( <u>D</u> ) ツー	-ル( <u>T</u> ) י	ウィンドウ₩)	_≷12i	ティ( <u>C</u> )	ヘルプ(出)		
16	• 🔠	• 📔 🔛	🗿   🐰 i	<b>b</b> 🖪 🤊 -	(° - 🗐 -	<b>B</b> ,	Debug	ţ	▼ Win32			- 🖄	TIME	• 7
	speed	13d.f スタ	ートページ	]		L				•	× V	<mark>見</mark> ーション :	エクスプローラー	עפע-בעע 🖕 🕂 🗙
₩-)\	C	FFTE:	A FAST	FOURIER TRA	NSFORM PAG	CKAGE	1	Deb	ug モー	- <b>ド</b>		] ] VII.5	Sature Texat	1 Յումերի)
Г Н	C	(0) (	OPYRIGHT	T SOFTWARE,	2000-2004,	, ALL	RIGHTS	RESERVE	ED			🚧 291  - 📑 FF	TE	0 7071717

#### ■ ソリューションのクリーン

Visual Studio 2005 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。

🏶 FFTE40 – Microsoft Visual Studio		
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) ツール(T) ウィンドウ(W) コミ	1ニティ©) ヘルプ( <u>H</u> )	
👬 🕶 🏣 🖌 🛃 🍠 🐰 🐴 🙈 👘 🗸 👹 ນາມ-ອອນຫຍັມທິເສີ Otrl+Shift+B 💈	👻 📑 T	TME 🔹 🚆
speed3d.f スタート ページ     ソリューションのリビルド(R)	▼ × עוער עוער ד	クスプローラー ソリューション 🖢 🚽 🗙
FFTE のビルド(型)	ערבעע 🌄 🔰	ョン 'FFTE40' (1 プロジェクト)
C (C) COPYRIGHT SOFTWARE, FFTE のリビルド(E)	🔲 🖃 🚰 FFTI	E nolude Files
C DAISUKE TAKAHASHI FFTE のクリーン(M)		] param.h
G UNIVERSITY OF TSUKUB, プロジェクトのみ(少) ト	- 🚞 R	lesource Files ource Files
C 1-1-1 TENNODAI, TSUKI C E-MAIL: daisuke@cs.t: バッチビルド①		y tests
ビー C 構成マネージャ(Q)		speed3d.f
き C ZFFT3D SPEED TEST PROGRAL (参 コンパイル(M) Ctrl+F7		kernel.f
C FORTRAN77 SOURCE PROGRAM		zfft3d.f
		✓ エクスプローラ 図 クラス ビュー
	プロパティ	<del>~</del> ‡ ×
IMPLICIT REAL*8 (A-H,O-Z) PARAMETER (NDA=33554432)	×	•
	> 2↓ □	
出力	<b>4</b> ×	
出力元の表示(2)、ビルド 💿 😽 🗐 🗐 🗐 🗐		
クリーン開始: プロジェクト: FFTE, 構成: Debug Win32 Deleting intermediate and output files for project 'FFTE', configuration 'Debug'	~	
======== クリーン: 1 正常終了、0 失敗、0 スキップ ========		
	_ •	
	2	
	1 50	1.#\$ #3
エロホレンクリーンが失いでいるした。 4 行	199	「大于 一押八 」

#### ■ 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。

![](_page_14_Picture_6.jpeg)

# ■ 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。

🏕 FFTE40 – 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)	<i><b>ī</b></i> :	「ッグ( <u>D</u> ) ツール(T) ウィンドウ( <u>M</u> )	_≷1_74©	C) ヘルプ(H)
i 🖥 • 🗃 • 💕 📕 🥔 i X 📭 🖭 i M • M • 🚚		ウィンドウ(型)		🕨 💌 📷 TIME 🔍 👻
speed3d.f スタート ページ		デバッグ開始( <u>S</u> )	F8	5 ューション エクスプローラー ソリューション 🖢 🚽 🗙
C C FETE: A FAST FOURIER TRANSFORM R	=D	デバッグなしで開始(日)	Ctrl+F8	5
C C C		プロセスにアタッチ( <u>P</u> )		<u>     ソリューション</u> 'FFTE40' (1 プロジェクト)
C (C) COPYRIGHT SOFTWARE, 2000-200	1	例外02	Ctrl+Alt+E	E FFTE
C DAISUKE TAKAHASHI	<u>S</u>	ステップ インΦ	F11	1 h param.h
C GRADUATE SCHOOL OF SYSTEMS A	(I	ステップ オーバー(①)	F10	0 Resource Files
C 1-1-1 TENNODAI, TSUKUBA, IBA	1		FS	9 ar burce riles
	1	ブレークポイントの作成(B)		speed3d.f
C ZFFT3D SPEED TEST PROGRAM		すべてのブレークポイントの削除(D)	Otrl+Shift+F9	9 - F kernel.f
C FORTRANZZ SOURCE PROGRAM	~			📄 📄 zfft3d.f
			_	マンジェーション エクスプローラ 3000000000000000000000000000000000000
C WRITTEN BY DAISOKE TAKAHASHI			>	ਹੋਰ//ੋਜ <b>ਦ ₽ ×</b>
出力		<i>v</i>	- 1 X	·
出力元の表示(S): ビルド	-	3 4 5 5		2↓ □
			~	
			~	
			>	
	_			
ビルト正市総了			l 行	1列 1文字 挿入 🦷

C:¥WINDOWS¥system32¥CMD.E	XE		- 🗆 X
KatoTest			
NX,NY,NZ =			
100,100,100			
NX = 100 NY =	100 NZ =	100 TIME =	
0.2241065586404147	497.9773937766156	MFLOPS	
FORTRAN STOP			
続行するには何かキーを押して	ください		
			-
•			• //

#### ■ 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。

MEETEIN - Microsoft Viewal Studio		
	A 195416	
ノアイル(E) 編集(E) 表示(E) ノロシェクト(E) ビルト(E) デバック(E) ツール(E) ら	インドリ <u>W</u> / Jミュ_テイビ/ /	
	Win32	🔹 🎯 IIME 🔹 🖉
과 · · · · · · · · · · · · · · · · · ·	עע <b>א ד</b> ע אין א	1ーションエクスプローラー ソリューション 🖢 👻 🗙
COMPLEX*16 A(NDA)	Image: A state of the state	
DIMENSION LNX(3),LNY(3),LNZ(3)		ソリューション 'FFTE40' (1 プロジェクト)
ほし ブレークポイントとするためにクリック		📴 FFTE
		Diparamh
print*, "KatoTest"		🛁 Resource Files
WRITE(6,*) ' NX,NY,NZ ='		🖻 📴 Source Files
READ(5,*) NX, NY, NZ		ereed3d f
CALL FACTOR(NY,LNY)		F fft235.f
CALL FACTOR(NZ,LNZ)		🖳 📄 kernel.f
CALL INIT(A,NX*NY*NZ)		E zfft3d.f
CALL ZFFT3D(A, NX, NY, NZ, 0)	_ 2	ノリューション エクスプローラ 💁 クラス ビュー
CHEL ZFFTOD(H, NA, NI, NZ, -1)	, Žo,	/(ティ → ╄ ×
		•
		2↓ □
Deleting intermediate and output files for project 'FFTE', configuration	'Debug'	
======= クリーン: 1 正常終了、0 失敗、0 スキップ =========		
	×	
	>	
□◎ エァーー覧   三 出刀		
正常にクリーンが実行されました。	4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。

🏶 FFTE40 – 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(T) ウィンドウ(W) コミュニティ(	<ol> <li>ヘルブ(H)</li> </ol>
i 🛐 • 🛅 • 📂 📕 🍠 i 🐰 🖬 隆 🔊 • (* • 🚚	ウィンドウω	🕨 🔹 🎯 TIME 🔹 🚆
スタート ページ speed3d.f*	<ul> <li>デバッグ開始(S)</li> <li>F!</li> </ul>	5 ຊーション エクスプローラー ソリューション 🖢 👻 🗙
COMPLEX#16 A(NDA)	=▶ デバッグなしで開始(H) Ctrl+F!	5
DIMENSION LNX(8),LNY(8),LNZ(8)	🔜 プロセスにアタッチ( <u>P</u> )	>ソリューション 'FFTE40' (1 プロジェクト)
SAVE A REAL*8 TIME1.TIME2.TIME0	例外⊗ Ctrl+Alt+I	FFTE
	ステップイン型 F11	I b param.h
I print*, Katolest	(ヨ ステップオーバー(0) F11	Resource Files
WRITE(6,*) ' NX,NY,NZ ='	ブレークポイントの設定/解除(G) F!	) Gurce Files
CALL FACTOR(NX,LNX)	ブレークポイントの作成(B)	speed3d.f
CALL FACTOR(NY,LNY)	すべてのブレークポイントの削縮金(D) Ctrl+Shift+F!	E fft235.t
	○ すべてのブレークポイントを無効にする(N)	zfft3d.f
CALL ZFFT3D(A, NX, NY, NZ, 0)		ママソリューション エクスプローラ 図 クラス ビュー
CALL ZFFT3D(A,NX,NY,NZ,-1)	×	70//ティ <b>↓ ₽ ×</b>
		-
出力	• # ×	2
	· · · · · · · · · · · · · · · · · · ·	
Deleting intermediate and output files for proje	ct 'FFTE', configuration 'Debug'	
======= クリーン: 1 正常終了、0 失敗、0 スキ・	· J =======	
<	>	
🔓 エラー一覧 🔳 出力		
אעדב	36 行	1列 1文字 挿入

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。

🐢 FFTE40 ( <del>7</del>	バッグ中	) – Mi	crosoft Visua	I Studio								
ファイル(F) 編	隼(E) ま	s±w	プロジェクト(P)	ビルド(B)	デバッグ(D)	<b>ツール(</b> 1	<ol> <li>ウィンドウ(W)</li> </ol>	7≷1-±4(C)	ヘルプ(H)			
		l v		a				2422710	T an	TME		
		6		(a + 4a	× ⊑≱   🕨 [		*			TIME		
speed3d.f									<b>→</b> ×	κ V	リューション エクスプロ… 🤜	, џ х
COM	PLEX*16	A(ND)	4)	- F					1	<b>~</b>		_
DIM	ENSION	LNX(3	, ).LNY(3).LNZ	(3)					_		🌄 ソリューション 'FFTE4	0'(1プロ
SAV	ΕA				テバッ	5		-	×	Ē	🖣 🚰 FFTE	
REA	L*8 TIM	E1,TI	ME2,TIME0			I 🖬 🖬	I 🔶 🔁 💭	🖆   16 進   🗔	•		😑 🗁 Include File	s
pri	nt*.″Ka	toTi	est″				7	<u>ک</u>	_		🔤 🛅 paramh	
C C							~ / / / /	12		-	Resource F	iles
I WRI	TE(6,*)	1 NX	NY,NZ ='								Source File:	s
REA CAL	D(5,*)   EACTO	NX,NY R(NY I	NZ NX)									ed3d.f
CAL	L FACTO	R(NY,	NY)			テハ	ツソ 進行	の前側			F fft235.f	
CAL	L FACTO	R(NZ,	.NZ)								— 📝 kernel.f	
C CAL	I INIT(	A NY#1	IV#N7)								📄 zfft3d.f	
CAL	L ZFFT3	D(A, N)	(,NY,NZ,0)									
CAL	L ZFFT3	D(A,N	(,NY,NZ,-1)									
LOO	IP=1						_			~		
<								)	>	<	<b></b>	>
コーカル						ųχ	コマンド ウィンドウ					ч×
名前		値	型			~						~
🥥 i		0	int	eger*4 i;								
😑 🥥 Inx		2	int	eger*4 lnx(	1:3);							
- 🥥 lnx (1	)	2	int	eger*4 lnx(	1:3);							
— 🧳 lnx (2	)	0	int	eger*4 lnx(	1:3);		<b>愛数</b> 個	[(シンボ	ルの値	[) <b>(</b> ,	)表示	
🖵 🥥 lnx (3	)	2	int	eger*4 lnx(	1:3);							
🗎 🌮 Iny		2	int	eger*4 Iny(	1:3);							
🗄 🥥 Inz		2	int	eger*4 Inz(	1:3);	~	2					5
2 100D 司白新恋逝 📰	n-tul	II III (Data)	⊥int ∓ 1	808°°4 1000			3.1並75中1.屁屎	== ブレーカポイト	ر. برید 💶 ا	ドロン		山山
HI H WORLD'S LACE	u 700	101-78-9	21				C + O II O MARK			1	- <u></u>	(/LLL ]
コマンド												

# 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

「ビルド」 -> 「構成マネージャ」を実行します。

🏕 FFTE40 – 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	_) ツール(T) ウィンドウ(⊻	V _%1_7-(0	2) ヘルプ(日)	)	
א אין דער 🚔 א 😭 א אין אין א אין אין אין אין אין אין אין	レド( <u>B</u> ) Ctrl+Shift+B 2		- 🖄	TIME	- 2
スタート ページ speed3d.f ソリューションの	ジルド(B)	• ×	ソリューション	エクスプローラー ソリューショ	ע <b>ד ד ד</b> צ
G FETE: A FAST FOURIER TRAI	/-ン( <u>C</u> )	~			
C (0) 000 μητουτ 000 τωμη Μάζη FFTE ΦΕμμικ	)		- בעע 🜄	ション 'FFTE40' (1 プロジェ	56)
C (C) CUPYRIGHT SUFTWARE, FFTE のリビル	<u>E</u> )		E- 🎯 FF	TE Include Files	
C DAISUKE TAKAHASHI FFTE のクリーン	N			b param.h	
G UNIVERSITY OF TSUKUB, プロジェクトのみ	⊅ ►			Resource Files	
び 1-1-1 TENNODAI, TSUKI C E-MAIL: daisuke@cs.t: バッチビルド①				i tests	
C 構成マネージャ	<u>)</u>			speed3d.f	
で ZFFT3D SPEED TEST PROGRA(参コンパイル(M))	Ctrl+F7			🖻 kernel.f	
C FORTRAN77 SOURCE PROGRAM				🖻 zfft3d.f	
			ערבווע⊊י	ョン エクスプローラ 💁 クラス	Ľı−
C WRITTEN DI DAISUKE TAKANASHI		>	プロパティ		<b>-</b> ₽×
出力		<b>-</b> ₽ ×			-
出力元の表示(5): ビルド - 📄	à 🗈 📑 🖃		8∎2↓ (0		_
		~			
		~			
		>			
17		-			
ビルド正常終了		1 行	1列	1 文字	挿入

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。

構成マネージャ								?×	
アクティブ ソリューション構成(©):				アクティブ ソリューション プラットフォーム(P):					
Release			Win32					*	
プロジェクトのコンテ	スト(ビルドまたは配置する)	プロジェクト構成をチェ	ック)( <u>R</u> ):						
プロジェクト		構成		プラット	トフォーム		ビルド		
FFTE		Release		💌 Win32		~	<ul> <li>Image: A set of the set of the</li></ul>		
	L								
- Delege	「「本西士ス」	自勇	か 的 に R の	elease	に変更され	れます			
Release	に変更りる								
							 開じ	3	

次に、「プロジェクト」 -> 「…のプロパティ」を選択し実行します。

<b>%</b>	FF	ГЕ40	- Miere	osoft Visu	al S	Studio										
-	ゆイル	۱Œ)	編集( <u>E</u> )	表示⊙	プロ	ジェクト(P)	ビルド( <u>B</u> )	デバッグ( <u>D</u> )	ツーノ	νD	ウィンドウ())	⊐≷⊒⊒∓≁©	シーヘルプ(日	)		
1	7 -	-	· 💕 🔒	🖉 🐰 🕻	•	新しい項目	の追加( <u>W</u> )	Ctrl+Shift+	A	e	▼ Win32		- 🕅	TIME		▼ <sup>1</sup> ∓
		スター	トページ〉	speed3d.f		既存項目(	D追加(G)	Shift+Alt+	A			<b>-</b> ×	ソリューション	/ エクスプローラー `	ソリューション '	. <b>-</b> ∓ ×
サーバ		C C	FFTE	: A FAST	-	スタートアッ	プ プロジェクト	(2設定(A)				_	E Mar	-?/=`/ 'FFTF40' :	1 ፓበ፡ንታታኑ)	_
GI –		C C	(0)	COPYRIGHT		FFIE (07	0/(77( <u>P</u> )	HEE HER		ESEF	RVED	=	🔤 📅 FI	FTE FTE	0.202220	
סלג		Č C		DAISUKE 1 GRADUATE	AKA SCH	HASHI DOL DE SY	STEMS AN	) INFORMA	TTON	ENGI	INFERING	- 11		param.h		
S.	Į	Č		UNIVERSIT	ΥÖ	F TSUKUB	1							Resource Files	;	
<b>X</b> 9 9-		C C		E-MAIL: d	INUD lais	ai, ISUKU uke®cs.t:	JBA, IBAR sukuba.ac	4КІ 385-85 .jp	573,	JAPA	AN	- 1		tests	df	
-ルポッ		C C	ZEFT	3D SPEED	TES	T PROGRAM	4					- 1		fft235.f		

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。

FFTE <b>プロパティ ページ</b> 構成( <u>C</u> ): アクティブ(Release)	✓ ブラットフォーム(P): アクティブ(Wi	n32)	<ul> <li>【</li> <li>【</li> <li>構成マネージャ(0)…</li> </ul>
<ul> <li>構成プロパティ</li> <li>General</li> <li>Debueging</li> <li>Fortran</li> <li>General</li> <li>Optimization</li> <li>Preprocessor</li> <li>Language</li> <li>Output</li> <li>Floating Point Options</li> <li>Command Line</li> <li>Linker</li> <li>Besources</li> <li>Build Events</li> <li>Custom Build Step</li> </ul>	Optimization Optimizations Vectorization Inlinine Use Frame Pointer Processor-Specific Optimization Loop Unroll Count Auto-Parallelization Maxin Coptimization Select option for overall code optimi	Maximize Speed (-fastsse) Default No No Default Default No mum Speed (-fastsse) ていることを確認する	se) と
		ОК	キャンセル 適用(A)

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。

🏶 FFTE40 – Microsoft Visual Studio						
ファイル(E) 編集(E) 表示(V) ブロジェクト(P) ビルド(B)	Ŧ	バッグ(①) ツール(① ウィ	心ドウ₩	_≷1_7-( <u>0</u>	)<ルプ(H)	
👔 • 🔤 • 💕 🛃 🍠 🐰 🗈 🛍 🖛 🖓 • 🕅 •		ウィンドウW			🕨 🖃 TIME	
speed3d.f スタート ページ	•	デバッグ開始( <u>S</u> )		F5	ューション エクスプローラー ソリュージ	ν=ν' <b>↓ ₽ Χ</b>
C EETE: A EAST EQUIPTED TRANSFORM E		- デバッグなしで開始( <u>H</u> )		Ctrl+F5		_
		プロセスにアタッチ(P)			📕 🖉 ンリューション 'FFTE40' (1 プロジ	፲ታኑን
C (C) COPYRIGHT SOFTWARE, 2000-200	4	例外⊠		Ctrl+Alt+E	FFTE	
C DAISUKE TAKAHASHI	5	ステップ インΦ		F11	h param.h	
C UNIVERSITY OF TSUKUBA	[]	ステップ オーバー(Q)		F10	- Carlos Resource Files	
C 1-1-1 TENNODAI, TSUKUBA, IBA	F	ブレークポイントの設定/解	峄[余(G)	F9	i burce riles	
	1	ブレークポイントの作成(B)	)		speed3d.f	
C ZFFT3D SPEED TEST PROGRAM		すべてのブレークポイントの	)肖·邶余(D)	Ctrl+Shift+F9	E trt235.f	
C EORTRANZZ SOURCE PROCEAM	^				📄 📄 zfft3d.f	
C					🖏 シリューション エクスプローラ 🧟 かき	57. Kii -
C WRITTEN BY DAISUKE TAKAHASHI				>	プロパティ	<b>-</b> ₽×
						•
出力テの表示(S) どルド	<b>.</b>			¥ * ^	2↓ □	
En state and a state of the sta		8   40 00   🗙 🗖				
				~		
<				>		
🖧 エラー一覧 🔳 出力						
ビルド正常終了			1	行	1列 1文字	挿入

## ■ 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。

FFTE <b>プロパティ ページ</b> 構成( <u>C</u> ): アクティブ(Release)	<ul> <li>ブラットフォーム(P): アクティブ(Wi</li> </ul>	n32)	<ul> <li></li></ul>
<ul> <li>構成プロパティ</li> <li>General</li> <li>Debugging</li> <li>Fortran</li> <li>General</li> <li>Optimization</li> <li>Preprocessor</li> <li>Language</li> <li>Output</li> <li>Floating Point Options</li> <li>Command Line</li> <li>Exercise</li> <li>Build Events</li> <li>Custom Build Step</li> <li>General</li> </ul>	Optimization Global Optimizations Vectorization Julian Use Frame Pointer Processor-Specific Optimization Loop Unroll Count Auto-Parallelization Auto-Parallelization	Maximize Speed (-fastsse) Default No No Default Default Yes	
		ОК	キャンセル 適用( <u>A</u> )

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

FFTE ታወለティ ページ	?
構成(): アクティブ(Release) ● 構成プロパティ - General - Debugging - Fortran - General - Optimization - Preprocessor - Unit - Floating Point Options - Command Line - Command Line - Linker - Resources - Build Events - General	▼ ブラットフォーム(P): アクライブ(Win32) ● 構成マネージャ(Q) Fortran Diabet Fortran 95 Process OpenMP Directives Yes           Process OpenMP Directives           Enable OpenMP 25 language extensions. (-mp)
	OK         キャンセル         適用(A)

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。

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)

FFTE プロパティ ページ	
構成(2): アクティブ(Release) ■ 構成プロパティ Ceneral ● Debugans ● General ● Optimization ● Preprocessor Language ● Output ● Floating Point Options ● Command Line ■ Linker ■ Resources ■ Build Events ■ Outom Build Step ■ General	▼ プラットフォーム(P: アクティブ(Win32) Command Command Arguments Working Directory Environment Margo Environment Yes Command Arguments Command Arguments Working Directory Environment Yes Command Arguments Command Argumen
	the existing environment. OK キャンセル 適用(A)

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 を起動する ようにしてください。

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

# 3 PVF コンパイラの起動(コマンド・ライン)

# 3.1 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」-> 「PGI Visual Fortran Tools」-> 「PVF Command Prompt (\*\*bit)」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。

![](_page_21_Picture_5.jpeg)

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。

![](_page_22_Picture_1.jpeg)

# 3.2 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 pgf95 (pgf90) 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」では、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)

🚳 PGI Visual Fortran 7.0-4 (32-bit)	- 🗆	×
C:¥Documents and Settings¥kato>cd C:¥		-
C:¥>cd PGI		
C:¥PGI>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 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 〈DIR〉 . 2007/05/28 13:25 24,576 test.dwf 2007/05/28 13:25 143,960 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		
C:¥PGI>pgf90 -fastsse -Minfo test.f		
C:¥PGI>test.exe hello!		
C:¥PGI>		-

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 7.1-4 (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> 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、\*.obi(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 Winodws 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 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)

# 3.3 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

# 4 その他

## 4.1 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¥PGl¥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 8¥redist

# 4.2 ヘルプ

**PVF** コンパイラのドキュメントは、Visual Studio 2005 の「ヘルプ」-> 「カテゴ リから検索」を選択して現れる「Microsoft Document Explore」内で参照できます。

![](_page_25_Picture_10.jpeg)

以 上