ISSN 0389-4010
UDC 681.31

航空宇宙技術研究所報告

TECHNICAL REPORT OF NATIONAL AEROSPACE LABORATORY

TR-915

スーパーコンピュータVPシステムによるプログラムの高速化技術

吉田正廣・中村絹代
内田啓一郎・棚倉由行

1986年10月

航空宇宙技術研究所
NATIONAL AEROSPACE LABORATORY

目　　次

スーパーコンピュータVPシステムによるプログラムの高速化技術*

吉田正廣** 中村絹代**
内田啓一郎** 棚倉由行***

Programming Techniques for High-Speed Processing on the Supercomputer FUJITSU VP-system

Masahiro YOSHIDA, Kinuyo NAKAMURA,
Keiichiro UCHIDA and Yoshiyuki TANAKURA

ABSTRACT

In this paper, various methods of effective FORTRAN programming for supercomputers are presented. These programming methods are designed to increase vectorization rates and to facilitate high-speed processing of FORTRAN programs. The methods have been used to improve the following NAL application software packages: the FVMCAS program for analyzing invicid compressible flow through three-dimensional cascades and the USTF3 program for unsteady transonic potential flow over oscillating three-dimensional wings. The vectorized codes are executed on the supercomputer FUJITSU VP-400. Computing times compared with those of the original programs show the excellent usefulness of the vectorizing methods.

1. 緒　言

近年，半導体素子技術のめざましい発展のもとに，スーパーコンピュータと呼ばれる科学技術計算用の超高速計算機の発表が相次いでいる。

コンピュータを高速化するためには，勿論高速素子，高密度実装技術の開発が重要であるが，これだけに頼ると自ずと限界があるため，スーパーコンピュータではベクトル処理の手法が取り入れられ，処理をできるだけ並列化する事により，その高速化を達成している。つまり，科学技術計算において頻繁に現れる配列データに対する同一の繰り返し計算を高速に実行するのである。繰り返し計算をベクトル処理するための機能はコンパイラに組み込まれているがそれはかなりの部分プログラムに依存するので，プログラムによっては，ベクトル処理機能が十分働かず，従来の汎用機と同程度の実効性能しか得られないこともある。従って，最大性能が汎用大型計算機に比べて数十倍に達するとしてもユーザはその性能を十分に引き出すために，問題の持っている並列処理可能性を十分に活用するアルゴリズムを開発し，コンパイラがベクトル化しやすいようなプログラムコーディング技術を検討して，ベクトル演算の比率（ベクトル化率）を高めなければならない。

* 昭和61年4月4日受付
** 計算センタ
*** 富士通株式会社

そこで本報告において，プログラムコーディングについて計算機のアーキテクチャを念頭においた改良技術を航技研の二つの汎用プログラムFV MCASおよびUSTF3を例にとって説明する。

なおこれは，昭和60年度に富士通㈱との間で行われた共同研究「VPシリーズ(VP400)による数値シミュレーション技術の高速化に関する研究」の成果に基づくものである。

以下，第2章にVPシステムの概要を，第3章にプログラムの分析手法，第4章に使用した汎用プログラムの概要と高速化のために行ったチューニングの内容とその効果および高速化のための作業についてのポイントをまとめる。

2. システム概要

2.1 機械の性能

本研究に於いて実測に使用したスーパーコンピュータFACOM VP-400について，その構成と性能概要を述べる。

2.1.1 ハードウェアの構成

FACOM VP-400の本体装置は，ベクトル処理装置(VPU:Vector Processor Unit)，主記憶装置(MSU:Main Storage Unit)，チャネル装置(CHP:CHannel Processor)，サービスプロセッサ(SVP:SerVice Processor)から構成される。

(1) ベクトル処理装置

ベクトル処理装置は，システムの制御およびスカラ命令の実行を行うスカラユニット(SU)とベクトル命令の実行を行うベクトルユニット(VU)からなる。

ベクトルユニットは，命令制御部，演算パイプ

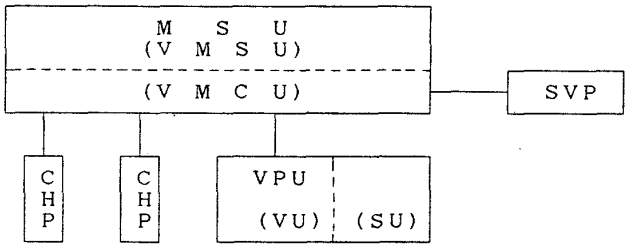


図1 ハードウエアの構成

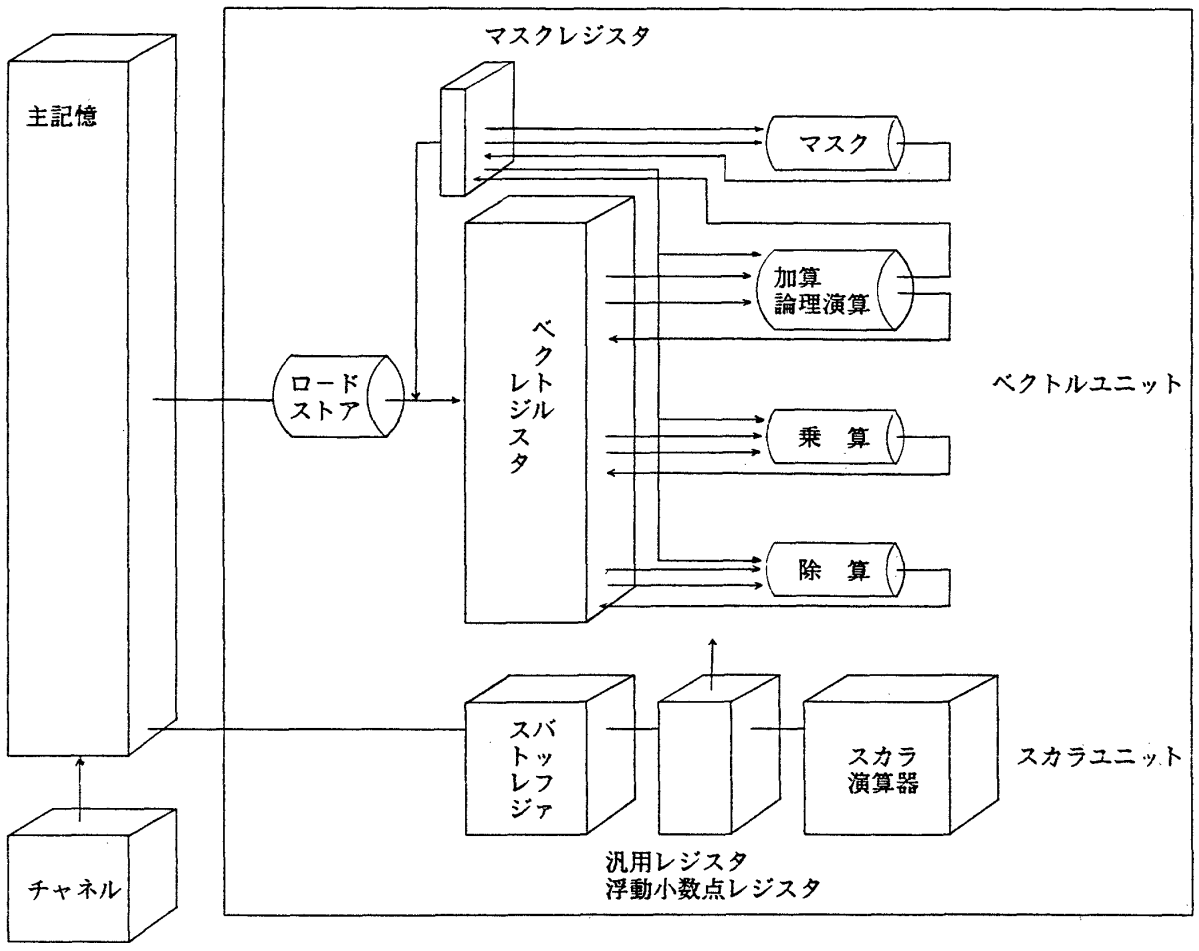


図2 ベクトル処理装置

ライン，ベクトルレジスタ，マスクレジスタなどから構成される。

図2にベクトル処理装置の構成概念図を示す。

(2) 主記憶装置

主記憶装置は，64，128，256MBの容量の主記憶構成を実現している。

記憶素子には，64Kビットスタティック RAMを採用している。

(3) チャネル処理装置

チャネル処理装置は2台まで装備可能であり，各チャネル処理装置には，チャネルを16台まで接続できる。

(4) サービスプロセッサ

サービスプロセッサは，ベクトル処理装置とは独立したサブシステムであり，プロセッサ・ディスプレイ装置，フロッピィディスク装置などから構成される。

2.1.2 ハードウェア性能

FACOM VP-400はマシンサイクル14ns（ただし演算パイプラインでは7ns）で最大性能が1140 MFLOPSのスーパーコンピュータである。性能諸元は表1に示す通りである。

(1) ハードウェア技術

FACOM VP-400ハードウェアは，400ゲート及び1300ゲートの高集積度で，かつゲート遅延時間350psのLSIを使用して設計されており，ベクトルレジスタとバッファストレージには，アクセス時間5.5nsの4Kビット／モジュールを使用してい

表1　FACOM VP-400の性能諸元

項目			性能
最大性能			1140MFLPS
マシンサイクル			14ns（ただし演算パイプラインは7ns）
命令数	スカラ		195
	ベクトル		83
ベクトル処理装置	レジスタ	汎用	16個（32ビット）
		浮動小数点	8個（64ビット）
		制御	16個（32ビット）
		ベクトル	128Kバイト
		スカラ	2Kバイト
	データ形式		論理　1，64ビット 固定小数点　32ビット 浮動小数点　32，64ビット
	パイプライン数		5本
	バッファストレージ		64Kバイト
主記憶装置	容量		64／128／256Mバイト
	インタリーブ数		128／256
チャネル処理装置	CHP接続台数		1／2　台
	チャネル接続台数		16／32　台
	スループット		24／48　Mバイト／秒

る。

(2) パイプラインの並列実行

アーキテクチャとしてパイプライン処理方式によりベクトル演算の高速化を図っている。パイプライン1段当たりの処理時間は，マシンサイクルに等しい7nsである。さらに演算パイプラインとして，ロード/ストアパイプライン1本，加算パイプライン1本，乗算パイプライン1本，除算パイプライン1本，マスクパイプライン1本の合計5本のパイプラインがあり，これらのうち，最大限4本のパイプラインが並列に可動する。また，異なるパイプラインを使用する命令間で，前の結果を受けて次のパイプラインに連鎖して動作することができる。命令の並列実行は，ベクトル命令間だけでなく，スカラ命令とベクトル命令間の並列実行も可能である。

(3) ベクトルレジスタ

ベクトル演算のために，128Kバイトのベクトルレジスタを使用している。ベクトルレジスタはデータの要素数に応じ，8Kバイト×64語をベクトルレジスタ長の最小単位としてレジスタを連結することができ，8, 16, 32, 64, 128, 256個の6種のレジスタの構成で使用できる。

(4) 条件付ベクトル演算

ベクトルレジスタに対応してマスクレジスタが用意されている。このマスクレジスタの値によってベクトルデータの演算を制御することができ，それによりベクトル化の範囲を拡げている。

(5) ベクトル命令

スカラハードウェア命令195種に加えて，ベクトル命令83種がある。ベクトル命令には，一般ロード/ストアや四則演算などのほかに，マクロ演算や編集演算などがある。又，等間隔の配列要素の参照や間接アドレスによる参照が行える命令形式になっている。

(6) 主 記 憶

記憶素子として64Kビットのスタティック RAM を使用しており，実記憶として256Mバイトまで使用可能である。

(7) 高速主記憶転送

高速演算能力を持つベクトルユニットに大量のデータをなるべく途切れなく供給する必要があるが，主記憶とベクトルユニット間に8バイト幅のデータバスを8本設け，14nsに64バイトすなわち，4.5Gバイト/秒のメモリスループットを達成している。転送速度は，ベクトルユニットの最大演算能力1140MFLOPSに見合うように設定されている。つまりベクトルユニットが最大演算能力に見合うように256ウェイのインタリーブ数が設けられている。

2.2 言語の概要

FACOM VPシステムの主プログラミング言語は，FORTRAN言語である。

2.2.1 FORTRANシステム概要

FORTRANシステムは，標準的な言語である ANS FORTRAN 77に準拠したFORTRAN言語で記述されたプログラムを処理する。本システムは，最適化機能，効率的なプログラムにするためのチューニング機能，高速入出力機能等の機能をサポートしている。また，科学技術計算用のサブルーチンパッケージも用意されている。

2.2.2 処 理 概 念

FORTRANシステムは，機能面から次の四つに分類される。(図3参照)

言語処理系
チューニング支援系
デバッグ支援系
サブルーチンパッケージ

言語処理系は，FORTRANプログラムの翻訳，並びに実行を行うためのコンパイラ，ライブラリ等である。チューニング支援系は，FORTRANプログラムの実行効率を上げるための情報を与えて，プログラムの修正を助ける。デバッグ支援系は，ユーザと実行プログラムとの間で，会話形式で情報の交換を行ったり，実行状況の表示などを行うデバッグツールである。サブルーチンパッケージは，科学技術計算で用いられる標準的な数値解法のサブルーチンライブラリである。

FORTRAN 77言語(FORTRAN Ⅳ言語も含む)で記述されたプログラムは，FORTRAN 77コンパイラ，又はFORTRAN 77/VPコンパイラに入力することができる。

FORTRAN 77コンパイラは，入力されたFORT-

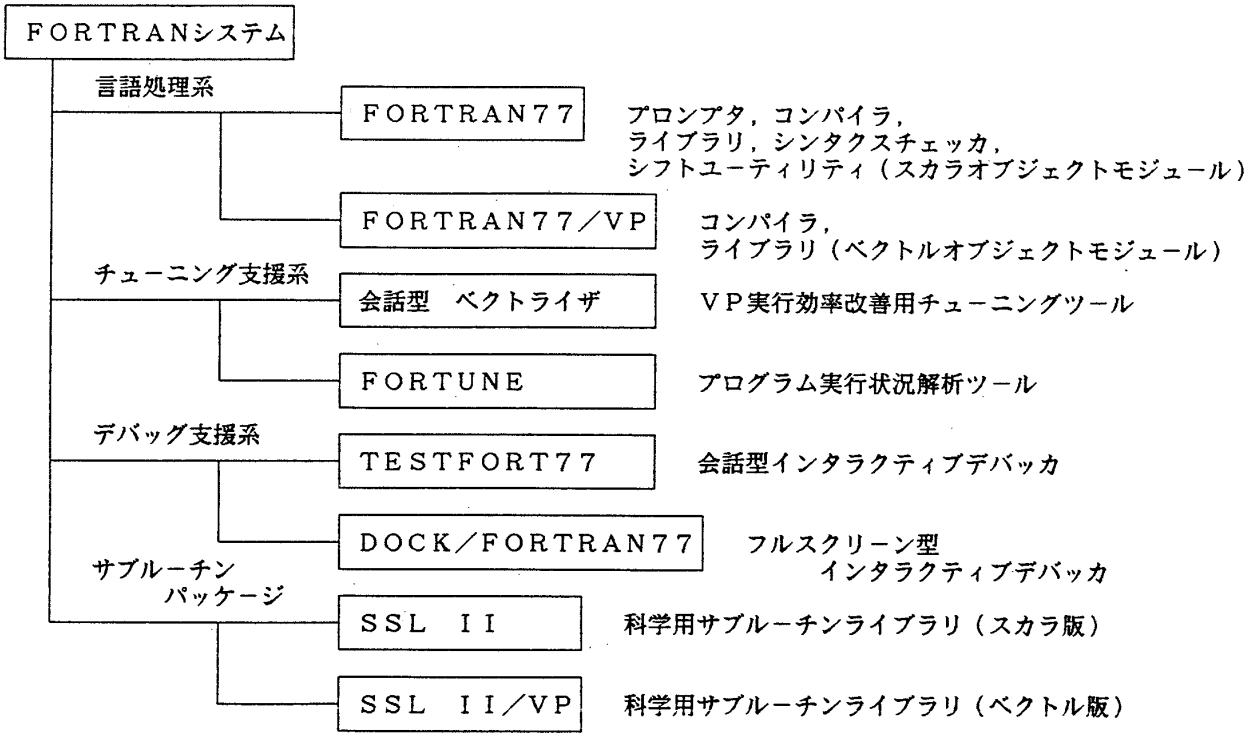


図3　FORTRANシステムの分類

開発過程	処　理　手　続　き	性　能
テスト実行	FORTRAN77コンパイラで翻訳，実行する。	スカラ性能でしかない。
本番実行	FORTRAN77／VPコンパイラで翻訳，実行する。	ベクトル効果発揮。
↓ チューンアップ	チューニングツールを用いてチューニングする。	FACOM　VPハードウェアの性能を最大限に引き出せる。

図4　プログラム開発過程

RANプログラムを，スカラ命令で構成されるオブジェクトモジュールに変換する。FORTRAN77/VPコンパイラは入力されたFORTRANプログラムを，スカラ命令並びにベクトル命令で構成されるオブジェクトモジュールに変換する。

VPシステムに適用するプログラムの開発過程を，図4に示す。

2.2.3　サポート機能

ここでは，主に，FORTRAN77/VPコンパイラのサポート機能の内，特にVPシステム向けの機能について述べる。

1)　最適化機能

FORTRAN77/VPコンパイラ（以下，VPコンパイラと呼ぶ）は，FORTRAN77コンパイラの持つ最適化機能に加えて，FACOM VPハードウェアの特性を活かすために，以下のような最適化機能を備えている。

a)　ベクトル化機能

VPコンパイラは，プログラム中のDOループ内の文が並列に処理できるかどうかを評価し，可能であれば，その並列処理可能部分をベクトル演算に変換する。DOループ内の四則演算，内積，組込み関数，条件付演算，リスト処理，集散処理などについてベクトル化することができる。

以下に，その例を示す。

例1：四則演算のベクトル化例

```
      DO 10 I=1, 100
```

```
      A(I)=B(I)+S*C(I)
      D(I,J)=(E(I,J)-F(I,J))/2.0
   10 CONTINUE
```

ベクトル化

⇩

ベクトル演算での処理表現

$A_i=B_i+S*C_i$, i=1,2,…,100

$D_{i,j}=(E_{i,j}-F_{i,j})/2.0$, i=1,2,…,100

例 2：IFなどの制御文を含むDOループのベクトル化例

```
      DO 50 I=1, N
      A(I,J)=SQRT(B(I)*C(I,J))
      IF(A(I,J).GT.ALPHA) THEN
        X=(A(I,J)+1.0)*S
      ELSE
        X=0.0
      ENDIF
      D(I,J)=COF*X+E(I,J)
   50 CONTINUE
```

ベクトル化

⇩

ベクトル演算での処理表現

$A_{i,j}=VSQRT(B_i*C_{i,j})$,i=1,2,…,N

$M_i=A_{i,j}.GT.ALPHA$,i=1,2,…,N

$X_i=(A_{i,j}+1.0)*S$;M_i,i=1,2,…,N

$L_i=.NOT.M_i$,i=1,2,…,N

$X_i=0.0$;L_i ,i=1,2,…,N

$D_{i,j}=COF*X_i+E_{i,j}$,i=1,2,…,N

ここで，VSQRTは，SQRTに対応するベクトル組込み関数である　変数Xは，DOループ中で一時的に使用される変数であり，コンパイラにより作業の配列X_i, i=1,2,…,Nに置き換えられてベクトル化される。また，MとLは，コンパイラが生成した作業用の論理型ベクトルデータである。「:M」は条件部またはマスク部と呼ばれ，Mの値の真/偽に応じて，左の演算の実行/非実行を意味する。

b) パイプラインパラレライズ機能

VPコンパイラは，FACOM VPハードウェアのパイプラインの並列実行性を効果的に利用するために，ハードウェア命令の単位で，それぞれの命令が並列に動作できるように演算順序の変更を行なう。

c) ベクトルテキストの最適化

テキスト最適化は，多数のFORTRAN文を対象として実行速度の向上を図る機能である。最適化には以下のような機能がある。

[共通式の削除]

共通式の削除とは，等しい演算結果をもつ共通な二つの演算に対して，後の演算では前の演算の結果を用いて後の演算を削除するものである。

例：

```
      DO 10 I=1, 1000
      X(I)=SIN(A(I))*B(I)
      Y(I)=SIN(A(I))*C(I)
   10 CONTINUE
```

⇩

```
      DO 10 I=1, 1000
      T=SIN(A(I))
      X(I)=T*B(I)
      Y(I)=T*C(I)
   10 CONTINUE
```

上記の例では，組込み関数SINの実行が，2回から1回に減少している。

[不変式の移動]

不変式とは，ループ内で演算してもループ外で演算してもその演算結果が変わらない演算である。不変式をループの外に移動して，不変式の実行回数を削減する。

[誘導変数の最適化]

誘導変数とは，ループ内で不変な値をもつ変数又は定数によってのみ回帰的に値が定義される整数型の変数であり，DO変数がその典型である。誘導変数の最適化とは，誘導変数とループ内で不変な値を持つ変数又は定数との演算を，新しい変数を導入してより高速な演算に変更することにより，ループ内の実行時間の短縮を図るものである。

例：

```
      DO 10 I=1, 1000
        :
      K=I*40+4
      X=FLOAT(I)
```

```
      ⋮
10  CONTINUE
      ⇓
    I1=44
    T2=1.0
    DO  10  I=1, 1000
      ⋮
    K=I1
    X=T2
      ⋮
    I1=I1+40
    T2=T2+1.0
10  CONTINUE
```

上記の例で，I1, T2はコンパイラが生成した変数であり，誘導変数を使用する演算I＊40+4, FLOAT(I)が，各々，より高速な加算命令に変更され，実行時間が短縮される。

[演算子強さの縮小]

演算子強さの縮小とは，ループ内にある演算をより高速な演算に変更して，ループ内の実行時間を短縮するものであり，例えば，定数による除算は，その逆数による乗算に変更される。

[単純代入の削除]

単純代入の削除とは，A=Bのような単純な代入文に対して，それ以降のAの参照をBで置き換えて，単純代入文を削除するものである。

[スカラ演算のくくりだし]

一般に演算の優先順位が等しい演算子は左から右に評価されるため，ベクトルデータが演算式中に現れると，その右側以降にスカラデータが現れてもベクトル演算で処理される。しかし，可換律を有する演算子で結ばれるベクトル演算項の変更を行い，ベクトル演算で実行しなくてもよい演算はスカラ演算で行なうように変更すれば実行時間を短縮することができる。

b) ベクトルレジスタの最適化

レジスタ最適化とは，レジスタを有効に利用することにより，実行時間の短縮を図るものであり，以下のものがある。

[レジスタ割付け最適化]

一般に，記憶域にあるデータを参照する命令よりも，レジスタ上にあるデータを参照する命令の方が処理時間が早い。レジスタ割付け最適化は，これに着目して，記憶域とレジスタ間のデータ転送を最少にとどめ，データを可能なかぎりレジスタに保持して，データの引用が効率よく行えるようにするものである。

[ベクトルレジスタ分割利用最適化]

FACOM VPハードウェアは，1ベクトル命令で取り扱う1ベクトルレジスタの容量(L)，及び使用するベクトルレジスタの個数(N)を可変にする機能，すなわち，ベクトルレジスタの分割利用機能をもっている。ただし，ベクトルレジスタの全容量(W)は一定である(W=N×L)。一般に，1ベクトルレジスタの容量が大きければ，長いベクトル長の演算が一度で行えるので，ベクトル命令単体の実行性能は向上する。一方，使用できるベクトルレジスタの個数が多ければ，記憶域とレジスタ間のむだなデータ転送が少なくなり実行性能は向上する。

しかし，ベクトルレジスタの全容量は一定であるから，プログラムによりその最適な分割方法を考慮する必要がある。ベクトルレジスタ分割最適化では，プログラム内のベクトル化される個々のDOループに対してベクトル長及びベクトルデータの個数によりベクトルレジスタの最適な分割利用を行い，実行速度の向上を図っている。

2) チューニング機能

VPコンパイラは，適切なベクトル化メッセージとチューニングメッセージを出力する機能を持っている。

3) パラレル入出力機能

一般に，一つの入出力処理は1台の入出力装置との間で行われ，入出力処理そのものの時間は，その入出力装置のハードウェアの性能に依存する。特に，大量データを入出力する場合には，入出力装置に対して多量のアクセス時間が必要となる。

パラレル入出力機能とは，大量のデータの入出

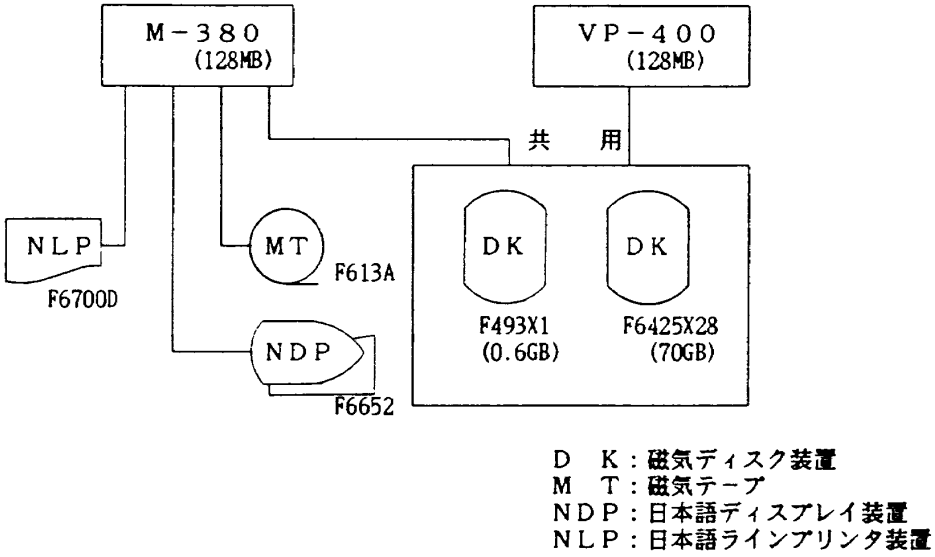


図5　システム機器構成図

力を複数台の入出力装置に対して並列に処理することによって，高速化を図る機能である。すなわち，大量の入出力データを分割（n分割）し，それぞれを複数台（n台）の入出力装置で独立に並列アクセスすることにより，処理時間を1/nに近づけようとするものである。但し，入出力装置を並列処理させるためI/Oパスの考慮が必要であり，また，入出力装置は直接アクセス記憶装置であることが必要である。

2.3　実行のハードウェア構成

実測に使われたシステム機器構成図を図5に示す。

3. プログラム高速化の概要と手法[1),2)]

3.1　分析の手法

2.2で述べた様に，VPコンパイラはプログラムを自動的にベクトル化するのである程度の高速化が図れる。しかし，個々のプログラムに応じたチューニングをすることによって実行性能をさらに改善することができる。この改善作業において，まずプログラムを分析することが必要である。以下にその手法について述べる。

3.1.1　分析処理の概要

最近の科学技術計算プログラムは大規模化している。このようなプログラム全体をベクトル化することはほとんど不可能に近く，仮りにできたとしても，それ程の効果が表われないところに労力をさくことは賢明なやり方ではない。従って，プログラム分析は次の事を目的に行う。

(a)　どこの場所をベクトル化すれば，高い性能が得られるかを決める。

(b)　ベクトル化のためのプログラムの変更方法を決定する。

この内，(b)については，3.2で述べ，ここでは(a)について述べる。

基本的な分析手順は以下の通りである。

1)　一般に科学技術計算のプログラムは数多くの副プログラムからなっているが，全体のCPU時間の90％以上は4～5個，多くても，10個程度の副プログラムで消費されている。またさらに1つの副プログラムをとっても，その一部の実行文で，ほとんどのCPU時間を消費している。分析作業用のツールFORTUNEを使って，このような副プログラム，又はその一部を見つけ出す。

2)　上記副プログラム等を，VPコンパイラにかけて，どの程度ベクトル化できるかを見る（ベクトル化率の算出）。また，現状のコーディングではベクトル化できないものがあれば，VPコンパイラの出すベクトル化情報から，その理由

を知る。以下，分析作業に於けるツールであるFORTUNE, VPコンパイラについて述べる。

3.1.2 FORTUNE

1) FORTUNEの構成

FORTUNEは解析の対照となるプログラムをFORTUNE原始プログラムに編集するエディタと，実行状況を解析するアナライザから構成されている。

(a) FORTUNEエディタ

エディタは，FORTUNEのオプションに従ってプログラムを解析し，原始プログラムを編集する。それと同時に，アナライザで必要となるオプション及び原始プログラムの編集情報を出力する。

(b) FORTUNEアナライザ

エディタによって編集された原始プログラムはFORTRANコンパイラによって翻訳され，リンケージエディタ又はローダによって，FORTRANライブラリ及びアナライザと結合編集されて，実行される。

アナライザはこのプログラムの実行開始直後及び実行終了直前に呼び出され，エディタから渡されたオプションに従って，エディタによって編集された原始プログラム，編集情報及び実行回数を記録したカウンタを入力しながらプログラムの実行の流れを解析し，実行回数，実行費用などの解析情報を原始プログラムのリストと共に出力する。

2) FORTUNEの機能

FORTUNEには種々の機能があるが，ここでは分析のために特に有用な機能について述べる。

(a) 実行文の実行回数と実行費用の出力

実行文に対して，その実行回数と，実行費用を出力する。なお実行費用とは単純な代入文を一回実行するのに必要な費用を1とした時の相対的な値であり，その実行文を実行するのに必要とする時間の目安となるものである。また，ベクトル化された実行文の場合，実行回数によりベクトル長が求められる。この機能により，1プログラム単位の内の実行費用の高い部分を知ることができる。

(b) プログラム単位ごとの実行回数と実行費用の出力

プログラム単位ごとの解析情報として，その実行回数と実行費用及び実行費用のプログラム全体に占める比率を出力する。ここで，プログラム単位の実行回数とは，そのプログラム単位が，他のプログラム単位から呼ばれた回数のことであり，プログラム単位の実行費用とは，そのプログラム単位内のすべての実行費用を合計したものである。この機能により，そのプログラム全体のうちで，実行費用の高いプログラム単位を知ることができる。

(c) IF文において，論理式が真となる場合の実行回数とその比率

論文IF文，ブロックIF文，及びELSE IF文に対しては，実行回数と実行費用の他に，それらの文の論理式が真の場合の実行回数とその比率を出力する。

(d) 解析するプログラム

FORTUNEのEDITオプションにより，実行時の振舞いを解析するプログラム単位を指定することができる。

3.1.3 FORTRAN 77/VP コンパイラ

VPコンパイラはベクトル化情報を出力する。ベクトル化情報としては，どのFORTRAN文が，ベクトル化されたかを示すベクトル化表示原始プログラムリストとベクトル化されたかどうか，あるいはどのようにベクトル化されたかを通知するベクトル化メッセージの2つがある。

1) ベクトル化表示原始プログラムリスト

ベクトル化表示原始プログラムリストには，原始プログラムのFORTRAN文に対応して，ベクトル化されたかどうか表示される。ベクトル化の表示は内部文番号(ISN)とFORTRAN文の間に次の記号で示される。

DO文に対する表示

V：このDOループに含まれるFORTRAN文は，すべてベクトル化されたことを示す。

M：このDOループに含まれるFORTRAN文には，ベクトル化されたものとされないものが混在していることを示す。

S：このDOループに含まれるFORTRAN文は，ベクトル化されなかったことを示す。

```
  ISN    V                    SOURCE
000001           SUBROUTINE  SUB(A,B,C,D,X,Y,Z,N)
000002           DIMENSION A(N),B(N),C(N),D(N),X(N),Y(N),Z(N)
000003   V       DO 10 I = 1 , N
000004   V       IF( A(I).GT.0.0 ) THEN
000005   V         B(I) = SQRT( A(I) )
000006   V       ENDIF
000007   V   10  CONTINUE
000008   M       DO 20 I = 1 , N
000009   V       X(I) = Y(I)*Z(I)
000010   S       C(I) = A(I-1)+D(I)
000011   S       A(I) = B(I)*C(I-1)
000012   V   20  CONTINUE
000013           RETURN
000014           END
```

図6　ベクトル化表示原始プログラムリスト

```
FORTRAN 77/VP VECTORIZATION MESSAGES: PROGRAM(SUB)
JND2011-I ISN:000004-000007 THE STATEMENTS IN THIS RANGE ARE VECTORIZED BY DO VARIABLE  I.
JND2051-I ISN:000005        THE STATEMENTS IN THIS RANGE ARE VECTORIZED BY GATHER/SCATER
                            METHOD.
JND2011-I ISN:000009        THE STATEMENTS IN THIS RANGE ARE VECTORIZED BY DO VARIABLE  I.
JND2011-I ISN:000012        THE STATEMENTS IN THIS RANGE ARE VECTORIZED BY DO VARIABLE  I.
```

図7　ベクトル化メッセージ

DO文以外の文に対する表示

V：ベクトル化されたFORTRAN文を示す。

M：ベクトル化されたものとされないものが混在しているFORTRAN文を示す。

S：ベクトル化されなかったFORTRAN文を示す。

空白：ベクトル化対象外のFORTRAN文を示す。

2)　ベクトル化メッセージ

ベクトル化メッセージは，通知メッセージの形式で表示され，ベクトル化されたかどうか，あるいはどのようにベクトル化されたかを知らせる。

3.2　実行性能改善の手法

3.2.1　VPにおけるプログラム高速化の要因

VPにおけるプログラムの相対処理速度（スカラ化）は，次の式で表わされる。

$$P=1/\{(1-V)+V/\alpha\} \qquad (1)$$

V：ベクトル化率

α：ベクトル/スカラ速度比

したがって，Vおよびαをできるだけ大きくすることが，プログラムの高速化につながる。

ここで，αは，計算機性能とともにプログラムの性質にも依存している。αに影響をあたえるプログラムの性質としては，次のようなものが考えられる。

ベクトル長：ベクトルデータの要素数の多い方がベクトル命令の処理効率が高い。

データ引用関係：VPでは，連続，等間隔，間隔の3種類のインデックスによるデータの引用が可能である。このうち，連続引用が最も効率が高い。

演算子数：一般にDOループ中の並列に実行できる演算子数が多い方が資源が有効に利用される。

3.2.2　ベクトルコーディング

VPにおいて，より高速に計算を実行するために，

ベクトル化の促進 — ベクトル化率(V)の向上
ベクトル演算の高速化 — ベクトル/スカラ速度比(α)の向上

を図ることが重要であり，そのためには，

最適化制御行を用いて，コンパイラに適切な情報を提供する。

プログラムの一部を書き替える。

などの方法がある。

(1) 最適化制御行

最適化制御行は，ベクトル化対象のDOループにおいて，ベクトル化されなかったDOループまたはステートメントに対して，ベクトル化のために必要な情報を与えたり，逆にベクトル化を抑制するため，また，ベクトル化されたDOループを更に高速に処理するための情報を与える事に使用される。最適化制御行は，行の第1桁から"＊VOCL"で始まり，第7桁以降に，情報の有効範囲と制御情報を書く。情報の有効範囲は，TOTAL（プログラム単位），LOOP（DOループ単位），STMT（ステートメント単位）のいずれかである。制御情報には以下の様なものがある。

(a) 変数の大小関係を示すもの。
(b) ベクトル化可能な配列であることを示すもの。
(c) IF文の真率を示すもの。
(d) DOループの繰返し回数を示すもの。
(e) ベクトル化の抑制を示すもの。
(f) ベクトル化を示すもの。

(2) ベクトル化の促進

ベクトル化対象のDOループにおいて，ベクトル化情報が"S"になる場合がある。これは，"S"のついた文がベクトル化されなかったことを意味するが，その原因として，

ベクトル化の基本条件は満たすが，自動ベクトル化のために必要な情報が不足している。

その文の要素が，ベクトル化の基本条件を満たさない。

などが考えられる。VPコンパイラは，ベクトル化できなかった原因をベクトル化メッセージとして出力するので，その内容に基づいてプログラムを修正すれば，ベクトル化の促進を図ることができる。

例えば，ループ中に現れるデータの定義/引用の関係は，ベクトル化の重要な鍵になる。回帰演算は本質的にベクトル化できないが，その他の場合でもデータの依存関係が明確でないために，ベクトル化できないことがある。次の例では，変数M,Nの値の関係に応じてベクトル化の可否が決まる。M≦Nであればベクトル化可能であり，この関係（M≦N）を明示する最適化制御行をループの先頭に挿入することによって，ベクトル化される。

```
0000010 S        DO  10  I=1,100
0000011 S        A(I+M) = A(I+N) + B(I)
0000012 S   10   CONTINUE

JND231I-I LNO:0000010-0000012 MAY CAUSE RECURSIVE REFERENCE (A BY N)
JND231I-I LNO:0000010-0000012 MAY CAUSE RECURSIVE REFERENCE (A BY M)
JND232I-I LNO:0000010-0000012 MAY CAUSE RECURSIVE REFERENCE (N AND M)
```

```
0000010   *VOCL LOOP,M.LE.N
0000011 V        DO  10  I=1,100
0000012 V        A(I+M) = A(I+N) + B(I)
0000013 V   10   CONTINUE

JND201I-I LNO:0000012-0000013 VECTORIZED BY DO VARIABLE I.
```

図8 ベクトル化促進の例

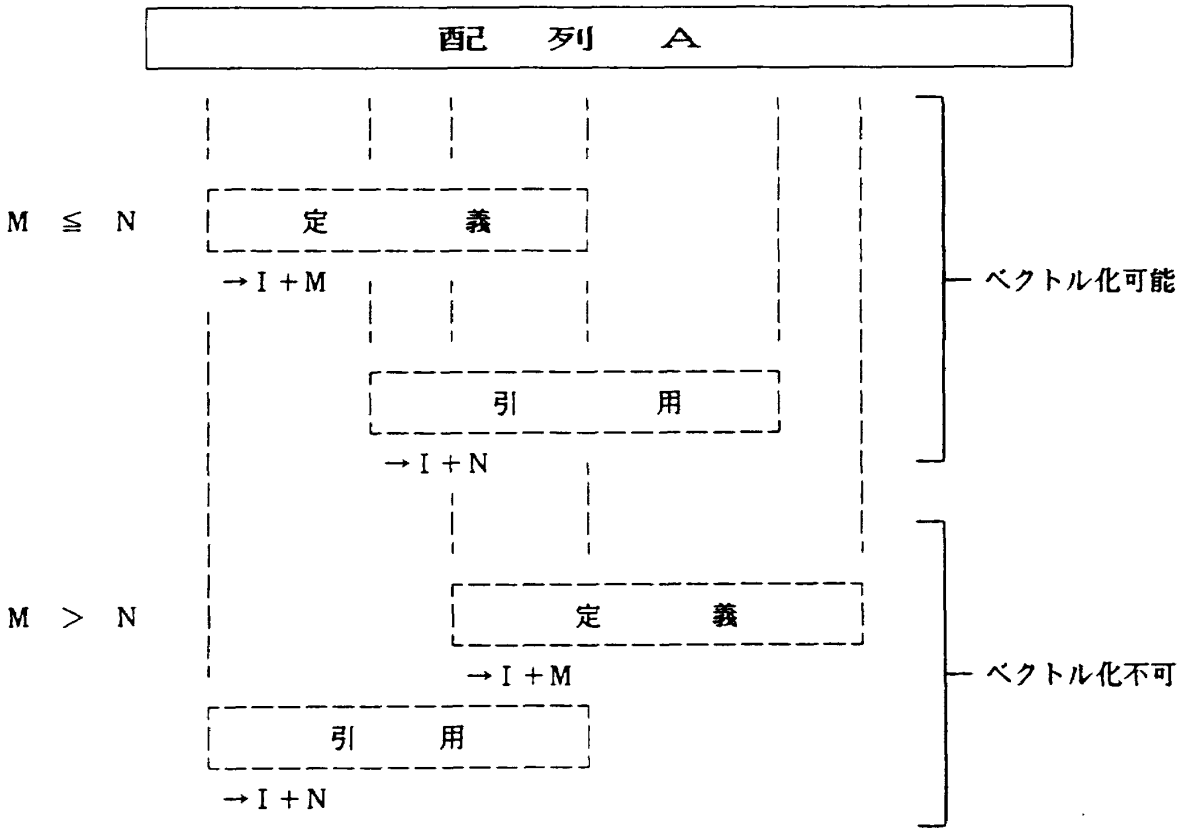


図9 データの定義/引用の範囲とベクトル化

この例で，データの定義/引用の範囲とベクトル化の可否の関係を図9に示す。

(3) ベクトル演算の高速化

ベクトル演算のDOループにおいて，ベクトル化情報が全て"V"になる場合でも，更に効率の高いオブジェクトを生成できる可能性がある。例えば，以下の様な方法で，ベクトル演算の高速化をはかることができる。

(a) IF文の真率を最適化制御行で指定する。

VPコンパイラは，IF文を含むDOループもベクトル化する。ベクトル化方式としては，

マスク付演算方式

収集/拡散方式

リストベクトル方式

を採用している。VPコンパイラは真率を仮定し，ループ中の演算密度やメモリアクセス頻度などに基づいて何れかの方式を用いたオブジェクトを生成する。しかし，予めIF文の真率の目安が分かっている場合には，最適化制御行にそれを指定することにより，適切な方式が選択される。一般に真率の値は低い方で0～30,高い方で70～100が判断の目安とされ，その他演算量により方式が選択される。

(b) DOループの繰返し回数を指定する。

DOループの繰返し回数を最適化制御行で指定すると，VPコンパイラはそれに応じて効率のよいベクトルレジスタの分割を行う。

(c) DOループの繰返し回数を可能な限り多くする。

ベクトル命令の実行時間は，一般的に次式で表現される。

$$\tau = \alpha * n + \beta \quad (\alpha << \beta) \cdots\cdots (2)$$

ここで，nは要素数，αは一要素あたりの実行時間，βはベクトル命令が完全に動作するまでの時間（立上がり時間）である。今，スカラ命令の実行時間を$\tilde{\alpha}$とすると

$$\alpha < \tilde{\alpha} < \beta \qquad (3)$$

になる。(2),(3)式より，ベクトル命令はnが小さい場合にはβが支配的になるために性能が期待できないが，nが大きい場合は効果的であることがわかる。すなわち，

```
000004  V        DO  10  I=1,N
000005  V        IF( M(I)  )   THEN
000006  V          A(I)=0.25*(B1(I+1,J)+B1(I-1,J)+B1(I,J+1)+B1(I,J-1))
000007  V          C(I)=0.25*(D1(I+1,J)+D1(I-1,J)+D1(I,J+1)+D1(I,J-1))
000008  V          E(I)=0.25*(F1(I+1,J)+F1(I-1,J)+F1(I,J+1)+F1(I,J-1))
000009  V          G(I)=0.25*(H1(I+1,J)+H1(I-1,J)+H1(I,J+1)+H1(I,J-1))
000010  V        ENDIF
000011  V   10   CONTINUE

FORTRAN 77/VP VECTORIZATION MESSAGES
JND201I-I ISN:000005-000011 THE STATEMENTS IN THIS RANGE ARE VECTORIZED
                            BY DO VARIABLE I.
JND204I-I ISN:000006-000009 THE STATEMENTS IN THIS RANGE ARE VECTORIZED
                            BY MASKED OPERATION METHOD.
```

```
000004  V        DO  10  I=1,N
          *VOCL STMT,IF(20)          真率は２０％と指定
000005  V        IF( M(I)  )   THEN
000006  V          A(I)=0.25*(B1(I+1,J)+B1(I-1,J)+B1(I,J+1)+B1(I,J-1))
000007  V          C(I)=0.25*(D1(I+1,J)+D1(I-1,J)+D1(I,J+1)+D1(I,J-1))
000008  V          E(I)=0.25*(F1(I+1,J)+F1(I-1,J)+F1(I,J+1)+F1(I,J-1))
000009  V          G(I)=0.25*(H1(I+1,J)+H1(I-1,J)+H1(I,J+1)+H1(I,J-1))
000010  V        ENDIF
000011  V   10   CONTINUE

FORTRAN 77/VP VECTORIZATION MESSAGES
JND201I-I ISN:000005-000011 THE STATEMENTS IN THIS RANGE ARE VECTORIZED
                            BY DO VARIABLE I.
JND206I-I ISN:000006-000009 THE STATEMENTS IN THIS RANGE ARE VECTORIZED
                            BY LIST VECTOR METHOD.
```

図10　IF文の真率を最適化行で指定した例

$$\frac{\widetilde{\alpha} * n}{\alpha * n + \beta} \underset{n \to \infty}{\Longrightarrow} \frac{\widetilde{\alpha}}{\alpha}$$

になる。

したがって，ベクトル化されるループの要素数を多くする必要がある。この要素数はDOループの繰返し回数によって定まる。次の例では，ベクトル化される内側のループの繰返し回数が多くなるように改めている。

例

```
S ┌──DO 10 I=1, M
S │    A(I)=0.0
V │┌──DO 20 J=1, N
V ││    A(I)=A(I)+B(I,J)*C(J)
V │└20  CONTINUE
S └─10  CONTINUE
```

⇩ M >> N

```
V ┌──DO 10 I=1, M
V │    A(I)=0.0
V └─10  CONTINUE
  ┌──DO  20 J=1, N
V │┌──DO 20 I=1, M
V ││    A(I)=A(I)+B(I,J)*C(J)
V │└20  CONTINUE
```

(4) ベクトルコーディングの留意点

VPコンパイラは，自動ベクトル化により，プログラムの高速化を図っている。しかし，ベクト

ル化されたプログラムでも，場合によっては実行効率が改善されないことがある。その原因として

ベクトル長が短い。

部分ベクトル化のオーバヘッド

などが考えられる。VPコンパイラは，翻訳時にこのような原因で実行効率改善が望めないことが明らかなときは，ベクトル化を抑止している。しかし，実際にはこれらのことは実行時に決まることが多いので，判断には限りがある。以下にその例と対応方法を示す。

1) ベクトル長が短い時

以下の例で，Nが非常に小さい（1又は2など）時は，最適化制御行でスカラ指定することにより，スカラ命令で実行した方が効率が良い。

例

```
V       DO  10  I=1, N
V       IF(M(I)) THEN
V         A(I)=B(I)+C(I)/D(I)
V       ELSE
V         A(I)=B(I)-C(I)
V       ENDIF
V   10  CONTINUE
              ⇩
 * VOCL LOOP,SCALAR
        DO  10  I=1, N
        IF(M(I))  THEN
          A(I)=B(I)+C(I)/D(I)
        ELSE
          A(I)=B(I)-C(I)
        ENDIF
    10  CONTINUE
```

2) 部分ベクトル化

部分ベクトル化されたDOループは，(1)ベクトル部分とスカラ部分間でのデータの受渡し，(2)スカラ部分のループ制御の増加等のオーバヘッドのために，十分な実効が得られないことがある。

次の例では，Nが小さいときは部分ベクトル化のオーバヘッドが大きくなり，実行効率の低下を招く。このような場合は，ソースプログラムのDOループをベクトル部分とスカラ部分とに分割するか，または最適化制御行を用いて抑止すれば良い。

例

```
S       DO  20  I=K, N
M       DO  20  J=1, N
M       A(J+1)=A(J)+C(J)/E(I)
V   20  CONTINUE
              ⇩
S       DO  20  I=K, N
V       DO  21  J=1, N
V       G(J)=C(J)/E(I)
V   21  CONTINUE
M       DO  20  J=1, N
S       A(J+1)=A(J)+G(J)
V   20  CONTINUE
```

4. プログラムおよび高速化のためのチューニング

この章では，実際に2本の汎用プログラム（FVMCASおよびUSTF3プログラム）を使って行った高速化のためのチューニング作業について述べる。

4.1 三次元翼間非粘性圧縮性流れの解析プログラム（FVMCAS）[3]

4.1.1 プログラム概要

(1) 理　論

本プログラムは，三次元翼間の定常な非粘性圧縮性流れ場を計算するものである。

この流れ場の解析に用いられる基礎方程式は，全温一定の関係式（$T_t = T(1+\frac{r-1}{2}M^2) = \mathrm{Const}$）を付加した三次元オイラー方程式

$$\frac{\partial \vec{\varphi}}{\partial t}+\frac{1}{r}\frac{\partial r\vec{F}}{\partial r}+\frac{1}{r}\frac{\partial \vec{G}}{\partial \theta}+\frac{\partial \vec{H}}{\partial z}=\vec{K} \quad (4)$$

$$\vec{\varphi}=\begin{bmatrix}\rho\\ \rho u_r\\ r\rho u_\theta\\ \rho u_z\\ \rho E_r\end{bmatrix}\vec{F}=\begin{bmatrix}\rho u_r\\ \rho u_r^2+p\\ r\rho u_r u_\theta\\ \rho u_r u_z\\ \rho u_r R_0\end{bmatrix}\vec{G}=\begin{bmatrix}\rho u_\theta\\ \rho u_\theta u_r\\ r\rho u_\theta^2+rp\\ \rho u_\theta u_z\\ \rho u_\theta R_0\end{bmatrix}$$

$$\vec{H}=\begin{bmatrix}\rho u_z\\ \rho u_z u_r\\ r\rho u_z u_\theta\\ \rho u_z^2+p\\ \rho u_z R_0\end{bmatrix}\vec{K}=\begin{bmatrix}0\\ p/r+\rho(u_\theta+r\omega)^2/r\\ -2r\omega\rho u_r\\ 0\\ 0\end{bmatrix}$$

となる。

$$E_r = E - r\omega(u_\theta + r\omega)$$
$$R_0 = H - r\omega(u_\theta + r\omega)$$
$$p = (r-1)[\rho E_r - \rho(q^2 - r^2\omega^2)/2]$$
$$q^2 = u_r^2 + u_\theta^2 + u_z^2$$

である。

(2) 計 算 法

計算法は，有限体積法（FVM）を用いる。(4)式の空間微分の項は，円柱座標での発散形式であるので，これを領域Vで積分すると

$$\frac{\partial}{\partial t}\iiint_V \vec{\varphi} r dr d\theta dz + \iint_S (\vec{F} r d\theta dz + \vec{G} dz dr + \vec{H} r dr d\theta) = \iiint_V \vec{K} r dr d\theta dz \quad (5)$$

となる。領域Vとして図11のような微小六面体をとり，時間微分を差分で置き換え，近似積分することにより

$$\Delta\vec{\varphi}\cdot|V| + \Delta t \sum_{i=1}^{6}(\vec{F}_i A_{ri} + \vec{G}_i A_{\theta i} + \vec{H}_i A_{zi}) = \Delta t \vec{K}|V| \quad (6)$$

が得られる。ここで$\vec{\varphi}$および$\vec{K}$の値は点Pで評価する。

(6)式に基づいて，適当な初期値$\vec{\varphi}^0$から始め，各時刻nでの増分$\Delta\vec{\varphi}^n$を求め

$$\vec{\varphi}^{n+1} = \vec{\varphi}^n + \Delta\vec{\varphi}^n \quad (7)$$

とすることにより時刻$n+1$での$\vec{\varphi}$の値が求まる。このようにして$\vec{\varphi}^n$が変化しなくなる（変化が十分小さくなる）まで計算を続け，そのときの$\vec{\varphi}^n$を定常解とする。

4.1.2 プログラムチューニング

FVMCASプログラムを，格子点数21×19×67点，イタレーション回数1000回の条件で実行すると，航技研計算センタのFACOMM-380計算機システムでは，1775.10秒（GOステップのみ）であった。これをそのままスーパーコンピュータVP-400システム（図5）で実行した場合，287.34秒かかり，M-380の6.18倍の性能が計測された。このプログラムを基にしていろいろ高速化のための技術を適用することにより，最終的には47.86秒となり，M-380の37.09倍（コンパイラのレベルを上げると40.05倍）にまで高速化することができた。表2にそのチューニング内容，VP400での実行時間およびM-380で実行したオリジナルプログラムとの倍率を示す。以下にチューニング内容およびそのチューニングを行う理由を詳しく述べる。

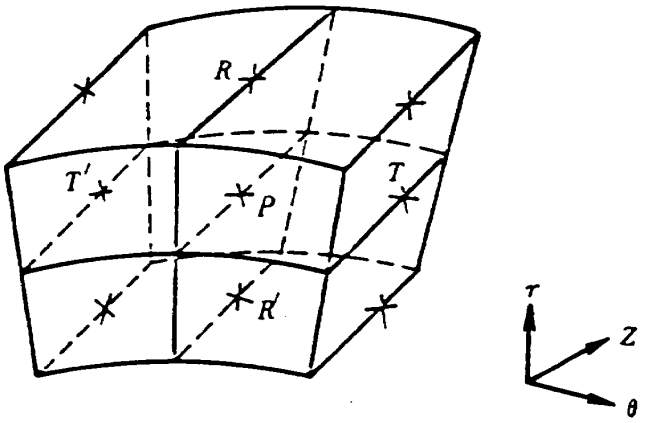


図11　積分領域

チューニング1

本プログラムのオリジナルソースプログラムを富士通のチューニング支援系ツール（会話型ベクトライザ，FORTUNE[4]）を使用し，イタレーション回数を100回として処理した結果を表3に示す。表から各サブルーチンのベクトル化率（V-RATE）をみると，FVMRRBルーチンが9.9%，SMOOTHルーチンが69.1%，FVMSOLルーチンが84.3%と他のサブルーチンと比べると低い。そこで最初のチューニングとしてこれら3つのルーチンのベクトル化率を上げることにした。

資料A-1にFVMRRBルーチンのDO11ループを示す。このDOループはWRITE文があるためにベクトル化されない。したがって，このDOループを計算の部分と出力の部分とに分割することにより，計算の部分をベクトル化できるようにした。SMOOTHルーチンのDO1010ループ（資料A-2）は複雑な処理形態になっており，これらもいくつかの処理単位に分割することにより，ベクトル部分を増した。FVMSOLルーチンにもSMOOTHルーチンのDO1010ループと類似した部分があり，これも同様に分割することにより，ベクトル化率を向上させた。また，このサブルーチンには，DO300ループ（資料A-3）と同様の形式で書かれた部分が多く，このDOループをみると，DO310ループとDO320ループはベクトル化され

表2 FVMCASのチューニング一覧表

番号	実行時間（秒）	倍率 :M380	チューニング状況 （ ）内はサブルーチン名	備考
	1775.10	——		スカラ(M380)
	287.34	6.18		オリジナル(VP400)
1	218.09	8.14	DOループの分割。(FVMRRB,SMOOTH,FVMSOL)	
2	186.36	9.47	配列の低次元化。	チューニングの対象をFVMSOL,FVMRINに限って行う。
3	174.00	10.20	IF文のベクトル化。低次元化の際の無駄な計算の削除。	
4	174.18	10.19	割算の計算式をDOループの外に移動。	
5	174.79	10.16	変数COEFM,WKの展開，一次元化。	
6	132.04	13.44	IF文のベクトル化。	
7	129.96	13.65	DOループの合体。	
8	101.64	17.46	IF文の削除。	
9	100.88	17.59	DOループの分割。	
10	100.79	17.61	IF文の修正。	
11	99.07	17.91	低次元化におけるマスクの削除。割算式をDOループの外に移動。	
	1872.42	——	チューニング1～11のスカラ版	スカラ(M380)
総合	47.86	37.09		
	44.32	40.05		FORTRAN/VP V10/L20

表3 チューニング前のベクトル化チューニング結果

サブルーチン名	V-COST	S-COST	V-RATE(%)
(全 体)	674×10^6	2526×10^6	91.9
FVMSOL	222×10^6	870×10^6	84.3
FVMUIN	78×10^6	310×10^6	99.8
FVMVIN	75×10^6	300×10^6	99.9
FVMWIN	75×10^6	300×10^6	99.9
FVMRIN	65×10^6	260×10^6	99.8
FVMRRB	47×10^6	51×10^6	9.9
SMOOTH	30×10^6	63×10^6	69.1
FVMURB	15×10^6	63×10^6	99.8
FVMVRB	15×10^6	61×10^6	99.8
FVMWRB	15×10^6	61×10^6	99.8

イタレーション回数は100回

* V-COST は各ルーチンをベクトル計算機で実行したときの，およその実行費用を示す。
** S-COST は各ルーチンをスカラ（汎用計算機）で実行したときの，およその実行費用を示す。
*** V-RATE は各ルーチンのベクトル化率を示す。

表4 チューニング1後のベクトル化チューニング結果

サブルーチン名	V-COST	S-COST	V-RATE(%)
(全 体)	5193×10^6	25095×10^6	91.9
FVMSOL	1080×10^6	8649×10^6	84.3
FVMUIN	770×10^6	3105×10^6	99.8
FVMVIN	751×10^6	2999×10^6	99.9
FVMWIN	751×10^6	2999×10^6	99.9
FVMRIN	652×10^6	2604×10^6	99.8
FVMRRB	108×10^6	393×10^6	96.1
SMOOTH	159×10^6	630×10^6	99.5
FVMURB	157×10^6	634×10^6	99.8
FVMVRB	153×10^6	618×10^6	99.8
FVMWRB	153×10^6	618×10^6	99.8

イタレーション回数は1000回

ているが，そのすぐ後の部分がスカラになっている。これもDOループを分割することにより，スカラ部分もベクトル化した。以上のように，DOループを分割することによって，ベクトル化率を向上させた。これらの処理結果を表4に示す。

この後のチューニング作業をまず使用頻度の高い代表的なプログラム形式を多く含むFVMSOLおよびFVMRINルーチンに絞って行うことにした。

チューニング2

ベクトル計算機を高速に使用するためには，ある程度のベクトル長が必要である。本プログラムは格子数が25×20×75までの大きさを処理できるが，実際のコーディングにおいてはベクトル長が長い所でも75要素，短かい所では20要素というように，VP400の高速性が十分に発揮できる様なベクトル長になっていない。一般にDOループの繰り返し回数は，変数で書くことが多く，二重三重のDOループなどは内側のループだけがベクトル化される。そこでチューニング2では，できるだけベクトル長を長くするために，配列の低次元化を行った。なお，低次元化の処理のできない配列は，最も長いベクトル長をもつ次元が最も内側のDOループとなるように置き換えた（資料A−1）。

低次元化によって，連続に処理できる量（ベクトル長）が増え，DOループの実行回数が減少するが，本プログラムのように配列の大きさが25×20×75で，実際に処理する配列の大きさが21×19×67と小さい場合は，その計算を行う部分と行なわない部分とを区別するマスク配列（資料A−4の場合だとMSO110）を作成し，区別する必要がある。

チューニング3

FVMSOLルーチンのDO100ループ（資料A−5）のような形式は，データなどをチェックするためによく使用されるコーディング形式であるが，DOループ内にWRITE文があるためにベクトル化されない。DOループをデータチェックの部分（IF文）とそのデータを出力する部分（WRITE文）に分けることにより，前者のIF文を含んだDOループをベクトル化した。

また，低次元化の際に余分な計算（Iを1から37500（25×20×75）までよい）を行っている部分を削除した。

チューニング4

スカラー計算機にも言えることであるが，割算は加算・乗算などに比べて処理速度が非常に遅い。また，その割算がDOループ内に使用されている場合は，さらに処理時間がかかる。このような場合，その割算式の値が逐次変わらない部分は，DOループの外側であらかじめその計算を行い，計算結果を配列に代入しておき，その値をDOループ内で使用するようにした。

チューニング5

FVMSOLルーチンのDO150ループ（資料A−6）の一次元化を行うために，配列変数$FX(i,j,k)$，$UA(i,j,k)$と$COEFM(k,l)$を同一次元にした。そこで，新しく配列変数$COEFM2(i,j,k,l)$を作り，配列変数$COEFM(k,l)$を$i\times j$個だけコピーした。

チューニング6

FVMSOLルーチンのDO111ループ（資料A−7）をさらに効率良く処理するために，DO114ループ内の変数WK1の判定処理をDO112ループ内で行いその数をカウントしておき，DO114ループに入る前にその値を判定し，もし零であればDO114ループを飛びこすような形式に修正した。このプログラムの場合，カウントされた値が零になる事が多いので，この処理は有効と考えられる。

チューニング7

FVMSOLルーチンの中にはベクトル長が等しいDOループがある。ベクトル処理される場合，個々のDOループに対してベクトル処理するための準備作業（一般に立ち立がり時間と呼ばれている）が必要とされる。したがって，ベクトル長の等しいDOループを合併することにより準備作業を減らす事ができ，また，並列処理できる演算要素が増えるので二重のメリットがある。この様なチューニングをFVMSOLルーチンに行った。

チューニング8

FVMSOLルーチンのDO113ループ（資料A－8）は，回帰的演算があるためにベクトル化されない。実際には，回帰的演算を含むDOループ内の変数Jに関する2つのIF文は全く意味を持っていない。したがって，この2つのIF文を削除することで，DOループのベクトル化を行った。

チューニング9

FVMSOLルーチンを更にベクトル化するためにチューニング1と同じDOループの分割を行った。

チューニング10

FVMSOLルーチンでの処理における不要なロード命令をなくすために，IF文を修正した。

チューニング11

チューニング2では低次元化のためにマスク配列を作成した。このマスク配列は計算する領域を区別するためのもので，その区別の必要のないDOループでは，判定作業を削除した。また，割算の実行回数を減らすために割算式をDOループの外側へ移動した。

総合チューニング

チューニング作業2～11は，2つのサブルーチンFVMSOLとFVMRINルーチンに限って行った。その他のルーチン（FVMUIN, FVMVIN, FVMWIN, FVMRRB, SMOOTH, FVMURB, FVMVRB, FVMWRBルーチン）は，上記のルーチンとほぼ同じコーディング形式を行っており，これらのルーチンにチューニング2～11の作業を行った。表5に総合チューニング後のベクトル化チューニング結果を示す。

4.2 三次元非定常ポテンシャル方程式解法プログラム（USTF3）[5]

4.2.1 プログラム概要

(1) 理　論

USTF3プログラムは遷音速流れ中で振動する三次元翼に作用する非定常空気力を計算するためのプログラムである。準線形の三次元非定常完全ポテンシャル方程式はデカルト座標系で次のよう

表5　総合チューニング後のベクトル化チューニング結果

サブルーチン名	V-COST	S-COST	V-RATE(%)
（全　体）	1971×10^6	15711×10^6	99.6
FVMSOL	502×10^6	6566×10^6	99.8
FVMUIN	152×10^6	1555×10^6	99.9
FVMVIN	137×10^6	1342×10^6	99.9
FVMWIN	137×10^6	1342×10^6	99.9
FVMRIN	117×10^6	1160×10^6	99.9
FVMRRB	103×10^6	398×10^6	98.3
SMOOTH	58×10^6	295×10^6	98.5
FVMURB	135×10^6	533×10^6	99.1
FVMVRB	130×10^6	518×10^6	99.2
FVMWRB	130×10^6	518×10^6	99.2

イタレーション回数は1000回

に書かれる。

$$(a^2-U^2)\phi_{xx}+(a^2-V^2)\phi_{yy}+(a^2-W^2)\phi_{zz}$$
$$-2UV\phi_{xy}-2VW\phi_{yz}-2WU\phi_{xz}$$
$$-2U\phi_{xt}-2V\phi_{yt}-2W\phi_{zt}-\phi_{tt}=0$$

$$U=\phi_x+\cos\alpha \qquad \alpha：迎角$$
$$V=\phi_y+\sin\alpha \qquad (8)$$
$$W=\phi_z$$
$$a^2=\frac{1}{M_\infty^2}-\frac{\gamma-1}{2}\left[2\phi_t+q^2-1\right]$$
$$q^2=U^2+V^2+W^2$$
$$C_p=2\left\{\left[1+\frac{\gamma-1}{2}M_\infty^2(1+2\phi_t-q^2)\right]^{\gamma/\gamma-1}-1\right\}/(\gamma M_\infty^2)$$

上の式において，物理量はすべて無次元化されている。流れが翼面に沿う条件（翼面の境界条件）は，

$$V=f_t+Uf_x+Wf_z$$

（$f<x, z, t>$：翼面の位置のy座標），

WAKEの境界条件（$y=0$ 平面）は，

$$\Delta\phi_t+\Delta\phi_x=0 \quad (y=0,\ x\geqq x+<z> \quad |z|\leqq S),$$

$z=0$の対象平面は

$$W=0 \qquad (z=0)$$

である。

(2) 計 算 法

x, y, z空間の物理領域を関数変換を使って，ξ，η，ζ空間の計算領域に変換する。図12(a)および(b)は物理空間の$x-y$平面および$x-z$平面の図である。図13(a)および(b)は計算空間の$\xi-\eta$平面および$\xi-\zeta$平面の図である。計算空間において(8)式の完全ポテンシャル方程式が時間依存の半陰的，陰的2段階差分法を用いて解かれる。この方法では，第1スイープでz平面毎にy方向の連立三項方程式をGauss消去法によって解き，x方向にスイープし，次に第2スイープで第1スイープで求まった解を用いて陽的に各点の解を求める。第2スイープの計算で1ステップ後の解が求まる。

4.2.2 チューニング

USTF3プログラムをステップ数200回のパラメータのもとで高速化した。格子点数は49×31×13である。これらの条件のもとでFACOMM-380計算機システムで実行した場合のGOジョブステップの実行時間は220.81秒で，主記憶使用量は3.3 MBである。これをVP400システムで実行した場合（JCLでVECTOR指定する），GOジョブステップの実行時間は306.58秒となり，SCALAR指定した場合の実行時間の1.38倍を必要とし，遅くなる。

USTF3プログラムは約60のサブルーチンから

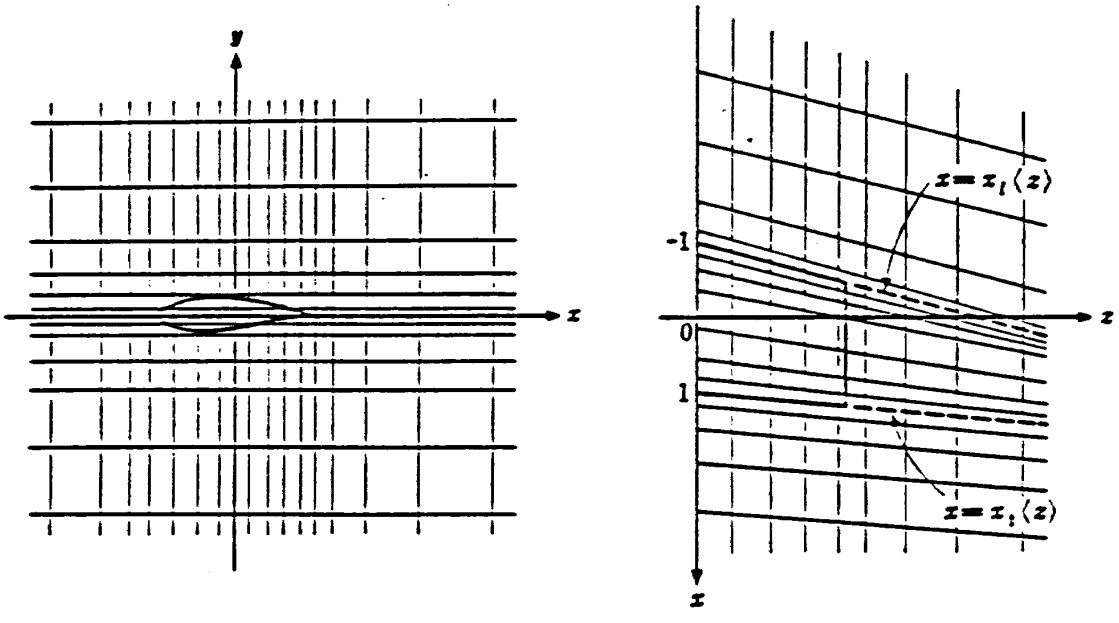


図12　物理空間平面

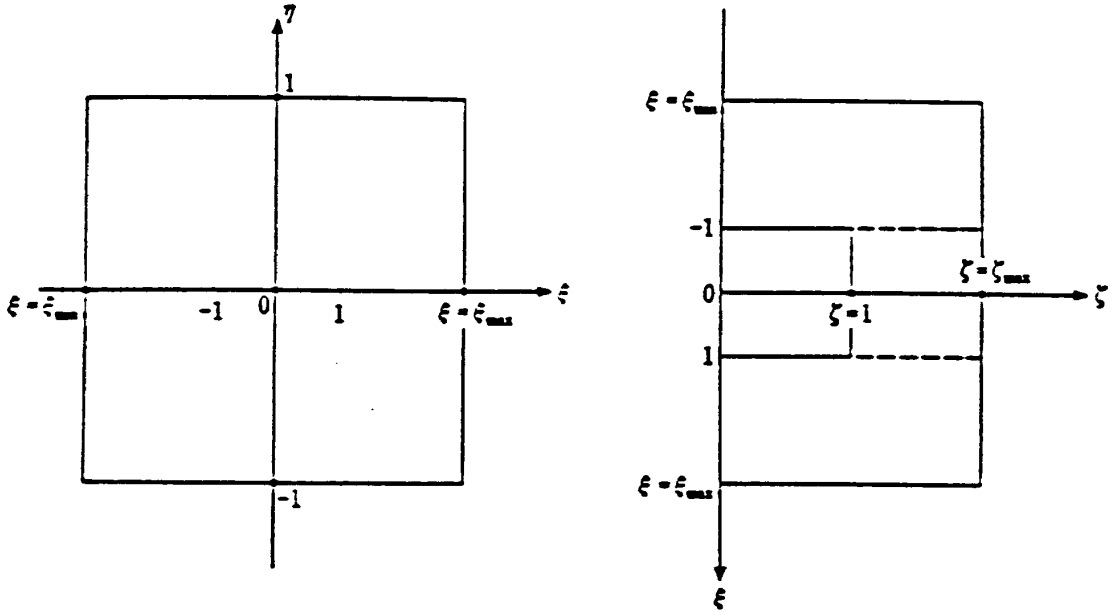


図 13　計算空間平面

成っているが，それらのサブルーチンのうち，中央処理装置（CPU）に対する負荷が 1％以上のものを表 6 に示した。表 6 のサブルーチンは，CCMAIN ルーチンを主ルーチンとするサブルーチン群であり，そこでは 4.2.1(2) の計算法で述べた差分方程式が図 13 の計算空間を 5 つの領域に分割して解かれている。また，SWEP 2 および SWEEP ルーチンで CPU 時間のほとんどが消費されていることがわかる。

従って，USTF 3 プログラムを高速化するためには表 6 に示したサブルーチンを対象にチューニングを施すことが重要である。

表 7 は USTF 3 プログラムのチューニング表である。チューニング内容，目的，チューニングを行った後の VP 400 システムでの GO ジョブステップの実行時間，および M 380 計算機システムにおける GO ジョブステップの実行時間を 1 とした場合の倍率を示す。全チューニング終了後の GO ジョブステップの実行時間は 24.76 秒となり，8.9 倍の高速化を得た。

なお，同プログラムの M-380 システムにおける GO ジョブステップの実行時間は 188.79 秒，主記憶使用量は 15 MB である。汎用計算機による実行時間も短縮されることがわかる。以下にチューニング内容の具体的説明および効果について述べる。必要に応じてチューニング事項を資料 B に示す。

表 6　USTF 3 プログラムの CPU 負荷

サブルーチン名	負荷（％）
SWEP 2	50．0
SWEEP	46．1
TRIDQ	1．7
CCMAIN	1．4

チューニング 1

SWEP 2 ルーチンは 1 タイムステップの第 2 スイープの陽的解法を行うプログラムである。このサブルーチンの中に次の例のようなコーディングが存在する。

```
      DO  100 I=1,N
      TRD(I)=CTC(I) * BA(I-1)
      DD(I)=1./(YAP(I)-TRD(I)*
                  DD(I-1))
      RS(I)=CTC(I) * DD(I-1)
  100 CONTINUE
```

表7　USTF3プログラムのチューニング表

NO	修正モジュール	チューニング内容	目　的	実行時間（秒）	倍率
1	SWEP2	ＤＯループ内の回帰的演算部分の分割	スカラ部分をベクトル化	135.53	1.63
2	CCMAIN	関数を呼び出し元に展開。*	スカラ部分をベクトル化	134.98	1.64
3	DDMAIN	配列変数の初期設定ＤＯループをブロックデータで実行。	計算の省力化		
4	SWEEP	サブルーチンＴＲＩＤＱを呼び出し元へ展開。	サブルーチン呼び出しの省力化		
5	CCMAIN	関数展開部のＩＦ文をＤＯループ外へ排出。	構造の単純化によるオーバヘッドの削減	132.72	1.66
6	SWEEP,SWEP2	IRH=4 の場合のみの計算を行うサブルーチンSWEEP4およびSWEP24を作成し、関連するＩＦ文を削除。	〃	101.75	2.17
7	SWEEP,SWEP2, SWEEP4,SWEP24	変数F,G,H1,H3 の計算を予め行い、配列にストアする。これを行うサブルーチンＦＧＨ１Ｈ３を作成。	計算の省力化	100.97	2.19
8	CCMAIN	関数を予び出し元に展開。	スカラ部分をベクトル化	100.35	2.20
9	SWEEP	ＤＯループ内のＩＦ文の削除または最適化制御行の挿入。	ベクトル化オーバヘッドの削除	77.52	2.85
１０	SWEP2	〃	〃	51.97	4.25
１１	SWEEP4,SWEP24	〃	〃	48.34	4.57
１２	SWEEP,SWEP2 SWEEP4,SWEP24	変数の配列化およびそれを行うサブルーチンＶＰＣＡＬの作成。サブルーチンの統合。	計算の省力化	26.96	8.22
１３	VPCAL	ＤＯループの次元（ｉ，ｊ，ｋ）を（１，ｋ）へ低次元化。	ベクトル長の拡大	24.76	8.92

＊　関数ＶＧの呼び出し元への展開が行われなかったため、ＤＯループはベクトル化されなかった。

オリジナルプログラムのＭ３８０システムにおけるＧＯジョブステップ実行時間　２２０．８１秒
オリジナルプログラムのＶＰ４００システムにおけるＧＯジョブステップ実行時間　３０６．５８秒

下線部のようなデータの定義/参照を回帰的なデータの定義/参照関係と言い，この場合DOループは完全にはベクトル化されない。従ってベクトル化されるようなコーディングに変える必要がある。ここでは資料B−1のようにDO 2ループの中にある線で囲った部分をDO2ループの外に出し，新しくDO9990 ループを作成することによって，DO2ループをベクトル化した。さらに，DO 9990 ループの中のDD の計算式のコーディングを変えることにより変数WのIF判定を削除してベクトル化オーバヘッドを削減した。

チューニング 2

DO ループの中に関数ルーチンが存在するとそのDO ループのベクトル化が妨げられる。CCMAIN ルーチンには関数FTD，FTX，FTZ，FTDC，FTXC，FTZC のためにベクトル化されないDOループがある。これらの関数をCCMAIN サブルーチンの中に展開することによりベクトル化をはかった。実際には関数VG を展開しなかったのでベクトル化されなかったが，関数の内部展開によってそれも含めて最適化が行われるメリットがあり，プログラムの高速化に効果的なテクニックなのでそれについて述べる。

資料B−2−1から2−3のように，関数FTD と

FTDC, FTXとFTXC, FTZとFTZCは線で囲った部分のみ異なり，しかも，FTDとFTXとFTZの関数ルーチンの構造は類似しているので，統合可能である。6つの関数ルーチンをまとめたものが資料B−2−4および2−5である。

また，CCMAINルーチン内の翼面の境界条件の計算に現れる資料B−2−6の変数Bの計算において，共通項を取り出して式の整理を行った。このチューニングでは共通項をくくり出すことにより共通項の計算回数が何分の1かになるといったメリットはないが，式を整理することにより複雑な式が単純化されて結果的には高速化される。また，単純化した式からヒントを得て高速化可能なコーディングに変更可能となるメリットもある。変数Bの計算は資料B−2−6から2−7のように書きかえられた。

チューニング3

DDMAINルーチンの中で資料B−3に示すような初期設定を行うDOループ群があり，これらによりSWEEPおよびSWEP2ルーチンで計算される領域が前もって零クリアされる。USTF3プログラムの実行においてこの零クリアは最初に一度だけ行れるのでDATA文でおきかえても差し支えない。DATA文による零クリアの方が計算機の仕事量は少ない。

チューニング4

SWEEPルーチンの中にそこで呼び出されているTRIDQルーチンを展開してサブルーチン呼び出し処理を省いた。ちなみにTRIDQルーチンは136200回呼び出されている。資料B−4にそのサブルーチンをSWEEPルーチン内に展開したプログラムを示す。

チューニング5

CCMAINルーチンの中に図14のフローとなる部分があるが，DO26ループ内のIF文のDO26

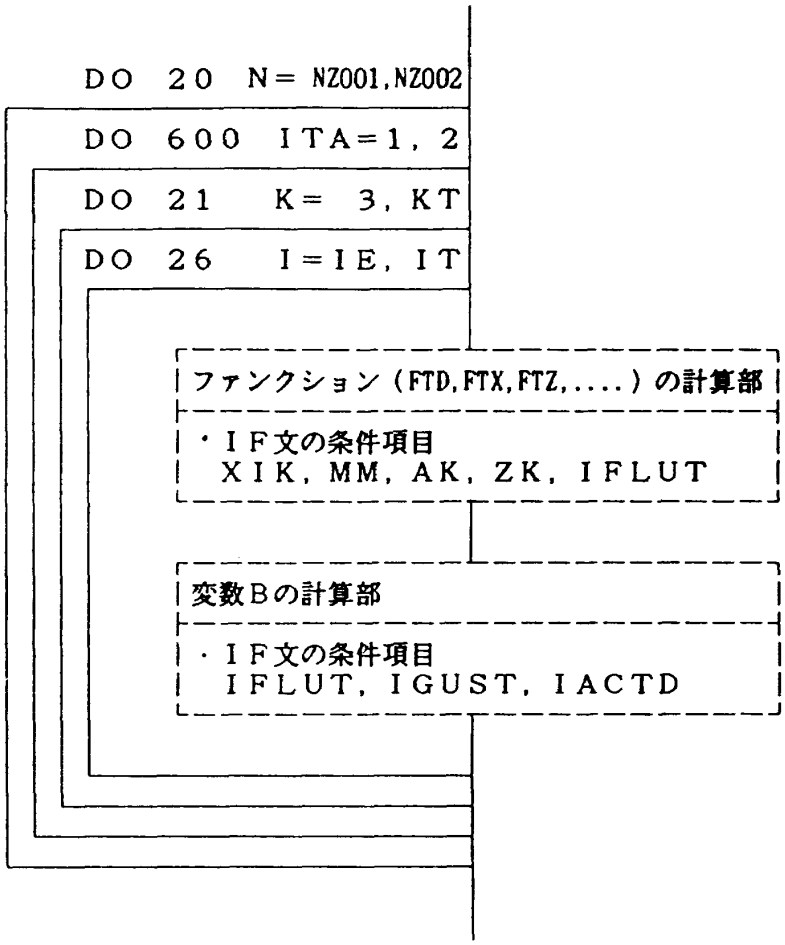


図14 CCMAINサブルーチンの一部

ループの外へ排出およびそれにともなって関数FTD, FTDC, FTX, FTXC, FTZ, FTZCと変数Bの計算式の整理を行った。

最初にIF文のDOループ外への排出について説明する。オリジナルプログラム内では複雑な形であらわれているので例を示して説明する。

```
      DO  10  I=1, N
        IF(M. EQ. 1) THEN
          Y(I)=A*X(I)
        ELSE
          Y(I)=B*X(I)+C
        ENDIF
10    CONTINUE
```

⇒

```
        IF(M. EQ. 1) THEN
          P=A
          Q=0. 0
        ELSE
          P=B
          Q=C
        ENDIF
      DO  10  I=1, N
        Y(I)=P*X(I)+Q
10    CONTINUE
```

この例ではDOループ内のIF文をDOループ外に出し，N回のIF判定を1回に減少させている。そして，仮変数PおよびQをつくりDOループ内では変数A, Bの代わりにP，変数Cの代わりにを使って配列Y(I)を求めている。これにより，ベクトル化されたDO10ループ内のIF文に対するベクトル化オーバヘッドが減少している。この方法はベクトル計算機だけでなく汎用計算機におけるプログラムの高速化についても役立つ方法である。

次に変数Bの計算機のコーディング変更について説明する。図14よりわかるように，IFLUT, IGUST, IACTDのパラメータにより変数Bの計算式が異なるが，全く異なるのではなく計算式の一部が異なるだけなので変数Bの式をいくつかのパートに分け，パラメータによりパートを加えることを指示する式に変更した。資料B－5の変更前のプログラムは既に1～4のチューニングで手が加えられたためわかりにくいと思われるので，オリジナルプログラムの変数Bの計算式について説明する。オリジナルの変数Bの計算式のうちの1つは次のコーディングとなっている。

```
B=-DET2*SIAL+(0.5*F(I,K)*DET2X*
  (PH2(IL+1, L-1, K, 2)-PH2(IL-1,
  L-1, K, 2))+(ETBU(IL, K)-ET(J-1))
  *0.5*F(I, K)*DETX*(PH2(IL+1,
  L, K, 2)-PH2(IL+1, L-1, K, 2)-PH2
  (IL-1, L, K, 2)+PH2(IL-1, L-1, K, 2))
  +DET2*COAL)*(FSXU(IL, K)+THT
  (K)+DWX(IL-2, K-2)+FTXD)+(H1
  (I, K)*PHXB+H3(K)*PHCB)*(FSZU
  (IL, K)+DWZ(IL-2, K-2)+FTZD)
  +DET2*(DWT(IL-2, K-2)+FTDD-
  -VG(XIK, ZK, T))
```

上の変数Bの計算式の —— 下線部は

```
VPPP1 = -DET2*SIAL
VPPP5 = 0.5*DET2X
VPPP6 = 0.5*DETX
VPPP7 = DET2*COAL
```

の形の仮変数をつくりDO20ループの外側に出した。SIAL, DET2X等はCCMAINルーチンの中で変化しないので問題はない。また，═══ 下線部は

```
VPPP2 = VPPP5*F(I, K)
        *(PH2(IL+1, L-1, K, 2)
        -PH2(IL-1, L-1, K, 2))
        +(ETBU(IL, K)-ET(J-1))
        *VPPP6*F(I, K)*(PH2
        (IL+1, L, K, 2)-PH2(IL+1,
        L-1, K, 2)-PH2(IL-1, L, K,
        2)+PH2(IL-1, L-1, K, 2))
        +VPPP7
VPPP3 = FSXU(IL, K)+THT(K)
VPPP4 = H1(I, K)*PHXB+H3(K)
        *PHCB
```

の形の仮変数をつくりDO26ループ内の変数Bの計算式の前におく。——，═══ 下線部以外の項に対して

DWX(IL-2, K-2)	VPIF (1)
FTXD	VPIF (2)
DWZ(IL-2, K-2)	VPIF (3)
FSZD	VPIF (4)

DWT(IL−2, K−2) VPIF (5)
FTDD VPIF (6)
VG(XIK, ZK, T) VPIF (7)

の対応関係をつくり，配列VPIFには予め0.または1.の値をDATA文で与えておき，上の左の項を計算上加えたり加えなかったりする。従って，変数Bの計算式は

```
B= VPPP1+VPPP2*(VPPP3+VPIF(1)
   *DWX(IL−2, K−2)+VPIF(2)*FTXD)
   +VPPP4*(FSZU(IL,K)+VPIF(3)
   *DWZ(IL−2, K−2)+VPIF(4)*FTZD)
   +DET2*(VPIF(5)*DWT(IL−2, K−2)
   +VPIF(6)*FTDD−VPIF(7)*VG(XIK,
   ZK, T))
```

となる。たとえば，VPIF(1)=0.0であればDWX(IL−2, K−2)は加えられない。

この修正により，変数Bの式に関するIF文をすべて削除できた。

変数Bの計算式の整理に関連し，共通式のくくり出しの原則を説明する。計算式の演算は左から行われるので，左からの演算式の共通のもののみ共通式としてコンパイラに認識される。したがって，離れている変数の演算式は共通式として認められない。共通式としてくくりだされる例およびくくりだされない例を以下に示す。

共通式としてくくりだされる例

例1 $\underline{A*B}*C$
$\underline{A*B}*D$

例2 $\underline{A_1*A_2*\cdots\cdots*A_n}*C$
$\underline{A_1*A_2*\cdots\cdots*A_n}*D$

$A*B$および$A_1*A_2*\cdots\cdots*A_n$は共通式としてくくりだされる。

共通式としてくくりだされない例

例3 $\underline{A}*C*\underline{B}=(A*C)*B$
$\underline{A}*D*\underline{B}=(A*D)*B$

例4 $C*\underline{A*B}=(C*A)*B$
$D*\underline{A*B}=(D*A)*B$

数学的には$A*B$は共通式であるが，演算の実行は左から行われるので演算としては共通式と認められない。したがって，例3および例4の場合には$A*B$は共通式としてくくりだされない。なお，例1に示す計算式のように変更すれば$A*B$は共通式としてくくりだされる。

関数FTX, FTD, FTZ, FTXC, FTDC, FTZCに対しても変数Bの計算式に施したものと同様の修正を行い，それにより各DOループ間で必要な処理が施されている。詳細は資料B−5を参照されたい。

これらすべての修正によりプログラムを単純化してオーバヘッドを削減した。資料B−5の5−1〜5−2は改造前，5−3〜5−6は改造後の処理を示す。

チューニング6

SWEEPおよびSWEP2ルーチンはCCMAINルーチン内で5分割された各々の計算領域の格子点での値を計算するために呼ばれる。それらのサブルーチン内に計算領域4に対してのみWAKEの境界条件の計算のために他の計算領域とは異なる計算を行う部分がある。そのために，計算領域4であるかの判定およびWAKE境界の位置であるかの判定を行うIF文が存在する。これらは1ケ所だけではなく数ケ所に及んでおり，計算領域が4でない時にも常にこのIF判定を行っている。

これらの無駄なIF判定を除くために，計算領域4の格子点での値を計算するためのサブルーチンとしてSWEEP4およびSWEP24のサブルーチンを新たに作成し，SWEEPおよびSWEP2ルーチンは計算領域1, 2, 3および5を計算するものとし，計算領域4に関するIF判定を削除した。資料B−6はSWEEP, SWEP2, SWEEP4およびSWEP24の修正前と修正後の部分プログラムである。なお，この修正に伴なってCCMAINルーチンでの計算領域4に対する呼び出しサブルーチン名の変更が行われた。

チューニング7

資料B−7−1に示されている線で囲った部分は，SWEEP関連（SWEEP, SWEP2, SWEEP4およびSWEP24ルーチン）のサブルーチン内で同様に現れており，SWEEP関連のサブルーチンが呼び出されるたびに1度計算される。計算領域毎に計算される位置が異なることおよび1イタレーションは2ステップから成ることから，SWEEP関連に

現れるこれらの計算をサブルーチンの外に出してCCMAINルーチンで予め計算しておき，サブルーチン内では既に計算された値を参照すれば，1/2の節約になる。資料B−7−1および7−2にSWEEP関連ルーチンの改造前および改造後を示し，資料B−7−3にSWEEP関連ルーチン内に共通に現れる計算を行うために新しく作成したFGH1H3ルーチンを示す。FGH1H3ルーチンはCCMAINルーチン中で1回呼び出される。

チューニング8

CCMAINサブルーチン内の関数VGをCCMAINルーチン内に展開し，関数VG内の共通項をDO600ループの外に出して，その後関数VGのコーディングを資料B−8のように変更した。これはチューニング2と同様の修正を関数VGに施したものである。

チューニング9

一般的にIF文の実行は他の演算命令の実行に比較して時間がかかる。従ってIF文を使わずにコーディングできる場合には使わない方が高速処理される。特にDOループ内でIF文が使用されている場合には，それを削除することにより大きな効果が期待できる。

SWEEP関連ルーチン内のIF文削除のために次の5つの処理を施した。資料B−9に従って説明する。

第一には対称平面の位置でのみ零の値を取るというコーディングが最内DOループで行われているのをその外側のループで対称平面上であるかどうかをIF判定し，最内DOループではIF判定を行わない計算式にかえた。

第二には最大値を求める組込み関数AMAX1の使用により，IF判定を書きかえた。

第三にはIFの算術式の正負値によりある変数の値が異った計算式に従う場合，既にある情報を使ってIF判定を除くように書きかえた。

第四にはIF判定の前に最適化制御行を入れることによりベクトル化を助ける働きを持たせる。この最適化制御行

```
*VOCL STMT, IF(5)
```

は次につづくIF判定の条件式の成立する比率（真率）が5％であるという指定である。真率の大小によりそれぞれ最適なオブジェクトプログラムを作成する。ただし，この真率を誤って指定しても計算結果に影響を与えることはない。

第五にはIF文の算術式の正負値によるその後の処理の相違が配列の添字の値の相違に関係づけられる場合であり，組込み関数SIGNを使用してIF文を削除した。流体計算を行う場合，流れの方向により差分式をかえることが多いのでこのコーディング法は有効である。

上に述べた5つのIF文に関する修正をSWEEPルーチンのDOループ内の全IF文に適用した。

チューニング10

チューニング9と同じ修正をSWEP2ルーチンを対象に行った。

チューニング11

チューニング9と同じ修正をSWEEP4およびSWEP24ルーチンを対象に行った。

チューニング12

CCMAINルーチンはポテンシャル方程式の主計算サブルーチンであり，そのフローは図15の通りである。そこで呼び出されるSWEEP関連ルーチンでは格子(i,j,k)でのポテンシャル計算を行っている。この計算においてSWEEP関連ルーチンでの共通計算をそれらのサブルーチンの呼ばれる前に予め計算しておき，SWEEP関連ルーチン内ではそれを参照するというフローに変更した。この修正により今まで同じ計算を2度行っていたものを1度に減らすことができるので，この計算の省略化はかなり有効なものである。

図16はチューニングを行った後のCCMAINルーチンフローを示したものである。SWEEP関連ルーチン内で計算される変数U, V, W, A2/Q2, PHYYC, PHZZC, PHXYC, PHYZC, PHXZCを共通項として，新サブルーチンVPCALでイタレーション毎に計算を行い，従来のSWEEP関連ルーチン内では既に計算された値を参照するプログラム構造にかえた。これらの計算の省力化は全計算領域に対して行われている。

ＣＣＭＡＩＮ，ＳＷＥＥＰ類のサブルーチンにおける計算

＜主計算のループ：ＣＣＭＡＩＮ＞

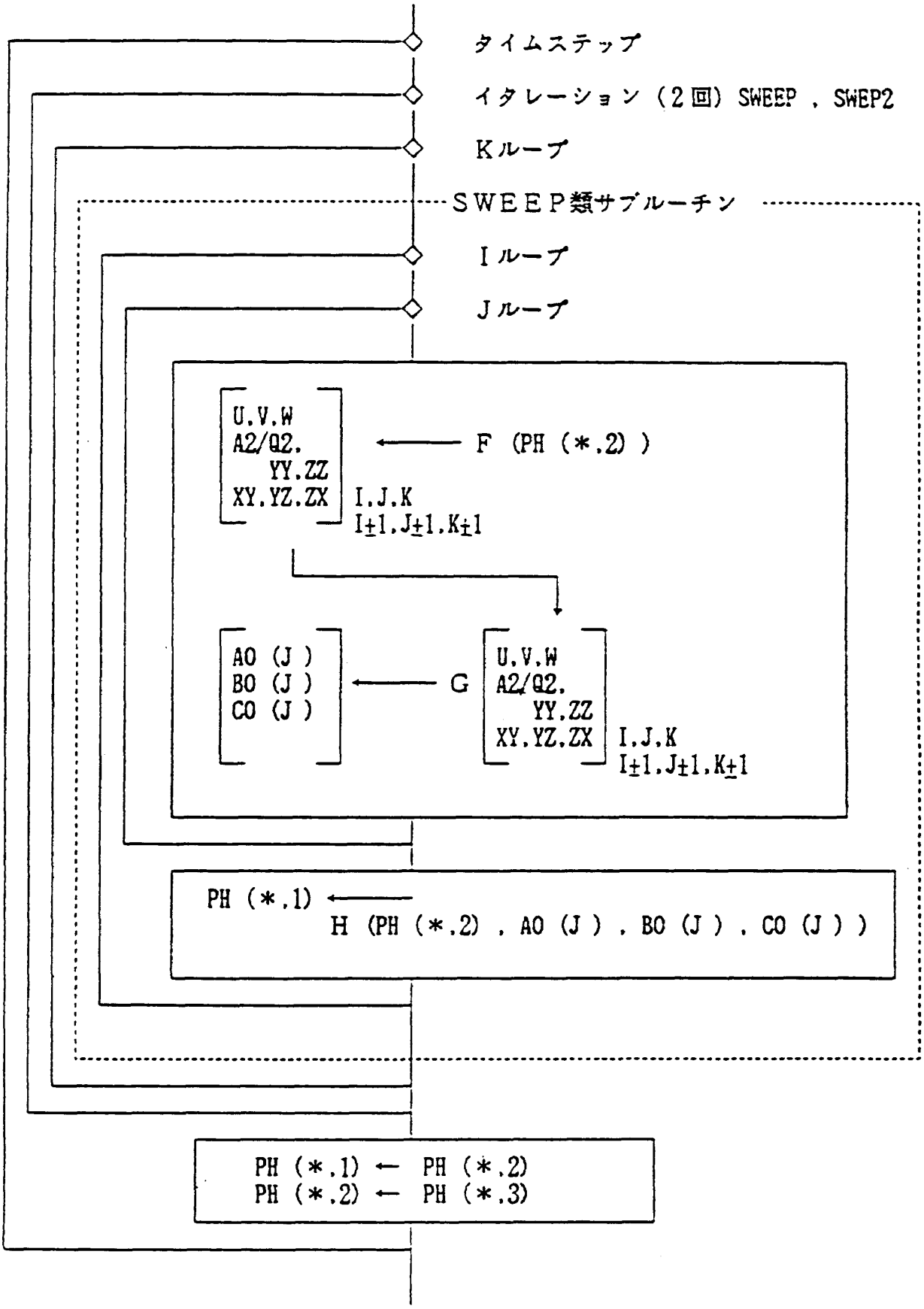


図 15　改造前のフロー

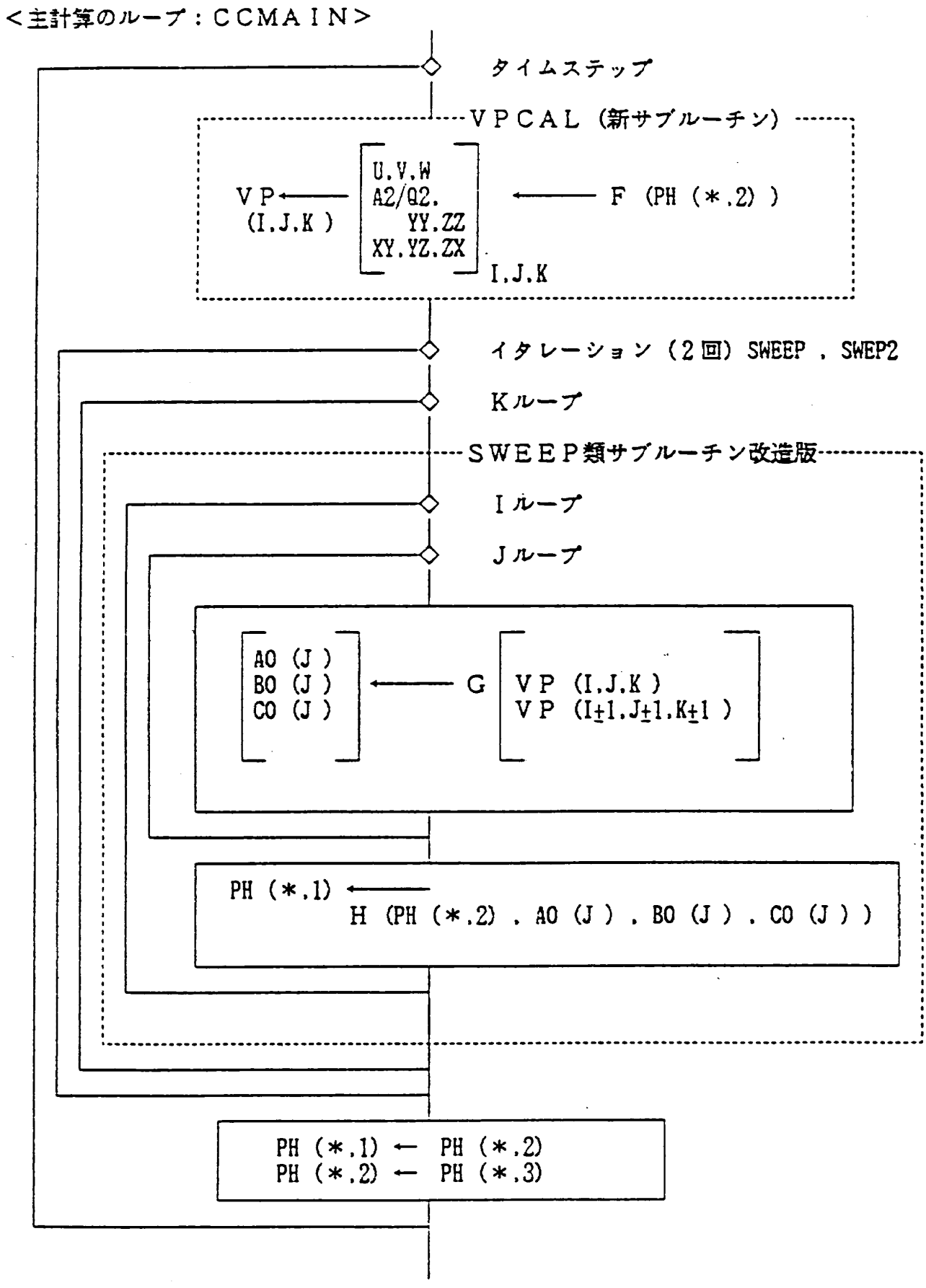


図16　改造後のフロー

資料B−12−1から12−3に新しく作成されたVPCALルーチンを示す。チューニング6においてWAKEの境界条件のためにSWEEP関連ルーチンが4つになったが，ここで再び統合し，SWEEPおよびSWEP2ルーチンの二つになった。WAKE境界条件のために他の計算領域と異なる計算を行う場合には，サブルーチンの最初の部分で予め作成する0, 1パターンの配列を乗ずることにより計算に加えたり加えなかったりする操作を行っている。資料B−12−4から12−6にSWEEPおよびSWEEP4を統合したサブルーチンを示す。同様にSWEP2およびSWEP24も統合した。

$$\left\{\left(\frac{a}{q}\right)^2-1\right\}\left\{U^2\underline{(\phi_{xx})^n_{ijk}}+V^2\underline{\underline{(\phi_{yy})^n_{ijk}}}\right.$$
$$+W^2\underline{\underline{(\phi_{zz})^n_{ijk}}}+2UV\underline{\underline{(\phi_{xy})^n_{ijk}}}+2VW\underline{\underline{(\phi_{yz})^n_{ijk}}}$$
$$\left.+2UW\underline{\underline{(\phi_{xz})^n_{ijk}}}\right\}+F^n_{ijk}-F^n_{i-1jk}+\underline{G^n_{ijk}}-\underline{G^n_{ij+1k}}$$
$$+\underline{H^n_{ijk}}-\underline{H^n_{ijk+1}}+\left(\frac{a}{q}\right)^2\left\{\left(q^2-U^2\right)\underline{(\phi_{xx})^n_{ijk}}\right.$$
$$+\frac{1}{2}\left(q^2-V^2\right)\left((\phi_{yy})^{n+1}_{ijk}+(\phi_{yy})^{n-1}_{ijk}\right)$$
$$+\left(q^2-W^2\right)\underline{\underline{(\phi_{zz})^n_{ijk}}}-2UV\underline{\underline{(\phi_{xy})^n_{ijk}}}$$
$$\left.-2VW\underline{\underline{(\phi_{yz})^n_{ijk}}}-2UW\underline{(\phi_{xz})^n_{ijk}}\right\}$$
$$-2U(\phi_{xt})^{n+1}_{ijk}-2V(\phi_{yt})^{n+1}_{ijk}-2W(\phi_{zt})^n_{ijk}$$
$$-(\phi_{tt})^{n+1}_{ijk}=0 \qquad (9)$$

$$\left\{\left(\frac{a}{q}\right)^2-1\right\}\left\{U^2(\phi_{xx})^{n+1}_{ijk}+V^2\underline{\underline{(\phi_{yy})^n_{ijk}}}\right.$$
$$+W^2\underline{\underline{(\phi_{zz})^n_{ijk}}}+2UV\underline{\underline{(\phi_{xy})^n_{ijk}}}+2VW\underline{\underline{(\phi_{yz})^n_{ijk}}}$$
$$\left.+2UW\underline{\underline{(\phi_{xz})^n_{ijk}}}\right\}+F^{n+1}_{ijk}-F^{n+1}_{i-1jk}+\underline{G^n_{ijk}}-\underline{G^n_{ij+1k}}$$
$$+H^n_{ijk}-H^n_{ijk+1}+\left(\frac{a}{q}\right)^2\left\{\left(q^2-U^2\right)\underline{(\phi_{xx})^n_{ijk}}\right.$$
$$+\left(q^2-V^2\right)(\phi_{yy})^{n+1}_{ijk}+\left(q^2-W^2\right)(\phi_{zz})^{n+1}_{ijk}$$
$$-2UV\underline{\underline{(\phi_{xy})^n_{ijk}}}-2VW\underline{\underline{(\phi_{yz})^n_{ijk}}}$$
$$\left.-2UW\underline{\underline{(\phi_{xz})^n_{ijk}}}\right\}-2U(\phi_{xt})^{n+1}_{ijk}$$
$$-2V(\phi_{yt})^{n+1}_{ijk}-2W(\phi_{zt})^{n+1}_{ijk}-(\phi_{tt})^{n+1}_{ijk}=0$$
$$(10)$$

上の(9), (10)式はそれぞれSWEEPおよびSWEP2ルーチンで解いている差分方程式である。このチューニングで行われた共通計算の省力化は二重下線部に対するものである。この他に一重下線部の計算も二重下線部の計算同様省力化可能である。本チューニングでは一重下線部のコーディングは複雑であったので対象としなかった。また，下線のない差分計算もその内の共通計算に対して同様の計算省力が可能である。

差分方程式の式の整理という観点から高速化を図ることも可能であるが，共同研究期間が短かったためこのアプローチからの高速化まで至らなかった。この場合にはSWEEPおよびSWEP2ルーチンを新たに作成する事になる。これはUSTF3プログラムの高速化に対する今後の課題となるであろう。

チューニング13

チューニング12で新しく作成した共通計算を行うサブルーチンVPCALの中で扱われる配列の次元を低次元化した。3次元(i,j,k)の配列の次元を2次元(l,k)にすることにより最内DOループの実行回数すなわちベクトル長を拡大してベクトル計算部分の有効処理をはかった。

PHn (i,j,k) ⇒ PH n (l,k)
VPn (i,j,k) ⇒ VP n (l,k,M)
$n=1\sim5$, $M=1\sim9$

の低次元化および，それにともなってチューニング7で扱われた配列も同様に次のように低次元化した。

F (i,k)		VPC n $(l,k,1)$
G (j)		VPC n $(l,k,2)$
H1 (i,k)		VPC n $(l,k,3)$
H3 (k)		VPC n $(l,k,4)$
FIMVP (i,k)	⇒	VPC n $(l,k,5)$
GJVP (j)		VPC n $(l,k,6)$
H1IVP (i,k)		VPC n $(l,k,7)$
H1KVP (i,k)		VPC n $(l,k,8)$
H3KVP (i,k)		VPC n $(l,k,9)$

$n=1\sim5$

これらの修正は全計算領域に対して行った。資料B−13−1および13−2に新サブルーチンVPCAL

の修正前と修正後のコーディング例を示す。また，資料B−13−3に新たに作成されたVPGRIDルーチンを示す。

4.3 考　察

以上のチューニングで述べてきたように，われわれは高速化のためのいろいろな手法を考え，実際に2つの汎用プログラムに適用した。これらの結果から高速化のためのプログラム技術についてまとめると，

(1) ベクトル化の対象は，一般にDOループで書かれた配列演算であるので極力DOループの形式で書く。

(2) DOループの中にベクトル化に不向きなステートメント（入出力命令，ユーザ関数，サブルーチンコール等）がある場合，それらのステートメントを分けることにより，ベクトル化部分を増す。

(3) 通常，IF文を含んだDOループもベクトル化されるが，IF文の無い記述をする方がより高速化が図れる。

(4) サブルーチンおよび関数ルーチンの呼び出しの省力化のために，それらを呼び出し元に展開する。呼び出し回数が多い場合は効果がある。

(5) プログラム構造を単純化する。

(6) DOループの中の配列を低次元化して，ベクトル長を長くすることにより，ベクトル計算機の効果を発揮させる。

(7) ベクトル計算機のハードウェアの並列演算処理能力を活かすために，DOループ内の演算の数を増やす。→ DOループの統括。

(8) DOループを統括することと，配列の低次元化をすることにより，ベクトル処理のための立ち上がり時間（スカラ命令）を削減する。

になる。

スーパーコンピュータを有効に使用するためには，(1)〜(5)の点について心掛ける必要がある。それからさらに高速化を行うためには，(6)〜(8)の点が重要なポイントになる。また，VPシステムでは，高速な処理方式の選択（IF文の真率指定，DOループ長指定によるベクトルレジスタの分割）が指定できる最適化制御行が使用でき，高速化が図れる。

今後，スーパーコンピュータ用のプログラムを作成する時には，上記に述べたことを十分に考慮してスーパーコンピュータの能力を十分に活用する必要がある。

5. 結　言

本報告では，実際にスーパーコンピュータVPシステムを使用し，プログラムの高速化技術の検証を行った。その結果，FVMCASプログラムで40.05倍，USTF3プログラムで8.92倍の成果が得られ，高速化技術の有効性が示された。それぞれの既存プログラムは計算機のリソースに制限があるため，一度行えばよい計算を何度も計算しなければならないこともあったが，現在では計算機のハードウェアおよびソフトウェア技術の発達により最先端のスーパーコンピュータでは最大256 MBの主記憶が使用可能なので，これを十分に有効利用して高速化を図ることが可能である。また，これらのプログラム高速化技術は更に高速なスーパーコンピュータ（パイプライン処理方式）が出現しても十分に利用できるものである。

計算高速化の1つの有力な手段であるベクトル化の促進のためのコーディング技術については，現在の時点においてかなりの成果を得た。もう1つの有力な手段である並列計算法については，今後検討すべき重要なテーマである。

本稿を終るにあたり，高速化のために対象プログラムを提供していただいた機体一部磯貝紘二技官，計算センター福田正大技官，プログラムチューニングに関して御協力をいただいたファコムハイタック株式会社の畠間晴夫氏，森重博司氏，中馬美利氏他，関連各部門の方々に紙上を借りて感謝の意を表する。

参考文献

1) 富士通㈱：OS Ⅳ/F4 MSP FORTUNE使用手引書

2) 富士通㈱：OS Ⅳ/F4 MSP FORTRAN77/VP使用手引書

3) 福田：三次元翼間の非粘性圧縮性流れの数値

計算，航空機計算空気力学シンポジウム論文集，航技研，SP−1，1983，pp.63−70

4) 富士通㈱：OSⅣ/F4 MSP 会話型ベクトライザ使用手引書

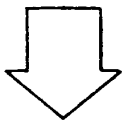
5) Isogai, K : Calulation of Unsteady Transonic Potential Flow Over Oscillating Three-Dimensinal Wings, TR706T, National Aerospace Laboratory, Japan, March 1982.

資料 A

———— 部が修正・追加部分

A－1

```
            :
            :
S       DO 11 K = 2 , NZ
S       DO 11 J = 2 , NT-1
S       DLTFLX  = SIGF*(   FR(IF,J,K)*SJKR(IF,J,K-1)
       .                 + FT(IF,J,K)*SJKT(IF,J,K-1)
       .                 + FZ(IF,J,K)*SJKZ(IF,J,K-1)    )
S       DLTFLX  = DLTFLX + 0.125*
       .        (  (  -FR(IF2,J+1,K)+6.0*FR(IF,J+1,K)+3.0*FR(IW,J+1,K) )
       .                                                  *SKIR(I,J+1,K-1)
       .        (  (  -FR(IF2,J-1,K)+6.0*FR(IF,J-1,K)+3.0*FR(IW,J-1,K) )
       .                                                  *SKIR(I,J-1,K-1)
       .        (  (  -FT(IF2,J+1,K)+6.0*FT(IF,J+1,K)+3.0*FT(IW,J+1,K) )
       .                                                  *SKIT(I,J+1,K-1)
       .        (  (  -FT(IF2,J-1,K)+6.0*FT(IF,J-1,K)+3.0*FT(IW,J-1,K) )
       .                                                  *SKIT(I,J-1,K-1)
       .        (  (  -FZ(IF2,J+1,K)+6.0*FZ(IF,J+1,K)+3.0*FZ(IW,J+1,K) )
       .                                                  *SKIZ(I,J+1,K-1)
       .        (  (  -FZ(IF2,J-1,K)+6.0*FZ(IF,J-1,K)+3.0*FZ(IW,J-1,K) )
       .                                                  *SKIZ(I,J-1,K-1)
S       IF( VOL(I,J,K-1).LE.0.0 .AND. K.LE.3 )  THEN
            KJI  =  K-1
S           WRITE(6,500) I, J, KJI, VOL(I,J,K-1)
  500       FORMAT(3(5X,3I5,E15.7))
S           VOL(I,J,K-1) = VOL(I,J, 3 )
        ENDIF
S       PN(I,J,K) = 0.125*( -PO(IF2,J,K)+6.0*PO(IF,J,K)+3.0*PO(IW,J,K) )
       .          - DLTT*DLTFLX/VOL(I,J,K-1)
S   11  CONTINUE
            :
            :
```

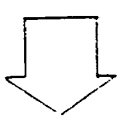


```
      DIMENSION DLTFLX(25,20,75)
           :
           :
S     DO 11 K = 2 , NZ
V     DO 11 J = 2 , NT-1
V     DLTFLX(I,J,K) = SIGF*(   FR(IF,J,K)*SJKR(IF,J,K-1)
     .                       + FT(IF,J,K)*SJKT(IF,J,K-1)
     .                       + FZ(IF,J,K)*SJKZ(IF,J,K-1)    )
V     DLTFLX(I,J,K) = DLTFLX(I,J,K) + 0.125*
     .        (   ( -FR(IF2,J+1,K)+6.0*FR(IF,J+1,K)+3.0*FR(IW,J+1,K) )
     .                                                *SKIR(I,J+1,K-1)
     .          - ( -FR(IF2,J-1,K)+6.0*FR(IF,J-1,K)+3.0*FR(IW,J-1,K) )
     .                                                *SKIR(I,J-1,K-1)
     .          + ( -FT(IF2,J+1,K)+6.0*FT(IF,J+1,K)+3.0*FT(IW,J+1,K) )
     .                                                *SKIT(I,J+1,K-1)
     .          - ( -FT(IF2,J-1,K)+6.0*FT(IF,J-1,K)+3.0*FT(IW,J-1,K) )
     .                                                *SKIT(I,J-1,K-1)
     .          + ( -FZ(IF2,J+1,K)+6.0*FZ(IF,J+1,K)+3.0*FZ(IW,J+1,K) )
     .                                                *SKIZ(I,J+1,K-1)
     .          - ( -FZ(IF2,J-1,K)+6.0*FZ(IF,J-1,K)+3.0*FZ(IW,J-1,K) )
     .                                                *SKIZ(I,J-1,K-1)
V  11 CONTINUE
S     DO 12 K = 2 , NZ
S     DO 12 J = 2 , NT-1
S     IF( VOL(I,J,K-1).LE.0.0 .AND. K.LE.3 )  THEN
          KJI  =  K-1
S         WRITE(6,500) I, J, KJI, VOL(I,J,K-1)
  500     FORMAT(3(5X,3I5,E15.7))
S         VOL(I,J,K-1) = VOL(I,J, 3 )
S     ENDIF
S  12 CONTINUE
S     DO 13 K = 2 , NZ
V     DO 13 J = 2 , NT-1
V     PN(I,J,K) = 0.125*( -PO(IF2,J,K)+6.0*PO(IF,J,K)+3.0*PO(IW,J,K) )
     .          - DLTT*DLTFLX(I,J,K)/VOL(I,J,K-1)
V  13 CONTINUE
           :
           :
```

A－2

```
      :
      :
      I       = 1
      DO 1000 II = 1 , 2
S     DO 1010 K  = 2 , NZP
      DO 1010 J  = 1 , NT
      DENS       = RA(I,J,K)
      WRI        = 1.0 / DENS
      WRU        = WRI*UA(I,J,K)
      WRV        = WRI*VA(I,J,K)
      WRW        = WRI*WA(I,J,K)
      VLO2       = WRU**2 + WRV**2 + WRW**2
      TMP        = TMPTN - AKAP1H*VLO2
      QQ         = VLO2
      IF( TMP .LE. 0.0  )  THEN
       IF( DENS.LT. 0.20 )  DENS = 0.25
       TMPA      = TMPTN
       IF( DENS.GE. 1.00 )  DENS = 0.99
       DO 1220 KK = 1 , 50
       PRS       = AKAPI*DENS*TMPA
       QQ        = 2.0*AKAP1H*TMPA*( ( PRSTN/PRS )**AKAPEI - 1.0 )
       TMPB      = TMPTN - AKAP1H*QQ
        IF( ABS( TMPA-TMPB ).LT.1.0E-5 )  THEN
          TMP      = TMPB
          GO  TO  1225
        ENDIF
       TMPA      = TMPB
 1220  CONTINUE

       TMP       = TMPTN*0.4
       GO  TO  1230
 1225  CONTINUE
       RAV       = SQRT( QQ / VLO2 )
       WRU       = RAV*WRU
       WRV       = RAV*WRV
       WRW       = RAV*WRW
      ENDIF
 1230 CONTINUE
      PRS        = AKAPI*DENS*TMP
      IF( PRS.GT.PRSTN ) GO TO 1050
      WORK       = ( PRSTN / PRS )**AKAPEI
      TMP        = TMPTN / WORK
      DENS       = AKAPA*PRS / TMP
      SPD        = DENS*SQRT( TMP*( WORK-1.0 )/AKAP1H )
     .                / SQRT( WRU**2 + WRV**2 + WRW**2 )
      RA(I,J,K) = DENS
      UA(I,J,K) = SPD*WRU
      VA(I,J,K) = SPD*WRV
      WA(I,J,K) = SPD*WRW
      GO  TO  1010
 1050 CONTINUE
      WORK       = ( TMPTN / TMP )**AKAPE
      PRS        = PRSTN / WORK
      DENS       = AKAPA*PRS / TMP
      UA(I,J,K) = DENS*WRU
      VA(I,J,K) = DENS*WRV
      WA(I,J,K) = DENS*WRW
 1010 CONTINUE
      I = NR
 1000 CONTINUE
      :
      :
```



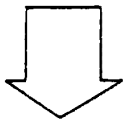
```
      DIMENSION DENS(25,20,75),wRI (25,20,75),wRU (25,20,75)
     .         ,WRV (25,20,75),wRw (25,20,75),VLO2(25,20,75)
     .         ,TMP (25,20,75),QQ  (25,20,75),PRS (25,20,75)
     .         ,TMPA(25,20,75),TMPB(25,20,75),RAV (25,20,75)
           :
           :
      I       = 1
      DO 1000 II = 1 , 2
S     DO 1010 K  = 2 , NZP
V     DO 1010 J  = 1 , NT
V     DENS(I,J,K) = RA(I,J,K)
V     wRI (I,J,K) = 1.0 / DENS(I,J,K)
V     wRU (I,J,K) = wRI(I,J,K)*UA(I,J,K)
V     wRV (I,J,K) = wRI(I,J,K)*VA(I,J,K)
V     wRW (I,J,K) = wRI(I,J,K)*wA(I,J,K)
V     VLO2(I,J,K) = wRU(I,J,K)**2 + wRV(I,J,K)**2 + WRw(I,J,K)**2
V     TMP (I,J,K) = TMPTN - AKAP1H*VLO2(I,J,K)
V     QQ  (I,J,K) = VLO2(I,J,K)
V1010 CONTINUE
S     DO 1011 K  = 2 , NZP
      DO 1011 J  = 1 , NT
      IF( TMP(I,J,K).LE. 0.0  )  THEN
       IF( DENS(I,J,K).LT. 0.20 )  DENS(I,J,K) = 0.25
       TMPA(I,J,K)       = TMPTN
       IF( DENS(I,J,K).GE. 1.00 )  DENS(I,J,K) = 0.99
       DO 1220 KK = 1 , 50
       PRS (I,J,K) = AKAP1*DENS(I,J,K)*TMPA(I,J,K)
       QQ  (I,J,K) = 2.0*AKAP1H*TMPA*(I,J,K)
     .                          *( ( PRSTN/PRS(I,J,K) )**AKAPE1 - 1.0 )
       TMPB(I,J,K) = TMPTN - AKAP1H*QQ(I,J,K)
        IF( ABS( TMPA(I,J,K)-TMPB(I,J,K) ).LT.1.0E-5 )  THEN
          TMP(I,J,K)       = TMPB(I,J,K)
          GO  TO  1225
        ENDIF
       TMPA(I,J,K)      = TMPB(I,J,K)
 1220  CONTINUE

       TMP (I,J,K)      = TMPTN*0.4
       GO  TO  1230
 1225  CONTINUE
       RAV(I,J,K) = SQRT( QQ(I,J,K) / VLO2(I,J,K) )
       WRU(I,J,K) = RAV(I,J,K)*WRU(I,J,K)
       WRV(I,J,K) = RAV(I,J,K)*WRV(I,J,K)
       WRw(I,J,K) = RAV(I,J,K)*WRw(I,J,K)
      ENDIF
 1230 CONTINUE
 1011 CONTINUE
S     DO 1012 K  = 2 , NZP
V     DO 1012 J  = 1 , NT
V     PRS(I,J,K) = AKAP1*DENS(I,J,K)*TMP(I,J,K)
V     IF( PRS(I,J,K).GT.PRSTN )  THEN
V     WORK         = ( TMPTN / TMP(I,J,K) )**AKAPE
V     PRSX         = PRSTN / WORK
```

```
V       DENSX          = AKAPA*PRSX / TMP(I,J,K)
V       UA   (I,J,K)   = DENSX*wRU(I,J,K)
V       VA   (I,J,K)   = DENSX*wRV(I,J,K)
V       WA   (I,J,K)   = DENSX*wRw(I,J,K)
V       ELSE
V       WORK           = ( PRSTN / PRS(I,J,K) )**AKAPEI
V       TMPX           = TMPTN / WORK
V       DENSX          = AKAPA*PRS(I,J,K) / TMPX
V       SPD            = DENSX*SQRT( TMPX*( WORK-1.0 )/AKAP1H )
     .                 / SQRT( WRU(I,J,K)**2 + WRV(I,J,K)**2 + wRw(I,J,K)**2 )
V       RA(I,J,K)  = DENSX
V       UA(I,J,K)  = SPD*wRU(I,J,K)
V       VA(I,J,K)  = SPD*wRV(I,J,K)
V       WA(I,J,K)  = SPD*wRw(I,J,K)
V       ENDIF
V  1012 CONTINUE
        I = NR
  1000  CONTINUE
            :
            :
```

A－3

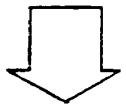
```
                :
                :
          IBF      = 1
S         DO 300 J = 1 , NT
S         DO 300 I = 1 , NR
V         DO 310 K = 1 , NZP
V         WK1( K ) = UA(I,J,K)*UR(I,J,K)
V         WK2( K ) = UA(I,J,K)*UT(I,J,K)
V         WK3( K ) = UA(I,J,K)*UZ(I,J,K)
V         BF(I,J,K) = ( PRS(I,J,K) + RA(I,J,K)*UT(I,J,K)**2 )/R(I,J,K)
V   310 CONTINUE
V         DO 320 K = 2 , NZ
V         FX(I,J,K) = COEFM(K,1)*WK1(K-1) + COEFM(K,2)*WK1( K )
          .                 + COEFM(K,3)*WK1(K+1)
V         FY(I,J,K) = COEFM(K,1)*WK2(K-1) + COEFM(K,2)*WK2( K )
          .                 + COEFM(K,3)*WK2(K+1)
V         FZ(I,J,K) = COEFM(K,1)*WK3(K-1) + COEFM(K,2)*WK3( K )
          .                 + COEFM(K,3)*WK3(K+1)
V   320 CONTINUE
S         FX(I,J, 1 ) = WK1( 1 )
S         FX(I,J,NZP) = WK1(NZP)
S         FY(I,J, 1 ) = WK2( 1 )
S         FY(I,J,NZP) = WK2(NZP)
S         FZ(I,J, 1 ) = WK3( 1 )
S         FZ(I,J,NZP) = WK3(NZP)
S   300 CONTINUE
                :
                :
```



```
                :
                :
          IBF      = 1
S         DO 300 J = 1 , NT
S         DO 300 I = 1 , NR
V         DO 310 K = 1 , NZP
V         WK1(  K  ) = UA(I,J,K)*UR(I,J,K)
V         WK2(  K  ) = UA(I,J,K)*UT(I,J,K)
V         WK3(  K  ) = UA(I,J,K)*UZ(I,J,K)
V         BF (I,J,K) = ( PRS(I,J,K) + RA(I,J,K)*UT(I,J,K)**2 )/R(I,J,K)
V   310 CONTINUE
V         DO 320 K = 2 , NZ
V         FX(I,J,K) = COEFM(K,1)*WK1(K-1) + COEFM(K,2)*WK1( K )
          .                 + COEFM(K,3)*WK1(K+1)
V         FY(I,J,K) = COEFM(K,1)*WK2(K-1) + COEFM(K,2)*WK2( K )
          .                 + COEFM(K,3)*WK2(K+1)
V         FZ(I,J,K) = COEFM(K,1)*WK3(K-1) + COEFM(K,2)*WK3( K )
          .                 + COEFM(K,3)*WK3(K+1)
V   320 CONTINUE
S   300 CONTINUE
V         DO 301 J = 1 , NT
V         DO 301 I = 1 , NR
V         FX(I,J, 1 ) = WK1( 1 )
V         FX(I,J,NZP) = WK1(NZP)
V         FY(I,J, 1 ) = WK2( 1 )
V         FY(I,J,NZP) = WK2(NZP)
V         FZ(I,J, 1 ) = WK3( 1 )
V         FZ(I,J,NZP) = WK3(NZP)
V   301 CONTINUE
                :
                :
```

A－4

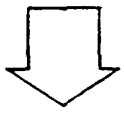
```
                :
                :
S       DO 110 K = 1 , NZP
S       DO 110 J = 1 , NT
V       DO 110 I = 1 , NR
V       WKRI       = 1.0 / RA(I,J,K)
V       UR(I,J,K) = WKRI*UA(I,J,K)
V       UT(I,J,K) = WKRI*VA(I,J,K)
V       UZ(I,J,K) = WKRI*WA(I,J,K)
V   110 CONTINUE
                :
                :
```



```
        DIMENSION MSD110(25,20,75)
                :
        DO 10 I = 1 , 37500
        MSD110(I,1,1) = 0
     10 CONTINUE
        DO 20 K = 1 , NZP
        DO 20 J = 1 , NT
        DO 20 I = 1 , NR
        MSD110(I,J,K) = 1
     20 CONTINUE
                :
                :
V       DO 110 I = 1 , 37500
V       IF( MSD110(I,1,1).EQ.0 ) GO TO 110
V       WKRI       = 1.0 / RA(I,1,1)
V       UR(I,1,1) = WKRI*UA(I,1,1)
V       UT(I,1,1) = WKRI*VA(I,1,1)
V       UZ(I,1,1) = WKRI*WA(I,1,1)
V   110 CONTINUE
                :
                :
```

A－5

```
                :
                :
S       DO 100 K = 1 , NZP
S       DO 100 J = 1 , NT
S       DO 100 I = 1 , NR
S       IF( RA(I,J,K).GT.0.0 ) GO TO 100
S       WRITE(6,'(2X,''DENSITY BECOMES NEGATIVE'',5X,''R('',I2
     .            '','',I2,'','',I3,'')='',E11.4,5X,''KAISU='',I5)')
     .             I,J,K,RA(I,J,K),KAISU
S       IF( J.EQ. 1 ) RA(I,J,K) = RA(I,J+1,K)
S       IF( J.EQ.NT ) RA(I,J,K) = RA(I,J-1,K)
S  100 CONTINUE
                :
                :
```

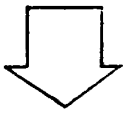


```
        DIMENSION MSD999(25,20,75)
                :
V       DO 10 I = 1 , 37500
V       MSJ999(I,1,1) = 0
V    10 CONTINUE
S       DO 20 K = 1 , NZP
S       DO 20 J = 1 , NT
V       DO 20 I = 1 , NR
V       MSJ999(I,J,K) = 1
V    20 CONTINUE
                :
                :
        I11F00  = 0
V       DO 9990 = 1 , 37500
V       IF( MSJ999(I,1,1).EQ.0 ) GO TO 9990
 *VOCL STMT,IF(100)
V       IF( RA(I,1,1).LE.0.0 ) I11F00 = I11F00 + 1
V 9990 CONTINUE
        IF( I11F00.EQ.0 ) GO TO 9991

S       DO 100 K = 1 , NZP
S       DO 100 J = 1 , NT
S       DO 100 I = 1 , NR
S       IF( RA(I,J,K).GT.0.0 ) GO TO 100
S       WRITE(6,'(2X,''DENSITY BECOMES NEGATIVE'',5X,''R('',I2
     .            '','',I2,'','',I3,'')='',E11.4,5X,''KAISU='',I5)')
     .             I,J,K,RA(I,J,K),KAISU
S       IF( J.EQ. 1 ) RA(I,J,K) = RA(I,J+1,K)
S       IF( J.EQ.NT ) RA(I,J,K) = RA(I,J-1,K)
S  100 CONTINUE
  9991 CONTINUE
                :
                :
```

A－6

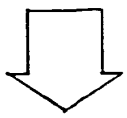
```
                :
                :
S       DO 150 K = 2 , NZ
V       DO 150 I = 1 , 500
V       IF( MSO150(I,1,K).EQ.0 ) GO TO 150
V       FX(I,1,K) = COEFM(K,1)*UA(I,1,K-1) + COEFM(K,2)*UA(I,1,K)
       .          + COEFM(K,3)*UA(I,1,K+1)
V       FY(I,1,K) = COEFM(K,1)*VA(I,1,K-1) + COEFM(K,2)*VA(I,1,K)
       .          + COEFM(K,3)*VA(I,1,K+1)
V       FZ(I,1,K) = COEFM(K,1)*WA(I,1,K-1) + COEFM(K,2)*WA(I,1,K)
       .          + COEFM(K,3)*WA(I,1,K+1)
V   150 CONTINUE
                :
                :
```



```
        DIMENSION COEFM2(25,20,75,3)
                :
V       DO 88 K = 2 , 66
S       DO 88 L = 1 , 3
V       DO 88 I = 1 , 500
V       COEFM2(I,1,K,L) = COEFM(K,L)
V    88 CONTINUE
                :
                :
V       DO 150 I = 1 , 500
V       IF( MSO150(I,1,1).EQ.0 ) GO TO 150
V       FX(I,1,K) = COEFM2(I,1,1,1)*UA(I,1,K-1)+COEFM2(I,1,1,2)*UA(I,1,K)
       .          + COEFM2(I,1,1,3)*UA(I,1,K+1)
V       FY(I,1,K) = COEFM2(I,1,1,1)*VA(I,1,K-1)+COEFM2(I,1,1,2)*VA(I,1,K)
       .          + COEFM2(I,1,1,3)*VA(I,1,K+1)
V       FZ(I,1,K) = COEFM2(I,1,1,1)*WA(I,1,K-1)+COEFM2(I,1,1,2)*WA(I,1,K)
       .          + COEFM2(I,1,1,3)*WA(I,1,K+1)
V   150 CONTINUE
                :
                :
```

A－7

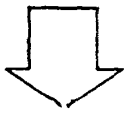
```
                :
                :
S       DO 111 J = 1 , 19
S       DO 111 I = 1 , 21
V       DO 112 K = 1 , 67
V       VLO2(I,J,K) = UR(I,J,K)**2 + UT(I,J,K)**2 + UZ(I,J,K)**2
V       WK1 (I,J,K) = AKAPI*RA(I,J,K)*( TMPTN - AKAP1H*VLO2(I,J,K) )
V   112 CONTINUE
S       DO 114 K = 1 , 67
S       IF( WK1(I,J,K).LE.0.0 )  THEN
S         DENS = RA(I,J,K)
S       IF( DENS.LT.0.15 )  DENS = 0.15
S         TMPA = TMPTN
        DO 90 JJ = 1 , 50
        PR       = AKAPI*DENS*TMPA
        QQ       = 2.0*AKAP1I*TMPA*(( PRSTN/PR )**AKAPEI - 1.0 )
        TMPB     = TMPTN - AKAP1H*QQ
        IF( ABS( TMPA-TMPB ).LT.1.0E-5 ) THEN
          TMP          = TMPB
          RAV          = SQRT( QQ/VLO2(I,J,K) )
          UR (I,J,K) = RAV*UR(I,J,K)
          UT (I,J,K) = RAV*UT(I,J,K)
          UZ (I,J,K) = RAV*UZ(I,J,K)
          WK1(I,J,K) = PR
        ENDIF
          TMPA       = TMPB
     90 CONTINUE
S         WK1(I,J,K) = WK1(I,J,K-1)
S       ENDIF
S   114 CONTINUE
                :
                :
```



```
            :
            :
S       DO 111 J = 1 , 19
S       DO 111 I = 1 , 21
S       I1IF0    = 0
 *VOCL LOOP,REPEAT(67)
V       DO 112 K = 1 , 67
V       VLO2(I,J,K) = UR(I,J,K)**2 + UT(I,J,K)**2 + UZ(I,J,K)**2
V       WK1 (I,J,K) = AKAP1*RA(I,J,K)*( TMPTN - AKAP1H*VLO2(I,J,K) )
V       IF( WA1(I,J,K).LE.0.0 ) I1IF00 = I1IF00 + 1
V  112 CONTINUE
S       IF( I1IF00.EQ.0 ) GO TO 9994
S       DO 114 K = 1 , 67
S       IF( WK1(I,J,K).LE.0.0 )  THEN
S         DENS = RA(I,J,K)
S       IF( DENS.LT.0.15 )  DENS = 0.15
S         TMPA = TMPTN
        DO 90 JJ = 1 , 50
        PR       = AKAP1*DENS*TMPA
        QQ       = 2.0*AKAP11*TMPA*(( PRSTN/PR )**AKAPEI - 1.0 )
        TMPB     = TMPTN - AKAP1H*QQ
        IF( ABS( TMPA-TMPB ).LT.1.0E-5 ) THEN
          TMP          = TMPB
          RAV          = SQRT( QQ/VLO2(I,J,K) )
          UR (I,J,K) = RAV*UR(I,J,K)
          UT (I,J,K) = RAV*UT(I,J,K)
          UZ (I,J,K) = RAV*UZ(I,J,K)
          WK1(I,J,K) = PR
        ENDIF
          TMPA         = TMPB
    90 CONTINUE
S         WK1(I,J,K) = WK1(I,J,K-1)
S       ENDIF
S  114 CONTINUE
  9994 CONTINUE
            :
            :
```

A－8

```
            :
            :
        DO 111 J = 1 , NT
            :
            :
 *VOCL LOOP,REPEAT(65)
M       DO 113 K = 2 , 66
V       DRUCK = COEFP(K,1)*WK11(I,J,K-1)
       .      + COEFP(K,2)*WK11(I,J, K )
       .      + COEFP(K,3)*WK11(I,J,K+1)
 *VOCL STMT,IF(99)
V       IF( DRUCK.GT.0.0 ) THEN
V         PRS(I,J,K) = DRUCK
V       ELSE
 *VOCL STMT,IF(100)
M         IF( J.EQ. 1 ) PRS(I,J,K) = PRS(I, J ,K-1)
 *VOCL STMT,IF(0)
M         IF( J.EQ.NT ) PRS(I,J,K) = PRS(I,J-1, K )
M         PRS(I,J,K) = PRS(I,J-1, K )
V       ENDIF
V  113 CONTINUE
            :
            :
   111 CONTINUE
            :
            :
```



```
            :
            :
        DO 111 J = 1 , NT
            :
            :
 *VOCL LOOP,REPEAT(65)
V       DO 113 K = 2 , 66
V       DRUCK = COEFP(K,1)*WK11(I,J,K-1)
       .      + COEFP(K,2)*WK11(I,J, K )
       .      + COEFP(K,3)*WK11(I,J,K+1)
 *VOCL STMT,IF(99)
V       IF( DRUCK.GT.0.0 ) THEN
V         PRS(I,J,K) = DRUCK
V       ELSE
 COMMENT**************************************************
 C*VOCL STMT,IF(100)                                      *
 C         IF( J.EQ. 1 ) PRS(I,J,K) = PRS(I, J ,K-1)      *
 C*VOCL STMT,IF(0)                                        *
 C         IF( J.EQ.NT ) PRS(I,J,K) = PRS(I,J-1, K )      *
 COMMENT**************************************************
V         PRS(I,J,K) = PRS(I,J-1, K )
V       ENDIF
V  113 CONTINUE
            :
            :
   111 CONTINUE
            :
            :
```

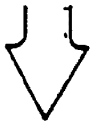
資料 B

B－1

チューニング1

```
      DO 2 J=JID,JFD
                :
                :
                :
                :
                :
```

```
      IF(W) 210,200,200
  200 DD=-2.0*W*H3K*DTC
      GO TO 201
  210 DD=2.0*W*H3K*DTC
  201 AAS=-(ABQ-1.0+AMU)*U2*DTX2*FI*(FIP+FIM)-AMUQ*UM2Q*DTX2*F(I-1,K)*
     1FIM-ABQ*(U2+V2)*DTC2*H3K*(H3KP+H3KM)+BO(JJ)+DD
      BBS=FO(JJ)-AO(JJ)*PH(IL,JL-1,KL,1)-CO(JJ)*PH(IL,JL+1,KL,1)
      PH(IL,JL,KL,1)=BBS/AAS
    2 CONTINUE
```



```
      DIMENSION W(61),ABQ(61),AMU(61),U2(61),V2(61),AMUQ(61),UM2Q(61)
                :
                :
                :
                :
                :
      DO 2 J=JID,JFD
                :
                :
                :
                :
                :
    2 CONTINUE
```

```
      DO 9990 J=JID,JFD
         JL   = J-JO+1
         JJ   = J-JID+1
         DD   = -2.0*ABS(W(J))*H3K*DTC
         AAS  = -(ABQ(J)-1.0+AMU(J))*U2(J)*DTX2*FI*(FIP+FIM)
     -          -AMUQ(J)*UM2Q(J)*DTX2*F(I-1,K)*
     -          FIM-ABQ(J)*(U2(J)+V2(J))*DTC2*H3K*(H3KP+H3KM)+BO(JJ)+DD
         BBS  = FO(JJ)-AO(JJ)*PH(IL,JL-1,KL,1)-CO(JJ)*PH(IL,JL+1,KL,1)
         PH(IL,JL,KL,1)=BBS/AAS
 9990 CONTINUE
```

B－2－1

チューニング 2

関数 FTD と FTDC は□□部分のみ異なる。

```
      FUNCTION FTD(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 FTD=-AK*THO*COS(AK*T)*((X-APX)*COS(RAMDP)-Z*SIN(RAMDP))
      GO TO 4
    2 FTD=-AK*THO*COS(AK*T)
      GO TO 4
    3 IF(MM-3) 5,5,6
    5 FTD=THO*AK*COS(AK*T)*(Z+APR)
      GO TO 4
    6 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 7
      FTD=0.0
      GO TO 4
    7 XH=Z*TAN(RAMDH)+APH
      IF(X.GE.XH) GO TO 8
      IF( MM .EQ. 4 ) FTD=0.0
      IF( MM .EQ. 5 )
     1FTD=-((X-APH)*COS(RAMDH)-Z*SIN(RAMDH))*THO *AK*COS(AK*T)
      GO TO 4
    8 IF( MM .EQ. 4 )
     1FTD=-((X-APH)*COS(RAMDH)-Z*SIN(RAMDH))*THO *AK*COS(AK*T)
      IF( MM .EQ. 5 ) FTD=0.0
    4 RETURN
      END

      FUNCTION FTDC(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      COMMON /ACTGST/IGUST,IACT,DELF(3)
      COMMON /GRID3/DXAI,DET,DCH,DTX,DTX2,DTE,DTE2,DTC,DTC2,DTXE,DTXC,
     1DTEC,DT
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 FTDC=-AK*THO*COS(AK*T)*((X-APX)*COS(RAMDP)-Z*SIN(RAMDP))
      GO TO 4
    2 FTDC=-AK*THO*COS(AK*T)
      GO TO 4
    3 IF(MM-3) 5,5,6
    5 FTDC=THO*AK*COS(AK*T)*(Z+APR)
      GO TO 4
    6 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 7
      FTDC=0.0
      GO TO 4
    7 XH=Z*TAN(RAMDH)+APH
      DELDT=(DELF(1)-DELF(3))/(2.0*DT)
      IF(DELDT.LT.1.0E-5) DELDT=0.0
      IF(X.GE.XH) GO TO 8
      IF( MM .EQ. 4 ) FTDC=0.0
      IF( MM .EQ. 5 )
     1FTDC=-((X-APH)*COS(RAMDH)-Z*SIN(RAMDH))*DELDT
      GO TO 4
    8 IF( MM .EQ. 4 )
     1FTDC=-((X-APH)*COS(RAMDH)-Z*SIN(RAMDH))*DELDT
      IF( MM .EQ. 5 ) FTDC=0.0
    4 RETURN
      END
```

B－2－2

関数FTXとFTXC

```
      FUNCTION FTX(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      COMMON /CONIDF/CONDO
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 IF(AK-1.0E-4) 5,5,6
    5 FTX=-THO
      GO TO 4
    6 FTX=-THO*SIN(AK*T)*COS(RAMDP)
      GO TO 4
    2 FTX=0.0
      GO TO 4
    3 IF(MM-3) 7,7,8
    7 FTX=0.0
      GO TO 4
    8 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 9
      FTX=0.0
      GO TO 4
    9 XH=Z*TAN(RAMDH)+APH
      IF(X.GE.XH) GO TO 10
      IF( MM .EQ. 4 ) FTX=0.0
      IF( MM .EQ. 5 )
     1FTX=-COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
      GO TO 4
   10 IF( MM .EQ. 4 )
     1FTX=-COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
      IF( MM .EQ. 5 ) FTX=0.0
    4 RETURN
      END
      FUNCTION FTXC(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      COMMON /ACTGST/IGUST,IACT,DELF(3)
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 IF(AK-1.0E-4) 5,5,6
    5 FTXC=-THO
      GO TO 4
    6 FTXC=-THO*SIN(AK*T)*COS(RAMDP)
      GO TO 4
    2 FTXC=0.0
      GO TO 4
    3 IF(MM-3) 7,7,8
    7 FTXC=0.0
      GO TO 4
    8 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 9
      FTXC=0.0
      GO TO 4
    9 XH=Z*TAN(RAMDH)+APH
      IF(X.GE.XH) GO TO 10
      IF( MM .EQ. 4 ) FTXC=0.0
      IF( MM .EQ. 5 )
     1FTXC=-COS(RAMDH)*DELF(1)
      GO TO 4
   10 IF( MM .EQ. 4 )
     1FTXC=-COS(RAMDH)*DELF(1)
      IF( MM .EQ. 5 ) FTXC=0.0
    4 RETURN
      END
```

B－2－3

関数FTZとFTZC

```
      FUNCTION FTZ(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      COMMON /CONIDF/CONDO
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 FTZ=THO*SIN(AK*T)*SIN(RAMDP)
      GO TO 4
    2 FTZ=0.0
      GO TO 4
    3 IF(MM-3) 5,5,6
    5 FTZ=THO*SIN(AK*T)
      GO TO 4
    6 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 7
      FTZ=0.0
      GO TO 4
    7 XH=Z*TAN(RAMDH)+APH
      IF(X.GE.XH) GO TO 8
      IF( MM .EQ. 4 ) FTZ=0.0
      IF( MM .EQ. 5 )
     1FTZ= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
      GO TO 4
    8 IF( MM .EQ. 4 )
     1FTZ= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
      IF( MM .EQ. 5 ) FTZ=0.0
    4 RETURN
      END

      FUNCTION FTZC(X,Z,T)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /AIRF/TAU,AK,THO,APX,APR,APH,RAMDP,RAMDH,ZH1,ZH2,MM
      COMMON /PLANA/S,RAMD,SRAM
      COMMON /ACTGST/IGUST,IACT,DELF(3)
      ZH1D=S*ZH1
      ZH2D=S*ZH2
      IF(MM-2) 1,2,3
    1 FTZC=THO*SIN(AK*T)*SIN(RAMDP)
      GO TO 4
    2 FTZC=0.0
      GO TO 4
    3 IF(MM-3) 5,5,6
    5 FTZC=THO*SIN(AK*T)
      GO TO 4
    6 IF(Z.GE.ZH1D .AND. Z.LE.ZH2D) GO TO 7
      FTZC=0.0
      GO TO 4
    7 XH=Z*TAN(RAMDH)+APH
      IF(X.GE.XH) GO TO 8
      IF( MM .EQ. 4 ) FTZC=0.0
      IF( MM .EQ. 5 )
     1FTZC= SIN(RAMDH)*DELF(1)
      GO TO 4
    8 IF( MM .EQ. 4 )
     1FTZC= SIN(RAMDH)*DELF(1)
      IF( MM .EQ. 5 ) FTZC=0.0
    4 RETURN
      END
```

B－2－4

関数FTD, FTDC, FTX, FTXC, FTZおよびFTZCを統合したプログラム

```
      XIK = X( I, K )
      ZK  = Z( K )
      IF( MM .LE. 1 )  THEN
            FTDD= -AK*THO*COS(AK*T)*((XIK-APX)*COS(RAMDP)
     1                                          -ZK*SIN(RAMDP))
        IF( AK .LE. 1.0E-4 )  THEN
            FTXD= -THO
        ELSE
            FTXD= -THO*SIN(AK*T)*COS(RAMDP)
        ENDIF
            FTZD= THO*SIN(AK*T)*SIN(RAMDP)
      ELSEIF( MM .EQ. 2 )  THEN
            FTDD= -AK*THO*COS(AK*T)
            FTXD= 0.0
            FTZD= 0.0
      ELSEIF( MM .EQ. 3 )  THEN
            FTDD= THO*AK*COS(AK*T)*(ZK+APR)
            FTXD= 0.0
            FTZD= THO*SIN(AK*T)
      ELSEIF( MM .GE. 4 )  THEN
        ZH1D = S * ZH1
        ZH2D = S * ZH2
        IF( ZH1D .LE. ZK .AND. ZK .LE. ZH2D )  THEN
          XH = ZK * TAN( RAMDH ) + APH
          IF( XIK .GE. XH )  THEN
            IF( MM .EQ. 4 )  THEN
              IF( IFLUT .LE. 1 )  THEN
                DELDT = ( DELF(1)-DELF(3))/(2.0*DT)
                IF( DELDT .LT. 1.0E-5 )  DELDT = 0.0
                FTDD= -((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))*DELDT
                FTXD= -COS(RAMDH)*DELF(1)
                FTZD= SIN(RAMDH)*DELF(1)
```

B－2－5

```
          ELSE
            FTDD=-((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))
     1                                              *THO*AK*COS(AK*T)
            FTXD= -COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
            FTZD= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
          ENDIF
        ELSEIF( MM .EQ. 5 )  THEN
          FTDD= 0.0
          FTXD= 0.0
          FTZD= 0.0
        ENDIF
      ELSE
        IF( MM .EQ. 4 ) THEN
          FTDD= 0.0
          FTXD= 0.0
          FTZD= 0.0
        ELSEIF( MM .EQ. 5 )  THEN
          IF( IFLUT .LE. 0 )  THEN
            DELDT = ( DELF(1)-DELF(3))/(2.0*DT)
            IF( DELDT .LT. 1.0E-5 )  DELDT = 0.0
            FTDD= -((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))*DELDT
            FTXD= -COS(RAMDH)*DELF(1)
            FTZD= SIN(RAMDH)*DELF(1)
          ELSE
            FTDD=-((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))
     1                                              *THO*AK*COS(AK*T)
            FTXD= -COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
            FTZD= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
          ENDIF
        ENDIF
      ENDIF
    ELSE
          FTDD= 0.0
          FTXD= 0.0
          FTZD= 0.0
    ENDIF
  ENDIF
```

B－2－6

オリジナルプログラムの変数Bの計算式

```
      IF(IFLUT-1) 5000,5000,5001
5000  IF(IGUST-1) 5500,5501,5500
5501  IF(IACTD-1) 5502,5503,5502
5503  XIK=X(I,K)
      ZK=Z(K)
      FTXD=FTXC(XIK,ZK,T)
      FTDD=FTDC(XIK,ZK,T)
      FTZD=FTZC(XIK,ZK,T)
      B=-DET2*SIAL+(0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2
     1))+(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)-PH2(IL+1,
     2L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))+DET2*COAL)*
     3(FSXU(IL,K)+THT(K)+DWX(IL-2,K-2)+FTXD)+(H1(I,K)*
     4PHXB+H3(K)*PHCB)*(FSZU(IL,K)+DWZ(IL-2,K-2)+FTZD)
     5+DET2*(DWT(IL-2,K-2)+FTDD-VG(XIK,ZK,T))
      GO TO 5002
5502  XIK=X(I,K)
      ZK=Z(K)
      B=-DET2*SIAL+(0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2
     1))+(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)-PH2(IL+1,
     2L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))+DET2*COAL)*
     3(FSXU(IL,K)+THT(K)+DWX(IL-2,K-2))+(H1(I,K)*
     4PHXB+H3(K)*PHCB)*(FSZU(IL,K)+DWZ(IL-2,K-2))
     5+DET2*(DWT(IL-2,K-2)-VG(XIK,ZK,T))
      GO TO 5002
5500  IF(IACTD-1) 5504,5505,5504
5505  XIK=X(I,K)
      ZK=Z(K)
      FTXD=FTXC(XIK,ZK,T)
      FTDD=FTDC(XIK,ZK,T)
      FTZD=FTZC(XIK,ZK,T)
      B=-DET2*SIAL+(0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2
     1))+(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)-PH2(IL+1,
     2L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))+DET2*COAL)*
     3(FSXU(IL,K)+THT(K)+DWX(IL-2,K-2)+FTXD)+(H1(I,K)*
     4PHXB+H3(K)*PHCB)*(FSZU(IL,K)+DWZ(IL-2,K-2)+FTZD)
     5+DET2*(DWT(IL-2,K-2)+FTDD)
      GO TO 5002
5504  B=-DET2*SIAL+(0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2
     1))+(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)-PH2(IL+1,
     2L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))+DET2*COAL)*
     3(FSXU(IL,K)+THT(K)+DWX(IL-2,K-2))+DET2*DWT(IL-2,K-2)+(H1(I,K)*
     4PHXB+H3(K)*PHCB)*(FSZU(IL,K)+DWZ(IL-2,K-2))
      GO TO 5002
5001  CONTINUE
      XIK=X(I,K)
      ZK=Z(K)
      FTXD=FTX(XIK,ZK,T)
      FTDD=FTD(XIK,ZK,T)
      FTZD=FTZ(XIK,ZK,T)
      B=-DET2*SIAL+(0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2
     1))+(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)-PH2(IL+1,
     2L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))+DET2*COAL)*
     3(FSXU(IL,K)+FTXD+THT(K))+DET2*FTDD
     4+(FSZU(IL,K)+FTZD)*(H1(I,K)*PHXB+H3(K)*PHCB)
5002  CONTINUE
```

B－2－7

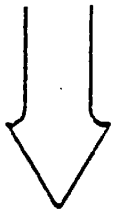
修正後の変数Bの計算式

```
C...CALCULATE CONSTANT TERM
      VPPP1 = -DET2*SIAL
      VPPP2 = 0.5*F(I,K)*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2))
     1          +(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)
     2          -PH2(IL+1,L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))
     3          +DET2*COAL
      VPPP3 = FSXU(IL,K)+THT(K)
      VPPP4 = H1(I,K)*PHXB+H3(K)*PHC3
      VPPP5 = FSZU(IL,K)+DWZ(IL-2,K-2)
C...CALCULATE B IF BLANCH
      IF( IFLUT .LE. 1 )  THEN
        IF( IGUST .EQ. 1 )  THEN
          IF( IACTD .EQ. 1 )  THEN
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2) + FTXD )
     1          + VPPP4 * ( VPPP5 + FTZD )
     2          + DET2 * ( DWT(IL-2,K-2) + FTDD - VG(XIK,ZK,T) )
          ELSE
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2)          )
     1          + VPPP4 *   VPPP5
     2          + DET2 * ( DWT(IL-2,K-2)          - VG(XIK,ZK,T) )
          ENDIF
        ELSE
          IF( IACTD .EQ. 1 )  THEN
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2) + FTXD )
     1          + VPPP4 * ( VPPP5 + FTZD )
     2          + DET2 * ( DWT(IL-2,K-2) + FTDD                  )
          ELSE
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2)          )
     1          + VPPP4 *   VPPP5
     2          + DET2 *   DWT(IL-2,K-2)
          ENDIF
        ENDIF
      ELSE
            B = VPPP1 + VPPP2 * ( VPPP3                    + FTXD )
     1          + VPPP4 * ( FSZU(IL,K) + FTZD )
     2          + DET2 * FTDD
      ENDIF
C*************************************************************************
```

B－3

チューニング3

```
      IF( NFCIN .NE. 0 )                          GO TO 1800
      DO 20 N=1,3
      DO 11 I=1,I1
      DO 11 J=1,J1
      DO 11 K=1,K1
   11 PH1(I,J,K,N)=0.0
      DO 12 I=1,I2
      DO 12 J=1,J2
      DO 12 K=1,K2
      PH2(I,J,K,N)=0.0
   12 PH3(I,J,K,N)=0.0
      DO 14 I=1,I4
      DO 14 J=1,J4
      DO 14 K=1,K4
   14 PH4(I,J,K,N)=0.0
      DO 15 I=1,I5
      DO 15 J=1,J5
      DO 15 K=1,K5
   15 PH5(I,J,K,N)=0.0
      DO 16 I=1,I1
      DO 16 K=1,K1
   16 DPH(I,K,N)=0.0
      DELF(N)=0.0
   20 CONTINUE
      DO 30 I=1, 4
      DO 30 J=1,20
      QQ(J,I)=0.0
   30 CONTINUE
      DO 7500 I=1,4
      DO 7500 N=1,5000
 7500 WSEN(I,N)=0.0
 1800 CONTINUE
```

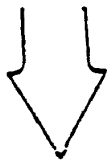


```
      BLOCKDATA  INIBL
*IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON / COMPH /  PH1(18,61,19,3),    PH2(35,36,19,3),
     1                  PH3(35,36,19,3),    PH4(22,61,19,3),
     2                  PH5(67,61,17,3)
      COMMON / COMAER / BQP(20,1000),  QQ(20,4),  NBQP,
     1                  GAFP(20,1000), IBQP(1000)
      COMMON / WAKE /   DPH(22,19,3)
      COMMON / ACTGST / IGUST, IACT, DELF(3)
      COMMON / WSENSR / WSEN(4,5000)
C
      DATA              PH1,                PH2,
     1                  PH3,                PH4,
     2                  PH5
     3       /          62586   * 0.0 ,     71820   * 0.0 ,
     4                  71820   * 0.0 ,     76494   * 0.0 ,
     5                  208437  * 0.0                      /
      DATA              QQ       /    80 * 0.0  /
      DATA              DPH      /  1254 * 0.0  /
      DATA              DELF     /     3 * 0.0  /
      DATA              WSEN     / 20000 * 0.0  /
      END
```

B－4

チューニング4

```
      SUBROUTINE TRIDQ(A,B,C,F,X,JZ)
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(100),B(100),C(100),F(100),X(100)
      JZ1=JZ-1
      C(1)=C(1)/B(1)
      F(1)=F(1)/B(1)
      DO 1 J=2,JZ1
      B(J)=B(J)-C(J-1)*A(J)
      F(J)=(F(J)-F(J-1)*A(J))/B(J)
      C(J)=C(J)/B(J)
    1 CONTINUE
      B(JZ)=B(JZ)-C(JZ1)*A(JZ)
      X(JZ)=(F(JZ)-F(JZ1)*A(JZ))/B(JZ)
      DO 2 J=1,JZ1
      JD=JZ-J
      X(JD)=F(JD)-C(JD)*X(JD+1)
    2 CONTINUE
      RETURN
      END
```



```
   81 AO(JT)=AO(JT)+CO(JT)*AZ1(IL)
      BO(JT)=BO(JT)+CO(JT)*AZ2(IL)
      FO(JT)=FO(JT)-CO(JT)*AZ3(IL)
      FO(1)=FO(1)-AO(1)*PH(IL,JID-JO,KL,1)
   82 CONTINUE
C*****************************************************************
C       SUBROUTINE TRIDQ WO TENKAI SURU  HISTORY (1)
C*****************************************************************
C     CALL TRIDQ(AO,BO,CO,FO,XO,JT)
      JTM1 = JT - 1
      CO(1) = CO(1) / BO(1)
      FO(1) = FO(1) / BO(1)
      DO 1110  J = 2, JTM1
        BO(J) = BO(J) - CO(J-1) * AO(J)
        FO(J) = ( FO(J) - FO(J-1) * AO(J) ) / BO(J)
        CO(J) = CO(J) / BO(J)
 1110 CONTINUE
      BO(JT) = BO(JT). - CO(JTM1) * AO(JT)
      XO(JT) = ( FO(JT) - FO(JTM1) * AO(JT) ) / BO(JT)
      DO 1120  J = 1, JTM1
        JD = JT - J
        XO(JD) = FO(JD) - CO(JD) * XO(JD+1)
 1120 CONTINUE
C*****************************************************************
```

TRIDQルーチン展開部

B－5－1

チューニング5

修正前の変数Bの計算式

```
C...CALCULATE CONSTANT TERM
      VPPP1 = -DET2*SIAL
      VPPP2 = 0.5*F(I,K)*DETZX*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2))
     1          +(ETBU(IL,K)-ET(J-1))*0.5*F(I,K)*DETX*(PH2(IL+1,L,K,2)
     2          -PH2(IL+1,L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))
     3          +DET2*COAL
      VPPP3 = FSXU(IL,K)+THT(K)
      VPPP4 = H1(I,K)*PHXB+H3(K)*PHCB
      VPPP5 = FSZU(IL,K)+DWZ(IL-2,K-2)
C...CALCULATE B IF BLANCH
      IF( IFLUT .LE. 1 )  THEN
        IF( IGUST .EQ. 1 )  THEN
          IF( IACTD .EQ. 1 )  THEN
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2) + FTXD )
     1           + VPPP4 * ( VPPP5 + FTZD )
     2           + DET2 * ( DWT(IL-2,K-2) + FTDD - VG(XIK,ZK,T) )
          ELSE
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2)          )
     1           + VPPP4 *   VPPP5
     2           + DET2 * ( DWT(IL-2,K-2)         - VG(XIK,ZK,T) )
          ENDIF
        ELSE
          IF( IACTD .EQ. 1 )  THEN
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2) + FTXD )
     1           + VPPP4 * ( VPPP5 + FTZD )
     2           + DET2 * ( DWT(IL-2,K-2) + FTDD                 )
          ELSE
            B = VPPP1 + VPPP2 * ( VPPP3 + DWX(IL-2,K-2)          )
     1           + VPPP4 *   VPPP5
     2           + DET2 *   DWT(IL-2,K-2)
          ENDIF
        ENDIF
       ELSE
            B = VPPP1 + VPPP2 * ( VPPP3                   + FTXD )
     1           + VPPP4 * ( FSZU(IL,K) + FTZD )
     2           + DET2 * FTDD
       ENDIF
C*****************************************************************
```

B－5－2

修正前の関数ルーチンの計算部分

```
      XIK = X( I, K )
      ZK  = Z( K )
      IF( MM .LE. 1 )  THEN
            FTDD= -AK*THO*COS(AK*T)*((XIK-APX)*COS(RAMDP)
     1                                              -ZK*SIN(RAMDP))
        IF( AK .LE. 1.0E-4 )  THEN
            FTXD= -THO
        ELSE
            FTXD= -THO*SIN(AK*T)*COS(RAMDP)
        ENDIF
            FTZD= THO*SIN(AK*T)*SIN(RAMDP)
      ELSEIF( MM .EQ. 2 )  THEN
            FTDD= -AK*THO*COS(AK*T)
            FTXD= 0.0
            FTZD= 0.0
      ELSEIF( MM .EQ. 3 )  THEN
            FTDD= THO*AK*COS(AK*T)*(ZK+APR)
            FTXD= 0.0
            FTZD= THO*SIN(AK*T)
      ELSEIF( MM .GE. 4 )  THEN
        ZH1D = S * ZH1
        ZH2D = S * ZH2
        IF( ZH1D .LE. ZK .AND. ZK .LE. ZH2D )  THEN
          XH = ZK * TAN( RAMDH ) + APH
          IF( XIK .GE. XH )  THEN
            IF( MM .EQ. 4 )  THEN
              IF( IFLUT .LE. 1 )  THEN
                DELDT = ( DELF(1)-DELF(3))/(2.0*DT)
                IF( DELDT .LT. 1.0E-5 )  DELDT = 0.0
                FTDD= -((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))*DELDT
                FTXD= -COS(RAMDH)*DELF(1)
                FTZD= SIN(RAMDH)*DELF(1)
              ELSE
                FTDD=-((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))
     1                                              *THO*AK*COS(AK*T)
                FTXD= -COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
                FTZD= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
              ENDIF
            ELSEIF( MM .EQ. 5 )  THEN
              FTDD= 0.0
              FTXD= 0.0
              FTZD= 0.0
            ENDIF
          ELSE
            IF( MM .EQ. 4 )  THEN
              FTDD= 0.0
              FTXD= 0.0
              FTZD= 0.0
            ELSEIF( MM .EQ. 5 )  THEN
              IF( IFLUT .LE. 0 )  THEN
                DELDT = ( DELF(1)-DELF(3))/(2.0*DT)
                IF( DELDT .LT. 1.0E-5 )  DELDT = 0.0
                FTDD= -((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))*DELDT
                FTXD= -COS(RAMDH)*DELF(1)
                FTZD= SIN(RAMDH)*DELF(1)
              ELSE
                FTDD=-((XIK-APH)*COS(RAMDH)-ZK*SIN(RAMDH))
     1                                              *THO*AK*COS(AK*T)
                FTXD= -COS(RAMDH)*(THO*SIN(AK*T)+CONDO)
                FTZD= SIN(RAMDH)*(THO*SIN(AK*T)+CONDO)
              ENDIF
            ENDIF
          ENDIF
        ELSE
              FTDD= 0.0
              FTXD= 0.0
              FTZD= 0.0
        ENDIF
      ENDIF
```

B－5－3

○ DO 20 ループより外側での処理

（改造後）

```
      COMMON / AIRF / TAU,AK,THO,APX,APR,APH,RAMOP,RAMOH,ZH1,ZH2,MM
      COMMON / PLANA / S, RAMO, SRAM
      COMMON / CONIOF / CONOO
      DIMENSION     KVPIF( 7, 5 ), VPIF( 7 )
C*****************************************************************
      DIMENSION       JI     (67)    ,JF     (67)
      DATA I1,J1,K1/15,61,19/,I2,J2,K2/35,36,19/,I5,J5,K5/67,61,17/
      DATA I4,J4,K4/22,61,19/
C*****************************************************************
      DATA  KVPIF  /  1,  1,  1,  1,  1,  1,  1,
     1                1,  0,  1,  0,  1,  0,  1,
     2                1,  1,  1,  1,  1,  1,  0,
     3                1,  0,  1,  0,  1,  0,  0,
     4                0,  1,  0,  1,  0,  1,  0     /
C       OUTER LOOP VARIABLE DEFINISION   HISTORY (2)
C
      VPPP1   =  - DET2 * SIAL
      VPPP5   =  0.5 * DET2X
      VPPP6   =  0.5 * DETX
      VPPP7   =  DET2 * COAL
C
      MVPZ4 = 1
      MVPZ5 = 1
      VPTAN = 0.0
C
      IF( MM .EQ. 4 )   THEN
                        MVPZ5   =  0
                        VPTAN   =  TAN( RAMOH )
      ELSEIF( MM .EQ. 5 )   THEN
                        MVPZ4   =  0
                        VPTAN   =  TAN( RAMOH )
      ENDIF
C
      ZH1D   =  ZH1 * S
      ZH2D   =  ZH2 * S
C
      IF( IFLUT .LE. 1 )   THEN
        IF( IGUST .EQ. 1 )   THEN
          IF( IACTD .EQ. 1 )   THEN
             KKKVP   =  1
          ELSE
             KKKVP   =  2
          ENDIF
        ELSE
          IF( IACTD .EQ. 1 )   THEN
             KKKVP   =  3
          ELSE
             KKKVP   =  4
          ENDIF
        ENDIF
      ELSE
             KKKVP   =  5
      ENDIF
C
      DO  1100  I  =  1,  7
        VPIF( I )  =  FLOAT( KVPIF( I, KKKVP ) )
 1100 CONTINUE
```

作業配列

変数Bの計算式の項の条件の設定

DO変数に関与しない定数項

関数計算部での条件による係数の設定

変数Bの計算部での条件式

B－5－4

DO 20 ループと DO 600 ループの間での処理（改造後）関数計算部の各項の係数の設定

```
      IF( MM .LE. 1 )   THEN
            VPFTD1 = - AK * THO * COS( AK * T )
            VPFTD2 =  1.0
            VPFTD3 =  APX
            VPFTD4 =  COS( RAMDP )
            VPFTD5 =  SIN( RAMDP )
        IF( AK .LE. 1.0E-4 )   THEN
            VPFTX1 = - 1.0
            VPFTX2 =   THO
        ELSE
            VPFTX1 = - COS( RAMDP )
            VPFTX2 =  THO * SIN( AK * T )
        ENDIF
            VPFTZ1 = SIN( RAMDP )
            VPFTZ2 = THO * SIN( AK * T )
      ELSEIF( MM .EQ. 2 )   THEN
            VPFTD1 = - AK * THO * COS( AK * T )
            VPFTD2 =  0.0
            VPFTD3 = -1.0
            VPFTD4 =  1.0
            VPFTD5 =  0.0
C
            VPFTX1 =  0.0
            VPFTX2 =  0.0
C
            VPFTZ1 =  0.0
            VPFTX2 =  0.0
      ELSEIF( MM .EQ. 3 )   THEN
            VPFTD1 = - AK * THO * COS( AK * T )
            VPFTD2 =  0.0
            VPFTD3 =  APR
            VPFTD4 =  1.0
            VPFTD5 =  1.0
C
            VPFTX1 =  0.0
            VPFTX2 =  0.0
C
            VPFTZ1 =  1.0
            VPFTZ2 =  THO * SIN( AK * T )
      ELSEIF( MM .GE. 4 )   THEN
        IF( IFLUT .LE. 1 )   THEN
          DELDT = ( DELF(1) - DELF(3) ) / ( 2.0 * DT )
          IF( DELDT .LE. 1.0E-5 )   DELDT = 0.0
            VPFTD1 = - DELDT
            VPFTD2 =  1.0
            VPFTD3 =  APH
            VPFTD4 =  COS( RAMDH )
            VPFTD5 =  SIN( RAMDH )
C
            VPFTX1 = - COS( RAMDH )
            VPFTX2 =  DELF( 1 )
C
            VPFTZ1 =  SIN( RAMDH )
            VPFTZ2 =  DELF( 1 )
        ELSE
            VPFTD1 = - AK * THO * COS( AK * T )
            VPFTD2 =  1.0
            VPFTD3 =  APH
            VPFTD4 =  COS( RAMDH )
            VPFTD5 =  SIN( RAMDH )
C
            VPFTX1 = - COS( RAMDH )
            VPFTX2 =  THO * SIN( AK * T ) + CONDO
C
            VPFTZ1 =  SIN( RAMDH )
            VPFTZ2 =  THO * SIN( AK * T ) + CONDO
        ENDIF
      ENDIF
```

B－5－5

DO21ループとDO26ループの間の処理およびDO26ループの処理（改造後）

```
      DO 25 J=3,JWP5
   25 PH3(IL,J,K,1)=PH1(I,J,K,1)
   23 CONTINUE
C*****************************************************************
C       HISTORY (2)
C*****************************************************************
      ZK   =  Z ( K )
      XH   =  ZK * VPTAN + APH
      IF( ZH1D .GT. ZK .OR. ZK .GT. ZH2D )  THEN
        MVPZ4  =  MVPZ4 * MVPZ5
        MVPZ5  =  MVPZ4
      ENDIF
C*****************************************************************
```

関数計算部での条件式の係数化

B－5－6

```
C       DO 26 I=IE,IT
        DO 1010  I = IE, IT
        IL=I-IE-3
        DO 27 J=JWM5,JZ2
        IF(Y(J).GT.YBU(IL,K)) GO TO 28
     27 CONTINUE
     28 JI(IL)=J
        JF(IL)=JZ2
 1010 CONTINUE
C
        DO 1011  I = IE, IT
        IL = I - IE + 3
        J  = JI( IL )
C*****************************************************************
        L=JI(IL)-JWM5+1
        PHXB=0.5*DET2X*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2))+0.5*DETX*
       1(ETBU(IL,K)-ET(J-1))*(PH2(IL+1,L,K,2)-PH2(IL+1,L-1,K,2)-
       2PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))
        PHCB=0.5*DET2C*(PH2(IL,L-1,K+1,2)-PH2(IL,L-1,K-1,2))+0.5*DETC*
       1(PH2(IL,L,K+1,2)-PH2(IL,L-1,K+1,2)-PH2(IL,L,K-1,2)+
       2PH2(IL,L-1,K-1,2))*(ETBU(IL,K)-ET(J-1))
        A=-1.5*GBU(IL,K)*DET+(ETBU(IL,K)-ET(J-1))*GBU(IL,K)
C       MODIFIED HISTORY (1) (2)
C*****************************************************************
C...COMPUTE FUNCTION TERM
        XIK = X( I, K )
        FTDD = VPFTD1 * (( XIK * VPFTD2 - VPFTD3 ) * VPFTD4
       1                                             - ZK * VPFTD5 )
        FTXD = VPFTX1 * VPFTX2
        FTZD = VPFTZ1 * VPFTZ2
          IF( XIK .GE. XH )  THEN
            MVPZ0  =  MVPZ4
          ELSE
            MVPZ0  =  MVPZ5
          ENDIF
        IF( MVPZ0 .LE. 0 )   THEN
          FTDD  =  0.0
          FTXD  =  0.0
          FTZD  =  0.0
        ENDIF
C...CALCULATE CONSTANT TERM
        VPPP2 = VPPP5*F(I,K)*(PH2(IL+1,L-1,K,2)-PH2(IL-1,L-1,K,2))
       1         +(ETBU(IL,K)-ET(J-1))*VPPP6*F(I,K)*(PH2(IL+1,L,K,2)
       2         -PH2(IL+1,L-1,K,2)-PH2(IL-1,L,K,2)+PH2(IL-1,L-1,K,2))
       3         +VPPP7
        VPPP3 = FSXU(IL,K)+THT(K)
        VPPP4 = H1(I,K)*PHXB+H3(K)*PHCB
C...CALCULATE B IF BLANCH
        B  =  VPPP1 + VPPP2 * ( VPPP3 + VPIF(1) * DWX( IL-2, K-2 )
       1       + VPIF(2) * FTXD ) + VPPP4 * ( FSZU( IL, K )
       2       + VPIF(3) * DWZ( IL-2, K-2 ) + VPIF(4) * FTZD )
       3       + DET2 * ( VPIF(5) * DWT( IL-2, K-2 ) + VPIF(6) * FTDD
       4       - VPIF(7) * VG( XIK, ZK, T ) )
C*****************************************************************
        AZ1(IL)=2.0*GBU(IL,K)*(-DET+ETBU(IL,K)-ET(J-1))/A
        AZ2(IL)=GBU(IL,K)*(0.5*DET-ETBU(IL,K)+ET(J-1))/A
C  26 AZ3(IL)=B/A
        AZ3(IL)=B/A
 1011 CONTINUE
```

DOループ 26 の分割
（DO 1010 および DO 1011）

関数の計算部

変数Bの計算部

B－6

チューニング6

SWEEPおよびSWEP2ルーチン

```
      IF(IRH-4) 9,10,9
   10 IF(J.EQ.JW) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
      IF(J.EQ.JWM1) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
    9 PHXYC=0.25*DTXE*FI*GJ*PHXYD
```

SWEEPおよびSWEP2ルーチン

```
CIRH  IF(IRH-4) 9,10,9
C   10 IF(J.EQ.JW) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
C     IF(J.EQ.JWM1) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
      PHXYC=0.25*DTXE*FI*GJ*PHXYD
```

SWEEP2およびSWEP24ルーチン

```
CIRH  IF(IRH-4) 9,10,9
      IF(J.EQ.JW) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
      IF(J.EQ.JWM1) PHXYD=PHXYD-DPH(IL+1,K,2)+DPH(IL-1,K,2)
      PHXYC=0.25*DTXE*FI*GJ*PHXYD
```

B－7－1

チューニング7

修正前のSWEEP関連ルーチン

```
      KL=K-KO+3
      H3K=H3(K)
      H3KP=0.5*(H3(K+1)+H3(K))
      H3KM=0.5*(H3(K)+H3(K-1))
      H3KPP=0.5*(H3(K+2)+H3(K+1))
      H3KMM=0.5*(H3(K-1)+H3(K-2))
      DO 1 I=I1,I2
      IL=I-IO+3
      JID=JI(IL)
      JFD=JF(IL)
      JT=JFD-JID+1
      FI=F(I,K)
      FIP=0.5*(F(I+1,K)+F(I,K))
      FIM=0.5*(F(I,K)+F(I-1,K))
      FIMM=0.5*(F(I-1,K)+F(I-2,K))
      H1IK=H1(I,K)
      H1IPK=0.5*(H1(I+1,K)+H1(I,K))
      H1IMK=0.5*(H1(I,K)+H1(I-1,K))
      H1IMM=0.5*(H1(I-1,K)+H1(I-2,K))
      H1IKP=0.5*(H1(I,K+1)+H1(I,K))
      H1IKM=0.5*(H1(I,K)+H1(I,K-1))
      HHIKM=H1(I,K-1)
      HHIPK=(H1(I+1,K-1)+H1(I,K-1))*0.5
      HHIMK=(H1(I,K-1)+H1(I-1,K-1))*0.5
      HHIKP=H1(I,K+1)
      HDIPK=0.5*(H1(I+1,K+1)+H1(I,K+1))
      HDIMK=0.5*(H1(I,K+1)+H1(I-1,K+1))
      HHIMM=0.5*(H1(I-2,K-1)+H1(I-1,K-1))
      HHIMP=0.5*(H1(I-2,K+1)+H1(I-1,K+1))
      DO 2 J=JID,JFD
      JL=J-JO+1
      JJ=J-JID+1
      GJ=G(J)
      GJP=0.5*(G(J+1)+G(J))
      GJM=0.5*(G(J)+G(J-1))
      GJPP=0.5*(G(J+2)+G(J+1))
      GJMM=0.5*(G(J-1)+G(J-2))
```

B－7－2

修正後のSWEEP関連ルーチン

```
      COMMON/ FGIIIVP / FIMVP(67,32),GJVP(61),H1IVP(67,32),H1KVP(67,32),
     +                  H3KVP(32)
C*****************************************************************************
      DIMENSION JI(67),JF(67),PH(IR,JR,KR,3),AO(61),BO(61),CO(61),FO(61)
     1,XO(61)
      KL=K-KO+3
      H3K     =   H3    ( K    )
      H3KP    =   H3KVP( K    )
      H3KM    =   H3KVP( K-1  )
      H3KPP   =   H3KVP( K+1  )
      H3KMM   =   H3KVP( K-2  )
      DO 1 I=I1,I2
      IL=I-IO+3
      JID=JI(IL)
      JFD=JF(IL)
      JT=JFD-JID+1
      FI      =   F     ( I  , K    )
      FIP     =   FIMVP( I  , K    )
      FIM     =   FIMVP( I-1, K    )
      FIMM    =   FIMVP( I-2, K    )
      H1IK    =   H1    ( I  , K    )
      H1IPK   =   H1IVP( I  , K    )
      H1IMK   =   H1IVP( I-1, K    )
      H1IMM   =   H1IVP( I-2, K    )
      H1IKP   =   H1KVP( I  , K    )
      H1IKM   =   H1KVP( I  , K-1  )
      HHIKM   =   H1    ( I  , K-1  )
      HIIPK   =   H1IVP( I  , K-1  )
      HIIMK   =   H1IVP( I-1, K-1  )
      IIIIKP  =   H1    ( I  , K+1  )
      IIDIPK  =   H1IVP( I  , K+1  )
      IIDIMK  =   H1IVP( I-1, K+1  )
      IIIIMM  =   H1IVP( I-2, K-1  )
      IIIIMP  =   H1IVP( I-2, K+1  )
      DO 2 J=JID,JFD
      JL=J-JO+1
      JJ=J-JID+1
      GJ      =   G( J )
      GJP     =   GJVP( J    )
      GJM     =   GJVP( J-1  )
      GJPP    =   GJVP( J+1  )
      GJMM    =   GJVP( J-2  )
```

B－7－3

新たに作成されたFGH1H3ルーチン

```
      SUBROUTINE  FGH1H3
C
      COMMON / FGHHVP / FIM(2144), GJ(61), H1IM(2144), H1KM(2144),
     +                  H3K(32)
      COMMON / GRID1  / F  (2144), G (61), H1  (2144),
     +                  H3 (32),  X(2144), Y(61), Z(32)
C
      DO  1000  J = 1, 60
C
        GJ( J )  =  0.5 * ( G( J+1 ) + G( J ) )
C
 1000 CONTINUE
C
      DO  1100  L = 1, 2143
C
        H1IM( L )  =  0.5 * ( H1( L+1 ) + H1( L ) )
C
        FIM ( L )  =  0.5 * ( F ( L+1 ) + F ( L ) )
C
 1100 CONTINUE
C
      DO  1200  L = 1, 2077
C
        H1KM( L )  =  0.5 * ( H1( L+67 ) + H1( L ) )
C
 1200 CONTINUE
C
      DO  1400  K = 1, 31
C
        H3K( K )  =  0.5 * ( H3( K+1 ) + H3( K ) )
C
 1400 CONTINUE
C
      RETURN
      END
```

B－8－1

チューニング8

修正前の関数VG引用部および関数VG

```
C...CALCULATE CONSTANT TERM
       VPPP2 = VPPP5*F(I,K)*(PH3(IL+1,J+1,K,2)-PH3(IL-1,J+1,K,2))
      1          +(ETBL(IL,K)-ET(J+1))*VPPP6*F(I,K)*(PH3(IL+1,J+1,K,2)
      2          -PH3(IL-1,J+1,K,2)-PH3(IL+1,J,K,2)+PH3(IL-1,J,K,2))
      3          +VPPP7
       VPPP3 = FSXL(IL,K)+THT(K)
       VPPP4 = H1(I,K)*PHXB+H3(K)*PHCB
C...CALCULATE B IF BLANCH
       B  =  VPPP1 + VPPP2 * ( VPPP3 + VPIF(1) * DWX( IL-2, K-2 )
      1        + VPIF(2) * FTXD ) + VPPP4 * ( FSZL( IL, K )
      2        + VPIF(3) * DWZ( IL-2, K-2 ) + VPIF(4) * FTZD )
      3        + DET2 * ( VPIF(5) * DWT( IL-2, K-2 ) + VPIF(6) * FTDD
      4        - VPIF(7) * VG( XIK, ZK, T ) )
C*****************************************************************
```

```
      FUNCTION VG(X,Z,T)
      COMMON /GUST/TG,VGO,XGO,XGW,IGSTM
      IF(IGSTM.EQ.1) GO TO 1
      IF(IGSTM.EQ.2) GO TO 2
    1 IF(X.GT.T-TG+XGO) VG=0.0
      IF(X.LE.T-TG+XGO) VG=VGO
      GO TO 10
    2 IF(X.GE.-XGW+XGO+T-TG.AND.X.LE.XGO+T-TG)
     1VG=0.5*VGO*(1.0-COS(2.0*3.1415927*(X-XGO-T+TG+XGW)/XGW))
      IF(X.LT.-XGW+XGO+T-TG.OR.X.GT.XGO+T-TG)
     1VG=0.0
   10 RETURN
      END
```

B－8－2

修正後の関数VG展開部

```
C************* HISTORY (4)*****************************************:
      VPPPVG  =  T - TG + XGO                                        ループ外での処理
C*****************************!***********************************
      DO 600 ITA=1,ITAZ
      DO 21 K=3,KT
      DO 22 I=3,ILM1

                                :
                                :
                                :
                                :
                                :
                                :
                                :
                                :

  C...CALCULATE CONSTANT TERM
       VPPP2 = VPPP5*F(I,K)*(PH3(IL+1,J+1,K,2)-PH3(IL-1,J+1,K,2))
      1          +(ETBL(IL,K)-ET(J+1))*VPPP6*F(I,K)*(PH3(IL+1,J+1,K,2)
      2          -PH3(IL-1,J+1,K,2)-PH3(IL+1,J,K,2)+PH3(IL-1,J,K,2))
      3          +VPPP7
       VPPP3 = FSXL(IL,K)+THT(K)
       VPPP4 = H1(I,K)*PHXB+H3(K)*PHCB
  C************** HISTORY (4) ********************
       IF( XIK .LE. VPPPVG )  THEN
         IF( IGSTM .EQ. 1 )  THEN
           VPVG  =  VGO
         ELSE
           VPVG  =  AMIN1( (VPPPVG-XIK)/XGW, 1.0 )
           VPVG  =  0.5 * VGO * ( 1.0 - COS(2.0*3.1415927*VPVG) )
         ENDIF
       ELSE
           VPVG  = 0.0
       ENDIF
  C...CALCULATE B IF BLANCH
       B  =  VPPP1 + VPPP2 * ( VPPP3 + VPIF(1) * DWX( IL-2, K-2 )
      1        + VPIF(2) * FTXD ) + VPPP4 * ( FSZL( IL, K )
      2        + VPIF(3) * DWZ( IL-2, K-2 ) + VPIF(4) * FTZD )
      3        + DET2 * ( VPIF(5) * DWT( IL-2, K-2 ) + VPIF(6) * FTDD
      4        - VPIF(7) * VPVG )
  C*****************************************************************
```

B－9

(1) Kの値はサブルーチン内で変化しないのでDOループの外に排出する。

```
 DO 1 I=I1,I2
 DO 2 J=JID,JFD
        :
        :
        :
        :
        :

 W=H1IK*(PH(IL+1,JL,KL,2)-PH(IL-1,JL,KL,2))/(2.0*DXAI)+H3K*
1(PH(IL,JL,KL+1,2)-PH(IL,JL,KL-1,2))/(2.0*DCH)
 IF(K.EQ.3) W=0.0

        :
        :
        :
        :
        :
2 CONTINUE
1 CONTINUE
```

⇩

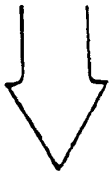
```
 VPK3   =   1.0
 IF( K .EQ. 3 )   VPK3  =  0.0
 DO 1 I=I1,I2
 DO 2 J=JID,JFD
        :
        :
        :
        :
        :

 W=H1IK*(PH(IL+1,JL,KL,2)-PH(IL-1,JL,KL,2))/(2.0*DXAI)+H3K*
1(PH(IL,JL,KL+1,2)-PH(IL,JL,KL-1,2))/(2.0*DCH)
  W = W * VPK3

        :
        :
        :
        :
        :
2 CONTINUE
1 CONTINUE
```

(2) IF文の代わりに関数を用いる。

```
IF(Q2.GE.A2) AMU=1.0-A2/Q2
IF(Q2.LT.A2) AMU=0.0
```



ただしQ2＞0.0.

```
AMU = AMAX1( 1.0-A2/Q2, 0.0 )
```

(3) IF文を削除し，毎回計算する。

```
    IF(Q2-A2) 800,801,801
801 CONTINUE
    FFI=AMU*(U2*PHXX+UV*PHXYC+UW*PHXZC)
    GGJ=AMU*(V2*PHYYC+UV*PHXYC+VW*PHYZC)
    HHK=AMU*(W2*PHZZC+UW*PHXZC+VW*PHYZC)
    GO TO 802
800 FFI=0.0
    GGJ=0.0
    HHK=0.0
802 IF(Q2M-A2M) 803,804,804
```

ただしQ2＜A2の場合AMU＝0.0となる。

```
FFI=AMU*(U2*PHXX+UV*PHXYC+UW*PHXZC)
GGJ=AMU*(V2*PHYYC+UV*PHXYC+VW*PHYZC)
HHK=AMU*(W2*PHZZC+UW*PHXZC+VW*PHYZC)
```

(4) 最適化制御行の利用

```
      IF(Q2M.GE.A2M) AMUM=1.0-A2M/Q2M
      IF(Q2M.LT.A2M) AMUM=0.0

  802 IF(Q2M-A2M) 803,804,804
  804 CONTINUE

                  :
                  :
                  :
                  :

      FFIM=AMUM*(UM2*PHXXB+UVM*PHXYB+UWM*PHXZB)
      GO TO 805
  803 FFIM=0.0
  805 CONTINUE
```

⇩ ただしAMUMは他で参照しない

```
       FFIM  = 0.0
*VOCL STMT,IF(5)
      IF( Q2M .GT. A2M )   THEN

                  :
                  :
                  :
                  :

      FFIM=(1.0-A2M/Q2M)*(UM2*PHXXB+UVM*PHXYB+UWM*PHXZB)
      ENDIF
```

(5) SIGN関数の利用によるIF文の削除

```
     IF(W) 59,60,60
  60 AA32=H3K*DTC*(PH(IL,JL,KL,2)-PH(IL,JL,KL-1,2)-PH(IL,JL,KL,3)+
    1PH(IL,JL,KL-1,3))
     GO TO 61
  59 AA32=H3K*DTC*(PH(IL,JL,KL+1,2)-PH(IL,JL,KL,2)-PH(IL,JL,KL+1,3)+
    1PH(IL,JL,KL,3))
  61 AA3=(AA31+AA32)*2.0*W
```

⇩

```
     KS  =  0.51 - SIGN( 0.5, W )
     KV  =  KL + KS
     AA32=H3K*DTC*(PH(IL,JL,KV,2)-PH(IL,JL,KV-1,2)-PH(IL,JL,KV,3)+
    1PH(IL,JL,KV-1,3))
     AA3=(AA31+AA32)*2.0*W
```

B－12－1

チューニング 12

新たに作成されたVPCALルーチン

```
C            VPCAL              TEMPORARY LEVEL=1     DATE=85.09.27
C*****************************************************************
C       HISTORY ........
C         (1) NEW SUBROUTINE
C              U,V,W,A2/G2,PHYYC,PHZZC,PHXYC,PHYZC,PHXZC
C         VP(*,1;2;3;     4;     5;     6;     7;     8;     9 )
C*****************************************************************
       SUBROUTINE VPCAL(PH1,PH1M,PH2,PH2M,PH3,PH3M,PH4,PH4M,PH5,PH5M,
      +                 VP1,     VP2,     VP3,     VP4,     VP5       )
*      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       COMMON /GRID1/F(67,32),G(61),H1(67,32),H3(32),X(67,32),Y(61),Z(32)
       COMMON /GRID2/JW,JWP1,JWM1,JWM2
       COMMON /GRID3/DXAI,DET,DCH,DTX,DTX2,DTE,DTE2,DTC,DTC2,DTXE,DTXC,
      1DTEC,DT
       COMMON /GRID4/C5,C6,Z1,XLMAX,XTMAX,ZWG(20),KTM1,KTP1,KZ1,KG
       COMMON /BC/AZ1(35),AZ2(35),AZ3(35)
       COMMON /WAKE/DPH(22,19,3)
       COMMON /AM/AM2,GAM1,GAM2,COAL,SIAL
       COMMON /IPL/ IE,IT,KT,ZS(32),ZSS(17),KKZ
       COMMON /COM002/ ILM1,  ILM2,  IS,    ITAZ,  ITM1,  ITP1,  IZ,
      1                IZ1,   JWM5,  JWP5,  JZ,    JZ2,   ZZ,
      2                KTP2,  KW1,   KZ2
C
       COMMON /FGHHVP/ FIMVP(67,32),GJVP(61),H1IVP(67,32),H1KVP(67,32),
      +                H3KVP(32)
C
       DIMENSION     PH1(18,61,19),    PH1M(18,61,19),
      +              PH2(35,36,19),    PH2M(35,36,19),
      +              PH3(35,36,19),    PH3M(35,36,19),
      +              PH4(22,61,19),    PH4M(22,61,19),
      +              PH5(67,61,17),    PH5M(67,61,17)
C
       DIMENSION VP1(18,61,19,9), VP2(35,36,19,9), VP3(35,36,19,9),
      +          VP4(22,61,19,9), VP5(67,61,17,9)
C
C                        ............ CONSTANT TERM
       RDXAI    =   0.5  /  DXAI
       RDET     =   0.5  /  DET
       RDCH     =   0.5  /  DCH
       RAM2     =   1.0  /  AM2
       RDT      =   2.0  /  DT
       DTXCVP   =   0.25 *  DTXC
       DTXEVP   =   0.25 *  DTXE
       DTECVP   =   0.25 *  DTEC
C
C                        ............ K LOOP
       DO  1000   KK = 2,  KT + 1
C
         K  =  KK
C                                      .......... PH1
         VPK3   =   1.0
         IF( KK .LE. 3 )  VPK3  =  0.0
```

B－12－2

```
      J1  =  3
      J2  =  JJ2
      I1  =  3
      I2  =  ILM1
C
      DO  1010  JJ = J1 - 1, J2 + 1
        J  =  JJ
      DO  1010  II = I1 - 1, I2 + 1
        I  =  II
C
        PH2X  =  PH1( I+1,J  ,K   ) - PH1( I-1,J  ,K   )
        PH2Y  =  PH1( I  ,J+1,K   ) - PH1( I  ,J-1,K   )
        PH2Z  =  PH1( I  ,J  ,K+1 ) - PH1( I  ,J  ,K-1 )
C
        VP1( I,J,K,1 ) = F( II,KK ) = PH2X = RDXAI + COAL
        VP1( I,J,K,2 ) = G( JJ )    = PH2Y = RDET  + SIAL
        VP1( I,J,K,3 ) = H1( II,KK )= PH2X = RDXAI
     +                 + H3( KK )   = PH2Z = RDCH
C
        VP1( I,J,K,3 ) = VP1( I,J,K,3 ) = VPK3
C
        Q2  = VP1( I,J,K,1 ) ** 2 + VP1( I,J,K,2 ) ** 2
     +                            + VP1( I,J,K,3 ) ** 2
        A2  = RAM2 - GAM1 * ( ( PH1( I,J,K ) - PH1M( I,J,K ) ) * RDT
     +                                                 + Q2 - 1.0 )
        VP1( I,J,K,4 ) = A2 / Q2
C
        VP1( I,J,K,5 ) = G( JJ ) * DTE2 *
     +        ( GJVP( JJ )   * ( PH1( I,J+1,K ) - PH1( I,J  ,K ) )
     +        - GJVP( JJ-1 ) * ( PH1( I,J  ,K ) - PH1( I,J-1,K ) ) )
C
        PHXZ1  = DTX2 *
     +     ( H1IVP( II  ,KK ) * ( PH1( I+1,J,K ) - PH1( I  ,J,K ) )
     +     - H1IVP( II-1,KK ) * ( PH1( I  ,J,K ) - PH1( I-1,J,K ) ) )
C
        PHZZ2  = DTXCVP * H1( II,KK ) * H3( KK ) *
     +                  ( PH1( I+1,J,K+1 ) - PH1( I-1,J,K+1 )
     +                  - PH1( I+1,J,K-1 ) + PH1( I-1,J,K-1 ) )
        PHZZ3  = DTXCVP * H3( KK ) *
     +  ( H1( II,KK+1 ) * ( PH1( I+1,J,K+1 ) - PH1( I-1,J,K+1 ) )
     +  - H1( II,KK-1 ) * ( PH1( I+1,J,K-1 ) - PH1( I-1,J,K-1 ) ) )
        PHZZ4  = DTC2 * H3( KK ) *
     +     ( H3KVP( KK   )   * ( PH1( I,J,K+1 ) - PH1( I,J,K   ) )
     +     - H3KVP( KK-1 )   * ( PH1( I,J,K   ) - PH1( I,J,K-1 ) ) )
C
        VP1( I,J,K,6 ) = H1(II,KK) *  PHXZ1
     +                 + PHZZ2 + PHZZ3 + PHZZ4
```

U

V

W

A2/Q2

PHYYC

PHZZC

B－12－3

```
C
      PHXY   =  DTXEVP = ( PH1( I+1,J+1,K ) - PH1( I-1,J+1,K )
     +                   - PH1( I+1,J-1,K ) + PH1( I-1,J-1,K ) )
C
      VP1( I,J,K,7 ) =  F( II,KK ) = G( JJ ) = PHXY      <-------- PHXYC
C
      PHYZ1  = PHXY    = H1( II,KK ) = G( JJ )
      PHYZ2  = DTECVP  = H3( KK ) =    G( JJ ) =
     +                   ( PH1( I,J+1,K+1 ) - PH1( I,J-1,K+1 )
     +                   - PH1( I,J+1,K-1 ) + PH1( I,J-1,K-1 ) )
C
      VP1( I,J,K,8 ) = PHYZ1 + PHYZ2                      <-------- PHYZC
C
      PHXZ2  = DTXCVP = H3( KK ) = F( II,KK ) =
     +                   ( PH1( I+1,J,K+1 ) - PH1( I-1,J,K+1 )
     +                   - PH1( I+1,J,K-1 ) + PH1( I-1,J,K-1 ) )
C
      VP1( I,J,K,9 ) = F( II,KK ) = PHXZ1 + PHXZ2         <-------- PHXZC
C
 1010 CONTINUE
```

計算領域 1 と同様

計算領域 2, 3, 4, 5 に対する処理

```
      RETURN
      END
```

B－12－4

サブルーチンSWEEPおよびSWEEP4を統合したサブルーチン

```
      SUBROUTINE SWPVP(IRH,IR,JR,KR,I1,I2,JI,JF,IO,JO,K,KO,PH,VP,VPC)
C*****************************************************************
C       HISTORY.......
C        (1) NEW SUBROUTINE
C             U,V,W,PHXYC,PHXZC,PHYZC,PHYYC,PHZZC,A2/G2
C              WO SUBR. VPCAL DE KEISAN ZUMI ON VP
C*****************************************************************
*       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        COMMON /GRID1/F(67,32),G(61),H1(67,32),H3(32),X(67,32),Y(61),Z(32)
        COMMON /GRID2/JW,JWP1,JWM1,JWM2
        COMMON /GRID3/DXAI,DET,DCH,DTX,DTX2,DTE,DTE2,DTC,DTC2,DTXE,DTXC,
       1DTEC,DT
        COMMON /BC/AZ1(35),AZ2(35),AZ3(35)
        COMMON /WAKE/DPH(22,19,3)
        COMMON /AM/AM2,GAM1,GAM2,COAL,SIAL
        COMMON/ FGHHVP / FIMVP(67,32),GJVP(61),H1IVP(67,32),H1KVP(67,32),
       +                 H3KVP(32)
        DIMENSION JI(67),JF(67),PH(IR,JR,KR,3),AO(61),BO(61),CO(61),FO(61)
       1,XO(61)
C***************** HISTORY (4) ***********************************
        DIMENSION VPJW(61,2),VP(IR,JR,KR,9),VPC(IR,JR,KR,9)
        DATA      VPJW      /   122 * 0.0 /
C
        VPK3  =  1.0
        IF( K .EQ. 3 )  VPK3  =  0.0
C
        IF( IRH .EQ. 4 )  THEN
         VPJW( JW  ,1 )  =  1.0
         VPJW( JWM1,2 )  =  1.0
        ELSE
         VPJW( JW  ,1 )  =  0.0
         VPJW( JWM1,2 )  =  0.0
        ENDIF
C
        KL=K-KO+3
C                                  ................. LOOP I
S       DO 1 I=I1,I2
C
          IL=I-IO+3
S         JID=JI(IL)
S         JFD=JF(IL)
S         JT=JFD-JID+1
C                                  ................. LOOP J
V         DO 2 J=JID,JFD
C
            JL=J-JO+1
            JJ=J-JID+1
C
V           AMU = AMAX1( 1.0-VP(IL,JL,KL,4), 0.0 )
C
V           U2  = VP(IL,JL,KL,1) ** 2
V           V2  = VP(IL,JL,KL,2) ** 2
V           W2  = VP(IL,JL,KL,3) ** 2
C
V           UVPHXY = VP(IL,JL,KL,1) * VP(IL,JL,KL,2) * VP(IL,JL,KL,7)
V           VWPHYZ = VP(IL,JL,KL,2) * VP(IL,JL,KL,3) * VP(IL,JL,KL,8)
V           UWPHXZ = VP(IL,JL,KL,3) * VP(IL,JL,KL,1) * VP(IL,JL,KL,9)
C
```

B－12－5

```
V            PHXX  = VPC(IL,JL,KL,1) * DTX2 *
     1          ( VPC(IL,JL,KL,3)*( PH(IL-1,JL,KL,2) - PH(IL,JL,KL,2) )
     2        - VPC(IL-1,JL,KL,3)*( PH(IL,JL,KL,2) - PH(IL-1,JL,KL,2) ))
C
V            U2PHXX = U2 * PHXX
V            V2PHYY = V2 * VP(IL,JL,KL,5)
V            W2PHZZ = W2 * VP(IL,JL,KL,6)
C
V            FFI  =  AMU * ( U2PHXX + UVPHXY + UWPHXZ )
V            GGJ  =  AMU * ( V2PHYY + UVPHXY + VWPHYZ )
V            HHK  =  AMU * ( W2PHZZ + UWPHXZ + VWPHYZ )
C
V            FFIM  = AMAX1( 1.0-VP(IL-1,JL,KL,4), 0.0 )
C
*VOCL STMT,IF(5)
V            IF( FFIM .GT. 0.0 )  THEN
C
V              PHXX8 = DTX2 * VPC(IL-1,JL,K,1) * VP(IL-1,JL,KL,1) ** 2 *
     1       ( VPC(IL-1,JL,KL,5)*( PH(IL,JL,KL,2) - PH(IL-1,JL,KL,2) )
     2       - VPC(IL-2,JL,KL,5)*( PH(IL-1,JL,KL,2) - PH(IL-2,JL,KL,2) ))
C
V              FFIM  = FFIM *( PHXX8 +
     1            VP(IL-1,JL,KL,1) * VP(IL-1,JL,KL,2) * VP(IL-1,JL,KL,7)
     2          + VP(IL-1,JL,KL,1) * VP(IL-1,JL,KL,3) * VP(IL-1,JL,KL,9) )
            ENDIF
C
V          JS = SIGN( 1.0, VP(IL,JL,KL,2) )
V          JV = JL - JS
C
V          GGJM = AMAX1( 1.0-VP(IL,JV,KL,4), 0.0 )
C
*VOCL STMT,IF(5)
V          IF( GGJM .GT. 0.0 )      GGJM  =  GGJM *
     1          ( VP(IL,JV,KL,2) * VP(IL,JV,KL,2) * VP(IL,JV,KL,5)
     2          + VP(IL,JV,KL,1) * VP(IL,JV,KL,2) * VP(IL,JV,KL,7)
     3          + VP(IL,JV,KL,3) * VP(IL,JV,KL,2) * VP(IL,JV,KL,8) )
C
V          KS  =  SIGN( 1.0, VP(IL,JL,KL,3) )
V          KV  =  KL - KS
C
V          HHKM  =  AMAX1( 1.0-VP(IL,JL,KV,4), 0.0 ) * VPK3
C
*VOCL STMT,IF(5)
V          IF( HHKM .GT. 0.0 )   HHKM  =  HHKM *
     1          ( VP(IL,JL,KV,3) * VP(IL,JL,KV,3) * VP(IL,JL,KV,6)
     2          + VP(IL,JL,KV,1) * VP(IL,JL,KV,3) * VP(IL,JL,KV,9)
     3          + VP(IL,JL,KV,2) * VP(IL,JL,KV,3) * VP(IL,JL,KV,8) )
C
V          PHXYZ  =  2.0 * ( UVPHXY + VWPHYZ + UWPHXZ )
C
V          AA1  =  ( VP(IL,JL,KL,4) - 1.0 ) *
     1           ( U2PHXX + V2PHYY + W2PHZZ + PHXYZ )
     2           - FFIM - GGJM - HHKM + FFI + GGJ + HHK
C
V          AA2  =  VP(IL,JL,KL,4) * ( - PHXYZ
     1           + ( V2 + W2 ) * PHXX + ( U2 + V2 ) * VP(IL,JL,KL,6) )
C
V          AA31 =  VPC(IL,JL,KL,3) * DTX
     1                          * ( PH(IL,JL,KL,2) - PH(IL-1,JL,KL,2)
     2                            - PH(IL,JL,KL,3) + PH(IL-1,JL,KL,3) )
C
V          KS = 0.51 - SIGN( 0.5, VP(IL,JL,KL,3) )
V          KS = KL + KS
C
V          AA32 = VPC(IL,JL,KL,4) * DTC
     1                        * ( PH(IL,JL,KS,2) - PH(IL,JL,KS-1,2)
     2                          - PH(IL,JL,KS,3) + PH(IL,JL,KS-1,3) )
```

B－12－6

```
V          AA3  = ( AA31 + AA32 ) * 2.0 * VP(IL,JL,KL,3)
V          AA   = - AA1 - AA2 + AA3
   C
V          DD = VPC(IL,JL,KL,6)    * ( PH(IL,JL+1,KL,3) - PH(IL,JL,KL,3) )
         1    - VPC(IL,JL-1,KL,6) * ( PH(IL,JL,KL,3) - PH(IL,JL-1,KL,3) )
   C
V          BB1 = DD - ( DPH(IL,K,1) + DPH(IL,K,3) ) *
         1  ( VPC(IL,JL,KL,6) * VPJW(J,2) - VPC(IL,JL-1,KL,6 ) * VPJW(J,1) )
   C
V          QQ   =  VP(IL,JL,KL,4) * ( U2 + W2 ) * 0.50 *
         1                                      VPC(IL,JL,KL,2) * DTE2
   C
V          BB2  =   2.0 * VP(IL,JL,KL,1) * DTX * VPC(IL-1,JL,KL,5) *
         1       ( - PH(IL-1,JL,KL,1) - PH(IL,JL,KL,2) + PH(IL-1,JL,KL,2) )
   C
V          JS = 0.51 - SIGN( 0.5, VP(IL,JL,KL,2) )
   C
V          BB3= VPC(IL,JL+JS-1,KL,6)
         1                    * ( -PH(IL,JL+JS,KL,2) + PH(IL,JL+JS-1,KL,2) )
V          BB3= BB3 + VPC(IL,JL+JS-1,KL,6) * VPJW(J,1+JS) *
         1                     ( - DPH(IL,K,1)       + DPH(IL,K,2) )
   C
V          QV   = 2.0 * VP(IL,JL,KL,2) * DTE
   C
V          BB = - BB1 * QQ + BB2 + BB3 * QV
         1           - 2.0 * PH(IL,JL,KL,2) + PH(IL,JL,KL,3)
   C
V          FO(JJ) = AA + BB
   C
V          VVP = 0.5 + SIGN( 0.5, VP(IL,JL,KL,2) )
   C
V          AO(JJ) = ( QQ + QV * VVP ) * VPC(IL,JL-1,KL,6)
   C
V          BO(JJ) = - QQ * ( VPC(IL,JL,KL,6) + VPC(IL,JL-1,KL,6) )
         1     - 2.0 * VP(IL,JL,KL,1) * DTX * VPC(IL-1,JL,KL,5) - 1.0
         2     - QV * VPC(IL,JL-1+JS,KL,6) * SIGN( 1.0, VP(IL,JL,KL,2) )
   C
V          CO(JJ) = ( QQ - QV * ( 1.0 - VVP ) ) * VPC(IL,JL,KL,6)
V       2 CONTINUE
   C
```

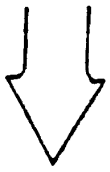
```
          RETURN
          END
```

B－13－1

チューニング 13

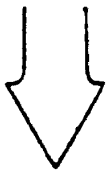
新サブルーチンVPCALの修正前および修正後

```
      COMMON /GRID1/F(67,32),G(61),H1(67,32),H3(32),X(67,32),Y(61),Z(32)
C
      COMMON /FGHHVP/ FIMVP(67,32),GJVP(61),H1IVP(67,32),H1KVP(67,32),
     +                H3KVP(32)
```



配列の同一化（新サブルーチンVPGRIDにより設定）

```
C
      COMMON /GRIDVP/ VPC1(1098,19,9), VPC2(1260,19,9),
     +                VPC3(1260,19,9), VPC4(1342,19,9),
     +                VPC5(4087,17,9)
C
      DIMENSION    PH1(18,61,19),   PH1M(18,61,19),
     +             PH2(35,36,19),   PH2M(35,36,19),
     +             PH3(35,36,19),   PH3M(35,36,19),
     +             PH4(22,61,19),   PH4M(22,61,19),
     +             PH5(67,61,17),   PH5M(67,61,17)
C
      DIMENSION VP1(18,61,19,9), VP2(35,36,19,9), VP3(35,36,19,9),
     +          VP4(22,61,19,9), VP5(67,61,17,9)
C
```

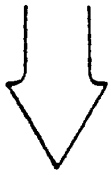


使用される配列の低次元化

```
C
      DIMENSION    PH1(1098,19),   PH1M(1098,19),
     +             PH2(1260,19),   PH2M(1260,19),
     +             PH3(1260,19),   PH3M(1260,19),
     +             PH4(1342,19),   PH4M(1342,19),
     +             PH5(4087,17),   PH5M(4087,17)
C
      DIMENSION VP1(1098,19,9), VP2(1260,19,9), VP3(1260,19,9),
     +          VP4(1342,19,9), VP5(4087,17,9)
C
```

B－13－2

```
      J1  =  3
      J2  =  JJ2
      I1  =  3
      I2  =  ILM1
C
      DO  1010  JJ = J1 - 1, J2 + 1
        J  =  JJ
      DO  1010  II = I1 - 1, I2 + 1
        I  =  II
C
        PH2X  =  PH1( I+1,J  ,K   ) - PH1( I-1,J  ,K   )
        PH2Y  =  PH1( I  ,J+1,K   ) - PH1( I  ,J-1,K   )
        PH2Z  =  PH1( I  ,J  ,K+1 ) - PH1( I  ,J  ,K-1 )
C
        VP1( I,J,K,1 ) = F( II,KK ) * PH2X * RDXAI + COAL
        VP1( I,J,K,2 ) = G( JJ )    * PH2Y * RDET  + SIAL
        VP1( I,J,K,3 ) = H1( II,KK )* PH2X * RDXAI
     +                   + H3( KK )   * PH2Z * RDCH
C
        VP1( I,J,K,3 ) = VP1( I,J,K,3 ) * VPK3
C
        Q2  = VP1( I,J,K,1 ) ** 2 + VP1( I,J,K,2 ) ** 2
     +                            + VP1( I,J,K,3 ) ** 2
        A2  = RAM2 - GAM1 * ( ( PH1( I,J,K ) - PH1M( I,J,K ) ) * RDT
     +                                                  + Q2 - 1.0 )
        VP1( I,J,K,4 ) = A2 / Q2
C
        VP1( I,J,K,5 ) = G( JJ ) * DTE2 *
     +          ( GJVP( JJ )   * ( PH1( I,J+1,K ) - PH1( I,J  ,K ) )
     +          - GJVP( JJ-1 ) * ( PH1( I,J  ,K ) - PH1( I,J-1,K ) ) )
```



計算領域１に対する配列の低次元化
（同様のことが計算領域２～５に対しても行われた。）

```
C
      L11  =  20
      L12  =  JJ2                * 18 + ILM1       + 1

      DO  1010   L = L11, L12

        PH2X  =  PH1( L+1    ,K   ) - PH1( L-1    ,K   )
        PH2Y  =  PH1( L+18   ,K   ) - PH1( L-18   ,K   )
        PH2Z  =  PH1( L      ,K+1 ) - PH1( L      ,K-1 )
C
        VP1( L  ,K,1 ) = VPC1( L,K,1 ) * PH2X * RDXAI + COAL
        VP1( L  ,K,2 ) = VPC1( L,K,2 ) * PH2Y * RDET  + SIAL
        VP1( L  ,K,3 ) = VPC1( L,K,3 ) * PH2X * RDXAI
     +                   + VPC1( L,K,4 ) * PH2Z * RDCH
C
        VP1( L  ,K,3 ) = VP1( L  ,K,3 ) * VPK3
C
        Q2  = VP1( L  ,K,1 ) ** 2 + VP1( L  ,K,2 ) ** 2
     +                            + VP1( L  ,K,3 ) ** 2
        A2  = RAM2 - GAM1 * ( ( PH1( L  ,K ) - PH1M( L  ,K ) ) * RDT
     +                                                  + Q2 - 1.0 )
        VP1( L  ,K,4 ) = A2 / Q2
C
        VP1( L  ,K,5 ) = VPC1( L,K,2 ) * DTE2 *
     +          ( VPC1( L,K,6 )
     +                       * ( PH1( L+18 ,K ) - PH1( L    ,K ) )
     +          - VPC1( L-18,K,6 )
```

B－13－3

新たに作成されたVPGRIDルーチン

（係数配列の次元およびサイズを同一にするプログラム）

```
      SUBROUTINE VPGRID
*     IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON /GRID1/F(67,32),G(61),H1(67,32),H3(32),X(67,32),Y(61),Z(32)
      COMMON /GRID2/JW,JWP1,JWM1,JWM2
      COMMON /GRID4/C5,C6,Z1,XLMAX,XTMAX,ZWG(20),KTM1,KTP1,KZZ,KG
      COMMON /IPL/ IE,IT,KT,ZS(32),ZSS(17),KKZ
      COMMON /COMOO2/ ILM1,  ILM2,  IS,    ITAZ,  ITM1,  ITP1,  IZ,
     1                IZ1,   JWM5,  JWP5,  JZ,    JZ2,   ZZ,
     2                KTP2,  KW1,   KZ2
C
      COMMON /FGHHVP/ FIMVP(67,32),GJVP(61),H1IVP(67,32),H1KVP(67,32),
     +                H3KVP(32)
C
      COMMON /GRIDVP/ VPC1(18,61,19,9), VPC2(35,36,19,9),
     +                VPC3(35,36,19,9), VPC4(22,61,19,9),
     +                VPC5(67,61,17,9)
C
C                     ............ VPC1
C
      DO  1100  K = 1, KT + 2
C
        DO  1100  J = 1, JZ2 + 2
C
          DO  1100  I = 1, ILM1 + 2
C
            VPC1( I, J, K, 1 ) = F( I, K )
            VPC1( I, J, K, 2 ) = G( J )
            VPC1( I, J, K, 3 ) = H1( I, K )
            VPC1( I, J, K, 4 ) = H3( K )
            VPC1( I, J, K, 5 ) = FIMVP( I, K )
            VPC1( I, J, K, 6 ) = GJVP( J )
            VPC1( I, J, K, 7 ) = H1IVP( I, K )
            VPC1( I, J, K, 8 ) = H1KVP( I, K )
            VPC1( I, J, K, 9 ) = H3KVP( K )
C
 1100 CONTINUE
```

配列VPC1と同様

配列VPC2, VPC3, VPC4, VPC5の処理

```
      RETURN
      END
```

航空宇宙技術研究所報告915号

昭和61年10月発行

発行所 航空宇宙技術研究所
東京都調布市深大寺東町7丁目44番地1
電話武蔵野三鷹(0422)47-5911(大代表)〒182

印刷所 株式会社 三興印刷
東京都新宿区信濃町12三河ビル

Printed in Japan