JAERI-M
85-142

大型プログラム開発支援ツール

1985年10月

飯田　鉱*・常松　俊秀・平山　俊雄
富山　峯秀・竹田　辰興・徳田　伸二
安達　政夫**・浅井　清

日本原子力研究所
Japan Atomic Energy Research Institute

JAERI-Mレポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所技術情報部情報資料課（〒319－11茨城県那珂郡東海村）あて，お申しこしください。なお，このほかに財団法人原子力弘済会資料センター（〒319-11茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

JAERI-M reports are issued irregularly.
Inquiries about availability of the reports should be addressed to Information Division, Department of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　日立高速印刷株式会社

JAERI－M 85－142

大型プログラム開発支援ツール

日本原子力研究所東海研究所計算センター
飯田 鉱*・常松 俊秀+・平山 俊雄++・富山 峯秀
竹田 辰興+・徳田 伸二+・安達 政夫**・浅井 清

（１９８５年８年１９日受理）

本報告は，１９８５年２月に開催された原子力コード研究委員会総合化専門部会における討論資料をまとめたものである。

(1) コードのデバック手法

核融合研究部理論解析研究室の長年にわたる多数のコード開発経験をもとに，プログラミング，デバッギングについての有効な手法が述べられている。

(2) 静的デバッグツール SDEBUIG

臨界プラズマ実験で開発したデバッグツールについて述べる。このツールは従来デバッグの難しかった長時間実行，大量出力ジョブのような大型コードのデバッグに有効である。

(3) プログラムの開発，保守，管理支援ツール等のプログラム開発手法

現在原研計算センターにおいて使用可能なプログラム開発，保守，管理のための支援ツールの一部を紹介する。

+ 核融合研究部
++ 臨界プラズマ研究部
* 富士通エフ・アイ・ピー株式会社
** 日本ソフトウェア開発株式会社

JAERI-M 85-142

Development Tools for Large-scale Computer Programs

Koh IIDA*, Toshihide TSUNEMATSU+, Toshio HIRAYAMA++,
Mineyoshi TOMIYAMA, Tatsuoki TAKEDA+, Shinji TOKUDA+,
Masao ADACHI** and Kiyoshi ASAI

Computing Center
Tokai Research Establishment, JAERI

(Received August 19, 1985)

A meeting of the Subcommittee for Nuclear Code Unification of the Nuclear Code Committee of JAERI was held in February 1985, and effective methods of development and debugging of computer programs were discussed. Three documents on subjects listed below were then introduced and they consist of three chapters in this report.

(1) Programming and debugging methods of computer programs,
(2) static program debugging tool : SDEBUG,
(3) development, maintenance and management tools of computer programs.

Based on experiences with a large amount of program developments at Plasma Theory Laboratory of JAERI, very useful programming and debugging methods are presented in the second chapter.

In the third chapter, new debugging tool that is useful for especially large-scale computer programs is introduced by JT-60 Program Office of JAERI.

Many tools which are used for development, debugging, maintenance and management of programs are introduced in the fourth chapter by Computing Center of JAERI.

Keywords: Computer, Nuclear Code, Software Engineering, Software Development, Software Maintenance, Program Debug, Debug, Debug Tool, SDEBUG

+ Department of Thermonuclear Fusion Research
++ Department of Large Tokamak Research
* Fujitsu F.I.P., INC.
** NIHON SOFTWARE KAIHATSU, INC.

目　　次

Contents

1. 序　　文

計算機の大型化は大規模なプログラムの開発を可能にしてきたが，一方で，その開発過程におけるエラーの修正や保守といった問題も多様化，複雑化してきている。このエラーや保守といった問題を無くすことができるものであろうか。ソフトウェアの開発において人間が重要な役割を担っている以上その成果であるプログラムに様々な誤りが存在するのは避けることができないであろう。このため内外の研究機関においてプログラムの効率的な開発，保守，管理技法の提唱，そのためのツールの開発が多数なされている。原研においてもいくつかの研究室において技法の研究，ツールの開発がなされてきている。これらの情報はプログラムの開発，利用者に携わる者にとり有効でありまた興味深い問題であるが，本報告ではデバックに焦点を絞りその一端を紹介する。

プログラムの開発者にとって興味のあるところはどのようにしたら効率よくプログラムが開発でき，かつ後々発生するであろうバグや機能の追加修正といった問題に容易に対処できるかということであろう。プログラムの開発過程はおおよそ次のようになる。

(1) プログラムの機能定義，
(2) 外部設計，
(3) 内部設計，
(4) プログラミング，
(5) テ ス ト，
(6) 実　　用。

実用化されたプログラムにバグや機能の変更修正が発生すると，部分的にしろ大々的にしろ(1)から繰返されることになる。この繰返しを無くするには(1)~(3)の過程を厳密に行う必要がある。しかし原研のような研究機関においてはその性格上困難な場合が多い。従って最初からプログラムの機能の追加，変更はあるものと考え後日の保守が容易なようにプログラムを作成しておくべきであろう。核融合部理論解析研究室では長年にわたり多数の大型コードを開発してきている。そのためコードの開発，保守，管理に対しては多くの苦心が払われている。第2章ではその経験をもとにプログラムの設計，デバッグ，検証に対し有効な手法を述べている。

プログラムのデバックにおいては支援ツールは強力な手段となる。現在メーカからもコンパイラ・レベルで指定できるDEBUGオプションやSDF（Symbolic Dump Facility），会話型でデバッグを行うTESTFORT−77やDOCKなど便利なツールが提供されている。しかしこれらのツールも長時間実行，大量出力，大容量メモリサイズなど計算センターの一般運用から外れてしまうような大型コードにおいては事実上使用できない。従ってデバッグ用出力文によるデバッグ方式を採らざるを得ず，しかもこれが思うにまかせないなど大型コードのデバッグではその情報を収集することにさえ難渋することが多い。第3章で述べられるSDEBUG（Statistic DEBUGer）はこれらの問題を解決するために開発されたツールで，プログラム実行中の任意の時点またはエラー発生時のメモリ・ダンプを採収し，それを利用者が会話型

式で検索し，必要な情報を取り出すものである。

プログラムの開発，保守，管理に付随する作業を手作業で行うには，あまりにも膨大な時間と手間がかかるものが多い。このためプログラム開発者はその目的に応じた様々なツールを作成している。その中から共通に利用できるものがいくつか計算センターに登録されており使用可能である。第4章では登録されているツールの中から使用すると便利であると考えられるものを選び出し紹介している。読者の中には既に御存知の方も多いと思われるが，より多くの方々にツールの存在と機能を理解していただくためここに掲載したものである。

2. コードのデバックについて

2.1 はじめに

"デバッグ"という言葉から思いつく作業は，通常，ソース・プログラムのトレースとか，変数内容のベタ打ち（ダンプ）であろう。事実，そういった作業がコードを正常に動かすまでに何度も繰返され，コード開発の大きな部分を占めている。しかしながら"正常"という言葉は，コード開発の各段階によって内容が違ってくる。開発の初期では，零除算，オーバーフローなどのエラーがなくなり，テスト・データに対して正しい結果を出すことであろう。狭い意味でのデバッグはこの段階を通過するための作業を意味すると思われるが，さらに，考えをすすめると，開発されたコードが実用上のパラメータに対して正しいと思われる（絶対に正しいかどうかは，場合によっては永遠に謎であろう）結果を与えるかどうか調べることもデバッグの一種と考えられる。特に，原研のような研究機関で開発されるコードは，ブラックボックスとして使われるよりもむしろ対象とする現象に応じて絶えず変更・拡張されながら開発されることが多く，後者の意味のデバッグが日常的となろう。このような観点から，"デバッグ"という作業を見直し，この作業を効率良く行うためのソフトウェア（デバッグ・ツール）の利用法について考察してみたい。

2.2 デバッグの種類

デバッグにはコードの開発段階に応じていくつかの種類がある。

(1) 言語レベルでのデバッグ

コンパイラにより言語文法の誤りを見つける作業で，一昔前には，この作業がデバックの大きな部分を占めていたが，言語処理系のエラーメッセージが良くなったため，現在ではほとんど負担にならなくなった。また，全画面方式の端末機の普及も大きな手助けになっている。

(2) 結合・編集時のデバッグ

コンパイラが作成したオブジェクト・モジュールを結合・編集してロード・モジュールを作成する時の誤りを見つける作業で，通常は，あまり意識されない。この段階で発見される誤りの多くは，ソース・プログラム中の配列宣言忘れであるが，大きなコードになると，いくつかのライブラリを結合する時，同じ名前のモジュールがいくつかのライブラリに存在する場合には，結合の順序によって違う実行結果になることがある。（もうすこしリンケージ・エディタのエラー出力が見易いと良いと思う時がある。）

(3) 実行時のデバッグ　その1

領域侵害によるデータ部，命令部への侵害，オーバーフロー，零除算等，明らかにプログラムが正常に動作しない時のデバッグである。原因は，例えば，配列サイズのオーバー，変数の未定義使用，引数対応の誤まり，COMMON 変数の対応の誤まり等が多い。デバッグの手段と

しては，エラーメッセージと共に出力されるトレースバッグ情報の利用，変数のダンプ等，丹念にプログラムを調べることであろう。デバッグツールとしては，変数のダンプツール[1]等が考えられる。

(4) 実行時のデバッグ その2

(1)から(3)の段階を経て，プログラムが結果を出しても，その結果が正しいとは限らない。このような場合が特に厄介で，単純なコーディングの誤りから，時には，数値解法や解くべき方程式の妥当性まで検討しなければならない。結局は，プログラムを丹念に調べたり変数の内容をダンプして誤りを見つけることになるが，その作業をできるだけ効率良く行う上で有効なデバッグツールについて述べたい。

2.3 デバッグを容易にするために

ここでは，プログラムの生産性を高めるという広い意味でのデバッグを効率良く行う上で有用である手段やソフトウェアについて述べる。この課題については，個人によってプログラム作成時の思想が違うと思われるが，ここでは筆者の経験と手持ちのソフトウェアについて述べたい。

2.3.1 プログラム構造の標準化

プログラム構造の標準化自体は直接デバッグと関係ないが，この作業をちゃんとやっておくとプログラムのトレースがしやすく，デバッグの効率が上るだけでなく，変更や拡張の際に役立つことが多い。標準化のしかたは，各人によって好みがあろうが，ここでは，参考までに筆者が外注でソフトウェアを作成する時に使う一種のガイドラインを思いつくままにあげる。

(1) （天才的）スパゲッティ化の禁止

不用意なGOTO文やＩＦ 文の禁止，計算単位でのブロック化。但し，ブロック化をあまり強くすると，ベクトル化がしにくいプログラムになるので注意を要する。

(2) 入出力部，ディフォルト値設定部の集中化

データ入力はNAMELＩST付き入力文を使い，複数個の入力文があっても一つのサブルーチンで行う。また，出力も極力一つのサブルーチンで行い，特に，外部ファイルとの入出力関係を明確にする。ディフォルト値の設定も一つのサブルーチンで行う。

(3) 複雑な引数関係の禁止

ルーチン間のデータの受渡しは，なるべくCOMMON ブロックを使い，一つの配列を切り分けて使うことは絶対にやめる。下位ルーチンへの引数の単純な引継ぎは避ける。配列の切り分け使用は，昔のプログラムではメモリサロズを変更するのに良く使われた技法であるが，FORTRAN７７ のＩNCLUDE機能やEOS[2] 等のプリプロセッサの利用により意味がなくなった。むしろ，下位ルーチンへのデータの引渡し法としてはデバッグしにくい方法といえる。

(4) 変数名の制限

変数名はできるだけ暗黙の型宣言に従う。また，COMMON 変数名と局所的変数名の先頭文字を区別し，ルーチン間のデータ受渡しが追跡しやすいよう配慮する。例えば，OLYMPUSシ

ステム[3]のように，局所的な実数はZで始まり，整数はI，Jで始まるというふうに，コンパイラで許される制限より厳しくした方が良いと思われる。

(5) ディフォルト値一覧の出力

プログラムがある程度完成したら，図2.1に示すような，ディフォルト値の一覧表の出力ルーチンをつけておくとよい。これは，古いロードモジュールのディフォルト値がどうなっているかとかメッシュサイズがいくらまで使えるとかが，そのロードモジュールの先頭を実行しただけでわかり，ロードモジュールの管理に便利である。

2.3.2 プログラムのトレースについて

プログラムのトレースには，大別してソース・プログラムでのトレースと色々なデータを使ったパラメータ・サーベイによるトレースがある。結果が明らかにおかしい時は，前者のトレースとか変数のダンプが有効であり，オーダー的には良いが何となくおかしく感じられる時は（これが一番頭を悩ますのであるが）後者のトレースが有効である。

ここでは，これらの作業のためのソフトウェアとしてソース・プログラム解析コードPLUTO[4]と図形処理コードARGUS[5]の例を示す。

表2.1にPLUTOの機能一覧を示すが，このコードは，ソース・プログラム解析機能としてはANALYSIS 77[6]と良く似た機能を持っているが，更に文書化機能を追加し，機械的な解析では出来ない面を利用者の対話的介入で補おうという思想で作られたコードである。図2.2～図2.6にPLUTOの出力例を示す。

また，パラメータ・サーベイによるトレースには結果の図形出力を比較するのが良いデバッグ法となる。特に，一つのデータから，1次元的断面を描いてみたり，等高線や鳥瞰図を描いてみたり，複数個の結果を並べて比較することによって出力結果のパラメータ依存性を視覚的に知ることができ，コードが正しく動いているかどうかの判定に役立つ。もちろんこの方法は，本来の解析にも強力な手段となる。表2.2にARGUSの機能一覧を，図2.7～図2.8にプログラム例と出力例を示す。

2.3.3 ベンチマーク・テスト

プログラムの成否を調べる一つの方法は，厳密解や既に得られている他の結果との比較をすることである。ある問題に関する専門的な経験の蓄積は，いかなるソフトウエアよりも強力なデバッグ手段となろう。あるコードを拡張し，必要条件として既に吟味されている結果との比較をする場合，関連コードを使った再計算や既にあるデータを参照する必要がある。知識の蓄積を系統的に管理するためには，コードや計算結果をデータベースとして保管管理しておくと便利である。特に，標準化コードと標準データセットを用意しておき，拡張コードで奇妙な結果が出た時，いつでも元に戻れるよう配慮しておくことは，コードのデバッグのみならず"現象解析のデバッグ"にとっても重要なことであろう。このような目的には，コード管理システムHARMONIA[7]，大容量数値データベースGAEAが利用できる。図2.9～図2.10にこれらの概念図を示すが，詳しくは，文献7，8を参照していただきたい。

2.4 まとめ

本稿では，コードのデバッグ一般について考察してみた。今迄に色々なコードを開発しその都度色々な方法でデバッグを行うと共に，各種のデバッグツールを利用，開発してきた。ソース・プログラムの解析やダンプツールなどは，確かに強力な道具ではあるが，一つの分野で長期的に研究をする場合，長い目で見れば，コードの標準化や過去のデータの蓄積からなる標準データベースを構築することがコード開発の際に非常に有効である。コードやデータの標準化とデバッグツールとのうまい組み合わせが相乗作用として働き，デバッグの効率を向上させると思われる。

常松　俊秀，竹田　辰興，徳田　伸二
（核融合研究部理論解析研究室）
安達　政夫
（日本ソフトウェア開発株式会社）

参考文献

1) 平山，静的デバッグ・ツール　SDEBUG，本報告書第3章
2) 竹田，常松，栗田，可変サイズ・プログラムのためのプリプロセッサ・システム EOS，　JAERI－M82－097（1982）．
拡張版についてはマニュアル準備中
3) K. V. Roberts，Comput．Phys．Communn．7（1974）237.
4) 常松，竹田，安達，プログラム文書化コードPLUTO，マニュアル準備中．
5) T. Takeda，T. Tsunematsu，S. Tokuda，Comput．Phys．Communn．34（1984）15.
6) 奈良岡，　FORTRAN 静的解析プログラムANALYSIS77，Computer 情報 No. 40.
7) 竹田，常松，岡田，TRITONシステム管理コードHARMONIA－F75　JAERI－M 9096（1980）．
新版マニュアルは準備中
8) 原子力コード総合化専門部会編，最近の原研コード・システム，JAERI－M82－208（1982）．

表 2.1　　ＰＬＵＴＯの機能一覧

Table 2.1　　Commands of PLUTO system

機能（コマンド名）	説　　明
ＳＴＲＵＣＴ	ソース・プログラムの木構造の解析
ＣＯＭＲＥＦ	ＣＯＭＭＯＮブロック、変数の相互参照解析
ＳＵＢＡＮＬ	プログラム単位毎のデータ受渡しの解析
ＡＲＧＲＥＦ	仮引数、実引数の対応関係の解析
ＭＶＳＵＢＳ	プログラムの構造的な転送
ＤＯＣＭＮＴ	プログラム単位毎の管理情報作成
ＣＯＭＰＡＲ	複数個のプログラムの比較
ＣＭＨＥＬＰ	使用法の説明（ユーティリティ）
ＧＰＳＭＡＮ	＊ＩＮＣＬＵＤＥの処理（ユーティリティ）
ＰＬＵＴＯＩ	マニュアルの作成、印刷
ＤＯＣ	会話処理によるマニュアルの作成、更新
ＰＬＩＯＵＴ	会話処理で作成されたマニュアルの印刷

表 2.2　　ＡＲＧＵＳの機能一覧

Table 2.2　　Functions of ARGUS routines

名称	機能	説　　明
ONEDIM	１次元図	y=f(x;α)の形のデータを作図する。αで指定される数の１次元図を表示する。
ONED3D	１次元図形の３次元表示	y=f(x;α)のかたちのデータを(x,α,y)空間の３次元図形とみなして平行透視図を陰線処理を行って作図する。
MESH2D	２次元メッシュ構成図	節点座標とメッシュ要素との関係テーブルを与えてメッシュ構成図を作図する。
FLOW2D	２次元流れ図	節点上での流れベクトルを作図する。
CONTOR	等高線図	z=f(x,y)なる曲面の等高線図を作図。与えるデータは、x-y格子データ、ランダム・データ、r-θ格子データである。
BRDEYE	鳥瞰図	等高線と同じデータに対して鳥瞰図を作図する。
MESH3D	３次元メッシュ構成図	ある３次元物体を五面体または六面体の要素に分割した時のデータをもとにして、３次元物体を構成し陰線処理をして作図する。作図にあたっては鳥瞰図と同じ透視変換を行う。
CMDISP	コメントのみの図	
HIST3D	３次元柱状グラフ	１価関数z=f(x,y)が規則的に有限個与えられた時、３次元柱状グラフを作図する。透視変換および陰線処理を含む。また、柱状上面にシンボルの作画が可能である。
その他		図の編集機能、合成機能、登録機能、カラー表示機能等。

```
DATE 85-01-31
************************************************************
*  ERATO ---- IDEAL MHD STABILITY ANALYSIS CODE            *
*    THIS CODE IS COMPOSED BY THE FOLLOWING MODULES.       *
*     1. CONSTRUCTING EQUILIBRIUM                          *
*     2. MAPPING MODULE  .....................(ERATO1)     *
*     3. VACUUM FIELD ENERGY CALCULATION ....(ERATOV)      *
*     4. MATRIX CALCULATION .................(ERATO3)     *
************************************************************

++ <<  DOCUMENT  >>  ============================++
II   WHEN  USE  THIS  PROGRAM, PAY ATTENTION TO  II
II  THE  FOLLOWING  MATTERS.                     II
++================================================++

*** THE ALLOWED MAXIMUM SIZES ARE AS FOLLOWS ***
+--------+------------+-------------------------------------------------+
I NPSI   I      101    I NUMBER OF PSI-CONTOUR LINES                      I
I NCHI   I       61    I NUMBER OF POLOIDAL MESH                          I
I NR     I      257    I NUMBER OF R-MESH                                 I
I NZ     I      131    I NUMBER OF Z-MESH                                 I
+--------+------------+-------------------------------------------------+

** I N P U T   D A T A   B Y   N A M E L I S T /NEWRUN/ **          (1/3)
+--------+------------+-------------------------------------------------+
I  VAR.  I  DEFAULT   I           C O N T E N T S                       I
+--------+------------+-------------------------------------------------+
I        I            I      ***  MESH SIZE  ***                        I
I NPSI   I        0   I NUMBER OF PSI-MESH.                             I
I NCHI   I        0   I NUMBER OF POLOIDAL MESH.                        I
I NV     I        0   I NUMBER OF PSI-MESH IN VACUUM FIELD.             I
I        I            I                                                 I
I        I            I DIMENSION SIZE OF NUMERICAL EQ. DATA PSI(R,Z).  I
I NR     I      257   I NUMBER OF R-MESH.                               I
I NZ     I      129   I NUMEBR OF Z-MESH.                               I
+--------+------------+-------------------------------------------------+
I        I            I      ***  CONDITION PARAMETERS  ***            I
I NLGREN I    FALSE   I SELECTION OF VACUUM FIELD ENERGY CALCULATION.   I
I        I            I  IF  NLGREN = TRUE  : GREEN FUNCTION METHOD     I
I        I            I      NLGREN = FALSE : VECTOR POTENTIAL METHOD.  I
I REXT   I 1.000D+00  I =(RADIUS OF CONDUCTOR WALL)/(PLASMA RADIUS).    I
I        I            I  IF  REXT =< 1.0  :  FIXED BOUNDARY CONDITION   I
I        I            I      REXT  > 1.0  :  FREE BOUNDARY CONDITION.   I
I        I            I                                                 I
I NCORD  I        1   I SELECTION OF MAPPING MODULE.                    I
I        I            I  IF  NCORD = 1 : NATURAL CO-ORDINATES           I
I        I            I                  NCORD=1   ---> NATURE=TRUE     I
I        I            I      NCORD = 2 : EQUI-ARC MAPPING               I
I        I            I      NCORD = 3 : PSI-THETA MAPPING.             I
I        I            I                  NCORD=2,3 ---> NATURE=FALSE    I
I        I            I                                                 I
I NTCASE I        4   I SELECTION OF EQUILIBRIUM CASE AND TOROIDAL      I
I        I            I CURRENT                                         I
I        I            I  IF  NTCASE= 1,2 :  SOLOVEV EQUILIBRIUM         I
I        I            I      NTCASE= 3,4 :  NUMERIACL EQUILIBRIUM.      I
I        I            I  IF  NTCASE=1,2,3: JT= R*DIV(GRAD(PSI)/R**2)    I
I        I            I      NTCASE= 4   : JT=-R*D(P)/DPSI-T*D(T)/DPSI/RI
+--------+------------+-------------------------------------------------+
```

図2.1　ディフォルト値一覧の例

Fig. 2. 1　Example of output of default values

```
      1------------2------------3------------4--------

 1    *MAIN01 --.-- PRESET ----- RESETR :
                |
 2              .-- INPUT  ----- PAGE   :
                |
 3              .-- MESH   :
                |
 4              .-- EQUIL  --.-- BLINES :
                |            |
 5              |            .-- IVAR   :
                |            |
 6              |            .-- RVAR   :
                |            |
 7              |            .-- RARRAY ----- BLINES :
                |
 8              .-- MATRIX --.-- MATA   ----- RESETR :
                |            |
 9              |            .-- MATB   ----- RESETR :
                |
10              .-- EIGVAL --.-- EJGSAC :
                             |
11                           .-- BLINES :
                             |
12                           .-- RARRAY ----- BLINES :

     NO REFERENCE
      * MAIN01 *

     UNDEFINED ENTRY
      * EJGSAC *
```

図2.2　　ＰＬＵＴＯの出力例（ＳＴＲＵＣＴ）

Fig. 2. 2　　Example of PLUTO system (STRUCT)

```
<<::: TRITON SYSTEM PLUTO   I  COMMON CROSS REFERENCE LIST  I

   C-NAME   I C C C C  I
            I O O O O  I
            I M M M M  I
            I E E W M  I
            I S Q K A  I
P-NAME      I H U   T  I
------------I----------I
            I          I
* MAIN01 *  I O O O O  I
            I          I
* EIGVAL *  I O O O O  I
            I          I
* EQUIL  *  I O O O O  I
            I          I
* INPUT  *  I O O O O  I
            I          I
* MATA   *  I O O O O  I
            I          I
* MATB   *  I O O O O  I
            I          I
* MATRIX *  I O O O O  I
            I          I
* MESH   *  I O O O O  I
            I          I
* PRESET *  I O O O O  I
            I          I
* BLINES *  I          I
            I          I
* RARRAY *  I          I
            I          I
* RESETR *  I          I
            I          I
* RVAR   *  I          I
            I          I
* IVAR   *  I          I
            I          I
* PAGE   *  I          I
            I          I
----------------------
```

図2.3.1　　ＰＬＵＴＯの出力例（ＣＯＭＭＯＮの相互参照表）

Fig. 2. 3. 1　　Example of PLUTO system
(Cross-reference table of COMMON blocks)

```
<<:::   TRITON SYSTEM PLUTO   I  COMMON CORRESPONDING LIST  I


COMMON   /COMESH/

                    0       0       0       0       0       0
* MAIN01 *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* EIGVAL *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* EQUIL  *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* INPUT  *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* MATA   *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* MATB   *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* MATRIX *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* MESH   *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
* PRESET *    RI     ,RIN    ,CDIST  ,SDIST  ,WDIST  ,NMESH
```

図 2.3.2　PLUTOの出力例（COMMONの変数対応表）

Fig. 2.3.2　Example of PLUTO system (Correspondence table of COMMON variables)

```
<<:::   TRITON SYSTEM PLUTO   I  COMMON DEFINISION LIST  I   (COMREF)   :::>>          DATE 84

.BLOCK.  .VARIABLE.  .MODULE.  .ID-SEQ.  .STATEMENT-IMAGE.

/COMEQU/   BTR      * EQUIL  * 00000720  BTR(1)=ZU(1)
                               00000730  BTR(2*NMESH+1)=ZU(1001)
                               00000790  BTR(J)=ZU(I-1)+(ZU(I)-ZU(I-1))*(RI(J)-(ZR-ZDR))/ZDR
           T        * EQUIL  * 00000900  T(I)=RO*RO*Q(I)*BTR(I)
           DT       * EQUIL  * 00000930  DT(I)=RO*RO*(DQ(I)*BTR(I)+Q(I)*ZDBTR)
           Q        * EQUIL  * 00000220  Q(I)=QF(ZR)
           DQ       * EQUIL  * 00000240  DQ(I)=DQF(ZR)
           P        * EQUIL  * 00000230  P(I)=PF(ZR)
                               00000350  P(I)=ZPO*P(I)
           AJZ      * EQUIL  * 00000940  AJZ(I)=RI(I)*ZDBTR+2.0*BTR(I)
           RHO      * EQUIL  * 00001010  RHO(I)=1.0
           GAMMA    * PRESET * 00000230  GAMMA=5.0/3.0
           RO       * PRESET * 00000320  RO=5.0
           P1       * PRESET * 00000250  P1=2.0
           P2       * PRESET * 00000260  P2=1.0
           PW       * PRESET * 00000270  PW=0.0
           QO       * PRESET * 00000280  QO=1.0
           Q1       * PRESET * 00000290  Q1=1.0
           Q2       * PRESET * 00000300  Q2=1.0
           Q3       * PRESET * 00000310  Q3=0.0
           Q4       * PRESET * 00000311  Q4=1.0
           BETAP    * PRESET * 00000240  BETAP=1.0
           M        * PRESET * 00000120  M=1
           N        * PRESET * 00000130  N=-1
```

図 2.3.3　PLUTOの出力例（COMMONの変数定義）

Fig. 2.3.3　Example of PLUTO system (Definitionof COMMON variables)

```
<<:::  TRITON SYSTEM PLOUT I  ARGUMENT CORRESPONDING LIST  I

|----------|
|  MATA    |  SUBROUTINE
|----------|

 PROG NAME    LINE NO   DUMMY ARGUMENTS  /  ACTUAL ARGUMENTS

                        KINT     K      -----仮引数

* MATRIX *          53  JINT     1      -----実引数

* MATRIX *          59  JINT     2

* MATRIX *          65  JINT     3
```

図 2.4.1　　ＰＬＵＴＯの出力例（仮引数と実引数の対応）

Fig. 2.4.1　　Example of PLUTO system
(Correspondence of arguments)

```
    ARGUMENT TRANSFER LIST
|---------|
| FUNC    |  FUNCTION
|---------|
REF-SUB NAME  LINE NO   DUMMY ARGUMENTS / ACTUAL ARGUMENTS
                          A        B        C        D        ...仮引数
* SUB1    *              (03)  3番目の実引数 (05)              ...
* SUB2    *                       (01)             (02)       ...
```

図 2.4.2　　ＰＬＵＴＯの出力例（仮引数の下位ルーチンへの受け渡し）

Fig. 2.4.2　　Example of PLUTO system
(Argument transfer table)

```
((:::   TRITON SYSTEM PLUTO   (  PROGRAM COMPARE LIST  )   (COMPAR)   :::))

                    * SETUP2 *  1 ST                                    * SETUP2 *  2 ND

MARK LINE NO        SOURCE CODE                                 LINE NO      SOURCE CODE

 =   (   224)       MUV(M)=PRV/HDV(M)**GAM                      (   246)     MUV(M)=PRV/HDV(M)**GAM
 R   (   225)*      NUV(M)=SDV(M)/HDV(M)                        (   247)     NUV(M)=SDV(M)/HDV(M)
 R   (   226)       NUV(M)=1.0/(4.*PI*PI*QQV(M))                (   248)*    NUV(M)=1.0/(4.*PI*PI*QQV(M))
 =   (   227) 510   CONTINUE                                    (   249) 510 CONTINUE
 I                                                              (   250)     IF(NFCT.LT.0) THEN
 I                                                              (   251)     SMBV(NBV)=0.
 I                                                              (   252)     DO 515 N=2,NBV
 I                                                              (   253)     JJ=NBV-N+1
 I                                                              (   254)     SMBV(JJ)=SMBV(JJ+1)+0.5*(SPB(JJ)+SPB(JJ+1))*
 I                                                              (   255) 515 CONTINUE
 I                                                              (   256)     DO 516 N=1,NBV-1
 I                                                              (   257)     SMV(2*N)=(SMBV(N)+SMBV(N+1))*0.5
 I                                                              (   258)     SMV(2*N-1)=SMBV(N)
 I                                                              (   259) 516 CONTINUE
 I                                                              (   260)     SMV(NV)=SMBV(NBV)
 I                                                              (   261)     ENDIF
 =   (   228)       TSMV=0.                                     (   262)     TSMV=0.
 =   (   229)       DO 520 N=1,NV                               (   263)     DO 520 N=1,NV
 D   (   230)       X=DFLOAT(N-1)/DFLOAT(NVM)
 =   (   231)       IF(N.EQ.1)HIV(N)=0.                         (   264)     IF(N.EQ.1)HIV(N)=0.
 =   (   232)       IF(N.GT.1)HIV(N)=HIV(N-1)                   (   265)     IF(N.GT.1)HIV(N)=HIV(N-1)
 =   (   233)      -              -0.5*(1./NUV(N)+1./NUV(N-1))*DSI (   266)    -              -0.5*(1./NUV(N)+1./NUV(N-1))*DSI
 I                                                              (   267)     IF(NFCT.GE.0) THEN
 I                                                              (   268)     X=DFLOAT(N-1)/DFLOAT(NVM)
 =   (   234)       SMV(N)=(1.D0-X**CPS(2))**CPS(3)             (   269)     SMV(N)=(1.D0-X**CPS(2))**CPS(3)
 I                                                              (   270)     ENDIF
 =   (   235)       IF(N.GT.1)TSMV=TSMV+0.5*(SMV(N)+SMV(N-1))*(V (   271)    IF(N.GT.1)TSMV=TSMV+0.5*(SMV(N)+SMV(N-1))*(V
 =   (   236) 520   CONTINUE                                    (   272) 520 CONTINUE
 =   (   237)       TSMV=(TSMV/VLV(NV))/(((BTV/RMAJ)**2)/2.)    (   273)     TSMV=(TSMV/VLV(NV))/(((BTV/RMAJ)**2)/2.)
 =   (   238)       DO 530 N=1,NV                               (   274)     DO 530 N=1,NV
 =   (   239) 530   SMV(N)=CPS(1)/TSMV*SMV(N)                  (   275) 530 SMV(N)=CPS(1)/TSMV*SMV(N)
 =   (   240)C-------------------------------------------------- (   276)C--------------------------------------------------
 =   (   241)       DO 540 N=1,NV                               (   277)     DO 540 N=1,NV
 =   (   242)       SAVE(N, 1)=HIV(N)                           (   278)     SAVE(N, 1)=HIV(N)
 =   (   243)       SAVE(N, 2)=HDV(N)                           (   279)     SAVE(N, 2)=HDV(N)
 =   (   244)       SAVE(N, 3)=SIV(N)                           (   280)     SAVE(N, 3)=SIV(N)
 =   (   245)       SAVE(N, 4)=SDV(N)                           (   281)     SAVE(N, 4)=SDV(N)
 =   (   246)       SAVE(N, 5)=SIW(N)                           (   282)     SAVE(N, 5)=SIW(N)
 =   (   247)       SAVE(N, 6)=SDW(N)                           (   283)     SAVE(N, 6)=SDW(N)
 =   (   248)       SAVE(N, 7)=VLV(N)                           (   284)     SAVE(N, 7)=VLV(N)
 =   (   249)       SAVE(N, 8)=CKV(N)                           (   285)     SAVE(N, 8)=CKV(N)
 =   (   250)       SAVE(N, 9)=SSV(N)                           (   286)     SAVE(N, 9)=SSV(N)
 =   (   251)       SAVE(N,10)=AAV(N)                           (   287)     SAVE(N,10)=AAV(N)
 =   (   252)       SAVE(N,11)=RRV(N)                           (   288)     SAVE(N,11)=RRV(N)
 =   (   253)       SAVE(N,12)=PDS(N)                           (   289)     SAVE(N,12)=PDS(N)
 =   (   254) 540   SAVE(N,13)=FDS(N)                          (   290) 540 SAVE(N,13)=FDS(N)
 =   (   255)C-------------------------------------------------- (   291)C--------------------------------------------------
 =   (   256)       CALL CLOCKM(MTIME)                          (   292)     CALL CLOCKM(MTIME)
 =   (   257)       SEC=1.D-03*DFLOAT(MTIME-MTIMEO)             (   293)     SEC=1.D-03*DFLOAT(MTIME-MTIMEO)
```

図2.5　PLUTOの出力例（ソース・プログラムの比較）

Fig. 2.5　Example of PLUTO system (Comparison of source programs)

```
<<:::   TRITON SYSTEM PLUTO   I  INSERTED DOCUMENT  LIST  I   (DOCMNT)   :::>>

PROGRAM NAME = 'MAIN01'   MODE = UPD
*.....*..*.........*.........*.........*.........*.........*.........*.........*

C*DOC*/S=================   DOCUMENTATION BY PLUTO   ==================00000100
C*                                                                     00000200
C*                                                                     00000300
C*      * MAIN01 -                                                     00000400
C*        - PRESET :                                                   00000500
C*          - RESETR :                                                 00000600
C*        - INPUT  :                                                   00000700
C*          - PAGE   :                                                 00000800
C*        - MESH   :                                                   00000900
C*        - EQUIL  :                                                   00001000
C*          - BLINES :                                                 00001100
C*          - IVAR   :                                                 00001200
C*          - RVAR   :                                                 00001300
C*          - RARRAY :                                                 00001400
C*            - BLINES :                                               00001500
C*        - MATRIX :                                                   00001600
C*          - MATA   :                                                 00001700
C*            - RESETR :                                               00001800
C*          - MATB   :                                                 00001900
C*            - RESETR :                                               00002000
C*        - EIGVAL :                                                   00002100
C*          - EJGSAC :                                                 00002200
C*          - BLINES :                                                 00002300
C*          - RARRAY :                                                 00002400
C*            - BLINES :                                               00002500
C*                                                                     00002600
C*            C-NAME|CCCC|                                             00002700
C*                  |0000|                                           00002800
C*                  |MMMM|                                           00002900
C*                  |EEWM|                                           00003000
C*                  |SQKA|                                           00003100
C*      P-NAME      |NU TI|                                          00003200
C*      ----------+----|                                             00003300
C*      * MAIN01 *|00001|                                            00003400
C*      * EIGVAL *|00001|                                            00003500
C*      * EQUIL  *|00001|                                            00003600
C*      * INPUT  *|00001|                                            00003700
C*      * MATA   *|00001|                                            00003800
C*      * MATB   *|00001|                                            00003900
C*      * MATRIX *|00001|                                            00004000
C*      * MESH   *|00001|                                            00004100
C*      * PRESET *|00001|                                            00004200
C*      * BLINES *|    1|                                            00004300
C*      * RARRAY *|    1|                                            00004400
C*      * RESETR *|    1|                                            00004500
C*      * RVAR   *|    1|                                            00004600
C*      * IVAR   *|    1|                                            00004700
C*      * PAGE   *|    1|                                            00004800
C*      ----------------|                                            00004900
C*                                                                   00005000
C*DOC*     * EIGVAL *                                                00005100
C*         * EIGVAL *                                                00005200
C*           WRITE(6,FMT='(///A,1PE12.4)')'EIGENVALUE=',ZEMIN        00005300
C*           WRITE(6,FMT='(//35X,A//5X,A,20X,A,19X,A,20X,A)')'EIGENVECTOR00005400
C*               ','RI','GZR','GZI','GZZ'                           00005500
C*           WRITE(6,FMT='(A,1PE12.4,10X,1PE12.4,10X,1PE12.4,10X,1PE12.4)00005600
C*               ')'',RI(2*I-1),GZR(I),GZI(I),GZZ(I)                 00005700
C*                                                                   00005800
C*         * EQUIL  *                                                00005900
C*           WRITE(6,FMT='(///A,2X,I4)')'(SUB:EQUIL)BTR**2<OATI=',I  00006000
C*           WRITE(6,FMT='(A,40X,A)')'1','************EQUILIBRIUM********00006100
C*               ****'                                               00006200
C*                                                                   00006300
C*                                                                   00006400
C*                                                                   00006500
C*DOC*/E=================   CREATED DATE  84-09-28   ==================00006600
C*                                                                   00006700
```

図 2.6.1　　ＰＬＵＴＯの出力例（文書化機能，主プログラム）

Fig. 2.6.1　　Example of PLUTO system
(Documentation of main program)

```
<<:::   TRITON SYSTEM PLUTO   I  INSERTED DOCUMENT  LIST  I   (DOCMNT)   :::>>

PROGRAM NAME = 'EIGVAL'   MODE = UPD
*.....*..*.........*.........*.........*.........*.........*.........*.........*

C*DOC*/S=================   DOCUMENTATION BY PLUTO   ==================00000100
C*                                                                     00000200
C*                                                                     00000300
C*      INPUT COM                                                      00000400
C*         AV      AX      EV      GZR     GZI     GZZ     M       NMESH    00000500
C*         QV      RI      RO      VEC                                 00000600
C*      OUTPUT COM                                                     00000700
C*         AV      AX      GZR     GZI     GZZ                         00000800
C*      PARAMETER                                                      00000900
C*         AMAT    : EJGSAC                                            00001000
C*         BMAT    : EJGSAC                                            00001100
C*         EV      : EJGSAC  RARRAY                                    00001200
C*         EW1     : EJGSAC                                            00001300
C*         EW2     : EJGSAC                                            00001400
C*         VEC     : EJGSAC                                            00001500
C*         NM3     : EJGSAC                                            00001600
C*         MS      : EJGSAC  RARRAY                                    00001700
C*         NEV     : EJGSAC                                            00001800
C*         NVEC    : EJGSAC                                            00001900
C*                                                                     00002000
C*DOC*/E=================   CREATED DATE  84-09-28   ==================00002100
C*                                                                     00002200
```

図 2.6.2　　ＰＬＵＴＯの出力例（文書化機能，副プログラム）

Fig. 2.6.2　　Example of PLUTO system
(Documentation of subprogram)

```
      CC=(1.0-BB*EE**2)/(1.0+BB)
      DO 50 J=1,NY
      DO 40 I=1,NX
      XX=X(I)
      YY=Y(J)
      PP(I,J)=2.0*(AA/EE)**2
     1       *(0.5*YY**2*(1.0+A(**2)*(1.0+CC*(XX**2-1.0-A(**2)
     2       /(1.0+A(**2))+EE**2*(XX**2-1.0-A(**2)**2/8.0)-1.0
      JJ=(J-1)/5
      JJ5=JJ*5+1
      IF(JJ5.EQ.J) PP1(I,JJ+1)=PP(I,J)
      III=(I-1)/2
      II2=II*2+1
      JJJ=(J-1)/2
      JJ2=JJJ*2+1
      IF((I2.EQ.I.AND.JJ2.EQ.J) PP2(III+1,JJJ+1)=PP(I,J)
   40 CONTINUE
   50 CONTINUE
      DO 70 J=1,NY2
      DO 60 I=1,NX2
      IF(PP2(I,J).GT.1.0) PP2(I,J)=1.0
   60 CONTINUE
   70 CONTINUE
C---------- INITIALIZE ARGUS-V4 ---------------------------------
      CALL ARGUS(0)
C---------- CDS ROUTINES ((STOR1D,STOR2N,STOR3H))----------------
      CALL STOR1D(ID(1),1,NX,7,X,PP1(1,7),41)
      CALL STOR2N(ID(3),NX,NY,X,Y,PP,41)
      CALL STOR3H(ID(5),NX2,NY2,X2,Y2,PP2,21)
C---------- CDS & DSP ROUTINES ((ONEDIM)) -----------------------
      DO 100 I=1,30
      ILINE(I)=I-(I/5)*5
      IF(ILINE(I).EQ.0) ILINE(I)=5
  100 CONTINUE
      CALL KLINE (ID(1),ILINE)
      CALL ONEDIM(ID(1),0)
C---------- CDS & DSP ROUTINES ((ONE03D)) -----------------------
      CALL ALPHA (ID(1),7,Y5(7))
      CALL ONE03D(ID(1),0)
C---------- CDS & DSP ROUTINES ((CONTOR, BROEYE)) ---------------
      HV(1)=-1.0
      DO 200 I=1,20
      HV(I+1)=HV(I)+0.1
  200 CONTINUE
      DO 210 I=1,9
      KIND(I)=1
  210 CONTINUE
      KIND(10)=2
      DO 220 I=11,21
      KIND(I)=3
  220 CONTINUE
      CALL ZSCALE(ID(3),-1.1,1.0)
      CALL HVALUE(ID(3),21,HV)
      CALL HKIND (ID(3),21,KIND)
      CALL CMTOPT(1)
      CALL CONTOR(ID(3),0)
      CALL BROEYE(ID(3),0)
C---------- CDS & DSP ROUTINES (( HIST3D)) ----------------------
      CALL VIEWPH(ID(5),30.0,0.0,-30.0,600.0,200.0,2)
      CALL HIST3D(ID(5),0)
C---------- TERMINATE ARGUS-V4 ----------------------------------
      CALL ARGUS(999)
      STOP
      END
```

図 2.7　　ARGUS のプログラム例

Fig. 2.7　　Example of ARGUS program

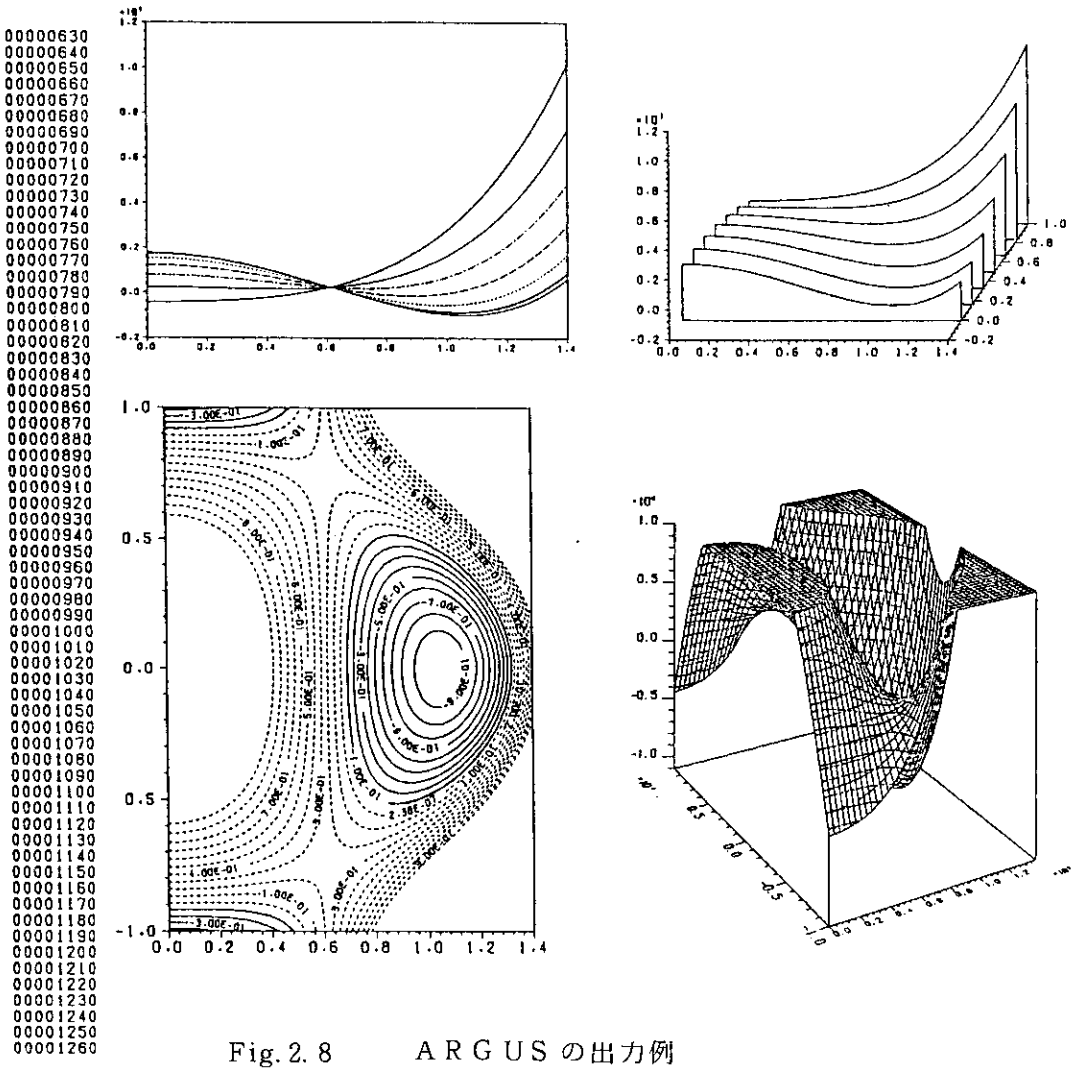


Fig. 2.8　　ARGUS の出力例

Fig. 2.8　　Example of output of ARGUS

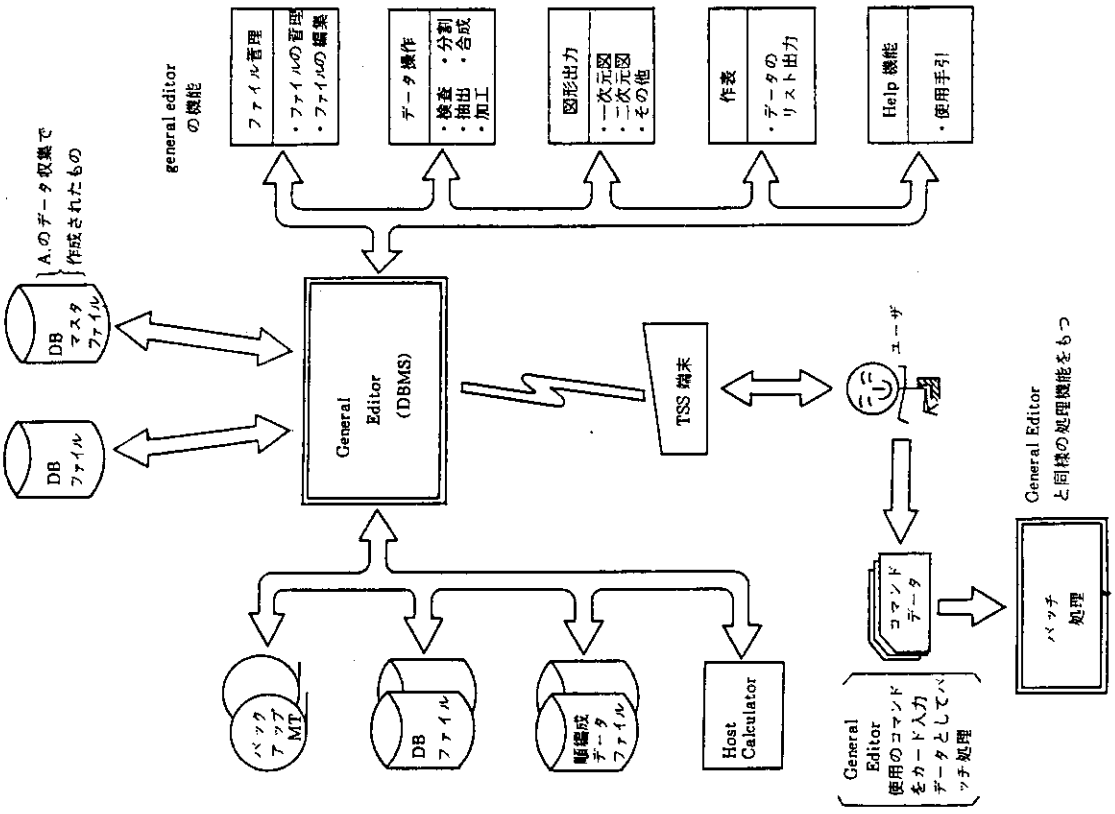


図 2.10　GAEAの概念図

Fig. 2.10　System structure of GAEA

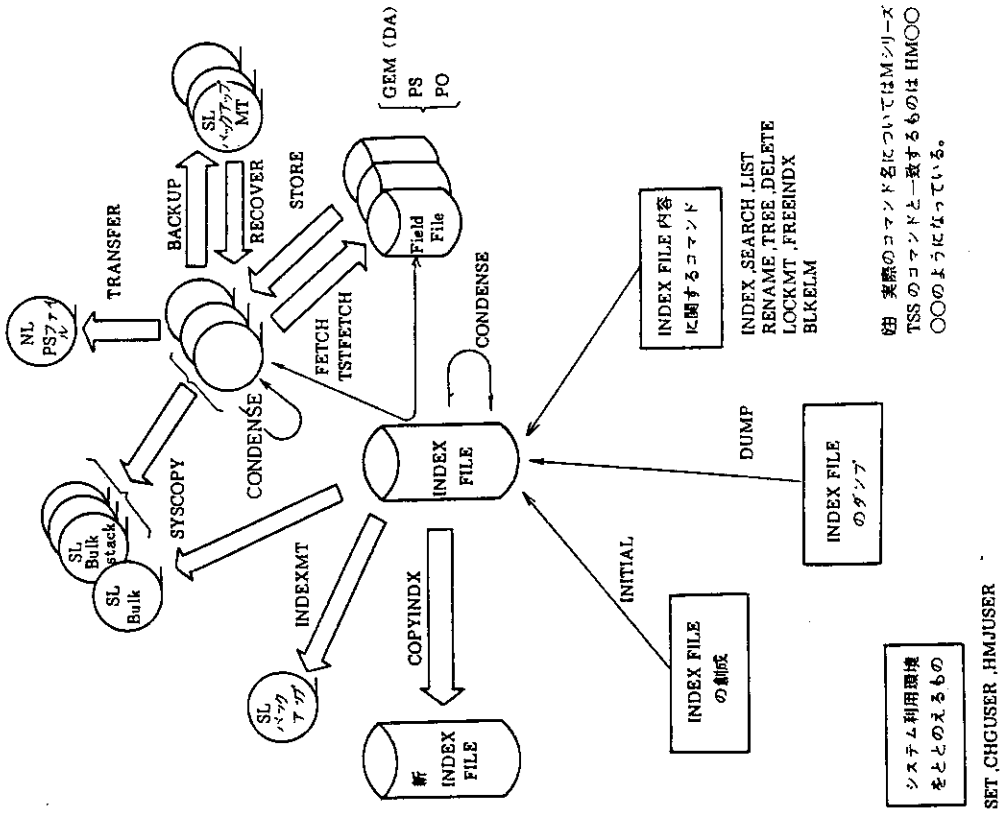


図 2.9　HARMONIAの概念図

Fig. 2.9　System structure of HARMONIA

3. 静的デバッグ・ツール：SDEBUG

3.1 はじめに

プログラムを開発する際に発生する虫（バグ）は多種類にわたり，中にはその発見が著しく困難なものもある。またプログラム（コード）が巨大化するにつれて，デバッグ作業を能率よく行うことが求められている。そのような要請に応えるダイナミックデバッグツールとして，「DOCK」あるいは「TESTFORT77」が既に富士通㈱より提供されている。しかし原研で開発されるプログラムは大容量，長時間実行，大量出力といったものが多く，この大規模プログラムに対しては前記のダイナミックデバッグツールでは事実上デバッグはできない。一方従来通りのデバッグ用出力文を挿入する方法では予想外に大量のデバッグ情報のため問題の部分まで到達しなかったり，的を外したためデバッグ文の位置や内容を変えて再度実行しなければならないなど容易に目的を達せられない場合が多い。以上の問題を解決するために，静的デバッグツール「SDEBUG」を開発した。「SDEBUG」はプログラムの任意の時点（通常はエラー発生時）の計算機のメモリーの内容を静的に走査して虫を発見することを目的としており，ロードモジュールを更新することなく変数の内容を任意に調べることができる。

3.2 SDEBUG の概要

実行しているプログラムにエラーが発生しそのプログラムが異常終了（ABEND)した場合，ABEND 時の計算機のコアの内容を調査してそのエラーの原因を発見しようとするのが本ツールの目的である。コアの内容を直接調査できるように，コンパイルマップとリンクマップを参照して，ABEND 時のメモリダンプ情報を各変数に対応づけてその値を出力するのが「SDEBUG」の機能である。

本ツールのユーザーは各自のデバッグの対象となるプログラムを実行させるJCLの中に，コンパイルマップ，リンクマップそしてメモリーダンプの情報をデータファイルに出力するパラメータを付け加えておけば，エラー発生後，直ちに「SDEBUG」を機動してデバッグ作業を開始できる。本ツールの特徴はただ一度の実行で全モジュールについての変数を順次調べて行ける点にあり，従来のデバッグ用の出力文を付加しては再リンクそして実行といった方法に較べて，能率的である。

「SDEBUG」のデバッグ作業の機能として以下のものがある。

(1) 指定した変数の値の表示
(2) 特定の番地の値の表示
(3) 特定の番地に対応するソースラインの表示
(4) ABEND 時のレジスタの中身の表示
(5) 特定モジュールの先頭番地の表示

(6) 調査対象となっている全モジュールで使用する全変数の値の出力。

3.3 SEEBUG の機能

○プログラムに関する情報

デバッグシステムの概念図を図3.1に示す。「SDEBUG」を使用する際必要となる調査対象プログラムに関する情報の採取をまず行う。プログラムのコンパイル（FORT77)時に，メモリマップリスト，オブジェクトリストそしてソースリストを，結合編集（LKED）時にモジュールマップを，実行（GO)時に調査したい時点でのメモリダンプを採用する。

○情報の整理

採集した情報を編集するための作業ファイルを作成する。作業ファイルには直接編成（ＤＡ）ファイルである。このＤＡファイルは容量が大きく，その初期化に時間がかかるため，あらかじめ初期化されているマスタファイルをコマンドINITMAST を使ってコピーすることにより，作業ファイル作成時間の短縮をはかっている。作業ファイルが既に存在していれば，以上の操作は行う必要はない。

次にメモリマップ等の情報がＤＡファイルに編集される。図3.2に編集内容の概念図を示す。プログラムが大きくなるとこの情報の整理に多くの時間がかかる。従って，エラーの内容から前もって調査しなくてはならないモジュールを選択し，コンパイルマップを限定しておくことがツールを効率よく使う上で必要である。もし当初のモジュールにエラーが無く，それら以外に調査する範囲が広がった場合でも，新たに調査対象となるモジュールのコンパイルマップのみを採取し，ＤＡファイルに追加することができる。

○デバッグ作業

デバッグ作業はSDEBUG コマンドを打つことにより始まる。SDEBUGはいくつかのコマンドを用意している。SDEBUG のコマンド体系を図3.3に示す。以下に各コマンドの内容と例を示す。

なお，SDEBUG 使用にあたっては次の制限事項がある。

a) 被解析プログラムはコンパイルエラーのないこと。
b) オーバーレイ構造のプログラムは保障されない。
c) プログラム内のモジュールの数は１０００個以下。
d) １モジュール内の変数の数は１３３０個以下。
e) FUNCTION の引数についてはその値を出力しない。
f) 特定のエラーについては，ERRORSET 文を使用する。

3.3.1 SDEBUG コマンド

デバッグ・ツールSDEBUG を起動する。

① パラメータ

メモリマップ及びメモリーダンプファイル指定のため表3.1で示す項目をパラメータで指定する。

Table 3.1 Input parameters of SDEBUG command
表3.1 SDEBUGコマンドの入力パラメータ

パラメータ名	内　　容	
FMAP	コンパイルマップ	@@FMAP・DATA
LMAP	リンクマップ	@@LMAP・DATA
MDUMP	メモリーダンプ	@@MDUMP・DATA

ファイルは，その存在がチェックされ，存在していなければ再度入力が要求される。

（例）
```
・SDEBUG FMAP ('J7019.@@FMAP.DATA') +
         LMAP ('J7019.@@LMAP.DATA') + ,
         MDUMP('J7019.@@MDUMP.DATA)
```

② コンパイル情報の追加

SDEBUGコマンドを入力すると次のメッセージが出力されることがある。

```
1=INITIALIZE／2=ADD／3=NO  ADD＝＝＝＞
```

このメッセージは，1度使用したDA作業ファイルを使用する時に表示されるもので1～3の内から選択する。1～3は次の意味をもつ。

1：作業ファイルの中味を全部消して，新しく使用する。

2：調査対象とするモジュールを追加する。

3：現在のDA作業ファイルの情報で再度デバッグ作業を行う。

3.3.2 LISTコマンド

```
≥LIST or L
```

このコマンドはサブコマンドをもち，それらが使用可能な状態（サブコマンドモード）にする。 ＞＞ を表示しサブコマンド入力待ちとなる。

3.3.2.1 SYMBOL サブコマンド

```
>>SYMBOL or SY
```

サブルーチン名とその中に出現する変数名を指定して，それに対応するメモリーダンプの値を出力する。サブルーチン名，変数名，ディメンジョンサイズは必要に応じて次のような形式で質問してくる。

```
ROUTINE NAME＝（ルーチン名）*1
VARIABLE NAME＝（変数名）or （変数名）：（タイプ名）*2
DIMENSION＝（サイズ）*3
値の表示 *4
```

＊1　○　SYMBOL サブコマンドを終了したい時は，何も入力しないでENTER キーを打鍵すると　>>　を表示しコマンド入力待ちとなる。

＊2　○　別のルーチンの検索を行ないたい時は，何も入力しないでENTER キーを打鍵すると次のサブルーチン名の入力を要求してくる。

○　入力した変数の型を陽に指定する場合は，セミコロンで区切ってタイプ名を入力する。タイプ名を指定しなければ，マップ情報に従う。

Table 3.2　Types used in SYMBOL command

表 3.2　SYMBOL コマンドで指定できるタイプ名

Type名	意　味
L＊1	1バイトの論理型
L	4バイトの論理型
I＊2	2バイトの整数型
I	4バイトの整数型
R	実数型
R＊8	倍精度実数型
R＊16	4倍精度実数型
C	複素数型
C＊16	倍精度複素数型
C＊32	4倍精度複素数型
CH＊n	文字型
Z	16進数表示

○　配列変数で変数名のみの入力は全配列の値が出される。

○　配列変数で出力範囲を指定する場合はFOTRAN のDO形並びに従う。ただし，増分値の指定は不可。

また，始値をm_1，終値をm_2とすると

$1 \leq m_1 \leq m_2$　でなくてはならない。

○　配列変数で出力範囲で指定した場合，出力は常に高次元からされる。

（例）（（A（I，J），I＝1，10），J＝1，20）……入力形式

⇓

（（A（I，J），J＝1，20），I＝1，10）……出力形式

＊3　○　DIMENSION は，配列変数の時のみ，入力できる。

○　入力方法はFORTRANの宣言文に従う。

（例）　A（10，20）

＊4　各タイプの出力有効桁数

Table 3.3 Significant figure for each type

表 3.3 各タイプの出力有効桁数

Type	有効桁数
I＊2	5
I＊4	10
R＊4	8
R＊8	8
R＊16	8
C＊8	8
C＊16	8
C＊32	8
L＊1	1
L＊4	1
CH＊n	n

o 配列の場合は常に1次元で出力される。

（例）

```
> L
>> SY
ROUTINE NAME = DUMMY
VARAIBLE NAME = B;CH*80
1 ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
VARIABLE NAME = A
DIMENSION = A(10)
   1             1,   2          2,   3          3,   4          4,
   5             5,   6          6,   7          7,   8          8,
   9             9,  10         10
VARIVBLE NAME = ((A(I,J),J=1,5),I=1,2)
DIMENSION = A(2,5)
   1             1,   2          3,   3          5,   4          7,
   5             9,   6          2,   7          4,   8          6,
   9             8,  10         10
VARIABLE NAME =
ROUTINE NAME = TEST
VARIABLE NAME = AAA
INVALID VARIABLE NAME
VARIABLE NAME = A
DIMENSION = (3)
1 1984/6 TEST RUN¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
2 1984/6 TEST RUN¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
3 1984/6 TEST RUN¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
VARIABLE NAME =
ROUTINE NAME
>>
```

3.3.2.2 LOCATION サブコマンド

>> LOCATION or LO

絶対番地を指定して，それに対応するメモリーダンプの内容を出力する。このサブコマンドを入力すると番地と出力形式の指定を次のように要求してくる。

LOCATION NO.？ （絶対番地）＊1

FORMAT？ （タイプ名） ＊2

値の表示 ＊3

＊1　○　LOCATION サブコマンドを終了したい時には，単にENTER キーを打鍵する。

　　　○　範囲指定ができる。

　　　　　　（開始番地）－（終了番地）

　　　○　番地は8桁以内の16進数で入力する。

　　　○　別の番地を表示したい時は，単にENTER キーを打つ。

＊2　○　指定可能なタイプ名は，SYMBOLサブコマンドと同一。

　　　○　タイプ名がブランクである事は許されない。

＊3　○　出力有効桁数はSYMBOLサブコマンドと同一。

（例）

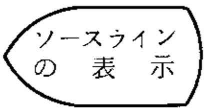
```
>> LO
LOCATION NO.? B5600
FORMAT? CH*80
1 ¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥
FORMAT?
LOCATION NO.?
>>
```

3.3.2.3　STATEMENT サブコマンド

>> STATEMENT or ST

絶対番地を指定して，それに対応するサブルーチン名，ソースラインを出力する。このサブコマンドを入力すると絶対番地の指定を次のように要求してくる。

LOCATION NO. ?（絶対番地）＊1



＊2

＊1　○　STATEMENT サブコマンドを終了する場合は，単にENTER キーを打つ。

　　　○　番地は8桁以内の16進数で入力する。範囲指定は不可。

＊2　○　ソースラインは指定番地の直近上位のものを出力する。

　　　○　ルーチン名とISNと継続行を含めたソースラインを表示する。

　　　○　ブロックデータ，コモン領域の表示は不可。

（例）

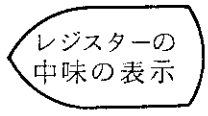
```
>> LO
LOCATION NO.? BD007
ROUTINE NAME = TEST     ISN =     17
      CALL ERRSET ( 209,1,10,2,1)
LOCATION NO.? BD3B2
ROUTINE NAME = TEST     ISN =     21

     DO 111 K=1,10
LOCATION NO.? BD3EA
ROUTINE NAME = TEST     ISN =     23
 111 CONTINUE
LOCATION NO.?
>>
```

3.3.2.4　REGISTER サブコマンド

≫ REGISTER or RE

ABEND 時のレジスターの内容を出力する。



＊1

＊1　○　レジスターは浮動小数点レジスターと汎用レジスターの中味を１０進数と，１６進数で表示する。

（例）

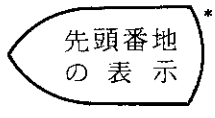
```
>> RE
F REG1   0 4215000000000000   2.100000000+01
F REG1   2 0000000000000000   0.0
F REG1   4 4110000004BDF000   1.00000002D+00
F REG1   6 4E000000000007D0   0.000000000D+15
G REG1    0 B0000000 **********     G REGI    1 800000FF **********
G REGI    2 00000008          8     G REGI    3 00000001           1
G REGI    4 000000FF        255     G REGI    5 00000004           4
G REGI    6 000DDCB0     908464     G REGI    7 000D9F18      392696
G REGI    8 000D9FA8     892840     G REGI    9 00000048          72
G REGI   10 000B07C4     722884     G REGI   11 000D6560      877920
G REGI   12 000C0D40     789824     G REGI   13 000C81C4      319652
G REGI   14 900C117A **********     G REGI   15 00000000           0
>>
```

3.3.2.5　ADDRESSサブコマンド

≫ ADDRESS or AD

モジュールの先頭番地とその長さを出力する。

ROUTINE NAME ？（モジュール名）　＊1



＊2

＊1　○　ADDRESS サブコマンドを終了する時は，単にENTER キーを打つ。

○　モジュール名とは，ルーチン名，コモンラベル名，あるいは，PROGRAM 文で与えた名前である。主プログラムにPROGRAM 文がない時は，モジュール名はMAINとなる。

＊2　○　モジュールの先頭の絶対番地と，その長さが１６進数で出力される。

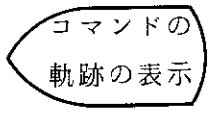
（例）

```
>> AD
ROUTINE NAME? TEST
START ADRESS = 000R2368 LENHGTH = 0000B442
ROUTINE NAME?
>>
```

3.3.3　PRINT コマンド

>PRINT or P

今まで実行した入力コマンドの一覧を出力する。



*1

*1 ○ コマンドに対する出力は表示しない。
入力されたコマンド名とそれらの入力パラメータについてのみ表示される。

○ LISTコマンドのサブコマンドSYMBOL, LOCATION, STATEMENT, REGISTER についてのみ出力される。

(例)

```
> P
 LIST
SYMBOL
DUMMY
B;CH*80
A
A(10)
((A(I,J),J=1,5),I=1,2)
A(2,5)
TEST
A
(3)
STATEMENT
LOCATION
 B5600
  CH*80
STATEMENT
 B5600
 B8700
 BA000
STATEMENT
 BD000
 BD3B2
 BD3EA
REGISTER
```

3.3.4 OUTPUT コマンド

> OUTPUT or O

コンパイルマップファイルに存在するモジュールの全変数の値をラインプリンタに出力する。出力プログラムはバッチジョブで動作する。

HAIRETSU HENSU NO KYOSEI SHITSURYOKU SU? (出力数) *1

ENTER YOUR CURRENT PASSWORD ===> (パスワード) *2

*3

*1 ○ ローカル変数で配列の場合，配列の何個分まで出力するかを入力する。入力した値がその配列より大きくてもチェックはしない。

*2 ○ パスワードを入力する。

*3 ○ バッチジョブで作動するプログラムのJCLがEDIT モードのフルスクリーンで表示されるので，修正後SUBMIT する。

（例）

```
//JCLG JOB
// EXEC JCLG
//SYSIN DD DATA,DLM='++'
// JUSER
 W.1 T.1 I.1 C.2 SRP
 OPTP PASSWORD
// EXEC PGM=OUTPUT
//STEPLIB DD DSN=J7019.DEBUGER.LOAD,DISP=SHR
//FT06F001 DD SYSOUT=*
//FT07F001 DD SYSOUT=*,
//  DCB=(RECFM=VBA,LRECL=137,BLKSIZE=14104)
//FT01F001 DD DSN=J3051.@@FUMKIL.DATA,DISP=SHR
//FT05F001 DD *
10
/*
++
//
```

* 4　OUTPUT コマンドの出力例

```
          ROUTINE NAME =AMOM

DAM   = 9.99999930-04,F     = 9.99998130-01,F0    = 9.99998130-01,I     =         0,X     = 9.99999930-01,Y     =-8.47565260-01,


          ROUTINE NAME =BDFPRC

N     =         0,NSPC1 =         0,NTOTL =      1600,SPN   = 4.00000000+00,


          ROUTINE NAME =BDFT1

FFS   = 0.0           ,IPS   =        31,J     =       401,JCNT  =         0,JN    =       400,JN1   =      1601,JN2   =      2000,
JP    =      2000,JPN   =       400,N     =         4,NN    =         1,NPLMT =      2000,R     = 8.68301190-01,
RFC   = 3.86145800+00,RRS   = 5.91888070-01,SBC   = 4.98132000-04,SKAX  = 9.59956680-01,THT   = 7.81300780-01,TKAI  = 2.00108270-01,
VXX   = 1.58113880+00,WRAND = 0.0           ,XJ    = 3.72694210+00,XRIN  = 1.97330530-01,YJ    = 1.01433870+00,ZZS   = 5.72556460-01,
BDD   =    1  4.66744460-04
FFF   =    1  3.97454430+00
RRR   =    1 -4.79274830-01
VRR   =    1  2.13300450+00
VS    =    1  2.13659710+00
VVV   =    1  4.56504730+00
VZZ   =    1  1.23850670-01
ZZZ   =    1 -8.71529770-01


          ROUTINE NAME =BRSYM

UNSTANDERD ARGUMENT; ORDER=1
IB    =         0,IBX   =         2,IRLN  =         0,IRZ1  =         0,IRZ2  =         0,RG    = 0.0
RGIN  = 0.0           ,


          ROUTINE NAME =BRZUNT

IP    =         0,JP    =         0,NP    =         0,NPI   =         0,NZBMX2=         0,PSIW  = 0.0


          ROUTINE NAME =BZSYM

UNSTANDERD ARGUMENT; ORDER=1
IB    =         0,IBX   =         0,IRZ1  =         0,IRZ2  =         0,RG    = 0.0           ,RGIN  = 0.0


          ROUTINE NAME =CLBLMX

DBFT  = 1.25000000-01,DBRA  = 2.50000000-01,ILL   =         0,ILLT  =         0,KPOINT=         0,M     =         0,
N     =         0,R     = 1.00000000+00,SLMAX = 3.30697930+00,SLMIN = 1.34037040+00,SLX   = 1.50496550+00,SLY   = 3.53038810+00,
THT   = 8.75000000-01,


 BFDX  = 6.48275450+00,BRDX  =-8.20456080-12,BRN1  = 0.0           ,BRN2  = 0.0           ,BRN3  = 0.0           ,BRN4  = 0.0           ,
 BRN5  = 0.0           ,BRN6  = 0.0           ,BZDX  = 8.05025800-01,BZN1  = 0.0           ,BZN2  = 0.0           ,BZN3  = 0.0           ,
 BZN4  = 0.0           ,BZN5  = 0.0           ,BZN6  = 0.0           ,DRR   = 1.06666670+01,DZZ   =-5.44340260-10,N1    =        11,
 UNSTANDERD ARGUMENT; ORDER=1
```

3.4 使 用 例

1) 'J3051.TSSMAC.CLIST(DDT)' を各ユーザーのTSSMAC ファイルに同一メンバ名でコピーする。.DDTと打鍵するよとにより，「SDEBUG」において用いるコマンドの運用が可能となる。

2) ユーザのJCL を修正して，コンパイルマップ，リンクマップ，そしてメモリーダンプの情報をデータファイルに出力できるようにする。図3.4 にその例を示す。ここで，各々のデータファイルは@@FMAP.DATA,@@LMAP.DATA 及び@@MDUMP .DATAの名前で一時ファイルで取られている。図3.4 で示されているJCLは，既に各データファイルが存在し，それらを繰返し使用する場合に対応している。新しくデータファイルを取る時は，コメントになっている部分を使用する。ここで注意すべき点は，EXEC GO文に実行時のオプションとしてABEND を指定する点である。このABEND コードを指定することによって，プログラムのいかなる終了に対してもメモリーダンプが出力される。

3) デバッグ作業

図3.5 に示すプログラムについて，デバッグ作業の例を示す。プログラムはAVTPとタイプする所をAVTI とタイプミスをしている。

a) 2) で述べたJCLの修正を行いBATCH 処理をすると，以下のエラーメッセージが出力される。

(例)

```
JZL209I-E FLOATING DIVIDE EXCEPTION IS DETECTED. OLD PSW=078D000FA014270
    INSTRUCTION=7D00D21C          LOCATION=001427CE
    GR0 =0000007C GR1 =0000007C GR2 =0000007C GR3 =00000001
    GR4 =0000001E GR5 =0000001F GR6 =0000001E GR7 =00000080
    GR8 =00000004 GR9 =0000007C GR10=00155CD1 GR11=00000000
    GR12=00142FF0 GR13=00141FF0 GR14=601427CA GR15=00000000
    FR0 =437D000017878000         FR2 =411CE14746B85200
    FR4 =3F81B4E866C16000         FR6 =0000000000000000
  ERROR OCCOURS AT MAIN      ISN 00025 LOC 001427CE OFFSET 000916
  MAIN       AT LOC 40141EBA CALLED FROM (O.S)
TAKEN TO (STANDARD) CORRECTIVE ACTION, EXECUTION CONTINUNING
    AV.NI =      3.000E+19
    AV.TI =      2.000E+03
    PNEUT =      1.944E-28
```

b) .DDT

.INITMAST (作業用DAファイルの確保)

c)

(例)

```
.SDEBUG
1=INITIALIZE/2=ADD/3=NO ADD ===> 1
> L
>> ST
LOCATION NO.?  1427CE <----------ゼロ DIVIDEが発生した番地の入力
ROUTINE NAME =MAIN     ISN =    25   } 入力番地に対応したソース・ラインの表示
      TIO = AVTPIN / AVTI * 1.E-3 }
0
LOCATION NO.?
>> SY
ROUTINE NAME = MAIN    <---------- Subroutine 名の入力
VARIABLE NAME = AVTI   <---------- 変数名の入力
   1  0.0              <---------- 変数の値を表示
VARIABLE NAME =
ROUTINE NAME =
>> END
> END
```

この結果，AVTI 値の値が0のために生じたエラーであることがわかる。

平　山　俊　雄
(臨界プラズマ実験室)

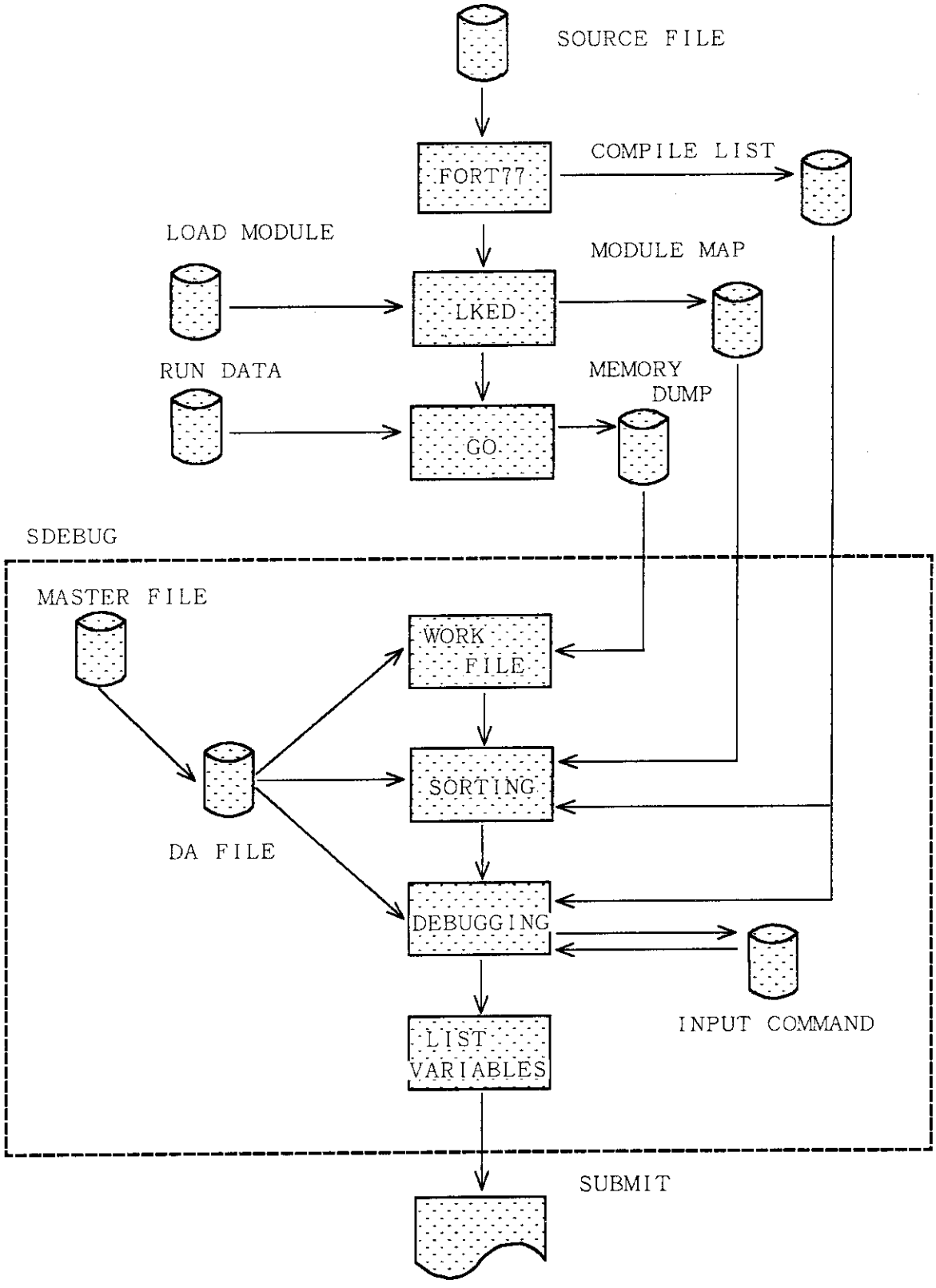


図 3.1　ディバックシステムの概念図

Fig. 3.1　Conceptive figure of debug system

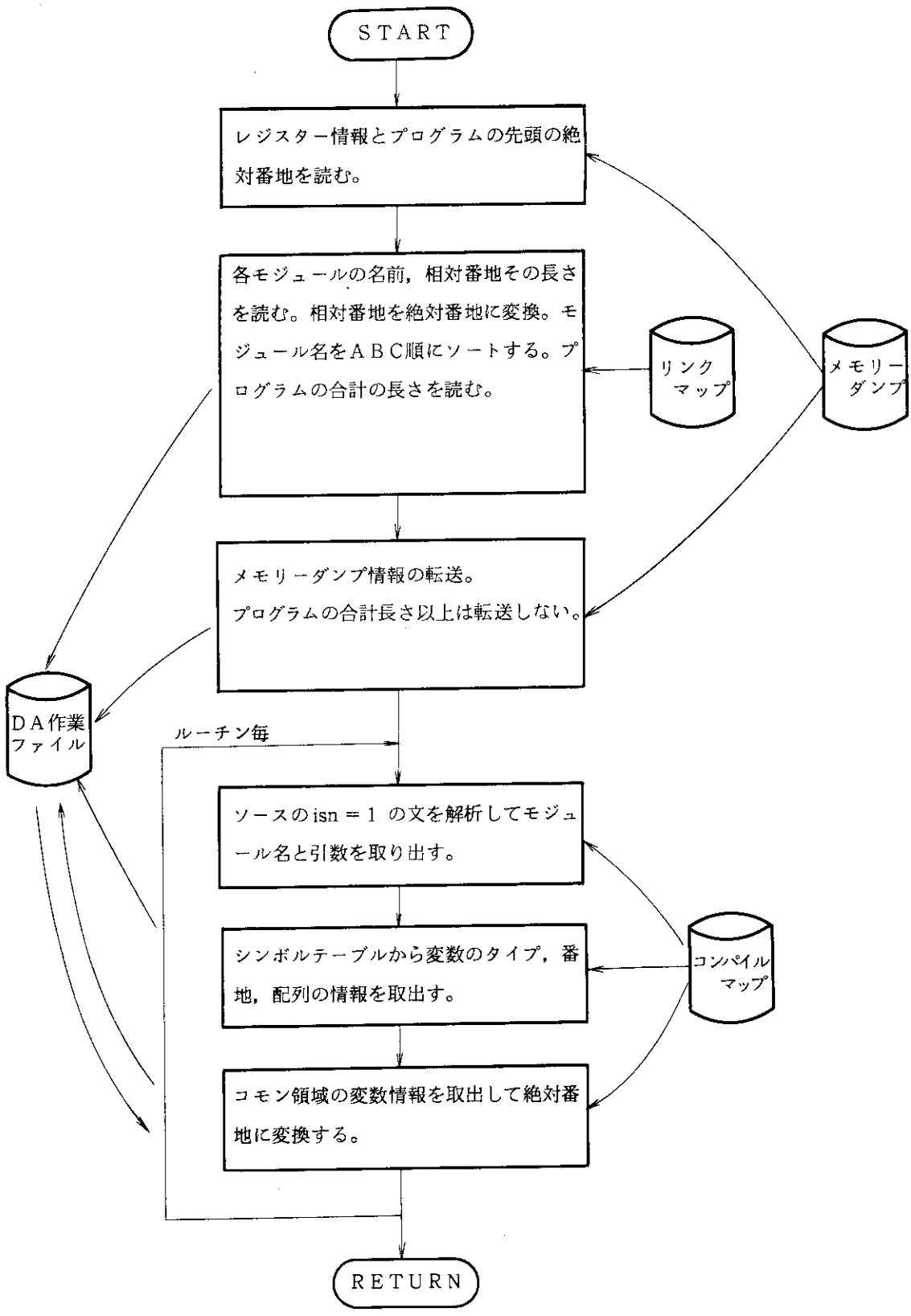


図 3.2　情 報 の 整 理

Fig. 3.2　Sorting information

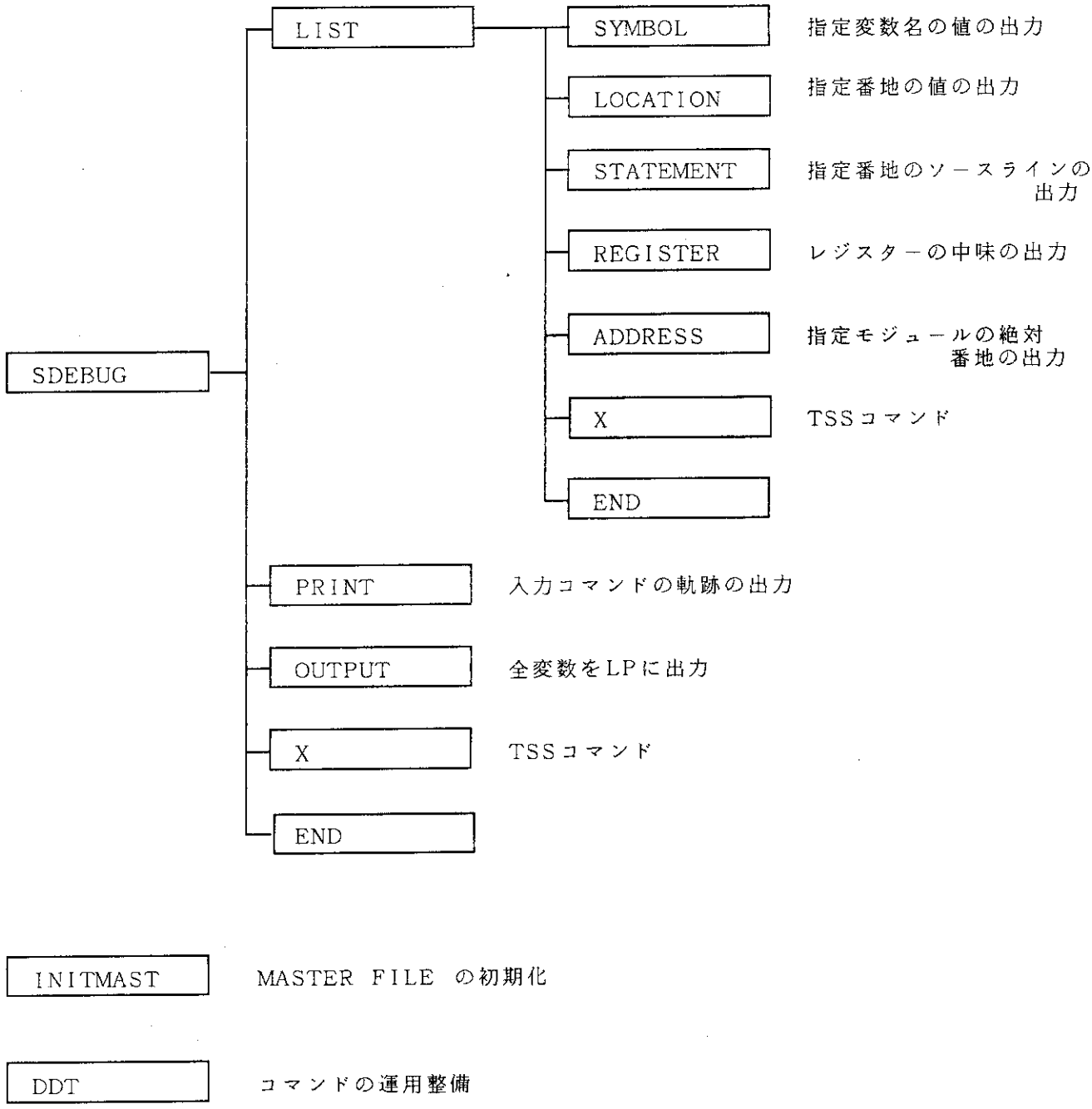


図3.3　コマンド体系

Fig. 3.3　Command system

```
JG.CNTL                        DATE     85/02/15(FRIDAY)      TIME 17:05:28  PAGE 000


**
**
**

  //JCLG   JOB                                                              00000010
  //JCLG   EXEC JCLG                                                        00000020
  //SYSIN  DD DATA,DLM='++'                                                 00000030
  // JUSER 90823051,TO.HIRAYAMA,0012.05                                     00000040
    T.0 C.0 I.1 W.0 SRP                                                   00000050
    OPTP  PASSWORD=                                                       00000060
  //*********************************************  COMPILE  *********    00000070
  // EXEC FORT77,SO='J3051.@NEUTRON',Q='.FORT',                            00000080
  // A='ELM(*) SOURCE LIST',B='MAP'                                        00000090
  //*SYSPRINT DD DISP=(NEW,CATLG),SPACE=(TRK,(100,50),RLSE),              00000100
  //*    UNIT=TSSWK,DCB=(BLKSIZE=19004,LRECL=125,RECFM=VBA),              00000110
  //*    DSN=J3051.@@FMAP.DATA                                            00000120
  //SYSPRINT DD DSN=J3051.@@FMAP.DATA,DISP=SHR                            00000130
  //*********************************************  LINK  ************    00000140
  // EXEC LKED77,                                                         00000150
  //          A='MAP',B='LIST'                                            00000160
  //*SYSPRINT DD DISP=(NEW,CATLG),SPACE=(TRK,(100,50),RLSE),              00000170
  //*    UNIT=TSSWK,DCB=(BLKSIZE=19004,LRECL=125,RECFM=VBA),              00000180
  //*    DSN=J3051.@@LMAP.DATA                                            00000190
  //SYSPRINT DD DSN=J3051.@@LMAP.DATA,DISP=SHR                            00000200
  //*****************************************************  RUN  ******    00000210
  // EXEC GO,A='ABEND'                                                    00000220
  //FT06F001 DD SYSOUT=*                                                 00000231
  //SYSIN DD *                                                           00000240
   3.0E19  2.0E3                                                        00000250
  /*                                                                    00000260
  //*********************************************  SYSUDUMP  ********    00000280
  //*SYSUDUMP DD DISP=(NEW,CATLG),                                       00000290
  //*    SPACE=(TRK,(300,100),RLSE),UNIT=TSSWK,                          00000300
  //*    DCB=(BLKSIZE=19004,LRECL=125,RECFM=VBA),                        00000310
  //*    DSN=J3051.@@MDUMP.DATA                                          00000320
  //SYSUDUMP DD DSN=J3051.@@MDUMP.DATA,DISP=SHR                          00000330
  ++                                                                    00000340
  //                                                                    00000350
```

図3.4　各マップの出力ファイルと得るためのＪＣＬの修正

Fig. 3.4 Revised JCL to take out each map file

```
TRON.FORT                    DATE    85/02/15(FRIDAY)     TIME 18:15:32  PAGE 000


**
**
**

C*****************************************************************        00000100
C*  MAIN ROUTINE TO CALCULATE NEUTRON YIELDING                   *        00000200
C*                                                               *        00000300
C*****************************************************************        00000400
C                                                                          00000500
      COMMON / AAA / DR(50),DP(50),TP(50),COEF(50)                         00000600
C                                                                          00000700
C ------------------------------  INPUT FOR AV. DENSITY AND AV. TEMP.      00000800
      READ(5,2100) AVDPIN,AVTPIN                                           00001100
 2100 FORMAT(2E7.1)                                                        00001200
C-------------------------------------------------------------------------00001300
C                                                                          00001400
C -- ( 1 ) --  PLASMA PARAMETERS                                           00001500
C                                                                          00001600
C ------------------------------   MAJOR RADIUS                            00001700
      RT = 3.03                                                            00001800
      PAI= 3.1415926                                                       00001900
C ------------------------------   MINOR RADIUS                            00002000
      RP = 0.95                                                            00002100
      NP1= 31                                                              00002200
      NM1= NP1 - 1                                                         00002210
      DX = RP/FLOAT(NM1)                                                   00002300
C                                                                          00002400
      DO 100 I=1, NP1                                                      00002500
         DR(I) = DX * FLOAT(I-1)                                           00002600
  100 CONTINUE                                                             00002700
C ------------------------------   PLASMA PARAMETERS                       00002800
      CLT = 2.0                                                            00002900
      CNT = 2.0                                                            00003000
C                                                                          00003100
      CLN = 4.0                                                            00003200
      CNN = 1.0                                                            00003300
C                                                                          00003400
      DO 200 I=1, NM1                                                      00003500
         TP(I) = ( 1.0 - (DR(I)/RP)**CLT )**CNT                            00003600
         DP(I) = ( 1.0 - (DR(I)/RP)**CLN )**CNN                            00003700
  200 CONTINUE                                                             00003800
         TP(NP1) = 0.0                                                     00003810
         DP(NP1) = 0.0                                                     00003820
C                                                                          00003830
      CALL UINTGS( TP, NP1, DR, AVTP )                                     00003900
      CALL UINTGS( DP, NP1, DR, AVDP )                                     00004000
      TIO = AVTPIN / AVTI * 1.E-3                                          00004100
      DNO = AVDPIN / AVDP                                                  00004200
      DO 220 I=1, NP1                                                      00004300
         TP(I) = TIO * TP(I)                                               00004400
         DP(I) = DNO * DP(I)                                               00004500
  220 CONTINUE                                                             00004600
C                                                                          00004700
C ---------------------------   NEUTRON YIELDING                           00004800


TRON.FORT                    DATE    85/02/15(FRIDAY)     TIME 18:15:32  PAGE 000
```

図3.5　虫のいるプログラム例

Fig. 3.5　Example of program with bug

4 プログラムの開発，保守，管理支援ツール

プログラムの開発が企画，開発され，やがてそのプログラムが廃棄されるまでをプログラムのライフ・サイクルと呼んでいる。このライフ・サイクルにおいては開発，実行のみではなくそれに付随する作業が大型コードになればなる程多く存在する。たとえば開発時のデバッグ，機能の変更，追加やプログラム変換のための動的，静的解析，また入出力データセットの管理など枚挙にいとまがない。プログラムのデバッグ，保守，管理のための作業は一般に多くの工数を必要とするから，これを効率よく処理するため計算機利用者により，作業目的に合ったツールが数多く開発されてきた。その中から一般に有効と思われるものが計算センターに登録されている。ここでは種々のツールの存在を認識していただくためプログラムの開発，保守，管理に有効と思われ，よく利用されているものを選択し，機能の概要を示した。この中にはメーカー提供のものも含まれている。具体的な利用法については参考資料に示したマニュアル類を参照されたい。

表4.1　プログラムの開発，保守，管理支援ツール一覧

Table 4.1 Table of tools used for development, debugging, maintenance and management of programs

	No.	プログラム名	機能	備考
ソフトウェア解析ツール	1	ANALYSIS－77	プログラムの静的解析を行い，TREE構造，COMMON参照表等を出力する。	(利) PS，PO，GEMファイルのプログラムが入力可能。
	2	FORTUNE	プログラムの動的解析を行い，実行回数，実行費用等を原始プログラムと共に出力する。	(欠) コンパイルオプションのAUTODBL使用不可。
	3	TOP10	VP化に必要な各情報（DOループのベクトル長，ベクトル化の可否指示記号V，M，S等）を出力する。	FORTUNE出力結果を入力とするが，原始プログラムも可能。ただし，出力項目が制限される。
	4	会話型ベクトライザー	プログラムをベクトル計算機で効率良く実行させるように改善するための作業を支援するシステム。	(欠) メモリ使用量が大きい。 (利) ベクトライザー配下でプログラムの修正ができる。
	5	TESTFORT77	プログラムを端末から，デバッグすることができる デバッグ支援システム。	(欠) オブジェクトモジュールからの実行。 (利) 決まりきったコマンドをコマンドプロシジャーにまかせられる。
	6	DOCK/FORT77	プログラムを端末から，デバッグすることができる デバッグ支援システム。	(利) DOCK/FORT77配下でプログラムの修正ができる。 (利) 実行モードS，L，M，Hが選択できる。
データセット保守管理用ツール	1	LPCOMP	出力結果を比較する。	(利) 値の検査もできる。 0.1E＋01＝1.0
	2	SFCOMP	原始プログラムを比較する。	
	3	JSGCOMPR	データセットの比較を行う。	(利) VBS形式の比較でBLOCK単位で行う。 (利) POファイルの比較が可能
	4	MTDUMP	MTの種々な情報（SL，NL か，ブロックサイズ，ブロック数，ファイル名等）を出力する。	(利) SL，NLの区別なし (利) ブロックNo指定の出力可能
	5	MTLABEL	MTのファイル名等の情報をMTラベルと同じ形に編集してNLPに出力する。	(欠) 入力MTはSLのみ
	6	SFMAKE	PSファイルをEND文字で区切り，それを1メンバーとしてPOかGEMファイルに変換する。	(欠) 入力FILEのブロックサイズは3200バイトまで。 (利) SFMKGEM，SFMKPO コマンドが用意されている。
	7	VIVAPO	POファイルに格能されているプログラムの取り扱いを容易にする。	(欠) 制御文PSTOPS，PSTOPO の入力ファイルは，レコード長80バイトに限る。
	8	MYBACKUP	指定した利用者のすべてのファイルをMTに退避する。	(利) 退避するファイルの範囲を指定できる。
	9	COMPACT	活字の大きさを約1／2に縮小してNLPに出力する。	(利) 印刷形式に4つのタイプがある。
	10	F77INC	インクルードファイル をソースファイルの1メンバーとして登録しておくことができるプリプロセッサ。	
デバック機能	1	エラー処理サブルーチン	実行時のエラー発生に対して，エラーモニタが行う処理を利用者が制御できるように用意されたサブルーチン。	
	2	シンボリックダンプ	実行時に異常終了するか，又はSDFDMPサブルーチンを呼び出した場合に，変数及び配列の値，属性等を出力する。	(利) 配列の出力範囲を指定できるSDFARYオプション又は，SDFARY サブルーチンが用意されている。
その他	1	SLC	FORTRAN HEプログラムの精度拡張を行う。	(欠) PSファイルのみ入力可能
	2	自動再リンケージユーティリティ	入力ロードモジュールの指定されたメンバーに対して，オプション指定された要求に合った結合編集を行う。	(利) オーバレイ構造をもつロードモジュールを結合編集して，再びオーバレイ構造のロードモジュールを作る時，オーバレイ制御文不要。

4.1 ソフトウエア解析ツール

4.1.1 ANALYSIS－77

FORTRANプログラムの静的解析を行い，プログラムの改良，保守作業の効率化を計る。

(イ) 機　　能

以下に示す１１項目のオプション機能が組み込まれている。

(1) 副プログラムの引用を含むFORTRAN文の印刷。

(2) COMMON と副プログラムの参照関係表（クロスリファレンス・テーブル）の印刷。

(3) 副プログラムの親ルーチン，子ルーチンの印刷。

(4) 副プログラム毎に，これを引用しているFORTRAN文と副プログラム名の印刷。

(5) 副プログラム毎に，引用している入出力文の印刷。

(6) 利用者が入力したFORTRAN変数を含むFORTRAN文と，副プログラム名の印刷。

(7) 副プログラム間の参照関係を木構造として印刷。

(8) 仮引数，COMMON 変数の中で陽に値がセットされているもの。或は副プログラムへの実引数として渡されるものがあれば，その旨のメッセージを印刷。

(9) 利用者が入力したFORTRAN文の種類に従って，対応するFORTRAN文とこれを含む副プログラム名を印刷する。

(10) 副プログラム中の注釈文を印刷する。

(11) ＤＯネスト構造，ＩＦ－THEN－ELSE 構造を明示しながら，すべてのFORTRAN文を副プログラム単位に印刷する。

(ロ) 出 力 例

以下に出力例を示す。

(1) 木 構 造

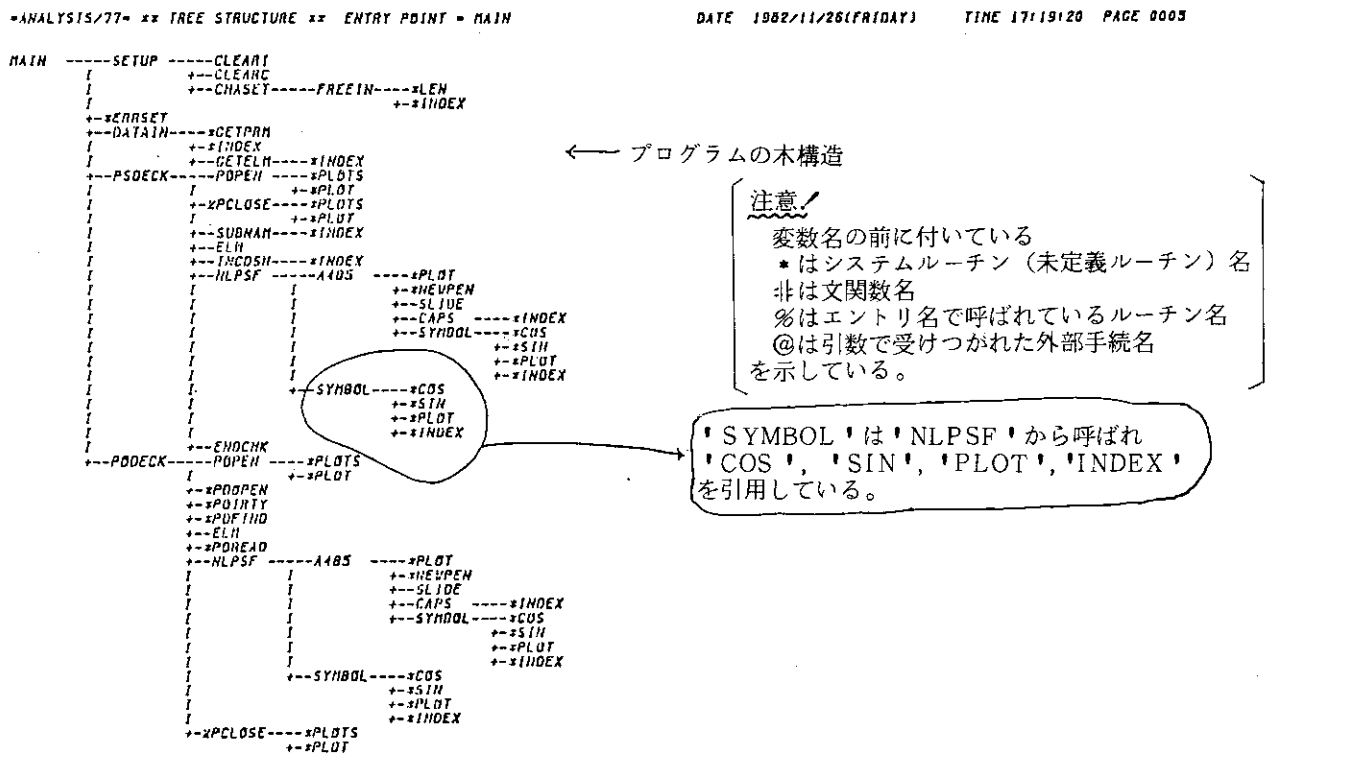


図 4.1　木構造出力例

Fig. 4.1 Example of output of ANALYSIS−77 (Tree structure)

(2) コモン参照表

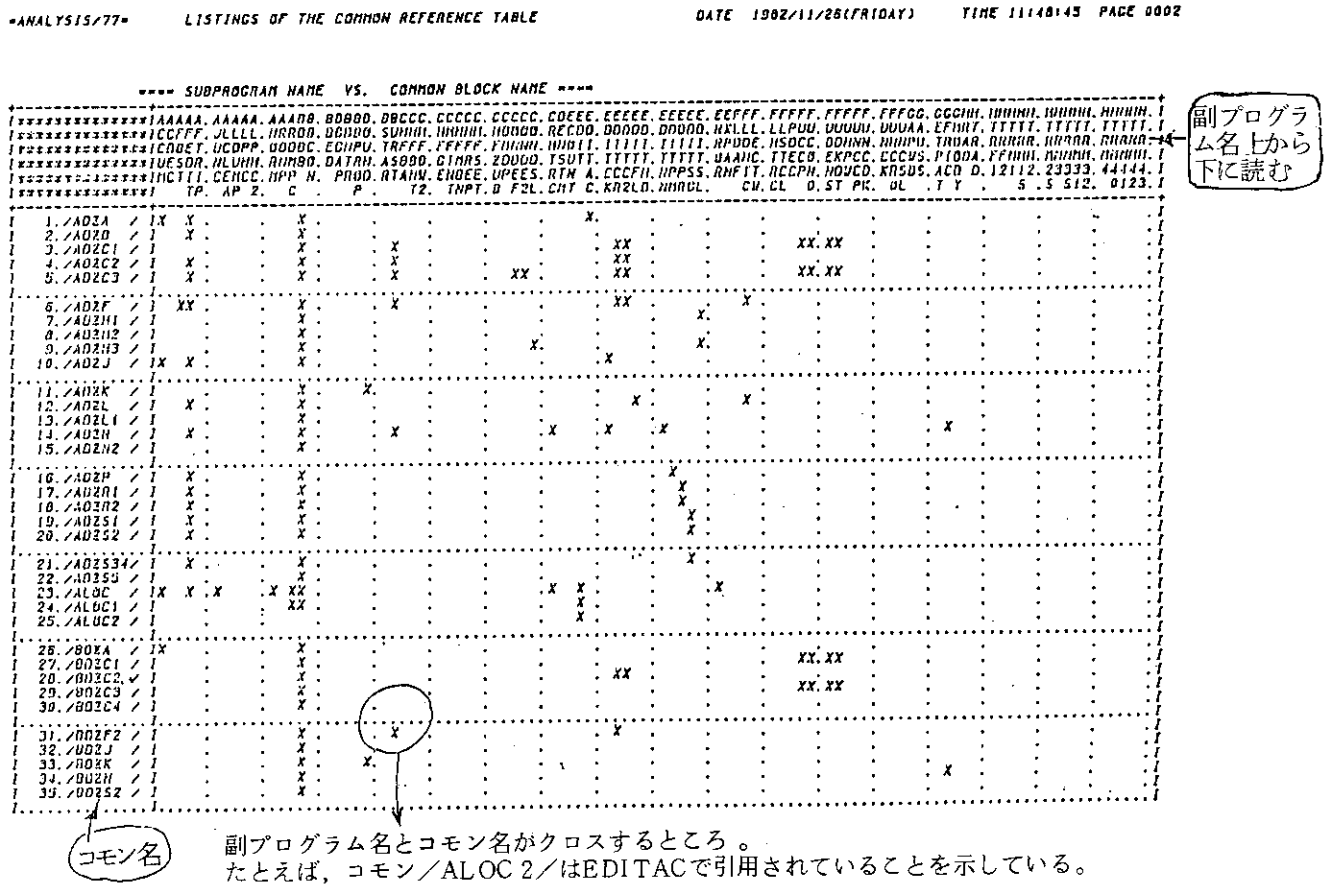


図 4.2　コモン参照表出力例

Fig. 4.2 Example of output of ANALYSIS−77 (Common section reference table)

(ハ) 使用上の注意

(1) 解析可能な副プログラム数は，最高６００である。

(2) 名前付COMMON の種類が３００以内のプログラムが解析可能である。

(3) 実行時のメモリー使用量は約７６０KBである。

(4) 入力ソースファイルが，ＰＯまたはＧＥＭファイルの場合は，制御カード（FT05F001）に，ＥＬＭ(*)または，ＥＬＭ（メンバー名）を指定する。

(5) 解析対象となるソースプログラムは，FACOM FORTRAN-HE，FORTRAN-77 で書かれているものとし，構文的な誤まりは，含まれていないものとする。

(ニ) 補　　足

(1) コマンド，プロシジャー '.ANALYSIS' も用意されている。

(2) ANALYSIS -77 と，ほぼ同等の機能を持ち，出力先をＮＬＰとするDOCTRAN-77 もある。但し，入力ソースファイルは，ＰＳ だけである。

4.1.2 FORTUNE

FORTRAN プログラムの実行時の振舞いを解析し，プログラムの改良あるいはテストに役立つ情報を出力する。

(イ) 機　　能

以下に示す３項目の機能を持っている。

(1) 実行文の実行回数と実行費用[注)] の出力

(2) プログラム単位ごとの実行回数と実行費用の出力

(3) 論理ＩＦ文，ブロックＩＦ文，ELSE-IF文 において論理式が真となる場合の実行回数とその真率の出力

また，FORTUNE オプションにより

(1) 解析するプログラム単位の選択

(2) 利用者出口ルーチンの呼出し

などが行える。

注) 実行費用；単純な代入文を１回実行するのに必要な費用を１としたとき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ＣＰＵ時間のおおまかな目安となるもの。

(ロ) 出 力 例

(1) 文単位の解析情報

```
          FORTUNE  V10L10                    DATE 84.10.27  T          PAGE     1
実行回数
=EXECUTIONS=     =TRUE=          =ISN=       =STATEMENT=                =SEQ=      実行コスト =COST=
                  IF文の真率
                                        C CHANGE OF PARITY              00013800
       224                       00665          JPI=JPI+1               00013900          448
       224          112(50.0%)   00666          IF (JPI.EQ.1) GO TO 9   00014000          560
                                        C INCREASE OF THE TOTAL ANGULAR 00014100
       112                       00667          AJ=AJ+1.                00014200          224
       112                       00668          IPJ=IPJ+1               00014300          224
                                        C REDUCTION OF MAXIMUM NUMBER OF 00014400
       112           0( 0.0%)    00669          IF (LO(125).AND.ITERR.LE.2 00014500       336
                                        C CHECKS OF CONVERGENCE         00014600
       112           0( 0.0%)    00670          IF (LO(128).AND.NML.EQ.0) 00014700        336
       112         102(91.0%)    00671          IF (R1.GE.CONJ.AND.AJ.LT.A 00014800       662
        10                       00672       12 AJMA=AJ-1.              00014900           20
        10           0( 0.0%)    00673          IF (LO(19).AND.LO(32))  00015000           20
        10          10(100.%)    00674          IF (.NOT.(LO(225).AND.LO(2 00015100       10030
        10                       00675       13 NDX=NNC+JTH             00015200           20
        10                       00676          NDY=NDX+JTH             00015300           20
        10                       00677          NMC=2*((NDY+JTX)/2)+1   00015400          120
        10                       00678          ID1=IDMT-NMC            00015500           20
                                        C FOR ARGUMENTS     SEE CALX AND 00015600
        10                       00679          CALL RESUL(W,W(NAI),W(NSPI 00015700        10
                                               1NFM),W(NFM),IPJ-1,W(NDONN) 00015800
                                               2DX),W(NDY),ID1,LO)      00015900
        10                       00680          NPLACE=MAXO(NPLACE,NMC+ID1 00016000        40
        10           0( 0.0%)    00681          IF (LO(84)) CALL HORA   00016100           10
```

図 4.3　文単位の解析情報出力例

Fig. 4.3 Example of output of FORTUNE (Analysis information of statements)

(2) プログラム単位の解析情報

```
          実行回数          実行コスト
=ROUTINE= =EXECUTIONS=       =COST=      %      0.....1.....2.....3.....4....
CALC                1            79952    0.0    I
CALX                2            19675    0.0    I
CAL1               10            55722    0.0    I
COLF               10           480935    0.0    I
COULFN              5             4250    0.0    I
COUPOT              0                    0.0    I
CPCC             1104       1900199412   30.5    I**********************
DEPHAM              1            23220    0.0    I
DERIV               0                    0.0    I
DFCGS           95004          6621940    0.1    I
DFCOUL             55           732788    0.0    I
DFCZO              55             6215    0.0    I
HORA                0                    0.0    I
HULT                0                    0.0    I
INCH              224       4226219896   67.7    I***********************************
INSH                0                    0.0    I
INSI                0                    0.0    I
INTI                0                    0.0    I
INVATA              0                    0.0    I
JLSB2             224          25455582    0.4    I
LECD                0                    0.0    I
LECDE               0                    0.0    I
LECT                1            32368    0.0    I
LGPL                7            46819    0.0    I
LILESQ              0                    0.0    I
*MAIN               1                2    0.0    I
OBSERV              1              134    0.0    I
PADE                0                    0.0    I
POTENT             10            25130    0.0    I
PSI                 0                    0.0    I
QUANT             224          17065318    0.3    I
REARG               0                    0.0    I
REDMAT             10            28060    0.0    I
RESTOR              0                    0.0    I
RESUL              10              730    0.0    I
```

図 4.4　プログラム単位解析情報出力例

Fig. 4.4 Example of output of FORTUNE (Analysis information of subprograms)

この出力例のプログラムでは，実行時間を費やす部分はルーチンCPCCとINCHに局在していることがわかる。したがって，プログラムの処理効率を改善するには，このルーチンを中心に改善すればその効果は大きい。

(ハ) 使用上の注意

(1) 被解析プログラムは，FACOM OSⅣのFORTRAN及びFORTRAN77で作られていること。また，原則として翻訳及び実行時に誤りがないものとする。

(2) 先頭の3文字が'JZZ'である英字名を使用していないこと。

(3) IF文，DO文，CALL文の数に対応するカウントの数が最大4096個である。

(4) プログラム中の末端にIF文があるDO LOOPがないこと。DO LOOPの末端にIF文があるときは，末端にCONTINUE文がくるように変更する。

(5) コンパイルオプションのAUTODBLは，指定できない。但し，原始プログラムがFORTRAN-HEの文法に従っている場合は，精度拡張ユーティリティSLCを使用し原始プログラム自身を倍精度に変換した後，FORTUNEの入力とすれば良い。

4.1.3 TOP10

FORTRANプログラムのベクトル化作業を効率良く行うために，副プログラム間の参照関係を考慮した巨視的な解析情報を提供する。

(イ) 機　能

FORTUNE結果が解析対象のとき，以下の9項目のオプション機能を持っている。

(1) DOループ構造，IF-THEN-ELSE構造の範囲およびループの深さを明示する。

(2) 副プログラムの引用部分にアンダーラインを引き見やすくする。

(3) FORTRAN77/VPコンパイラによるベクトル化の可否を示す記号（V，M，S）をソースリスト上に付加する。。

(4) DOループ単位の実行コスト（FORTUNEコスト）の算出を行う。

(5) DOループの平均回転長（ベクトル長）を表示する。

(6) 宣言文およびコメント文の省略が可能である。

(7) 副プログラムごとにFORTUNEコストより算出したベクトル化率を表示する。

(8) 副プログラム単位に，親ルーチンから引用された回数およびルーチンを引用した回数を表示する。

(9) すべての副プログラムに関しての実行回数，ソースライン数，実行コストおよびその比率をグラフ形式で出力する。

FORTRANソースプログラムが解析対象のときは，副プログラム単位の選択出力が可能で，出力項目(1)，(2)，(3)，(6)を含んでいる。

(ロ) 出力例

(1) ベクトル化情報

ベクトル化の可否

```
1------------S------DO 13 IC=1,NC                    ISN=0013     39746                              39746
1            S      IF (IC.EQ.IET) GO TO 13          ISN=0014    999218       39746( 3.9%)        2038182
1            S      IF (IPD(IC).GT.ISM) GO TO        ISN=0015    959472      183590(19.1%)        2102534
1            S      IF (LO(209)) GO TO 57            ISN=0016    775882        5020( 0.6%)         780902
1             C NO PREVIOUS CALCULATION OF COU       00002200
1            S      K1=NVI(1,IC,IET)                 ISN=0017    770862                           770862
1            S      K2=NVI(2,IC,IET)                 ISN=0018    770862                           770862
1 DOループ構造 S      IF (K1.GT.K2) GO TO 10           ISN=0019    770862      278732(36.1%)       1820456
1            S      K3=IPD(IC)                       ISN=0020    492130                           492130
1 ループの深さ S      IDP=MINO(IDP,K3)                 ISN=0021    492130                          1476390
1 2----------S------DO 9 K=K1,K2                      ISN=0022    492130                           492130
1 2          S      BT=AT(K)                         ISN=0023    492130                           492130
1 2          S      KT=IABS(KKT)                     ISN=0024    492130                           984260
1 2          S      LT=KT.EQ.KKT                     ISN=0025    492130                           984260
1 2 3--------V------DO 8 IS=K3,ISM                    ISN=0026    492130                           492130
1 2 3        V      WRE(IS)=WRE(IS)+CT*VRE(IS,       ISN=0027  24203408                        145220448
1 2 3        V      WIM(IS)=WIM(IS)+CT*VRE(IS,       ISN=0028  24203408                        145220448
1 2 3 4------V------IF (.NOT.LT) THEN                 ISN=0029  24203408  24203408(100.%)        48406816
1 2 3 4      V        WRE(IS)=WRE(IS)-CT*VIM(         ISN=0030  24203408                        145220448
1 2 3 4      V        WIM(IS)=WIM(IS)+CT*VIM(         ISN=0031  24203408                        145220448
1 2 3 +------V------END IF                           ISN=0032  24203408                              0
1 2 +--------V----8 CONTINUE                         ISN=0033  24203408                         24203408
1 2                                                                    DO      8;95.70%(    653984146)
1 2                                                                    AVERAGE DO-LOOP LENGTH =     (49)
1 +----------S----9 CONTINUE                         ISN=0034    492130                         492130
1                                                                      DO      9;96.21%(    657429056)
1                                                                      AVERAGE DO-LOOP LENGTH =      1
                                                                                                  平均ベクトル長
```

図 4.5　ベクトル化情報出力例

Fig. 4.5 Example of output of TOP10 (Vectorized information)

(2) 副プログラムごとの実行コスト及びヒストグラム

```
+-------------------------------------------------------------------------------------------------
I
I  NO.   ROUTINE  EXECUTIONS     COST      %     0.........1.........2.........3.........4.......
I
+-------------------------------------------------------------------------------------------------
I 0001   LPMAIE             6   284336    48.6   I*************************************************
I 0002   MAKPAS            346   134662    23.0   I***********************
I 0003   UNIRN            3177   112810    19.3   I*******************
I 0004   SERPOS            346    34244     5.9   I******
I 0005   INIT                6    10373     1.8   I*
I 0006   MAIN                1     8155     1.4   I*
+-------------------------------------------------------------------------------------------------
I                                584580
+-------------------------------------------------------------------------------------------------
```

図 4.6　コストのヒストグラム出力例

Fig. 4.6 Example of output of TOP10 (Histogram of cost)

(ハ) 使用上の注意

(1) メモリ使用量は，VP(MSG)を指定しないとき約1MBである。VP(MSG)を指定した時は，更にFORTRAN77/VPコンパイラかソースプログラムをコンパイルできるだけのメモリを必要とする。通常3MBあれば十分である。

4.1.4　会話型ベクトライザー

ＴＳＳ配下で，ＰＦＤの対話管理機能を利用しFORTRANプログラムがベクトル計算機（FACOM VP システム）で効率良く実行するようにソースプログラムを変更する作業を支援する。

(1) 機　　能

主な機能を以下に示す。

(1) FORTRANプログラムの実行解析情報として「どのプログラム」，「どのFORTRAN文」に最も実行時間が費されているかを知る。

(2) ベクトル化情報の表示

「最も実行時間が費やされている部分がベクトル化されているか」，「どのようにすればベクトル化できるか」がわかる。

(3) ベクトル化効果の表示

FORTRANプログラムの動的解析の情報と，ベクトル化情報により，ＶＰでのおよその高速化の効果を知ることができる。

(4) 原始プログラムの編集機能

ベクトル化情報やチューニングメッセージに基づき，最適化制御行の挿入や原始プログラムの修正を行うことができる。

(ロ) 使 用 例

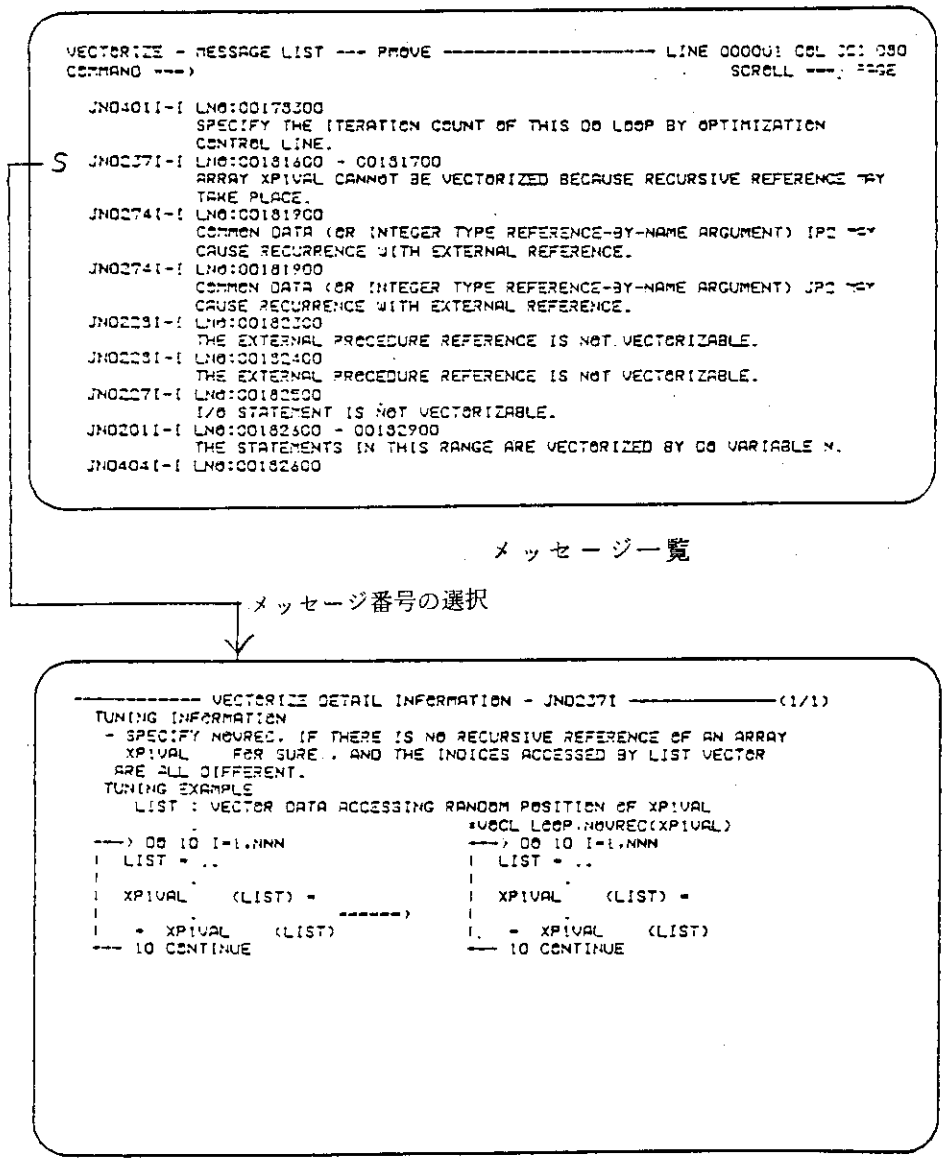


図 4.7　会話型ベクトライザのメンバー画面

Fig. 4.7 Display of information of conversational vectorizer

例えば，メッセージ一覧においてメッセージID を選択すると，そのメッセージに対応する処理方法やベクトライズ／チューニング例などが表示される詳細情報画面となる。

(ハ) 使用上の注意

(1) 必要とするメモリ使用量は，概算で次の式で求められる。

メモリ使用量＝８５０＋1.5＊Ｓ＋Ｖ（Ｋバイト）

Ｓ：プログラム単位内の文の数が最大であるものの文の数

Ｖ：次のいずれかの値である

高速モード　　：３００
デバッグモード：　０

(2) 原始プログラムがＧＥＭファイルの場合，その中にＧＥＭのINCLUDE制御文を含まないこと。

(3) FORTUNE解析情報を利用して原始プログラムを修正，編集する場合，ファイルはＰＳかＰＯの１メンバー，ＧＥＭの１モジュールにすべての副プログラムがそろっていなければならない。

4.1.5 TESTFORT77

ＴＳＳ端末を介して，FORTRANプログラムを実行させながら，ダイナミックなデバッグを行う。

(イ) 機　　能

TESTFORTコマンドが入力されるとTESTFORT77インタラクティブデバッグのサブコマンドをシステムが要求してくる。サブコマンドの機能にはおよそ次のものがある。

(1) 実行の中断

被デバッグプログラムの適当な文に実行中断点を設定し，制御がその文に達したとき，ユーザーにサブコマンドの入力を促す。

(2) 変数の値の印刷

被デバッグプログラムの実行途中における変数，配列の値を印刷することができる。

(3) 変数の値の変更

被デバッグプログラムの実行論理に無関係にサブコマンドで変数，配列に値を設定することができる。

(4) 実行の再開

先に設定した中断点から実行を再開したり，全く別の文から実行を開始したりすることができる。

(5) 条件の判定

FORTRANの論理ＩＦ 文と類似の機能で，中断点に到達したときに，あらかじめ提示された条件が判定され，その結果の'真'，'偽'に応じて，指定された一連のサブコマンドを実行させることができる。

(6) SUBCHK, ARGCHK

配列要素又は部分列式の引用がある度に，その添字式又は部分列式の指定が不正かどうかを検査する。また外部手続きの引用がある度に，引用先の実引数の対応が正しいかどうか，検査する。

(7) ソースプログラム

デバッグ作業中にソースプログラムの内容を，デバッグ作業を中断することなく印刷することができる。

(8) 文実行カウントの表示

被デバッグプログラム中の各文が何回実行されたかを表示させることができる。

(9) 端末割込み

アテンションキーを押すと，システムはサブコマンドを要求するので，その時点の被デバッグプログラムの状態を調査したり，ループを抜け出したりすることができる。

(ロ) 実 行 例

```
.FORT D.FORT77 PARM(TEST,OBJ(D)          ← TESTオプション
FORT77 D.FORT77 OBJ(D) TERM TEST ELM(*) BYNAME
FORTRAN 77 COMPILER ENTERED
END OF COMPILATION
*** NOW REPLACED OBJECT DATASET D ***
TESTFORT D   ← TESTFORTコマンド
AT 30 IN MAIN (TEMPORARY)
FREQ ON
AT 60
GO
 AT 60/10 IN MAIN
 TESTFORT
GO
 AT 60/10 IN MAIN
 TESTFORT
GO
 AT 60/10 IN MAIN
 TESTFORT
GO
 SUBSCRIPT OF 'IDATA ' OUT OF RANGE.  } SUBCHK検出
 AT 60/10 IN MAIN (TEMPORARY)
 TESTFORT
LIST I     ← Iの値を表示
 I=              4
 TESTFORT
LISTFREQ *   ← 各STATEMENT毎の実行回数を表示
 MAIN
   30                1                    40                1
   50                4                    60/10             3
   70                0                    90                0
 TESTFORT
SOURCE
 SOURCE FOR MAIN
 00000010 C ***************** TEST PROGRAM *****************
 00000020         INTEGER*4 IDATA(3)
 00000030         N = 10
 00000040         DO 10 I=1,N
 00000050         IDATA(I) = I
 00000060      10 CONTINUE
 00000070         WRITE(6,610) IDATA
 00000080    610 FORMAT(1H ,'IDATA=',3I5)
 00000090         STOP
 00000100         END
 TESTFORT
SET N=3   ← Nの値を3にリセットする
 TESTFORT
GO 40
 AT 60/10 IN MAIN
 TESTFORT
OFF 60   ← 中断点消去
 TESTFORT
GO       ← 再実行
 IDATA     1     2     3
 DEBUGGING PROGRAM TERMINATED. USER=0008, SYSTEM=0000
 TESTFORT
END
READY
```

図 4.8　TESTFORT 使用例
Fig. 4.8　Example of output of TESTFORT

(ハ) 使用上の注意

(1) TESTFORT コマンドに，FORTRAN のソースプログラムを翻訳する機能はない。従って，ユーザは，TESTFORT コマンド入力前に，被デバッグプログラムを，TEST オプション付でコンパイルしておく必要がある。

(2) 利用者はあらかじめ一定の手順の処理を行うコマンドプロシジャを作成しておくことができる。

4.1.7 DOCK／FORT77

ＴＳＳ端末を介して，FORTRANプログラムを実行させながら，ダイナミックなデバッグを行う。

(イ) 機　　能

(1) メニュープログレス方式

デバッグに必要な環境を設定するための入力情報の要求を，各種のメニューによって段階的に表示する。利用者は，メニューに従って，必要な情報を次々に指定すればよい。

(2) 実行モードによる実行速度の制御

利用者は，任意にプログラム単位ごとの実行モードを指定することができ，デバッグの終了したプログラム単位は高速で実行させ，じっくりデバッグしたいプログラム単位は，ゆっくり実行させることができる。実行モードには，次の４種類がある。

i) ステップモード

被デバッグプログラムの原始プログラムを画面に表示しながら，一文を実行するごとに静止モードとなる。

ii) 低速モード

被デバッグプログラムの原始プログラムを画面に表示しながら連続して文を実行し，特定の文（文番号付きの代入文，ブロックＩＦ文など）で静止モードとなる。

iii) 中速モード

プログラム単位の最初の実行文と最後の実行文を実行する直前で静止モードとなる。

iv) 高速モード

連続して文を実行する。原始プログラムは表示しない。

(3) プログラムの実行経過の画面表示

実行モードとしてステップモードまたは低速モードが指定されたプログラム単位については，実行の経過を実行文単位に，原始プログラムの形式でディスプレイ画面に表示される。実行経過の表示とは，実行中の文を，高輝度（カラーディスプレイでは赤色）で表示することをいう。

(4) 端末出力データの保存

プログラムの端末出力データ及びDOCK／FORT77 が出力する各種メッセージを内部的に生成したデータセットに保存する。利用者は，プログラムが静止モードとなった段階で，ＰＦキー若しくはサブコマンドによって，以前に出力された情報を参照することができる。

また，DOCK／FORT77 の終了時にリスト出力するかどうかも指定することができる。

(5) プログラムの再実行

プログラムの実行が終了した時，プログラムの先頭から再試行させることができる．再試行には，次のような方法がある。

i) 単にプログラムの先頭から再び実行させる。

ii) 原始プログラムを修正した後，再び実行させる。

iii) データセット名，オプションなどを変更した後再び実行させる。

(6) スナップ文

実行中のプログラムが静止モードとなった段階で，'スナップ文'と呼ばれる'文'を指定することによって，動的に実行状態を変更することができる。

スナップ文には，以下のものがある。

i) 代 入 文

ii) 論理ＩＦ文

iii) GOTO文

iv) ＳTOP文

v) WRＩTE 文

スナップ文が指定されると，動的に制御の移行や値の変更・参照ができるので，強力なデバッグ手段となる。

(7) モニタ条件

利用者はプログラムが静止モードとなった段階で，プログラム単位ごとにデータの値を監視する条件を定義することができる。これをモニタ条件という。モニタ条件が成立すると，DOCK／FORT77 はその旨をメッセージによって通知する。

モニタ条件の成立とは，次の２つのうちいずれかである。

(i) 指定された変数，配列要奏又は文字部分列の値が変化した場合

(ii) 指定された論理式の評価の結果が真になった場合

また，定義の方法によって，単に値の表示を行う場合と，値の表示を行った後，静止モードとする場合がある。

(8) SUBCHK，ARGCHK

配列要素又は部分列式の引用がある度に，その添字式又は部分列式の指定が適正かどうかを検査する。また外部手続の引用がある度に，引用元の実引数の対応が正しいかどうか検査する。

(9) 文実行カウントの表示

実行文ごとの実行回数を収集しておき，実行経過を表示する画面で文ごとの実行回数を表示する。

(ロ) 実 行 例

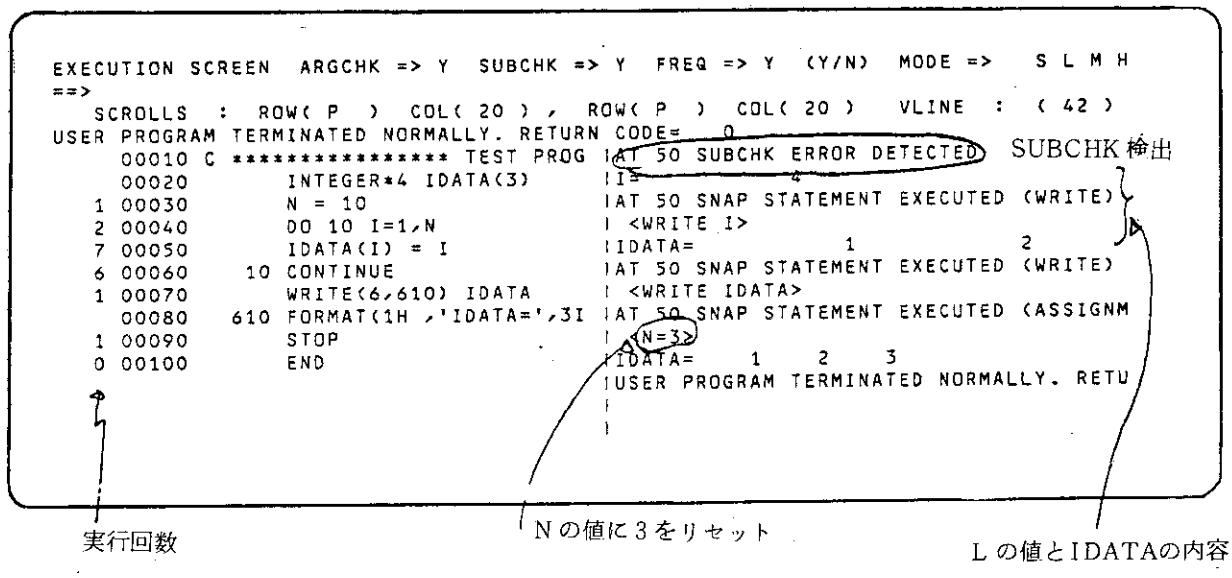


図4.9 ＤＯＣＫ使用例

Fig. 4.9 Example of display of DOCK

(ハ) 使用上の注意

(1) ＴＳＳリソース制限内（CPU2分，メモリ１ＭＢ）で処理可能なジョブであること。

使用メモリの目安は，約５００ステップのプログラムで５６０ＫＢである。

(2) コンパイル，リンケージでエラーがないこと。

4.2 データセット保守，管理ツール

4.2.1 LPCOMP

FORTRAN プログラムの変換の際，変換の妥当性をチェックするために，サンプルアウトプットとテストラン結果の比較を行う。この比較作業は人手によって行われるため，作業量の膨大さが問題となっている。したがって，これを計算機で行い，作業量を削除する。

(1) 機　　能

(1) 文字部の比較

(2) 数値部の比較

種々の数値形式の比較が考えられるが，主なものを以下に示す。

i) E 変換，D 変換，Q 変換を比べる場合，仮数部および指数部の数字のみが比較される。

0.2376E－06

0.2376Q－06

は等しいと見なされる。

ii) 桁数移動子（ピリオド）の位置にかかわらず，E，D，Q 変数の数値は比較される。

23.76E－08

0.2376E－06

は等しいと見なされる。

(3) アンダーラインの表示

比較されたデータ行の中で異なった部分をアンダーラインで表示する。

(ロ) 出 力 例

1項目には，プログラム制御パラメータが出力される。

```
***** CONTROL PARAMETERS FOR COMPA RISON OF PAIR FILES *****

     NO. OF CHECKED PAIR FILES        IS     1.


          *** FILE NO. =     1 ***

CHARACTER CHECK LENGTH .  :    136.
FIGURE CHECK LENGTH       :      0.
PAGE SKIP (FILE A)        :      0.
PAGE SKIP (FILE B)        :      0.


     L I S T I N G    I M F O R M A T I O N
"A" SIGN..... DATA FILE A
"B" SIGN..... DATA FILE B
     LISTING OF COMPARED DATA A & B
          1)  COLUMN  1          SIGN (A & B)
          2)  COLUMN  2-->3      LINE NUMBRE
          3)  COLUMN  5-->MAX    COMPARED DATA


     NAME OF INPUT FILE A: J9375.VP377.DATA
     NAME OF INPUT FILE B: J937>.VP384.DATA
```

図 4.10　LPCOMP出力例1

Fig. 4.10　Example of output of LPCOMP

アンダーライン付の出力例

```
     **A** NEW PAGE (FILE A)  .....PAGE     6 **A**      **B** NEW PAGE (FILE B)  .....PAGE     6 **B**

A 7 FIRST CALL TO HORA                  295324MS AVAILABLE
B 7 FIRST CALL TO HORA                  295398MS AVAILABLE

     **A** NEW PAGE (FILE A)  .....PAGE     8 **A**      **B** NEW PAGE (FILE B)  .....PAGE     8 **B**

A10 *** TOTAL TIME ***  0H  0MN 31S 979MS          DIFFERENCE SINCE LAST CALL  31979MS
B10 *** TOTAL TIME ***  0H  0MN 31S 876MS          DIFFERENCE SINCE LAST CALL  31876MS

     **A** NEW PAGE (FILE A)  .....PAGE    10 **A**      **B** NEW PAGE (FILE B)  .....PAGE    10 **B**

A10 *** TOTAL TIME ***  0H  1MN  3S 620MS          DIFFERENCE SINCE LAST CALL  31641MS
B10 *** TOTAL TIME ***  0H  1MN  3S 635MS          DIFFERENCE SINCE LAST CALL  31757MS

     **A** NEW PAGE (FILE A)  .....PAGE    12 **A**      **B** NEW PAGE (FILE B)  .....PAGE    12 **B**

A10 *** TOTAL TIME ***  0H  1MN 35S 315MS          DIFFERENCE SINCE LAST CALL  31695MS
B10 *** TOTAL TIME ***  0H  1MN 35S 390MS          DIFFERENCE SINCE LAST CALL  31670MS

     **A** NEW PAGE (FILE A)  .....PAGE    14 **A**      **B** NEW PAGE (FILE B)  .....PAGE    14 **B**

A10 *** TOTAL TIME ***  0H  1MN 39S 851MS          DIFFERENCE SINCE LAST CALL   4536MS
B10 *** TOTAL TIME ***  0H  1MN 39S 873MS          DIFFERENCE SINCE LAST CALL   4573MS

     **A** NEW PAGE (FILE A)  .....PAGE    16 **A**      **B** NEW PAGE (FILE B)  .....PAGE    16 **B**

A10 *** TOTAL TIME ***  0H  1MN 44S 241MS          DIFFERENCE SINCE LAST CALL   4490MS
B10 *** TOTAL TIME ***  0H  1MN 44S 434MS          DIFFERENCE SINCE LAST CALL   4546MS

     **A** NEW PAGE (FILE A)  .....PAGE    18 **A**      **B** NEW PAGE (FILE B)  .....PAGE    18 **B**

A10 *** TOTAL TIME ***  0H  1MN 49S 260MS          DIFFERENCE SINCE LAST CALL   4519MS
B10 *** TOTAL TIME ***  0H  1MN 49S 942MS          DIFFERENCE SINCE LAST CALL   4522MS

     **A** NEW PAGE (FILE A)  .....PAGE    20 **A**      **B** NEW PAGE (FILE B)  .....PAGE    20 **B**

A10 *** TOTAL TIME ***  0H  1MN 53S 354MS          DIFFERENCE SINCE LAST CALL   4494MS
B10 *** TOTAL TIME ***  0H  1MN 53S 432MS          DIFFERENCE SINCE LAST CALL   4533MS

     **A** NEW PAGE (FILE A)  .....PAGE    22 **A**      **B** NEW PAGE (FILE B)  .....PAGE    22 **B**

A10 *** TOTAL TIME ***  0H  2MN 24S 831MS          DIFFERENCE SINCE LAST CALL  31527MS
B10 *** TOTAL TIME ***  0H  2MN 25S 137MS          DIFFERENCE SINCE LAST CALL  31655MS
```

ラインNO

図4.11　LPCOMP出力例2

Fig. 4.11　Example of output of LPCOMP

(ハ) 使用上の注意点

(1) アンダーラインが表示されたデータは，重ね打ちの偽ターミナルの画面で見ることができない。これを避けるには，アンダーラインを打ち出さなければ良いが，これは入力パラメータで制御できるようになっている。

(2) 1ページを比較するのに約0.3秒かかるので，実行時間は，次式を考慮して行う。

実行時間＝0.3×比較ページ数〔秒〕

(3) 比較される入力ファイルはＰＳファイルを原則としているが，ＰＯ，ＧＥＭファイルも，ＪＣＬでコンカチネイトしておくことにより，９９９コのメンバーまで一度に比較することもできる。

4.2.2 SFCOMP

2本のソースプログラムの比較を行い，その相違箇所を明確にする。

(イ) 機　　能

2本のソースプログラムの比較を各サブルーチン単位（ユーザがＪＣＬで指定）に行い，異る箇所を出力する。サブルーチンの数は９９９以内である。又５００個ステートメント格納用テーブルを持ち，対応しない行をストアしていくので５００行以上異なる場合には，そのテーブルをスウィープアウトして，以下に現われるステートメントとの比較は行わない。

(ロ) 出 力 例

```
***** NO. OF FILE IS    1  *****
      NO. OF CHECK LENGTH IS   18


   *** FILE NO. =    1 ***
  3 F01   ((     1          -56,  28,  23  ,86,  99,   4,   3,4096,  -1,  15/      00000030))
 12 F01   ((      CALL SDFDMP(2,1)                                                  00000130))
 35 F01   ((      IF(ICNT2 .LT. 20) GO TO 5000                                      00000360))
 36 F01   ((C     WRITE(6,620) DATA                                                 00000370))
 37 F01   ((C 620 FORMAT(1H ,'AFTER DATA=',2015)                                    00000380))
          3 F02   ((     1          -56,  28,  23  ,86,  99,   4,   3,4100,  -1,  15/      00000030))
         12 F02   ((C     CALL SDFDMP(2,1)                                                  00000130))
         33 F02   ((C                                                                       00000331))
         36 F02   ((      IF(ICNT2 .LT. 21) GO TO 5000                                      00000360))
         37 F02   ((C                                                                       00000361))
         38 F02   ((      WRITE(6,620) DATA                                                 00000370))
         39 F02   ((  620 FORMAT(1H ,'AFTER DATA= ',2015)                                   00000380))

  *****  N U M B E R  O F  C A R D  *****
       F01 =       39            34(SAME)        5(DIF)
       F02 =       41            34(SAME)        7(DIF)
```

異なるものがFO1 では5カード
FO2 では7カード存在した。

図4.12　　SFCOMP出力例

Fig. 4.12　Example of output of SFCOMP

(ハ) 使用上の注意

(1) 使用メモリは１２８KBである。

(2) PO，GEMファイルの比較は，ＪＣＬでメンバ名を指定する。

たとえば，２つの区分データ・セットが

Ｊ９３７５．ＳＦ１．ＦＯＲＴ７７

Ｊ９３７５．ＳＦ２．ＦＯＲＴ７７

に格納されている。メンバーＡとＡＸ，ＢとＢＸ，ＣとＣＸ の比較を行う時，制御文は以下のように指定する。

```
//FT01F001 DD DSN=J9375.SF1.FORT77(A),DISP=SHR
//FT01F002 DD DSN=J9375.SF1.FORT77(B),DISP=SHR
//FT01F003 DD DSN=J9375.SF1.FORT77(C),DISP=SHR
//FT02F001 DD DSN=J9375.SF2.FORT77(AX),DISP=SHR
//FT02F002 DD DSN=J9375.SF2.FORT77(BX),DISP=SHR
//FT02F003 DD DSN=J9375.SF2.FORT77(CX),DISP=SHR
```

4.2.3 JSDCOMPR

データセットの比較を行うときに，使用者の負担を軽減し，比較作業を効率的に行う。

(イ) 機　　能

2つのデータセットをレコードごとに比較し，その結果をメッセージとして出力する。

比較は，論理レコードごとに行われるので，ブロックの大きさは問わない。ただし，可変長スパンドレコードは，ブロックごとに比較する。

比較する2つのデータセットは同じ編成のデータセットでなければならない。

(ロ) 出 力 例

可変長スパンドレコードのデータセットを比較した結果を以下に示す。

```
 COMPARE TYPORG=PS                                                    00000820
JSD253I *** WARNING *** RECORDS ARE COMPARED AT PHYSICAL BLOCK LEVEL
JSD221I *** MESSAGE *** RECORDS ARE UNEQUAL

    DDNAME = SYSUT1

① PHYSICAL RECORD NUMBER = 00000007  LOGICAL RECORD NUMBER WITHIN PHYSICAL RECORD = 00000000 ②
(000000) 0C6C0000 015C0200 00000020 00000030  00000033 00000036 00000039 0000003C   *.%...*.....................
(000020) 0000003F 00000042 00000045 00000048  0000004B 0000004E 00000051 00000054   *.....................+.......
(000040) 00000057 0000005A 0000005D 00000060  00000063 00000066 00000069 0000006C   *...........)...-...............%
(000060) 0000006F 00000072 00000075 00000078  0000007B 0000007E 00000081 00000084   *...7.............#...=.........
(000080) 00000087 0000008A 0000008D 00000090  00000093 00000096 00000099 0000009C   *...............................
(0000A0) 0000009F 000000A2 000000A5 000000A8  000000AB 000000AE 00000081 000000B4   *...............................
(0000C0) 00000087 0000008A 00000080 000000C0  000000C3 000000C6 000000C9 000000CC   *.....................C...F...I....
(0000E0) 000000CF 00000002 00000005 00000008  00000008 0000000E 000000E1 000000E4   *.......K...N...Q..............U
(000100) 000000E7 000000EA 000000ED 000000F0  000000F3 000000F6 000000F9 000000FC   *...X...........0...3...6...9....
(000120) 000000FF 00000102 00000105 00000108  00000108 0000010E 00000111 00000114   *...............................
(000140) 000001'' 0000011A 00000110 00000120  00000123 00000126 00000129 0000012C   *...............................

   DDNAME = SYSUT2
   PHYSICAL RECORD NUMBER = 00000007  LOGICAL RECORD NUMBER WITHIN PHYSICAL RECORD = 00000000
(000000) 0C6C0000 015C0200 00000573 00000500  00000620 0000068A 000006E7 00000744   *.%...*.....................X....
(000020) 000007A1 000007FE 00000858 000008B8  00000915 00000972 000009CF 00000A2C   *...........¥...................
(000040) 00000A89 00000AE6 00000B43 00000BA0  00000BFD 00000C5A 00000CB7 00000D14   *.......W.......................
(000060) 00000071 00000DCE 00000E2B 00000E88  00000EE5 00000F42 00000F9F 00000FFC   *...................V...........
(000080) 00001059 000010B6 00001113 00001170  000011CD 0000122A 00001287 000012E4   *.............................U
(0000A0) 00001341 0000139E 000013FB 00001458  00001485 00001512 0000156F 000015CC   *.........................?....
(0000C0) 00001629 00001686 000016E3 00001740  00001790 000017FA 00001857 000018B4   *...........T...  ..............
(0000E0) 00001911 0000196E 000019CB 00001A28  00001A85 00001AE2 00001B3F 00001B9C   *.......>.............S.........
(000100) 00001BF9 00001C56 00001CB3 00001D10  00001D60 00001DCA 00001E27 00001E84   *...9.............._............
(000120) 00001EE1 00001F3E 00001F9B 00001FF8  00002055 000020B2 0000210F 0000216C   *.............8.............%
(000140) 000021C9 00002226 00002283 000022E0  0000233D 0000239A 000023F7 00002454   *...I.....................7....

③                                                                    ④

END OF JOB - TOTAL NUMBER OF RECORDS COMPARED 00000007
```

図4.13　JSDCOMPR出力例

Fig. 4.13 Example of output of JSDCOMPR

① ブロック番号

② レコード番号　ただし，可変長スパンドレコードの場合はブロック単位に比較するのでレコード番号はセットされない。

③ 内容を16進表示する。

④ 内容を文字型で表示する。

(ハ) 使用上の注意

(1) 二つの区分データセットの複数のメンバーを一度に比較することができる。

制御文で指定すれば，メンバー名の異なるメンバと比較することもできる。制御文を省略したときは全メンバーを比較の対象とするが，比較の順序は，アルファベッド順に行われる。たとえば，データセットJ9375.S1.DATA 内にメンバーAX，BX，CX，データセットJ9375.S2.DATAに，メンバーXW，YW，ZWが存在する時，比較されるメンバーは

AX : XW
BX : YW
CX : ZW

となる。

(2) 区分データセットのレコード形式が可変長スパンドレコードの場合は，DD文にメンバ名パラメータを指定して，順データセットとして扱わなければならない。

(3) 区分データセットの比較において，一方が本名，他方が別名のとき，エラー（リターンコード＝8）となるが，レコードの比較処理は行われる。

4.2.4 MTDUMP

1つの磁気テープを読込み，種々の情報（SLテープ，NLテープの別，ブロックサイズ，ブロック数，ファイル名など）を得ることにより，磁気テープの内容の検査に役立てる。

(イ) 機　能

(1) 磁気テープを読んでSLテープかNLテープかを識別し，それぞれ以下の項目を出力する。

i) SLテープの場合

ボリューム通番
データセット名
ブロックサイズ
ブロック数
レコードフォーマット
データセット作成日
記録密度
データセットをMTに作成した時の計算機システムのOS名
データセットをMTに作成した時に使用したユーティリティ名
データセットが占めるMT上での長さ（単位はフィート）

ii) NLテープの場合

ブロックサイズ（ロングブロック長）
ブロックサイズ（ショートブロック長）
ブロック数

データセットをMTに作成した時に使用したユーティリティ名

(2) データセットの内容を文字型(EBCDIC)あるいは16進でダンプする。

i) SLテープの場合

各データセットごとに，HDR1，HDR2，データの最初の2ブロック，EOF1(あるいはEOV1)，EOF2(あるいはEOV2)をダンプする。

ii) NLテープの場合

各データセットごとに，最初の2ブロックをダンプする。

(ロ) 出力例

```
*SL*-------------------------*SL*
*SL*      XXXXX     %        *SL*   VOLUME NO. =SPDBK2
*SL*    %           %        *SL*   OWNER=JAERI
*SL*      XXXXX     %        *SL*
*SL*            %   %        *SL*   .DATE(1985-03-20)
*SL*      XXXXX     XXXXXX   *SL*   .TIME=13:16:39
*SL*-------------------------*SL*
```

NO.	DATA-SET-NAME	(BYTES) BLKSIZE	(BYTES) LRECL	BLOCKS	RECFM	CREATION	(BPI) DENSITY	VOL	SYSTEM/OS	UTILITY	(FEET) LENGTH
1.	SJEFMENU.PS.DATA	3000	100	61	FB	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	4.03
2.	SJEFMENU.GEM.DATA	1736	0	121	U	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	3.95
3.	SYSCORE.NEW.DATA	2020	2016	46	VS	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	2.20
4.	SPEEDI.PS.CLIST	3120	255	3	VB	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	0.27
5.	SPEEDI.PO.CLIST	2020	2016	8	VS	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	0.45
6.	SYSCMD.V03	1096	0	960	U	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	32.83
7.	SYSUTY.V03	1096	0	265	U	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	8.44
8.	EMER.V03	1096	0	797	U	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	27.91
9.	GSDOSE.FORT77	1096	0	196	U	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	?	6.48
10.	SYSCMD.M38.LOAD	6164	6160	128	VS	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	11.08
11.	SYSCMD.S33.LOAD	6164	6160	131	VS	1985/03/13(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	11.18
12.	SYSGRF.LOAD	6164	6160	434	VS	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	36.94
13.	WIND.DATA	4000	4000	400	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	31.44
14.	INCONC.DATA	4000	4000	501	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	39.35
15.	SFCONC.DATA	4000	4000	501	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	39.35
16.	PART.DATA	3000	150	753	FB	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	49.04
17.	DOSECUM.DATA	4000	4000	100	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	7.94
18.	SYSMAP4.V3.DATA	800	800	4503	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	160.71
19.	WEATHER.T795.DATA	400	400	901	F	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	?	27.43
20.	SYSGRF2.LOAD	6164	6160	1217	VS	1985/03/18(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	103.94
21.	SYSUTY.LOAD	6164	6160	119	VS	1985/03/20(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	10.64
22.	DOSETHY.DATA	4000	4000	100	F	1985/03/20(WED)	6250	SPDBK2	FACOM OSIV/F4	?	7.94
23.	SYSGRF.LOAD	6164	6160	182	VS	1985/03/20(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	15.73
24.	SYSGRN.LOAD	6164	6160	213	VS	1985/03/20(WED)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	18.02

```
NORMAL END OF * MTDUMP *   TOTAL LENGTH=  657 FEET
```

図4.14　MTDUMP出力例

Fig. 4.14　Example of output of MTDUMP

(ハ) 使用上の注意

(1) 読込み可能な最大ブロック長は32760バイトである。

(2) 最初のブロックを読んで，ブロックの先頭4バイトの内容がVOL1のとき，SLテープとして処理し，それ以外はNLテープとして処理する。

(3) SLテープの記録密度は，ヘッダーラベルの情報を出力するため，ラベルの記録密度の情報と，実際の記録密度が異なる場合がある。

たとえば，システムユーティリティJSGMTCPYでデッドコピーを行うとき，出力側の記録密度を1600BPI，入力側の記録密度を6250BPIと指定すると，データセットラベルもコピーするため，出力側磁気テープのMTDUMP出力結果は6250BPIとなってしまう。

4.2.5　MTLABEL

磁気テープに格納されているファイル名などの情報を通常の磁気テープラベルと同じ形に編集してＮＬＰに出力し，保存用磁気テープラベルの内容証明資料として役立てる。

(イ) 機　　能

標準ラベル（ＳＬ）磁気テープのラベル（VOL 1, HDR 1, HDR 2, EOF 1, EOF 2, EOV 1, EOV 2)の内容や，名ファイルの第１ブロックの内容を解析し，ファイル名，レコード長，ブロック長，レコード形式などを得る。これをもとに，情報の書き込まれたテープラベルを作成し，ＮＬＰに出力する。更に，詳細な情報をＡ４サイズの枠の中に編集してＮＬＰに出力する。

(ロ) 出 力 例

フルサイズ磁気テープの出力例を以下に示す。

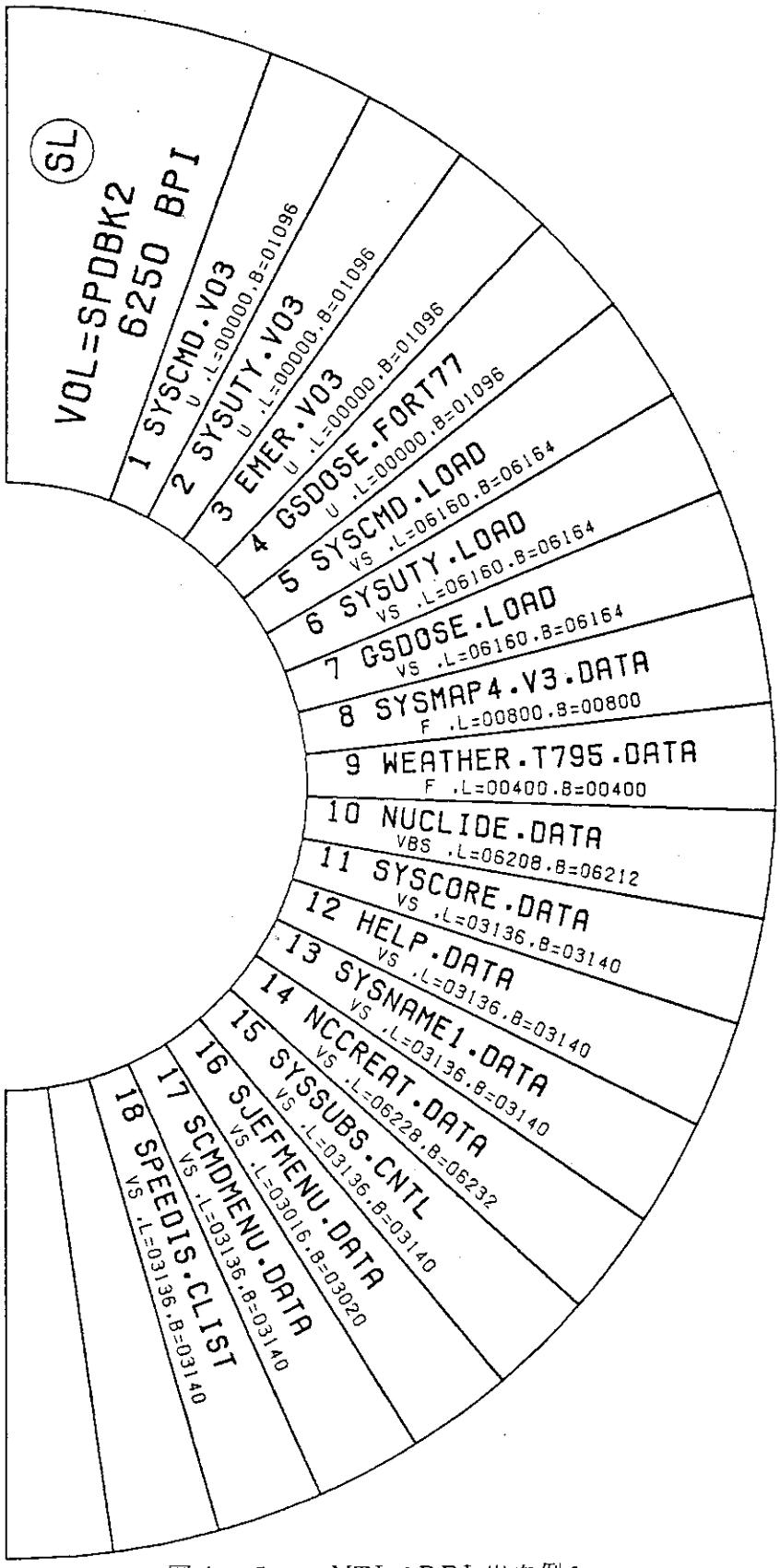


図4.15 MTLABEL出力例1

Fig. 4.15 Example of output of MTLABEL

詳細ファイル情報

SPDBK2

MEMO

NO.	DATA-SET-NAME	(BYTES) BLKSIZE	(BYTES) LRECL	BLOCKS	RECFM	CREATION	(BPI) DENSITY	VOL	SYSTEM/OS	UTILITY	(FEET) LENGTH	DATA SET SEQ. NO.
1.	SYSCMD.V03	1096	0	958	U	1985/02/28(THU)	6250	SPDBK2	FACOM OSIV/F4	GEMBKUP	38.05	1
2.	SYSUTY.V03	1096	0	265	U	1985/02/28(THU)	6250	SPDBK2	FACOM OSIV/F4	GEMBKUP	10.60	2
3.	EMER.V03	1096	0	797	U	1985/02/28(THU)	6250	SPDBK2	FACOM OSIV/F4	GEMBKUP	31.68	3
4.	GSDOSE.FORT77	1096	0	196	U	1985/02/28(THU)	6250	SPDBK2	FACOM OSIV/F4	GEMBKUP	7.87	4
5.	SYSCMD.LOAD	6164	6160	128	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	13.82	5
6.	SYSUTY.LOAD	6164	6160	256	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	27.54	6
7.	GSDOSE.LOAD	6164	6160	116	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	12.54	7
8.	SYSMAP4.V3.DATA	800	800	4603	F	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4		160.71	8
9.	WEATHER.F795.DATA	400	400	901	F	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4		27.45	9
10.	NUCLIDE.DATA	6212	6208	4	VBS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4		0.54	10
11.	SYSCORE.DATA	3140	3136	34	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	2.38	11
12.	HELP.DATA	3140	3136	25	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	1.78	12
13.	SYSNAME1.DATA	3140	3136	9	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	0.71	13
14.	NCCREAT.DATA	6232	6228	9	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	1.08	14
15.	SYSSUBS.CNTL	3140	3136	28	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	1.98	15
16.	SJEFMENU.DATA	3020	3016	81	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	5.39	16
17.	SCMDMENU.DATA	3140	3136	71	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	4.85	17
18.	SPEEDIS.CLIST	3140	3136	8	VS	1985/03/04(MON)	6250	SPDBK2	FACOM OSIV/F4	JSECOPY	0.51	18

TOTAL LENGTH= 349 FEET

図4.16　MTLABEL出力例2
Fig. 4.16　Example of output of MTLABEL

(ハ) 使用上の注意

(1) 被解析（入力）磁気テープは，標準ラベル（ＳＬ）磁気テープとする。

(2) 出力されるラベルには，フル，ハーフ，クォータ，スモールサイズの４種類の大きさがあり，それぞれテープの大きさにあう様に作成される。

(3) 詳細ファイル情報に出力される，データセットが占めるMT上での長さは，ショートブロックの長さをブロック長と同じとして計算するため，ファイルの長さは過大評価されている。

4.2.6　SFMAKE

順編成ファイルのFORTRAN原始プログラムを，扱い易い区分編成ファイル，またはＧＥＭファイルへ変換し，プログラムの変更，保守を容易にする。

(イ) 機　　能

順編成ファイルを入力とし，FORTRAN原始プログラムを'END'行ごとに，１つのモジュールとして分割する。すなわち，'END'行から次の'END'行までが１つのモジュールとなり，区分編成ファイルまたはＧＥＭファイルへ出力する。モジュール名は，SUBROUTINE名，FUNCTION名，BLOCK DATA 名が付けられる。なお，名前なしBLOCK DATA は'BLOCKD'，メインモジュールは，'ＭＡＩＮ０００ｎ'という名前が各々与えられる。

(ロ) 使用上の注意

(1) 入力データセットのBLKSIZEの最大値は３２００バイトまでが処理可能

(2) 出力データセットがＧＥＭファイルの場合は，イニシャライズされている必要がある。しかし，既にいくつかのモジュールが格納されている場合は，イニシャライズの必要はない。

(3) 入力データセットのサブルーチン名等に，ピリオド(.)があるとシャープ(#)に変換して，メンバ名もしくはモジュール名を作成する。

(4) SFMKGEM, SFMKPO コマンドも用意されている。

4.2.7 VIVAPO

分割型順編成（ＰＯ）ファイルに格納されているソースファイルの取り扱いを便利にまる。

(イ) 機　　能

以下に示す５つのオプションで指定できる機能を持っている。

(1) MEMBER オプション

ＰＯファイルのメンバーごとの名前とステップ数の一覧を編集して出力する。

(2) POTOPS オプション

ＰＯファイルの一部或いは全部のメンバーを順編成（ＰＳ）ファイルに転送する。

(3) POTOPOオプション

ＰＯファイルから他のＰＯファイルへ，一部或は全部のメンバーで転送する。

(4) POLIST オプション

ＰＯファイルの一部或いは全部のメンバーの内容をメンバー名のアルファベッド順に印刷する。

(5) PSTOPOオプション

ＰＳファイル上のFORTRAN原始プログラム（ ASSEMBLER プログラムを含んでいても良い）を，１副プログラム／１モジュールの形でＰＯファイルに格納する。メンバー名は原則として，副プログラム名が採られる。

(ロ) 使用上の注意

(1) 各オプション機能により，扱えるＰＯファイルのレコードフォーマットが異なる 以下に各機能ごとのレコードフォーマットの対応を示す。

Table 4.2 Function name and record format

表 4.2　機能と処理対象レコードフォーマット

機　　能	レコードフォーマット
MEMBER	U, FB, VB, F, V
POTOPS	FB, F
POTOPO	FB, F
PORIST	FB, F, VB, V
PSTOPO	FB, F

(2) MEMBERオプションの時，最大メンバー数は１０００まで取り扱い可能。

(3) POTOPSオプションの時，ＰＯファイルのレコード長は８０バイトのものに限る。

(4) POLIST オプションの時，実行完了コードが０以外は，ディレクトリ情報は出力されない。

(5) PSTOPOオプションの時，メンバー名は，基本的には，サブルーチン名，ファンクシ

ョン名，ブロックデータ名が採られる。同一名が，出現した時は，ＳＡＭＥ０００１～ と名前が付けられる。また，ＰＳ，ＰＯファイル共レコード長は８０バイトに限る。

4.2.8 MYBACKUP

利用者のすべてのファイルを一括してＭＴへ退避する。

(イ) 機　　能

利用者のファイルの全部，または一部を磁気テープに一括退避する。ＰＯファイル，ＧＥＭファイルはアンロード形式となる。また，退避するファイルの範囲を指定することもできる。

例　'ＡＡ，ＢＢ'

ＡＡ… とＢＢ… のファイルを退避する。

'Ａ－Ｈ'

Ａ～Ｈの範囲のファイルを退避する。

'－（ＡＡ，ＢＢ）'

ＡＡ… とＢＢ…のファイルを除いて退避する。

'－（Ａ－Ｈ）'

Ａ～Ｈの範囲のファイルを除いて退避する。

(ロ) 使用上の注意

(1) 磁気テープへ退避すると，ディスク上に存在した時のファイル編成（ＰＯ，ＰＳ，ＤＡ)が不明となるため，各ユーザーはオリジナルのファイル編成を覚えておく必要がある。

4.2.9 COMPACT

活字の大きさを縮小して印刷することにより，用紙の節約及び印刷行数，桁数の拡張に効果がある。

(イ) 機　　能

日本語ラインプリンタ（ＮＬＰ）を使用した縮刷プログラムで，文字の大きさは従来の約$\frac{1}{2}$である。出力形式には次の４つのタイプがある。

(1) タイプ１

従来の印刷リスト２ページ分をＮＬＰの１ページに下図の形式で出力する。ただし従来ページ分のラインカウント値は５２以内とする。

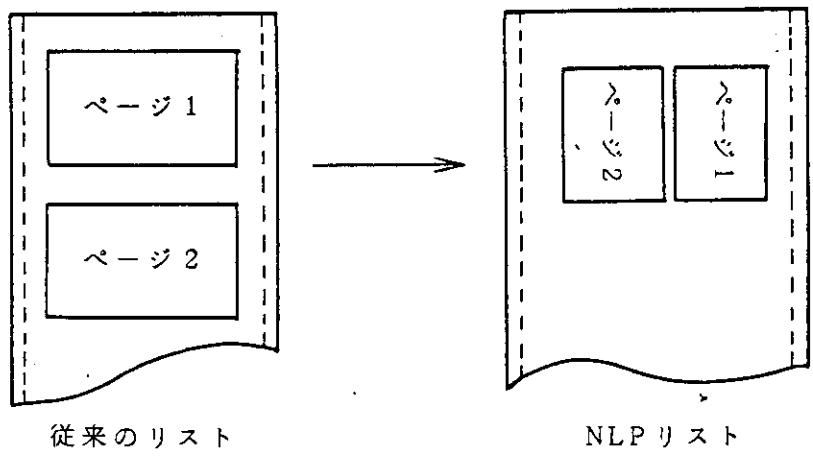


図4.17　タイプ1の出力形式
Fig. 4.17　Print format (type 1)

(2) タイプ2

ラインカウント値108以内の印刷リストをNLP1ページに下図の形式で出力する。ただし従来リストがベタ打ちの場合は，タイプ1，2は同じ出力結果となる。

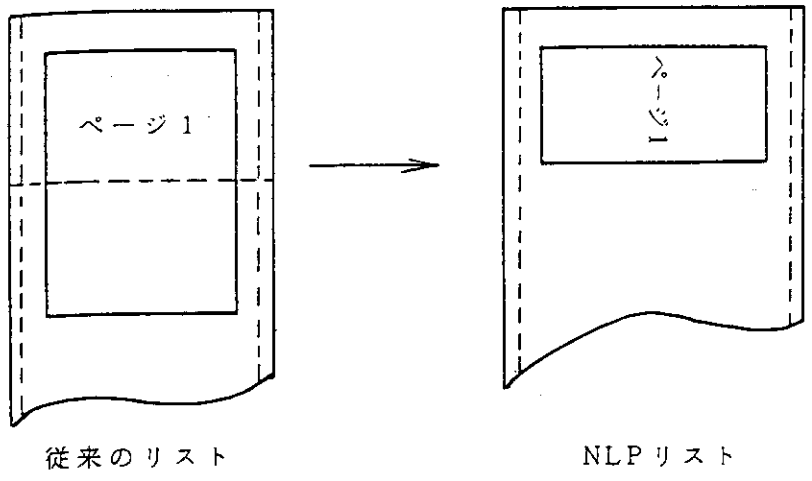


図4.18　タイプ2の出力形式
Fig. 4.18　Print format (type 2)

(3) タイプ3

従来の印刷リスト2ページ分をNLPリストの1ページに下図の形式で出力する。ただし1行100文字以内とし越えた分は印刷されない。

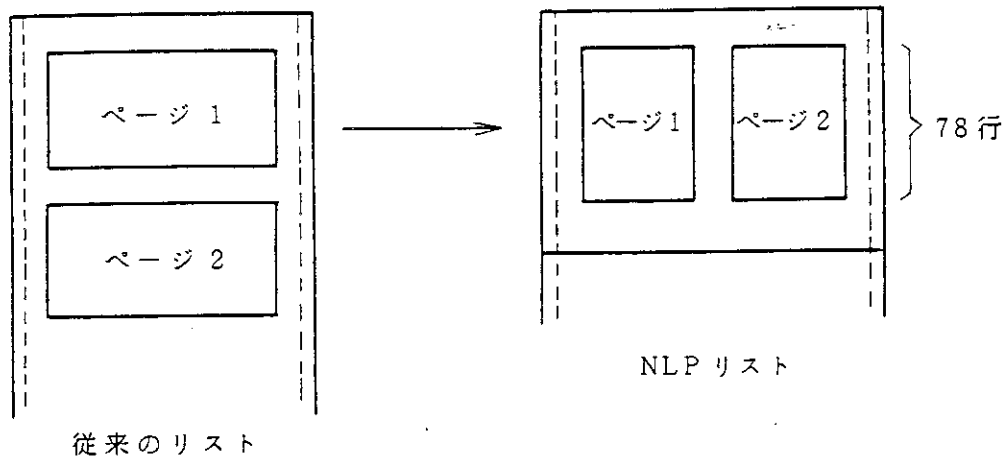


図4.19　タイプ3の出力形式
Fig. 4.19　Print format (type 3)

(4) タイプ4

ラインカウント値78以内の印刷リストをNLP1ページに下図の形式で出力する。1行216文字まで可能。

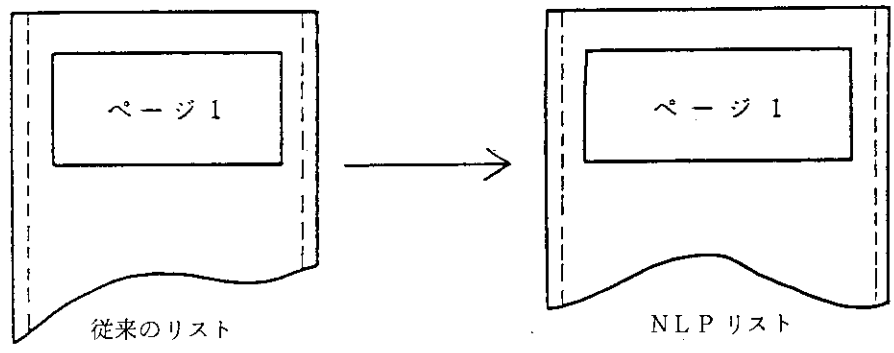


図4.20　タイプ4の出力形式
Fig. 4.20　Print format (type 4)

(ロ) 使用上の注意

(1) 原則としてPS(順編成)ファイルが入力可能であるが，・COMPACTコマンドを使用すれば，コマンド内でPSファイルに変換するので，PO(区分編成)及びGEMファイルの処理も可能である。

(2) ・COMPACTコマンドはバッチ処理のJCLを作成するものであり，実際の処理は，端末に表示されたJCLをSUBMITコマンドによりバッチジョブとして投入することにより行う。

(3) カナ文字・漢字データは出力できない。

(4) バッチ出力(SORP出力)をそのままCOMPACT化するには，・OUTDSコマンド

により出力結果をファイルに格納し，その後COMPACTコマンドにより行うことができる。

(ハ) 出 力 例

タイプ1の出力例

```
      INTEGER*4 IDATA(20)                                                00000010
      DATA IDATA/  10,   9, 231,  86,   1,  22, 111,  53,1024,  43,      00000020
     1           -56,  28,  23  ,86,  99,   4,   3,4096,  -1,  15/       00000030
C ************************************                                   00000040
      WRITE(6,610) IDATA                                                 00000050
  610 FORMAT(1H ,'BEFORE DATA=',20I5)                                    00000060
C                                                                        00000070
      CALL SORT(IDATA)                                                   00000080
C                                                                        00000090
      WRITE(6,620) IDATA                                                 00000110
  620 FORMAT(1H ,'AFTER DATA=',20I5)                                     00000120
      CALL SDFDMP(2,1)                                                   00000130
      STOP                                                               00000140
      END                                                                00000150
      SUBROUTINE SORT(DATA)                                              00000160
      INTEGER*4 DATA(1)                                                  00000170
C     DATA DATA/  10,   9, 231,  86,   1,  22, 111,  53,1024,  43,       00000180
C    1           -56,  28,  23  ,86,  99,   4,   3,4096,  -1,  15/       00000190
C ************************************                                   00000200
C     WRITE(6,610) DATA                                                  00000210
C 610 FORMAT(1H ,'BEFORE DATA=',20I5)                                    00000220
      ICNT2 = 1                                                          00000230
      ICNT3 = 2                                                          00000240
 5000 CONTINUE                                                           00000250
 6000 CONTINUE                                                           00000260
      IF(DATA(ICNT2) .GT. DATA(ICNT3)) GO TO 1000                        00000270
      IDATAX = DATA(ICNT3)                                               00000280
      DATA(ICNT3) = DATA(ICNT2)                                          00000290
      DATA(ICNT2) = IDATAX                                               00000300
 1000 CONTINUE                                                           00000310
      ICNT3=ICNT3 + 1                                                    00000320
      IF(ICNT3 .LT. 21) GO TO 6000                                       00000330
      ICNT2= ICNT2 + 1                                                   00000340
      ICNT3= ICNT2 + 1                                                   00000350
      IF(ICNT2 .LT. 20) GO TO 5000                                       00000360
C     WRITE(6,620) DATA                                                  00000370
C 620 FORMAT(1H ,'AFTER DATA=',20I5)                                     00000380
      RETURN                                                             00000390
      END                                                                00000400
```

図4.2.1　タイプ1の出力例

Fig. 4.21 Example of output of COMPACT (type 1)

4.2.10　F77INC

FORTRAN77のINCLUDE文を使用する場合，1つのプログラムに対してINCUDE文が属する原始プログラム（SOURCEファイル）とINCLUDE文によって組込まれる原始プログラム（INCLUDEファイル）の2つのファイルを用意しなければならない。

これに対してユーザから，プログラムの管理及び，ファイルの管理の面からもSOURCEファイルとINCLUDEファイルを1つのファイルに登録したいと言う要望が多くなってきている。

F77INCは，この要望に応えるために作成されたプリプロセッサである。

(イ) 機　　能

(1) F77INCを使用する場合，前もって以下に示すようなサブルーチン（以下，INCサブ

ルーチン）をSOURCE ファイルのメンバーとして作成しておく必要がある。F77INCは，このサブルーチンを解釈してINCLUDEファイルを作成する。

```
----*----1----*----2----*----3----*----4----*----5----*----6----*----7----
      SUBROUTINE   INCSO
CC
C     INCNAME    CHAR1
C     CHARACTER    INDD*8    , OUTDD*8   , INCMEM*8
CC
C     INCNAME    INTEG
C     INTEGER*4    RCA
CC
C     INCNAME    CHAR2
C     CHARACTER    INDD*8    , OUTDD*8   , MEMBER*8  , INCMEM*8
C     CHARACTER    DDMEM1*16 , DDMEM2*16 , SOURCE*80 , CHKDA*6
CC
      END
```

図4.2.2　INCサブルーチン
Fig. 4.22　Example of INC subroutine

このINCサブルーチンを作成するために，INCANA コマンドが用意してある。これは，既存のINCLUDEファイルをSOURCEファイルの任意のメンバーに登録するものである。

また，INC サブルーチンをINCLUDEファイルに変換するINCCREコマンドもある。

(ロ) 使 用 例

INC サブルーチンを含むソースファイルのコンパイルから実行までの一般的なJCLを示す。

```
//JCLG   JOB
//JCLG   EXEC   JCLG
//SYSIN DD      DATA,DLM='++'
// JUSER
   T.1 W.1 C.1 I.2
   OPTP   PASSWORD=XXXX
//FORT77 EXEC   F77INC,SO='JXXXX.XXXX',A='ELM(*)'
//LKED77 EXEC   LKED77
//RUN    EXEC   GO
++
//
```

図4.2.3　F77INC使用のためのJCL例
Fig. 4.23　Example of JCL to use F77INC

(ハ) 使用上の注意

(1) INCCRE コマンドによりINCLUDEファイルを作成するとコメント行はすべて省略される。したがってコメントだけで作成されているINCサブルーチンはINCLUDEファイルに登録されない。これは，コンパイル時にJZK…の警告メッセージとして出力されるが，実行にはまったく影響はない。

4.3 デバッグ機能

4.3.1 エラー処理サブルーチン

FORTRANプログラムの実行時のエラー発生に対して，その処理を利用者が制御，エラーモニタの機能を最大限に活用する。

(イ) 機　　能

以下に示すサブルーチンは，エラー制御表内のエラー項目をジョブステップの間だけ，動的に制御したり，トレースバックマップを出したり，利用者の用意したエラー処理をしたりするものである。

(1) ERRSET サブルーチン

このサブルーチンは，指定されたエラー識別番号に対応するエラー制御表内のエラー項目の個々の制御情報を変更したいときに使用する。ここで変更できる制御情報は，エラー打切り回数，メッセージの最大印刷回数，トレースバックマップの印刷の有無及びエラーの修正処理などである。

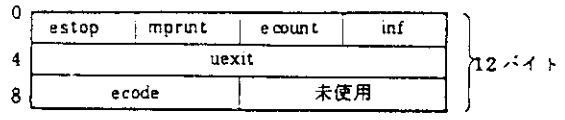
（引用の形式）

CALL ERRSET (errno , estop , mprint , trace , uexit , r)

errno　：エラー識別番号
estop　：エラー打切り回数
mprint ：メッセージ最大印刷回数
trace　：トレースバックマップ印刷の有無
uexit　：標準修正の有無及び利用者定義の修正サブルーチンの名前
r　　：エラー識別番号の範囲

(2) ERRSAV サブルーチン

このサブルーチンは指定されたエラー項目の先頭8バイトを，利用者が用意した8バイトの領域に退避させる。



estop ：エラー打切り回数．ゼロは無制限であることを示す．
mprint ：メッセージの最大印刷回数．ゼロは印刷しないことを示す．
ecount ：プログラムの実行中に発生したエラーの回数．
inf ：8ビットでエラーの処理のための制御情報を示す．
ビット0…制御文字が与えられているとき1，与えられていないとき0．
ビット1…利用者がこのエラー項目を修正可能なとき1，修正不可能なとき0．
ビット2…ecountが255を越えたとき1，255以下のとき0．
ビット3…診断メッセージと共にバッファの内容を印刷するとき1，バッファの内容を印刷しないとき0．
ビット4…未使用．
ビット5…メッセージを印刷するとき1，メッセージをmprintの値によって印刷するとき0．
ビット6…トレースバックマップを印刷するとき1，印刷しないとき0．
ビット7…未使用．
uexit ：利用者が用意した修正ルーチンの番地である．標準修正の場合は，最後のビットが1となる．
ecode ：エラーのレベルを示す．
4…Wレベルのエラー（エラーの発生した文を一部修正してプログラムの実行を続ける）を示す．
8…Eレベルのエラー（エラーの発生した文を無視する．実行は打切り回数（標準は10回)によって，続行又は中断する）を示す．
12…Sレベルのエラー（エラーの発生した文を無視する．実行は打切り回数（標準は1回）によって，中断又は続行する）を示す．

備考．エラー処理サブルーチンで参照できる項目は，先頭8バイトである．

図4.24　エラー情報格納領域
Fig. 4.24 Structure of error information table

（引用の形式）

CALL ERRSAV (errno , darea)

errno：エラー識別番号

darea：指定されたエラー項目を退避すべき8バイトの領域

(3) ERRSTR サブルーチン

このサブルーチンは，利用者が用意した領域にあるエラー項目の写しを，指定されたエラー識別番号に対応するエラー制御表内のエラー項目に格納する。

(引用の形式)

CALL ERRSTR (errno , darea)

errno : エラー識別番号

darea : 指定されたエラー項目を退避すべき8バイトの領域

(4) ERRTRA サブルーチン

このサブルーチンは現在実行中のプログラム単位までのトレースバックマップを出力したいときに使用する。

(引用の形式)

CALL ERRTRA

(5) ERRMON サブルーチン

このサブルーチンは，利用者のプログラムで検出するエラーの処理を行うときに使用する。エラーモニタでは，エラー制御表のエラー項目に従って，打切り回数，メッセージ出力及びエラー修正サブルーチンを呼ぶか否かの制御を行う。通常ERRSETサブルーチンと対で使用する。

(引用の形式)

CALL ERRMON (msg, ret , errno, data1, data2, ……)

msg : この配列の先頭4バイトに，診断メッセージの文字数，それに続いて診断メッセージを示す文字データを入れる。

ret : 復帰コード

errno : 検出したエラーのエラー識別番号

data1, data2 …… : 利用者が定義するエラー修正ルーチンに渡す変数名又は配列要素名

4.3.2 シンボリック・ダンプ

FORTRANプログラムの異常終了時に，原始プログラムをできる限り変更せずに，デバッグに必要な情報を出力する。

(イ) 機 能

シンボリックダンプ機能は，コンパイラオプションSDFを指定して翻訳したプログラムが，実行時に異常終了するか又はSDFDMPサブルーチンを呼び出した場合に，以下の情報を出力する。

(1) プログラム単位の実行順序

(2) プログラム単位ごとに変数及び配列の名前，値，属性。

(3) データセット情報

(ロ) 出 力 例

```
                    *** << SYMBOLIC DUMP FACILITY >> ***

      << CONTROL INFORMATION >>
           (SDFARY) ===> (20,1,1,1,1,1,1)

実行順序 MAIN      IS CALLED FROM O.S       AT ISN = NONE
         SORT      IS CALLED FROM MAIN      AT ISN = 5
  *** PROGRAM UNIT IS MAIN     ***
  NAME  TYPE LENGTH AUTO ATTR ADDRESS  ARRAY-SIZE DIM SUBSCRIPT-INFORMATION
  IDATA  I*4      4           00141DD0          80  1 (1:20)
      SUBSCRIPT  OR  SUBSTRING                    VALUE
      (1) - (20)                    (4096,1024,231,111,99,86,86,53,43,28,23,22,15,10,9,4,3,1,-1,-56)
  *** PROGRAM UNIT IS SORT     ***                VALUE                  値
  NAME  TYPE LENGTH AUTO ATTR ADDRESS
  ICNT2  I*4      4           00142040 0
  ICNT3  I*4 属性 4           00142044 0
  IDATAX I*4      4           00142048 0
  NAME  TYPE LENGTH AUTO ATTR ADDRESS  ARRAY-SIZE DIM SUBSCRIPT-INFORMATION
  DATA   I*4      4           00141DD0          4  1 (1:1)
      SUBSCRIPT  OR  SUBSTRING                   VALUE
      (1)                               4096
  *** FILE INFORMATION ***
       UNIT REFERENCE NUMBER = 6
データセット  DDNAME = FT06F001
 情報        LAST OPERATION IS WRITE
             ACCESS KIND = SEQUENTIAL          DEVICE TYPE = SYSIN/SYSOUT
             << DCB/ACB INFORMATION >>
             RECFM = FBA       LRECL = 137          BLKSIZE = 19043
             BUFNO = 002       DSORG = PS
```

図 4.2 5　　シンボリックダンプ出力例

Fig. 4.25　Example of output of symbolic dump

(ハ) 使用上の注意

(1) ＳＤＦ機能を制御するオプションとしては，コンパイラオプションと実行時オプションがある。

コンパイラオプションは，以下のものがある。

○ＳＤＦ　：ＳＤＦ用のオブジェクトを出力する。

○NOSDF ：ＳＤＦ用のオブジェクトを出力しない。

実行時のオプションは以下のものである。

ＳＤＦ機能を制御する実行時オプションは，コンパイラオプションＳＤＦを指定して翻訳したプログラムに対してだけ有効である。

○SDFARY＝ $n_1 * n_2 \cdots\cdots * n_\ell$　$1 \leqq \ell \leqq 7$　　配列の出力範囲を指定する。

○SDFDMP＝{ 0 | 1 | 2 }　　ＳＤＦ情報の出力を制御する。

(2) サービスルーチンとしてSDFARY サブルーチンと，SDFDMPサブルーチンがあるが，原始プログラムへの挿入箇所を注意しないと出力が膨大になる可能性がある。

4.4 その他

4.4.1 SLC

本プログラムは，FORTRAN−HE でコーディングされている原始プログラムの精度拡張行うものである。

(イ) 機　　能

精度拡張は，指定された型の基本実定数，変数，配列，組込み関数及び基本外部関数に対して，FORTRAN原始プログラムを精度が一段（或いは二段）高い型になるように書き換える事によって達成される。

主な機能を以下に示す。

(1) 基本実定数の精度拡張

実　数　型 ⟶ 倍精度実数型
倍精度実数型 ⟶ 4倍精度実数型
実　数　型 ⟶ 4倍精度実数型

(2) 型宣言文の精度拡張

実　数　型 ⟶ 倍精度実数型
複素数型 ⟶ 倍精度複素数型
倍精度実数型 ⟶ 4倍精度実数型
倍精度複素数型 ⟶ 4倍精度複素数型
実　数　型 ⟶ 4倍精度実数型
複素数型 ⟶ 4倍精度複素数型

(3) 関数の精度拡張

実数型関数名 ⟶ 倍精度実数型関数名
複素数型関数名 ⟶ 倍精度複素数型関数名
倍精度実数型関数名 ⟶ 4倍精度実数型関数名
倍精度複素数型関数名 ⟶ 4倍精度複素数型関数名
実数型関数名 ⟶ 4倍精度実数型関数名
複素数型関数名 ⟶ 4倍精度複素数型関数名

(ロ) 使 用 例

(1) 基本実定数の精度拡張

```
CALL SUB (0.0,  HBC,  3.0*OK)
                ↓
CALL SUB (0.0DC,  HBC,  3.0DO*OK)
```

(2) 型宣言文の精度拡張

```
IMPLICIT REAL (J-L)
          ↓
IMPLICIT REAL*8 (J-L)
```

(3) 関数の精度拡張

```
A=SQRT(X) + EXP(X)
          ↓
A=DSQRT(X) + DEXP(X)
```

(ハ) 使用上の注意

(1) ＳＬＣの処理単位は各副プログラムであって，副プログラム間のつながりは考慮しない。

(2) ＳＬＣによって変換されたフォートラン文は７３，７４，７５カラムに'%ＪＡ' 或は，'#ＪＡ' という識別記号をもっている。

(3) 文字列を挿入することによって，継続行の数が１９行を越えたときは，警告メッセージを出力するので，ＳＬＣ使用者は自分で，その部分を修正する必要がある。

(4) 入力となる原始プログラムは順編成（ＰＳ）ファイルのみである。

4.4.2 自動再リンケージユーティリティ

ロードモジュールライブラリのメンバを対象として，効果的に結合編集をすることができる。

(イ) 機　　能

このプログラムは基本的には２つの部分からなる。１つはリンケージエディタを呼び出す部分であり，他の１つはリンケージエディタに渡すオプションと制御文を生成する部分である。すなわち，自動再リンケージユーティリティは入力されたロードモジュールを解析して，得られたオプションや制御文をリンケージエディタに渡すことにより，ロードモジュールを結合編集するものである。したがって，ロードモジュールの結合編集の際に，リンケージエディタが必要とする制御情報を必要としない。更に，ロードモジュールライブラリの指定されたすべてのメンバーを一度のジョブステップで処理することができる。

(ロ) 使用上の注意

(1) 次のオプション又は制御文は，リンケージエディタ特有のもので，自動再リンケージユティリティでは使用できない。

- ○リンケージエディタのオプション　　NE
- ○リンケージエディタの制御文　　NAME, OVERLAY

飯田　鉱（富士通エフ・アイ・ビー株式会社）
富山　峯秀（計算センター）

参 考 資 料

- ○ ANALYSIS　COMPUTER情報No. 40
- ○ FORTUNE　COMPUTER情報No. 38
  FORTUNE 使用手引書　78SP－5360
- ○ TOPIO　COMPUTER情報No. 42
  JAERI MEMO 59－215
  原子力コードのベクトル化プログラミング〔I〕
- ○ 会話型ベクトライザー　COMPUTER情報No. 42
  会話型ベクトライザー使用手引書　78SP－5690
- ○ TESTFORT77　TESTFORT77 使用手引書78SP－5330
- ○ DOCK／FORT77　COMPUTER情報No. 37
  DOCK／FORT77使用手引書78SP－5340
- ○ JSGCOMPR　データセットユーティリティ使用手引書78SP－1681
- ○ MTDUMP　ジョブ制御文の利用手引
- ○ SFMAKE　ジョブ制御文の利用手引
- ○ VIVAPO　ジョブ制御文の利用手引
- ○ MYBACKUP　ジョブ制御文の利用手引
- ○ COMPACT　COMPUTER情報No. 40
- ○ DEBUGオプション　FORTRAN77 使用手引書78SP－5300
- ○ エラー処理サブルーチン　FORTRAN77 使用手引書78SP－5300
- ○ シンボリックダンプ　FORTRAN77 使用手引書78SP－5300
- ○ 自動再リンケージ・ユーティリティ
  リンケージエディタ／ローダ使用手引書78SP－1651

5. おわりに

本報告書は昭和６０年２月に開催された原子力コード研究委員会原子力コード総合化専門部会で提出討議された検討資料をまとめたものである。開発されるプログラムの大型化に伴い，そのライフ・サイクルにおけるデバッグ，保守，管理に多大の費用がかかるようになってきた。これらの作業を軽減する各種デバッグ，保守ツールは非常に有用なものであるので報告としてまとめることになった。会議に参加し，検討に加わっていただいた方々に感謝する次第である。

なお，原子力コード総合化専門部会のメンバーは次のとおりである。

部会名	氏名	機関名
原子力コード総合化専門部会	下桶敬則	(財)原子力工学試験センター 原子力安全解析所
〃	近藤悟	動力炉・核燃料開発事業団
〃	佐藤一雄	〃
〃	米倉徹	(財)原子力データセンター
〃	荒井長利	原研 多目的高温ガス実験炉設計室
〃	常松俊秀	〃 理論解析研究室
〃	○浅井清	〃 計算センター
〃	竹田辰興	〃 理論解析研究室
〃	内藤孝	〃 核燃料施設安全解析室
〃	鴻坂厚夫	〃 原子炉安全解析室
〃	茅野政道	〃 環境第１研究室
〃	中村康弘	〃 計算センター
〃	樋口健二	〃 〃
〃	平山俊雄	〃 ＪＴ－６０計画室
〃	杉原正芳	〃 炉設計研究室