JAERI-M
90-007

メニュー画面による入力データ作成支援システム：ＪＤＩＳＳ

１９９０年２月

前村　克己・石黒美佐子・土橋敬一郎・平塚　篤**

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

メニュー画面による入力データ作成支援システム：ＪＤＩＳＳ

日本原子力研究所東海研究所計算センター
前村　克己*・石黒美佐子・土橋敬一郎+・平塚　篤**

（１９９０年１月８日受理）

原子力コードの入力データ作成を会話的に行い，入力データ作成の繁雑さを解消したいという要望に応えるため，メニュー画面データ入力支援システム：ＪＤＩＳＳを開発した。ＪＤＩＳＳは，コードに依存しない，また，マルチウインドウなどの機能のない通常のＦＡＣＯＭ端末で利用できる汎用性のあるシステムである。問題に適応した既データを下敷として利用できる。また，既に入力した情報に基づいて動的に画面の制御が行えるなど人間工学的に見て使い易いシステムを目指した。

ＪＤＩＳＳは，内部的には，ＦＡＣＯＭの会話型プログラミング機能（ＩＰＦ）を駆使して作成している。コードの開発整備者が会話型入力画面を作成しようとする場合は，ＩＰＦの定義に煩わされることなく，作成したいメニューをディスプレイ上に直接打ち込むことによって画面作成を行うことができる。

本報告は，ＪＤＩＳＳの機能とＪＤＩＳＳの作成手法の概要について述べるとともに，ＪＤＩＳＳの使用法を例示する目的で，核設計コードシステムＳＲＡＣの入力画面作成方法について示す。

東海研究所：〒319－11　茨城県那珂郡東海村白方字白根２－４
+　原子炉工学部
*　原子力データセンター
**　外来研究員，富士通

JAERI Data Input Support System with Window/Menu: JDISS

Katsumi MAEMURA*, Misako ISHIGURO
Keiichiro TSUCHIHASHI+ and Atsushi HIRATSUKA**
Computing Center
Tokai Research Establishment
Japan Atomic Energy Research Institute
Tokai-mura Naka-gun Ibaraki-ken

(Received January 8, 1990)

JAERI Data Input Support System (JDISS) for nuclear codes has been developed in order to respond to the requirement for conversational data inputing environment delivered from troublesome data input tasks. JDISS is a general purpose window/menu designing tool which is code-independent and can be used on usual FACOM TSS terminals without multi-window facility. JDISS aims at easy to use based on ergonomic approach: An appropriate input model can be selected depending on user options, and window-menus to be processed can be automatically displayed according to preset data.

FACOM Interactive Programming Facility (IPF) is used as internal processing of JDISS. The user who intends to develop interactive input menus for a code can design a menu by inputing the lavout he wants, directly on the display.

In this report, features of JDISS and outline of applied techniques are described. How to use the JDISS is also shown by representing an example, where JDISS is applied to interactive input menu generation for a reactor core analysis code SRAC.

Keywords : Computer codes, Nuclear Codes, Input Menu, Screen, Window, Input Data, Interactive Programming Facility, SRAC Code, JDISS

+ Department of Reactor Engineering
* Nuclear Energy Data Center
** On leave from FUJITSU Ltd.

目　　次

Contents

1. はじめに

大規模原子力コードの処理実行に際して，入力→実行→出力のうち，近年のスーパーコンピュータ等の演算の高速化によって，従来ネックであった演算実行部分の処理時間が飛躍的に短縮され，その結果，処理の効率化が重要視されるのは，入力，出力部分となった。最近の大規模ソフトでは，エンジニアリング・ワークステーション（ＥＷＳ）の発展と相まって，ウインド画面からのデータ入力と，出力結果の画像化，データベース化が通常化し，プレ，及び，ポスト処理を重視した使い易いシステムが多くなっている。

原研においては，ユーザがマン・マシン・インターフェースの向上への要求を持ちながら，ワークステーションの導入が一般化していないなどハード／ソフト環境が整わないことから，実現されていないのが現状である。会話型入力に関しては，これまでに，ＦＡＣＯＭの「会話型プログラミング機能（ＩＰＦ）[1]」を使用してＯＲＩＧＥＮ－Ⅱコードなど個別の要求には応えてきた。出力処理については，３次元コンピュータ・グラフィクス・システムによって計算結果のアニュメーション化などを試行している。

その中で会話型入力については，ディスプレイ上に次々と現れるメニュー画面の指示に従って入力を行うだけで入力データを作成することができる「会話型入力データ作成システム」を利用すると，簡単に正しいデータを作成できるようになり，原子力コードの使い易さの向上に大きく寄与するものである。そのため，原子力コードの入力データをメニュー画面を介して作成したいという一般利用者の要求は年々増加してきている。

原研においては，それら会話型入力データ作成システムの開発はＩＰＦを使用して個々のコードに対処するのが一般的であった。しかし，ＩＰＦを使用して会話型入力データ作成システムを作るには，メニュー画面の定義や，その画面とのデータの受け渡しを行うプログラムの開発といったいくつかの複雑な手順を要し，そのためにはＩＰＦに精通している必要があった。このため，原子力コードの開発者であっても簡単にそのコードの会話型入力データ作成システムを作ることはできなかった。

そこで今回，メニュー画面による会話型入力データ作成システムを作る上での負担軽減を計り，ＩＰＦにあまり精通していない原子力コード開発者が，簡単に会話型入力データ作成システムの開発が行えるように，原子力コード共通に使用できるツールとして「メニュー画面による入力データ作成支援システム：ＪＤＩＳＳ（JAERI Data Input Support System）」の開発を行った。

今回のＪＤＩＳＳの開発は，上記の意味での原子力コード利用の高度化の一環としてなされた。コードの開発整備者にとっては，入力データを会話的に行うためのメニュー画面を容易に作成できること，またコードのユーザにとってはメニュー画面の指示に従ってマニュアルなしで自然な方法で入力データが作成できることを目的とした。

ＪＤＩＳＳ開発の方針は次のとおりである。

(1) 原子力コードと独立にメニュー画面作成が可能。既成の原子力コードに手を加える必要

がない。

(2) メニュー画面作成自身もメニュー画面の操作によってできる。作成したいメニューをディスプレー上に直接打ち込むことによって画面作成を行う。

(3) 原研で最もよく使用されているＦＡＣＯＭ端末，ＮＥＣパソコン（Ｆ6650インターフェース）で使用可能，将来は，マルチウインドを備えたＳＵＮなどのワークステーションでも利用可能とする。

(4) メニュー画面からのデータ入力も原子力コードと独立に行う（中間ファイル作成方式）。

(5) 人間工学的に見て使い易いものとする。

- 計算目的に合ったサンプル入力データを下敷として利用できる。
- 既に入力した値を用いて画面表示順序を制御できる。
- 自由形式入力（Ｔ１）と，変数の見出し付きでカラム毎に入力（Ｔ２）の２種類の表形式入力が可能。
- 入力データ妥当性のチェック（実数／整数，正／負，データのとる範囲など）が可能。（原子力コードの入力エラーチェック機能をＪＤＩＳＳに組み込むことも可能）

ＪＤＩＳＳシステムは個別のメニュー画面を定義する部分であるＪＤＩＳＳＰＨ１とメニューの表示順序を制御する部分（ＪＤＩＳＳＰＨ２）から成る。システム利用の概要はFig. 1.1で示すように

(1) ＪＤＩＳＳそのもの，

(2) ＪＤＩＳＳを利用してのメニュー画面作成，

(3) そのメニュー画面を利用しての原子力コードのデータ入力

の３段階から構成される。

本報告では，コード開発者の立場に立って，(2)ＪＤＩＳＳを利用してのメニュー画面作成を重点的に記述する。(1)については内部資料に留め，(3)については核設計コードＳＲＡＣ[2]のメニュー画面についてユーザ側に立った利用者マニュアルとして今後公開する予定である。

Fig. 1.2にＪＤＩＳＳシステムを使用して，メニュー画面による会話型入力データ作成システムを作り上げ，実際に原子力コードの入力作成で使用するまでの処理の流れを示した。

原子力コードの開発者（或いは整備者）は，あらかじめメニュー画面全体の設計を行い（Step 1），ＪＤＩＳＳＰＨ１によって個々のメニュー画面を作成する（Step 2）。引き続きＪＤＩＳＳＰＨ２を使用して，メニューの表示順序，エラー処理方法など規定する。それらの情報に基づいて会話型入力データ作成システムの「会話処理部」のプログラム・ソースが自動生成される（Step 3）。

一方，原子力コードのユーザはこの会話処理部を使用してデータ入力を実施するが，入力値は「中間ファイル」に保存される。中間ファイル上のデータFormatは必ずしも原子力コードのものと一致していないので，変換が必要となる。この場合のFormat変換プログラムは，個々のコード毎に原子力コードの開発者によって作成される（Step 4）。Step 3とStep 4によってユーザが使用可能な会話型入力データ作成システムが完成する。

原子力コードのユーザは，会話型入力データ作成システムを用いてデータ入力を行い，入力値は中間ファイルに一旦保存される（Step 5－1）。最後に，中間ファイルから原子力コードの

入力形式と整合を取るためのデータ変換（Step 5－2）を行う。

ＪＤＩＳＳをＳＲＡＣに適応した経験から評価すれば，ＪＤＩＳＳＰＨ１についてはほぼ当初の目的を達した。画面の順序を制御したり，意味論的入力エラーチェックを規定する部分が個々のコードに依存した処理として残っておりそれらをＪＤＩＳＳＰＨ２で自動化することは困難である。これらのコード依存部分はＪＤＩＳＳＰＨ２の入力データとして与える必要があるが，ＳＲＡＣのような大型コードでは制御方法の定義が繁雑となりコード開発整備者の負担はやや大きいものとなる。しかしながら，ＪＤＩＳＳシステムを使わずにＩＰＦで総てを解決するのと比較すると，遥かに開発に要する工数は軽減できる。

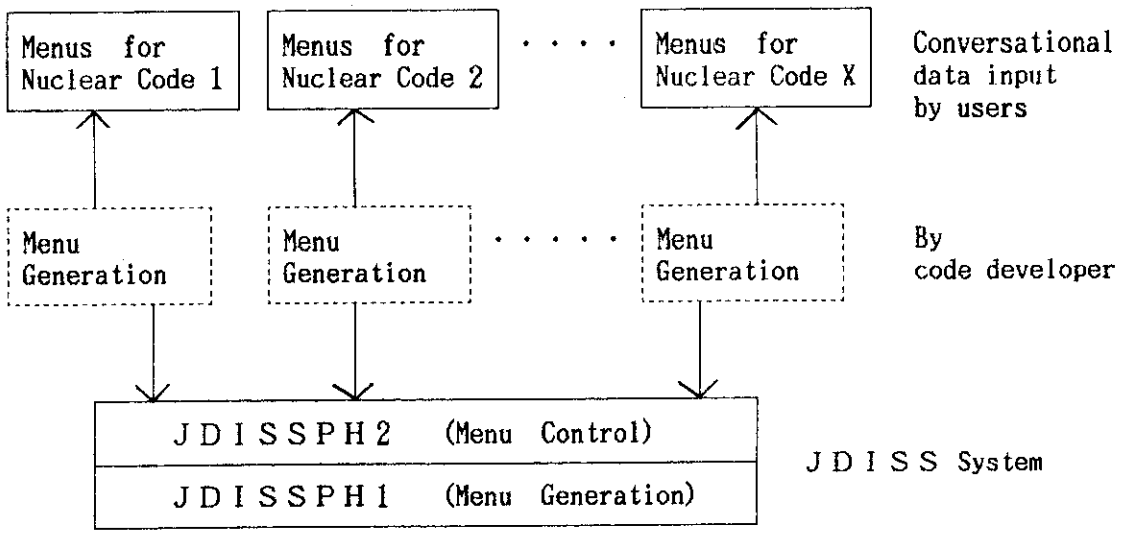


Fig. 1.1 JDISS Overview

処理の流れ

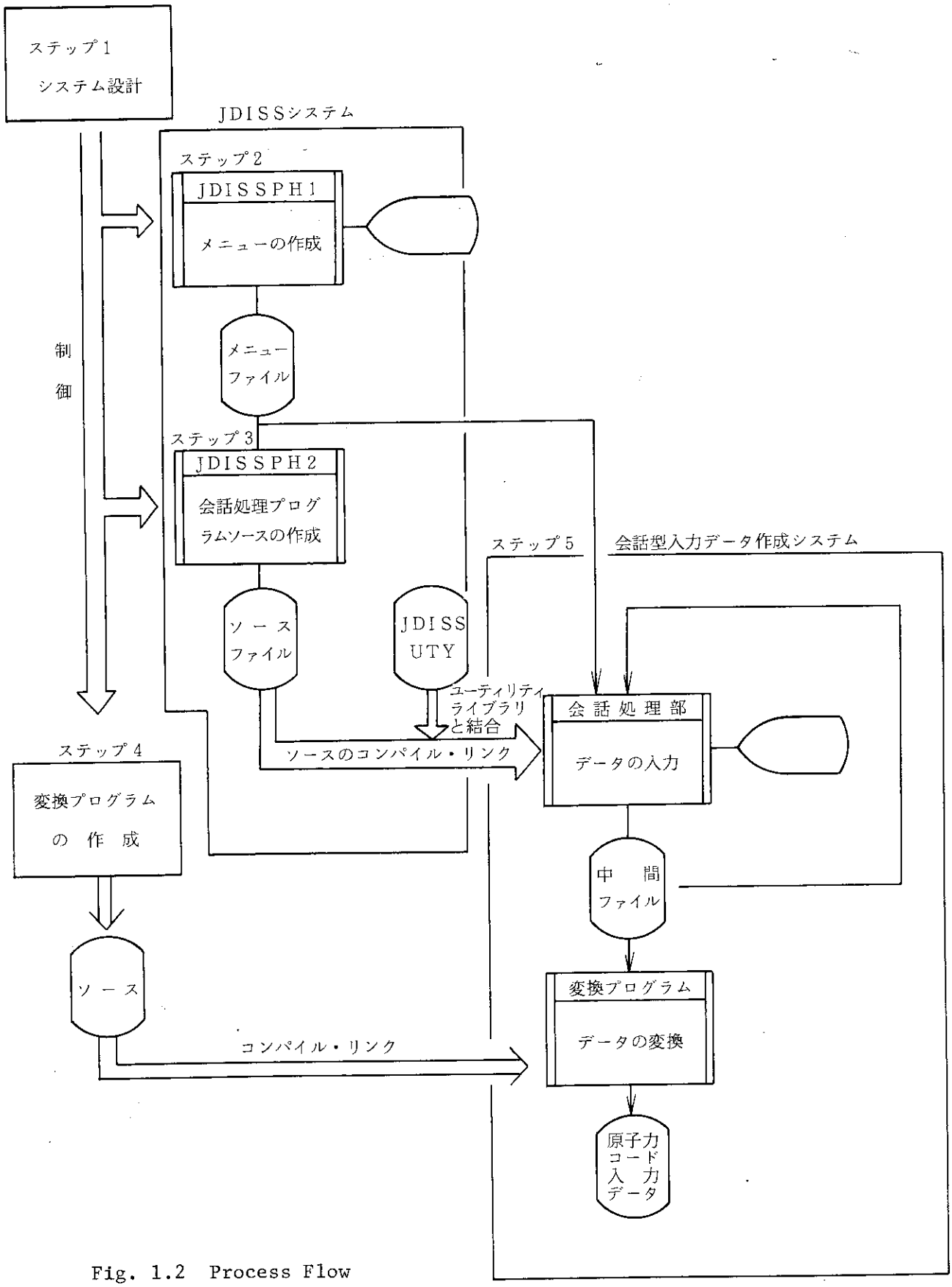


Fig. 1.2 Process Flow

2. ＪＤＩＳＳシステムの概要

2.1 システムの目的

既存の原子力コードを利用する際に，その入力データをディスプレー画面を通じて会話的な処理によって作成していくことは，原子力コードの使い易さの向上に大きく寄与するものである。

しかし，実際にそのような会話型入力データ作成システムを作成するには，ディスプレーに画面を表示させたり，制御を行うツールＩＰＦの機能に精通している必要があるが，それは誰にでもすぐに扱えるという性質のものではない。しかも，作成した会話型入力データ作成システムは，原子力コードの修正の度にメニュー画面の設計を含めた複雑な手順による修正が必要となり，その負荷はかなり大きいものである。

そこで，本システムの開発にあたっては，ＩＰＦの知識を持たない利用者でも，ディスプレー上で会話的に処理を行うことでメニュー画面の作成が行え，その制御プログラムも簡単な入力で自動的に作成できることを目的とした。

2.2 システムの持つ特徴

ＪＤＩＳＳシステムは，メニュー画面を作成するプログラムＪＤＩＳＳＰＨ１と，そのメニュー画面の制御プログラムを自動作成するプログラムＪＤＩＳＳＰＨ２の，２つのプログラムによって構成されているが，両者共に簡単な入力操作を行うだけで，目的のメニュー画面及びプログラムを得ることができる。本システムによって作成される会話型入力データ作成システムがＣＡＬＬするＩＰＦ関連のサブルーチンは，総て自動的にプログラム中に組み込まれるので本システムの利用者（主に原子力コードの開発者であることが予想される）はＩＰＦの知識を持つ必要はなく，原子力コード利用者が利用しやすい会話型入力データ作成システムの設計にのみ考えを傾注すればよい。

また，ＦＯＲＴＲＡＮ－77文法に則った入力データを作成することで，データの細かい処理を自由に行うことができるようになっている。

2.3 ＪＤＩＳＳＰＨ１プログラム

本プログラムは，ディスプレー上に表示するメニュー画面の作成を行う。いままでのＩＰＦによるメニュー画面の作成はメニューファイル上に，定められた形式でメニュー画面の定義を行うことによっていたが，その定義たるや，アトリビュートの定義，メニュー原型部の定義，と多岐にわたり極めて繁雑であった。しかも，メニュー原型部での入力部分と，アクション部の変数との対応などつい間違ってしまいそうな項目も数多く見受けられた。

そこで，ＪＤＩＳＳＰＨ１プログラムでは，メニュー画面の開発をディスプレー上で行い，自分の設計通りに入力することで目的のメニュー画面が得られるようにした。１枚のメニュー画面の作成は，画面のデザインを行うデザイン定義部分と，メニュー画面より入力するデータと変数の定義を行う変数定義部分より成り立っている。それらの実行はすべてディスプレー上で会話的に行い，設計者のデザイン通りに入力するだけでＩＰＦで使用するメニュー定義体が作成されてメニューファイル中に格納される。また，過去に本プログラムによって作成されたメニューを画面上に呼び出して，修正を加えて別のメニュー定義体として格納することも可能である。

本プログラムで作成されるメニュー画面は，入力すべきデータによって，下記のように３つのタイプに分けられる。

1) 一般画面

配列ではない一般の制御データを入力する。自由にデザインされた画面の入力域から入力された値を，入力データ中の対応する変数に格納するものである。

2) テーブル画面－１

多量の数データを一度に入力し，連続領域の文字型配列に格納する。主に，大きな配列などの入力に使用する。ある特定の値，ないし指定変数の値の数だけ，数値データの入力を促すメニューである。

3) テーブル画面－２

特定の形式に従ったデータを連続的に入力し，それらを文字型配列に格納する。主に一定のフォーマットを繰り返して入力するものに使用する。１枚のメニューに，ある特定の数，ないし変数の値だけデータ入力を行い，それを繰り返していくものである。

これらの詳細については４章で述べる。

作成されたメニュー画面は，指定されたメニューファイルに１画面１メンバーとして格納される。

2.4 ＪＤＩＳＳＰＨ２プログラム

本プログラムは，ＪＤＩＳＳＰＨ１で作成されたメニュー画面の関係を記述した簡単な入力データより，会話型入力データ作成システムのソースプログラムを作成するプログラムである。例えば，従来のＩＰＦでは，メニュー画面を呼び出すためにはＩＰＦＯＰＮ，ＩＰＦＭＩＯといった各種のサブルーチンをプログラム中に記述する必要があった。ＪＤＩＳＳＰＨ２では，各種ファイル名や入力ファイル・フォーマットを定義した定義文と，入力データに依ってどのメニューを表示するのかといったメニュー画面呼出順序や中間ファイルへのデータ書出しタイミングをＦＯＲＴＲＡＮで記述した制御文の２つを入力するだけで，メニュー画面の表示を制御するプログラムを自動的に作成する。その内容は，ＪＤＩＳＳＰＨ１で作成された各メニューを処理する各々のサブルーチン群とそれらを使って全体のメニュー表示の流れを制御する部分から成っている。これに加うるに，総てのメニューで定義された入力変数と，ＩＰＦ用制御データ及びプログラム制御データをＣＯＭＭＯＮとして格納しているＩＮＣＬＵＤＥ文を自

動的に作成する（Fig. 2.1 参照）。

2.5 会話型入力データ作成システム

メニュー画面による入力データ作成システムのことで，原子力コードのユーザによって利用される。

システムはＪＤＩＳＳＰＨ２によって自動的に作成される部分と原子力コード開発者が作成する部分の２つにより構成されている。前者はディスプレー上にメニュー画面を表示して会話的にデータの入力を次々と行い，その値を中間ファイルに格納するものである。後者はその中間ファイル中の値から，対象原子力コードの入力フォーマットに沿った入力データを作成するものである。このように，ＩＰＦを利用した会話処理部分と対象原子力コードの入力に依存する２つの部分を中間ファイルによって結合することでＩＰＦのサブルーチンを使用する会話処理部分を完全に自動作成することができるようになった。

自動作成されるプログラムの持つ主要機能を以下に示す。

i) 指定した中間ファイルの値を，デフォルト値として使用できる。これによって，原子力コード利用者は必要なデータのみを変更すればよく，過去に作成したデータを有効に再利用できる。

ii) 中間ファイルは，ブロックと呼ばれるいくつかの変数の組によって分割されており，ブロック単位にデフォルト値を与える中間ファイルを変えることができる。これによって，いくつかの中間ファイルを組み合わせて下敷とすることが可能となった。ブロックは通常コードシステムの１つの構成要素である単体コードに対する入力データを対象とする。

また，オプションによっては中間ファイルに格納するデータのロジックを組み込むことによって，中間ファイルに格納するデータを，その時の条件によって選択的に格納することができる。

iii) 内部の処理はセクションと呼ばれる，木構造を持つメニュー画面の組に分けられ，それらを単独に呼び出して処理を行う。セクションでの各々のメニュー画面の制御は以前にメニュー画面より入力された変数の値を用いて必要に応じて呼び出したり，飛ばしたりという制御を行っている。

iv) メニュー画面より入力される値は，メニュー画面を作成する段階での指定により，変数のタイプや範囲の妥当性のチェックが行える。また，画面それ自身の制御も入力値によって制御することができる。

v) ＰＦキーの操作によって，次のメニュー画面に進んだり，１つ前のメニュー画面に戻ったり，セクション呼び出しメニュー画面を表示させたりといった制御が行える。

以上が主な会話型入力データ作成システムの機能であるが，どのような順番で，如何なる条件の時にどのメニュー画面を呼び出すかを定義している木構造を，ＪＤＩＳＳＰＨ２で定義する制御文は，ほぼＦＯＲＴＲＡＮ－77の文法に従って記述されるので，その定義文中に各種の判定文を挿入することによって，さらにきめ細かい操作を行うことも可能である。

各機能の詳細については後述する。

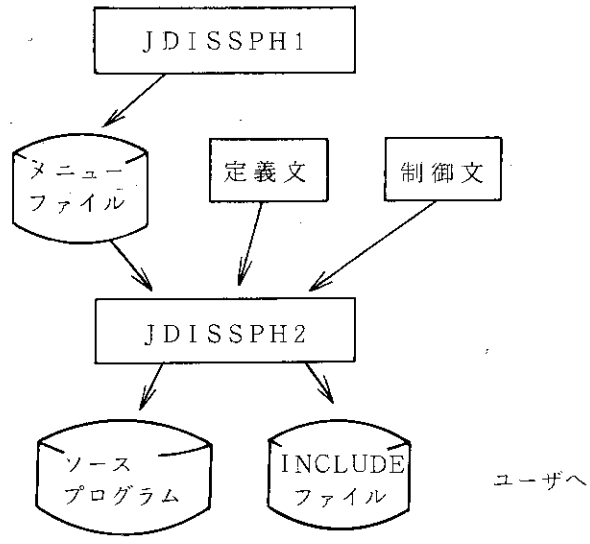


Fig. 2.1 Construction of JDISS

3. 処　理　の　流　れ

Fig.1.2にＪＤＩＳＳシステムを使用して，会話型入力データ作成システムを作りあげて，実際に稼働するまでの処理の流れを示した。各ステップで使用するプログラムの作成者，利用者はステップ毎に異なる。

(1) ステップ１　＜システム設計＞

設計者　：　原子力コードの開発者

対象原子力コード（本例ではＳＲＡＣ）の入力構造を分析して，コードの利用者が利用しやすい会話型入力データ作成システムの設計を行う。対象原子力コードの構造についてあまり詳しくない人を，会話型入力データ作成システムの利用対象者として考えていることから，まず画面を見ただけで入力するデータが何を意味するものであり，利用者の目的を果たすためにはどのような値を入力すべきであるのか，といった情報が一目で判るようなメニュー画面が次々に表示され，利用者はその指示に従って入力するだけで入力データが得られるような構造を持つシステムの設計を，綿密に行う必要がある。具体的な作業としては以下のようなものがある。

a) 作成する会話型入力データ作成システムの構造を決定する。どのような順番で，データをメニュー画面から入力するかを定める。オリジナルの入力フォーマットに拘束されることなく，会話型入力データ作成システムを利用する利用者が最も利用しやすいような入力の流れとすべきである。

b) メニュー画面のデザインを決める。メニュー画面には，用途に応じて３つのタイプが用意されているので，入力するデータによってその中から選択する。デザインは，そのメニュー画面を見ただけで必要な入力データが分かるようなコメントを簡潔に記述してあるようなものが望ましい。

c) 入力データを，意味のあるブロック単位に分割する。下敷となる中間ファイルをこのブロック単位で指定することになる。

d) 複数のメニュー画面からなる処理単位（＝セクション）に，処理を分割する。この処理単位を選択して，入力データ作成の処理を行うこととなる。ひと塊の木構造を持つ処理を１つのセクションとする。

今回のＳＲＡＣにおける適用では，中間ファイルのブロックとセクションを一致させることによって，処理毎に任意の中間ファイルを選択できるようにした。

(2) ステップ２　＜ＪＤＩＳＳＰＨ１を使用してのメニュー画面の作成＞

ＪＤＩＳＳＰＨ１作成者　：　ＪＤＩＳＳシステム開発者

〃　　利用者　：　原子力コードの開発者

ＪＤＩＳＳＰＨ１を実行して，ステップ１でデザインされた通りにメニュー画面を作成し

ていく。ＪＤＩＳＳＰＨ１は，会話型プログラムであるので，利用者はディスプレー画面に，そのメニュー画面のデザインを入力していくだけで，思い通りのメニュー画面を作っていくことができる。このとき，過去に作成したメニュー画面の一部を再利用してメニュー画面を作ることもできる。

作成したメニュー画面は，メニューファイルに格納される。

(3) ステップ３　＜ＪＤＩＳＳＰＨ２を使用しての会話型入力データ作成システムの会話処理部分のプログラムソースの作成＞

ＪＤＩＳＳＰＨ２作成者　：　ＪＤＩＳＳシステム開発者

〃　　利用者　：　原子力コード開発者

使用するメニューファイルや中間ファイルのフォーマットを指定した定義文とメニュー画面の木構造を記述した制御文を入力して，会話型入力データ作成システムの会話処理部分のプログラムソースを作成する。

(4) ステップ４　＜フォーマット変換部分の作成＞

フォーマット変換プログラム作成者　：　原子力コードの開発者

ステップ３で作成された会話処理部分を実行して得られる中間ファイル中に格納されているデータを，対象原子力コードの入力フォーマットに変換するプログラムを作成する。

(5) ステップ５　＜会話型入力データ作成システムの実行＞

会話型入力データ作成システム環境整備者　：　原子力コードの開発者

〃　　利　用　者　：　原子力コードのユーザ

ステップ３とステップ４で作成したプログラムを翻訳，実行して，会話型入力データ作成システムとして稼働させる。

まず，ステップ３で作成した会話型処理部分のプログラム中では，ユーティリィティー・ライブラリィー（ＪＤＩＳＳＵＴＹ）中のサブルーチンをコールしているので，リンク時に結合してロードモジュールを作成する。そして実行時には，ステップ２で作成したメニューファイルとシステムで用意したシステムメニュー・ファイルを使用して実行する。

次に，会話処理部分プログラムの実行で得られた中間ファイルを入力として，ステップ４で作成したフォーマット変換プログラムを実行して，求める入力ファイルを得る。

この２つのプログラムは中間ファイルによってインターフェースをとっているので必ずしも連続して実行する必要はない。会話処理部分を先に何度も実行して，いくつか中間ファイルを作成した後に，まとめてフォーマット変換プログラムを実行していくことも可能である。

4. メニュー画面定義の種類

ＪＤＩＳＳシステムで作成するメニュー画面と変数の関係は，原則として１対１の関係で結ばれている。すなわち１つのメニュー画面のある入力領域は，１つの変数と結ばれており，メニュー画面より入力された値が，条件によって選択的に変数に振り分けられることはない。また，配列に値が与えられることもない。（原子力コードの入力データとして配列を設定する必要があるときには，中間ファイル中には独立変数として定義されているものを，配列に置き換える処理をフォーマット変換プログラム中で行う。）

これによって，ＪＤＩＳＳシステムでの入力データ構造は簡潔なものとなり，初めてメニュー画面のデザインからメニュー画面制御プログラムの作成までの自動化が可能となった。

しかし，１つの入力域が１つの変数に対応するということは，多量の入力データを一度に扱う場合や，同種のデータを繰り返し入力する必要のある場合には不便を逃れ得ない。そこで，それら多量のデータを一度に取り扱えるメニューもテーブル画面として用意した。

会話型入力データ作成システム設計者は用意されたパターンから最適なものを選択して，メニュー画面のデザインを行う。

4.1 一般画面

一般画面は，アンダーラインによって示された入力可能領域に値を入力していくものである。入力された値は定義された変数に代入される（後述）。入力可能領域以外の領域には，入力すべき変数の説明が記述されており，そこには他のメニュー画面より入力された変数を表示させることもできる。Fig.4.1に一般画面の例を示す。

4.2 テーブル画面－１（Ｔ１）

テーブル画面－１は一度に多くの実数と整数型データを入力するものである。Fig.4.2にテーブル画面－１の例を示した。２－６行目が自由設定領域と呼ばれる領域であり，一般画面と同様な変数と，テーブルデータ制御変数が定義されている。テーブルデータ制御変数とは，７行目以降のテーブルデータ入力域（アンダーラインの引かれている領域）に入力すべきデータの個数を指定するものである。その値は，このテーブル画面－１で入力することも，他のメニュー画面で入力された値を使用することもできる。

テーブルデータ入力域に，原研科学計算ライブラリーＪＳＳＬ中のＲＥＡＧの入力規則に則ったフリーフォーマットで入力を行う。（ＲＥＡＧのフリーフォーマットについては参考文献[4]を参照のこと。）このとき，テーブルデータ制御変数によって指定された個数のデータを入力するまで入力が促がされる。また入力値に数字でないものがあった場合には，その旨表示される。もし，１画面に収まりきれない場合には，画面デザイン時に定められた最大行数まで，ＰＦ７・

ＰＦ８キィーでそれぞれ上下にスクロールを行いながらデータ入力できる。

入力されたデータは，指定された名称の76×n（ｎは入力された行数）の大きさを持つ文字型配列に，画面より入力された通りの形で中間ファイルに格納される。従って，中間ファイルから入力データに変換を行う時に，必要に応じてフォーマットを変換したり，実数型ないし整数型にデータタイプを合わせる必要がある。

4.3　テーブル画面－２（Ｔ２）

テーブル画面－２は，単純なフォーマットを持つ入力を繰り返し行うためのものである。Fig.4.3にテーブル画面－２の例を示した。２－７行目は自由設定領域であり，一般の変数とテーブル制御データを入力，表示する部分である。テーブル制御データとは，８行目以降のデータ入力の繰り返し回数を指定するものであり，自由設定領域で入力してもよいし，他のメニュー画面から入力された値を使ってもよい。この指定がない場合は，画面デザイン時に定められた最大値まで入力ができる。

８行目以降のテーブルデータは，８行目のレコード数指定行，９行目のコメント行，10行目～22行目までの繰り返しレコード行，の３つに分かれており，８～22行までのテーブルデータを一組として，それを指定の組数入力することができるようになっている。

８行目には，一般変数と，10～22行目に入力される繰り返しレコードの入力行数を指定する繰り返し制御変数を入出力する。繰り返し制御変数は，８行目で入力しても良いし，以前に入力された値を使用してもよい。またファンクション機能（後述）を使って計算値から求めることもできる。どの方法をとるかは画面の設計にかかっているが，８行目には必ずこの繰り返し制御変数を指定していなければならない。

10～22行目のデータは，任意のタイプの配列データをカラムワイズに入力するための領域である。最大13行まで同じフォーマットで入力するものである。この行に何行入力させるかは，８行目の繰り返し制御変数の値によって決まる。

８～22行目までのテーブル画面－２で入力されたデータは，９行目のコメント行を除き，１行中の変数間の空白を圧縮して，指定された名称の76×n（ｎは入力された行数）の大きさを持つ文字型配列として中間ファイルに格納される。従って中間ファイルから入力データに変換を行う時に，必要に応じてフォーマットを変換したり，実数型ないし整数型にデータタイプを合わせる必要がある。

一般画面の例

```
MEMBER-NAME | S09N01 | COMMENT | REACTION RATE CALCULATION

        <<  BLOCK 1    CONTROL  INTEGERS  >>            SECT. 09  PAGE 01

        1     NUMBER OF DETECTORS USED WITHOUT FILTER

        1     NUMBER OF DETECTORS USED WITH FILTER

        1     NUMBER OF CASES TO CALCULATE THE SPECTRUM INDEX

        1     SIGNED NUMBER OF MIXTURES WHICH CONTAIN THE NUCLIDES WITH
              IXMICR=1. CARE SHOULD BE TAKEN IN DEPLETION PROBLEM WHERE
              IXMICR IS AUTOMATICALLY ASSIGNED AND THIS EDIT WORKS AFTER
              THE FINAL BURN-UP STEP.

              NOTES: IXMICR ==> SECT.08 MATERIAL SPECIFICATION
                                PAGE.?? FLAG TO WRITE

   PF2 : BACK | PF3 : NEXT MENU | PF4 : RETURN
```

Fig. 4.1 Example of Usual Menu

テーブル画面－１の例

MEMBER-NAME	S10N03	COMMENT	CELL BURN-UP CALCULATION

PRIOD : BURN-UP STEPS IN ASCENDING ORDER (NOT COUNT THE INITIAL STEP)
IN UNIT SPECIFIED BY IUNIT. A MODIFICATION IS MADE FOR THE ACTION
AT THE BURN-UP STEP TO YIELD ONLY THE COMPOSITION, BUT TO SKIP THE
CELL CALCULATION. ENTRY PERIOD 1 , PIECES

01
02
03
04
05
06
07
08
09
10
11
12
13
14
15

PF2 : BACK	PF3 : NEXT MENU	PF4 : RETURN	PF7 : DOWN	PF8 : UP

Fig. 4.2 Example of Free-format Table(T1)

テーブル画面－２の例

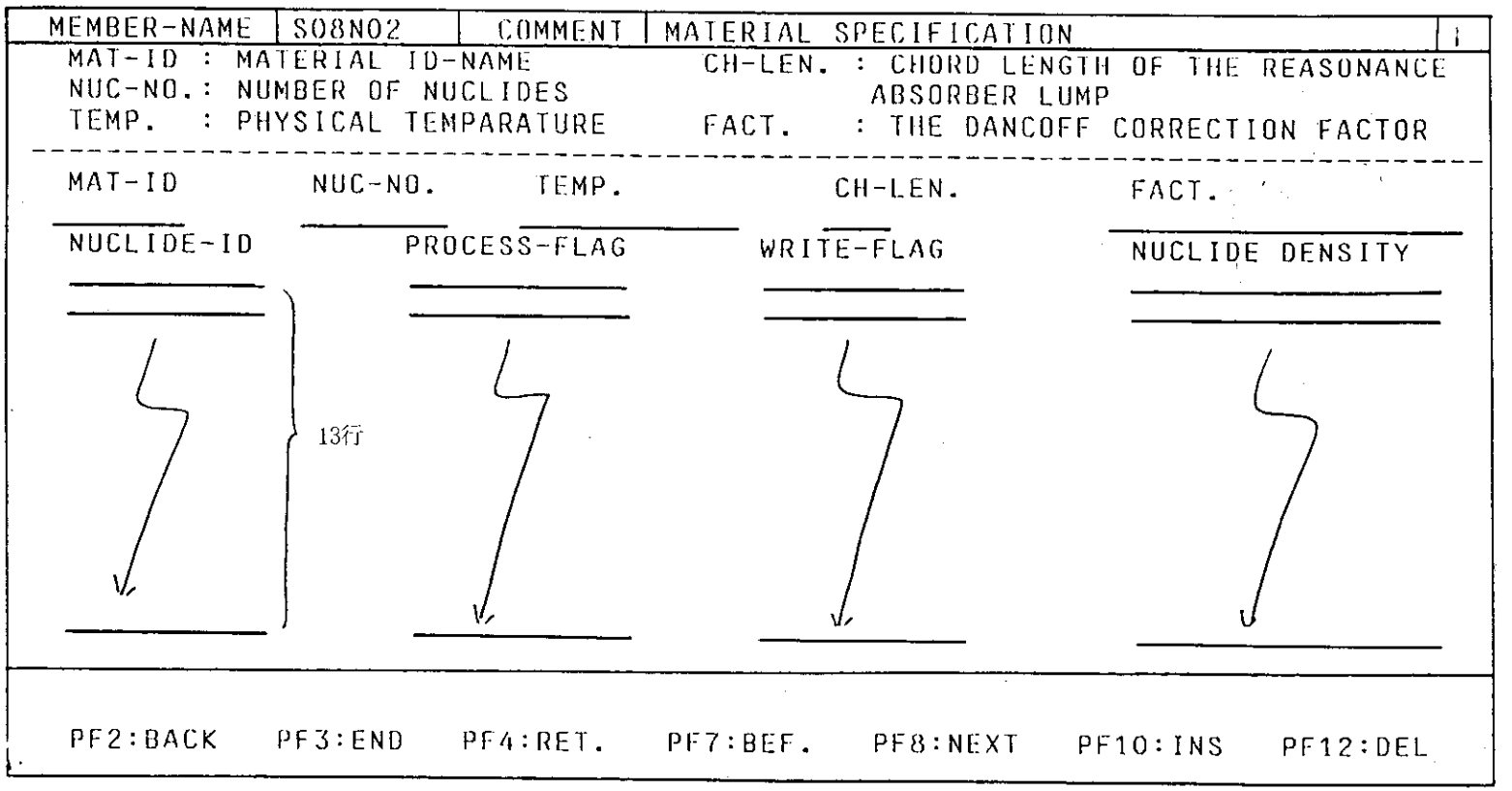


Fig. 4.3 Example of Colunm-wise Table(T2)

5. 中間ファイル

中間ファイルは，会話型入力データ作成システムのうち，会話処理部を用いて，（原子力コードのユーザ）がメニュー画面より入力したデータを格納するためのものである。ユーザは，データ変換プログラムを用いて，中間ファイルをさらに，原子力コード個有の入力形式に変換する必要がある（Fig. 2.1参照）。

会話処理部ではこの中間ファイルを作成するのみではなく，後述するように，各ブロック毎に会話処理プログラムを実行する上での下敷ファイルとしても使用する。

また，入力データの値によっては不要なデータも出てくるので，その際のロジックを￥ＳＡＶＥＦＴという名前のサブルーチンに組み込むことによって，必要なデータのみを中間ファイルに格納させることができる。

5.1 ブロック分割

中間ファイルは，その変数をいくつか集めてひと塊のデータの組としたブロックという概念によってその内部が分割されている。ブロックは，中間ファイル中の先頭カラムが“@”で識別された6文字以内の英字で始まる英数字によって名前付けられている。会話処理プログラムの実行時にはブロックを単位として，異なる中間ファイルを割り当てるので，意味を持ったまとまりをブロックとするべきである。

今回のＳＲＡＣへの適用の場合を例にとってみると，後述するセクションで入力する変数をひと塊にしてブロックとすることによって，各セクション毎に異なる中間ファイルの値をデフォルト値として採用できるようにした。

5.2 中間ファイルのフォーマット

5.2.1 属　性

中間ファイルの属性は，ＲＥＣＦＭ＝ＦＢ，ＬＲＥＣＬ＝８０，の行番号をつけないファイルである。１つの中間ファイルそれ自身はＰＳファイルであるが，１つの中間ファイルを１メンバーとしていくつかの中間ファイルをメンバーとするＰＯファイルとして管理することができる。

このファイルの属性は極めて一般的なものであるので取り扱いが簡単であり，ＰＦＤやＥＤＩＴコマンドによって直接ファイルの中身を確認したり修正を加えることもできる。

5.2.2 データの種類

中間ファイルに格納されているデータは，下記の５つのデータに分類することができる。

a) ブロックマーク

中間ファイル中におけるブロックの区切りを示す。

b) コメント行

各種のコメントである。各項目の間であればどこに存在してもよい。最終作成日付は会話処理プログラムが自動的に付加するが，その他に利用者がＰＦＤやＥＤＩＴコマンドによって任意に付け加えることもできる。また下敷ファイル中にある利用者が加えたコメント行は，会話型入力データ作成システムによって作成された新規の中間ファイル中にも全く同じようにコピーされる。

c) 一般変数格納行

一般画面中で使用される変数や，テーブル画面－１及びテーブル画面－２の自由設定領域中の変数名とその値が格納されている。

d) テーブル１データ格納行

テーブル画面－１より入力されたテーブルデータが，入力データ数等の制御情報と共に格納されている。

e) テーブル２データ格納行

テーブル画面－２より入力されたテーブルデータが，入力行数等の制御情報と共に格納されている。

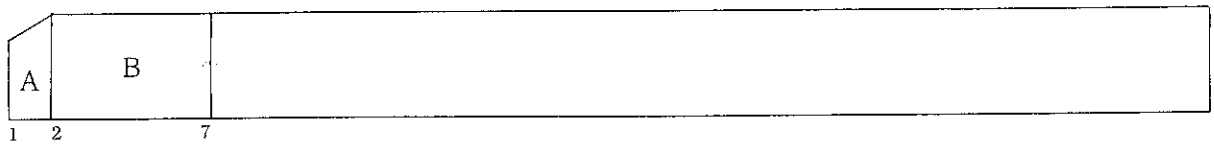
これらは，ファイル中のどこにどの順序で現れてもかまわないが，各ブロックに含まれる変数は，会話型入力データ作成システムにおいては常に同じものでなければならないし，中間ファイル中に存在するデータは使用する会話型入力データ作成システム中では必ず同じでなければならない。

5.2.3　各データのフォーマット

5.2.2で示した各データのフォーマットを以下に示す。また実際の中間ファイル例をFig. 5.1に示す。

i) ブロックマークのフォーマット

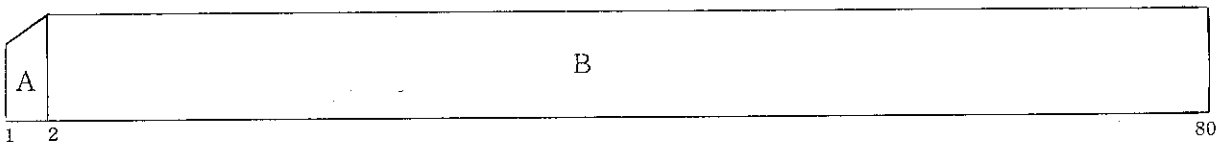
中間ファイル中で定義されるブロックの始まりとその名前を示す。これ以降のデータは，次のブロックマークが現れるまで，ここで定められた名称のブロックとして認識される。



記 号	カ ラ ム	内　　容
A	1 ～ 1	@記号で示されるブロックマーク開始記号。
B	2 ～ 7	ブロック名。英字で始まる６文字以内の英数字。

ii) コメントのフォーマット

各種のコメントである。項目と項目との間であればどこにあってもよい。システムでは先頭の日付のみコメントとして付加する。必要ならば利用者が他の部分に加えてもよい。

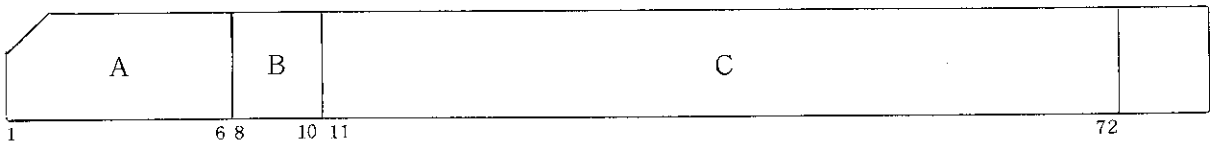


記号	カラム	内容
A	1～1	*記号で示されるコメント認識記号。
B	2～80	任意のコメント。

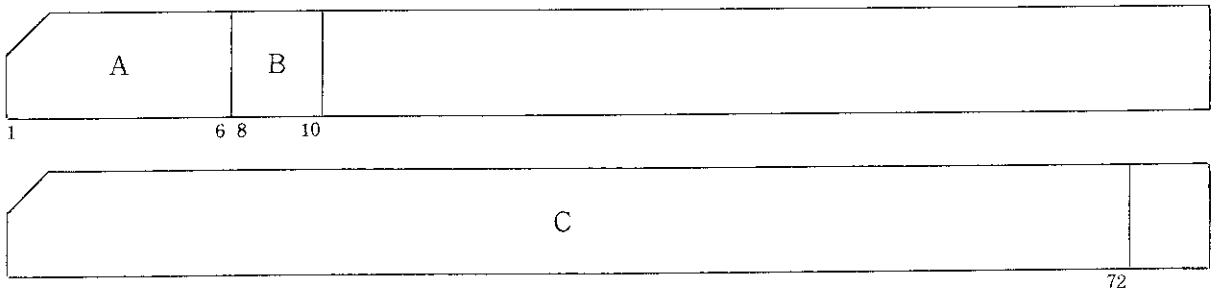
iii) 一般変数のフォーマット

一般変数を格納している。原則的に1変数1枚であるが，61バイト以上を占有する文字型変数の場合のみ2枚のカードを使用する。

1変数に1枚使用する場合



61バイト以上の文字型データを格納する場合



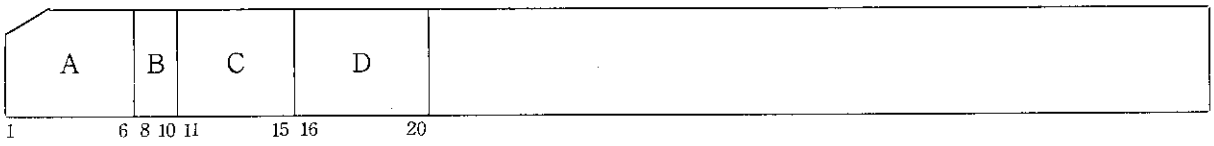
記号	カラム	内容
A	1～6	変数名。6文字以内の英字で始まる英数字。
B	8～10	変数のタイプを示す記号。 I－－ ： 整数型のデータであることを示す。 R－－ ： 実数型のデータであることを示す。 Cnn ： 文字型のデータであることを示す。 nnは文字型データの長さを表している。01から78までの値をとる。空白は許されず，1桁の場合も01，02のように書く。
C	11～72	実データ領域。実際に入力された値が記述されている。 ただし，文字型データで長さが60を超える場合には本欄は空白となり次のカードの先頭より記述される。

Ⅳ) テーブル画面－１データ（Ｔ１）のフォーマット

Ｔ１データは，テーブルの制御情報を記述したカード１枚と，複数枚の実データカードより構成されている。

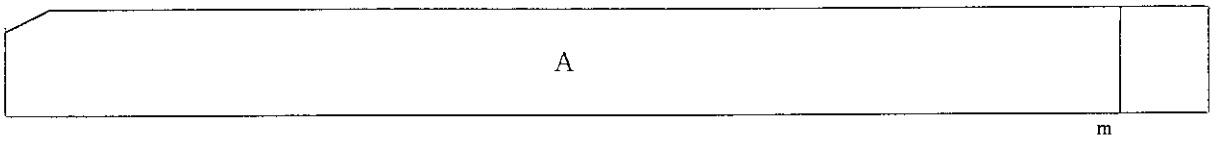
……制御情報カード……

Ｔ１データの先頭に１枚存在する。



記号	カラム	内容
A	1～6	Ｔ１データの認識名。会話処理プログラム中ではこの名前の配列に実データを格納している。
B	8～10	“Ｔ１　”と記述してあり，これによって以降のデータがテーブル１データであることを示している。
C	11～15	５桁の整数で，２枚目以降のカード中にデータの格納されている先頭からのカラム数を示す。現在は７２に固定。
D	16～20	５桁の整数で，２枚目以降のデータの枚数を示す。

……実データカード　上記Ｄ欄の枚数繰り返す。……



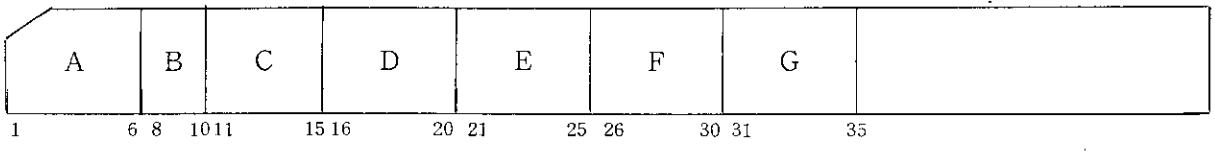
記号	カラム	内容
A	1～m mは上記C欄によって定められた長さ。	実際に入力されたＴ１データ。会話処理によってテーブル画面－１より入力されたままの形式で記述されている。本カードは上記Ｄによって定められた枚数だけ入力されている。

Ⅴ) テーブル画面－２データ（Ｔ２）のフォーマット

Ｔ２データは，テーブルの制御情報を記述した数枚のカードと，複数枚の実データカードより構成されている。

……制御カード１……

Ｔ２データの先頭に１枚存在する。



記 号	カ ラ ム	内　　容
A	1～6	Ｔ２データの認識名。会話処理プログラム中ではこの名前の配列に実データを格納している。
B	8～10	“Ｔ２　”と記述してあり，これによって以降のデータがＴ２データであることを示している。
C	11～15	5桁の整数で，実データカード中にデータの格納されている先頭からのカラム数を示す。現在は７２に固定。
D	16～20	5桁の整数で，実データの第１レコードと第２レコードの入力組数を格納している。
E	21～25	5桁の整数で，第１レコード中の変数の数を格納している。
F	26～30	5桁の整数で，第２レコード中の変数の数を格納している。
G	31～35	5桁の整数で，第２レコードの繰り返し回数を指定している変数が，第１レコードの何番目の変数であるかを示している。

……制御カード２……

第１レコード，第２レコードのフォーマットを格納している。第１レコード用１枚，第２レコード用１枚，計２枚が標準であるが，１つのレコード中の変数が２０を超えるごとに１枚ふえる。

初めに第１レコードのフォーマットを記述したカードがあり，その次に，第２レコードのフォーマットを記述したカードが存在している。

A	A	A	・・・

1 3 4 6 7 9

記 号	カ　ラ　ム	内　　容
A	3*(n−1)+1	データの型と各々の長さ。制御データ１中のＥないしＤで示された数だけ並んでいる。１枚のカードで記述できる個数は２０個までなので，それ以上の場合は，次のカードに続く。各々，変数の型や長さは以下のように記述されている。 Ｉｎｎ　：　整数型データであることを示している。 Ｒｎｎ　：　実数型データであることを示している。 Ｃｎｎ　：　文字型データであることを示している。 ｎｎは各々が占めるカラム数であり，００～７２までの値をとり，空白は許されず，１桁の場合も０１，０２のように記述する。

……実データカード　第１レコードと第２レコードの組を制御カード１中のＦ回繰り返す…

A

記 号	カ ラ ム	内　　容
A	1～m	実際に入力されたＴ２データの第１レコード。制御カード２で示されたフォーマットに従って記述されている。

記 号	カ ラ ム	内　　容
A	1～m	実際に入力されたＴ２データの第２レコード。制御カード２で示されたフォーマットに従って記述されている。このレコードは，第１レコード中の，制御カード１のＧで示された部分にある変数の値だけ繰り返す。

中間ファイルの例

```
*
* MIDWAY FILE  CREATED DATE : 89-09-29
*
@GENERL )ブロック名 ──────────────────────────────────────
TITLO   C53TEST OF SRAC INPUT PROGRAM.
CASEO   C04TEST
FLAG1   C01Y
IC1     I                   1
IC2     I                   1
IC3     I                   0
IC4     I                   0
IC5     I                   0
IC6     I                   1
IC7     I                   4
IC8     I                   0
IC9     I                   1
IC10    I                   0
IC11    I                   0
IC12    I                   4
IC13    I                   0
IC14    I                   0
IC15    I                   0
IC16    I                   0
IC17    I                   1
IC18    I                   0
IC19    I                   0
IC20    I                   1
FLAG11  C01Y                                    一つのブロック
FLAG2   C01N
FLAG3   C01Y
FLAG31  C01F
IFLG33  I                   1
FLAG4   C01Y
BSQ     R     1.2340002E+00
NEF     I                   0
NET     I                   0
NERF    I                   0
NERT    I                   0 ──────────────────────────────
@COLLIS
IGT     I                   1
NZ      I                   3
NR      I                   2
NRR     I                   2
NXR     I                   2
IBOUND  I                   0
NX      I                   4
NY      I                   4
NGR     I                   8
NDA     I                   0
IBETM   I                   0
IEDPIJ  I                   0
IPLOT   I                   0
NREG    T1    72      1   }テーブル－１データ
1 2 3
MAR     T1    72      2
1111
-22
IRX     T1    72      1
```

Fig. 5.1 Example of Intermediate File

```
1 2 3 4 5
IDB      I                    1
IGEOM    I                    2
MODEL    I                    3
RF       R     1.0000000E+01
RM       R     2.0000000E+01
@TUD
NRMAX    I                    5
IG2      I                    1
NXRO     I                   12
IBOUN2   I                    1
NK       T1     72     5
11
22
33
44
55
IK       T1     72     1
111 222 333 444 555
RX       T1     72     2
1.234 2.345 3.456
4.567 5.678
IXRTUD   T1     72     1
123 234 345 456 567
BSQ2     R     1.1100000E+01
XLAMD    R     2.2199997E+01
@MATRAL
NMAT     I                    3
MTCMP    T2     72     3     5     4     2
C08I03R12R12R12
C08I06I06R12
MAT001   0   0.002          0.003          0.004
MAT002   13  1.11           2.22           3.33
1                      111
2                      222
3                      333
4                      444
5                      555
6                      666
7                      777
8                      888
9                      999
10                     1000
11                     1111
12                     1222
13                     1333
MAT003   3
1111111
22222222
33333333
@BURNUP
NEP      I                   11
IBUNIT   I                    1
IBEDIT   I                    2
POWERL   R     1.1123400E+01
CVOL     R     2.2345596E+01
PERIOD   T1     72     2
1 2 3 4 5 6 7 8 9 10
```

テーブル－２データ

Fig. 5.1 (Coutinued)

6. セクション選択

一般に，会話形式で処理を行う場合Fig.6.1に示すように，いくつかのメニュー画面によって組み立てられた木構造が，複数集まって構成されている場合が多い。そこで，それら1つ1つをセクションと呼ばれる単位にまとめ，修正の必要のあるセクションのみ入力を行えるようにしたのがセクション選択画面である。また，このセクション自身が入力条件によっては不要なケースもあり，その際はセクション選択画面に表示されないようにした。

Fig.6.2にＳＲＡＣの例を示した。12に分割されたセクションのうち，ＧＥＮＥＲＡＬ－ＣＯＮＴＲＯＬによって指定されなかった，つまり，不必要なセクションに関しては表示されないようになっている。そして表示されているセクションのうち，選択されたセクションに関してはーＷＲＩＴＥ，選択されなかったセクションに関してはーＮＯＮＥの表示がされており，利用者は作業状況を一目で把握できるようになっている。

6.1 セクション選択の定義

Fig.6.2で示したメニュー画面はシステムによって自動的に作成されるものであり，利用者は特別に画面を意識する必要はない。それらの定義は，ＪＤＩＳＳＰＨ2の入力データの定義文と制御文で行う。

まず定義文では，使用するセクション選択機能の数，セクション選択機能呼び出し名，セクション選択定義名等を定義し，制御文では実際にセクション選択呼び出し機能名を名前に持つサブルーチンで各々のセクションの呼び出しを定義する。

A) セクション選択機能の数

セクション選択機能は，プログラム中で複数使用することができる。そこでその異なる定義を持つセクションの数を指定する。

B) セクション選択機能呼び出し名

ここで指定された名前を，制御文中の任意のセクション選択機能を使用する場所でコールすることによって，その機能が作動する。

C) セクション選択機能定義名

ここで指定された名前を持つサブルーチンを，制御文中に定義して，セクション表示の判定をそれまでの入力値によって行う。本サブルーチンの引数はセクション選択番号とリターンコードの2つである。

そしてまず，データより表示するセクションを選択し，その結果をＪＤＩＳＳＵＴＹサブルーチン群の￥ＢＬＫＳの引数として，それぞれセットする。(￥ＢＬＫＳの引数に関しては後述)そして￥ＢＬＫＳをＣＡＬＬするとFig.6.2のような画面が表示される。利用者がセクションを選択すると，その値が￥ＢＬＫＳのＩＮＯに格納されて戻ってくるので，その値を上位サブルーチン(システムで自動作成されるセクション選択機能呼び出し名を持つサブルーチン)に

引き渡し，つまりＲＥＴＵＲＮして，処理を終了する。その後の制御は上位ルーチンが行う。

￥ＢＬＫＳの呼び出し形式は，以下の通りである。

￥ＢＬＫＳ（ＩＮＯ，Ｎ，ＮＡＭ，ＣＯＭ，ＩＵ，ＩＲＴ）

ＩＮＯ	： 選択されたセクションナンバー	〔Ｉ＊４：出力〕
Ｎ	： 表示するセクションの数	〔Ｉ＊４：入力〕
ＮＡＭ（全セクション数）	： セクション名	〔Ｃ＊６：入力〕
ＣＯＭ（全セクション数）	： コメント	〔Ｃ＊40：入力〕
ＩＵ（Ｎ）	： セクション制御情報	〔Ｉ＊４：入力〕
ＩＲＴ	： リターンコード	〔Ｉ＊４：出力〕

以上の様子をFig.6.3～6.4で示す。

6.1.1　セクション選択機能定義サブルーチンの定義

入力データの値によって実行するセクション名を決定するサブルーチン。

a）サブルーチン名

制御文で指定した名前で，選択セクションナンバーとリターンコードを引数に持つ。

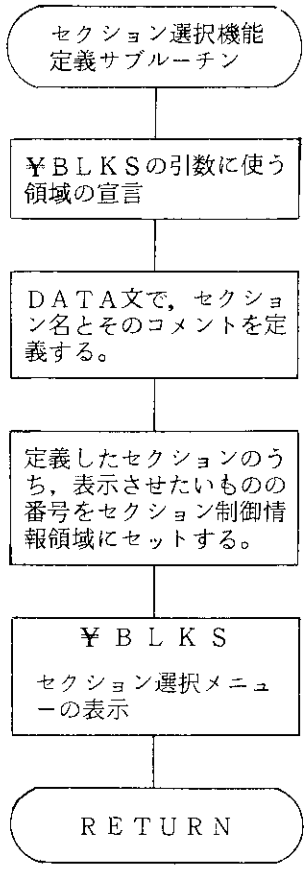
b）ＩＮＣＬＵＤＥ

制御文で指定したＩＮＣＬＵＤＥを指定。

c）配　　列

文字型配列に，セクション名，そのコメントを定義。

整数型配列に，使用するセクションナンバーを格納する。



6.2　ＰＦ２キィーの処理

ＪＤＩＳＳＰＨ２で作成されるプログラムではＰＦ２キィーを押すと，前のメニュー画面に戻るようになっている。これは単純にＣＡＬＬの連続であれば，比較的簡単な処理ですむが，ＩＦ文が存在する場合にはその条件を考慮して飛び先を決定しなくてはならないので（例えばＩＦ文の外から，中へ戻ることはできない等），各メニューの呼出し状況を￥ＣＬＦＬＧに格納して，飛び先を決定するようにプログラムを作成するようにした。

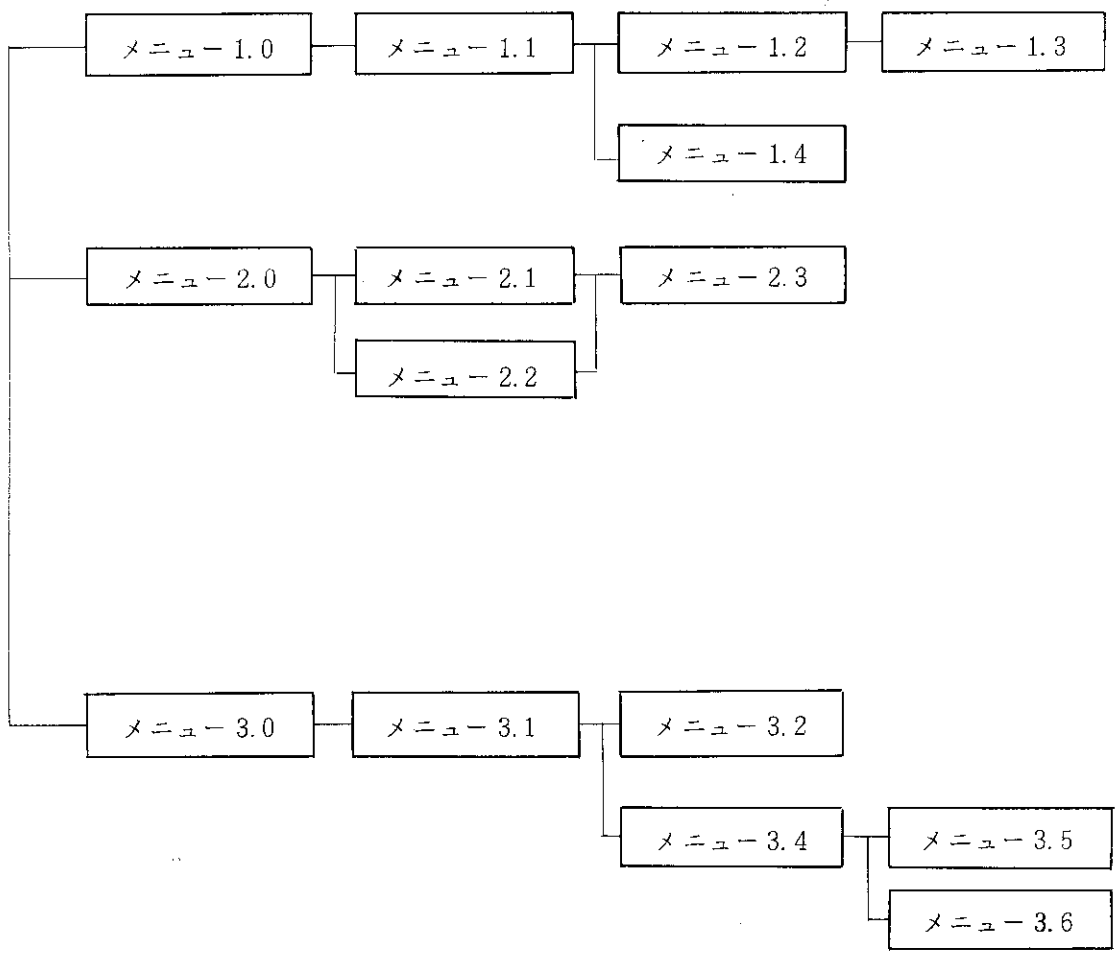


Fig. 6.1　Section (= Tree Structure of Menus)

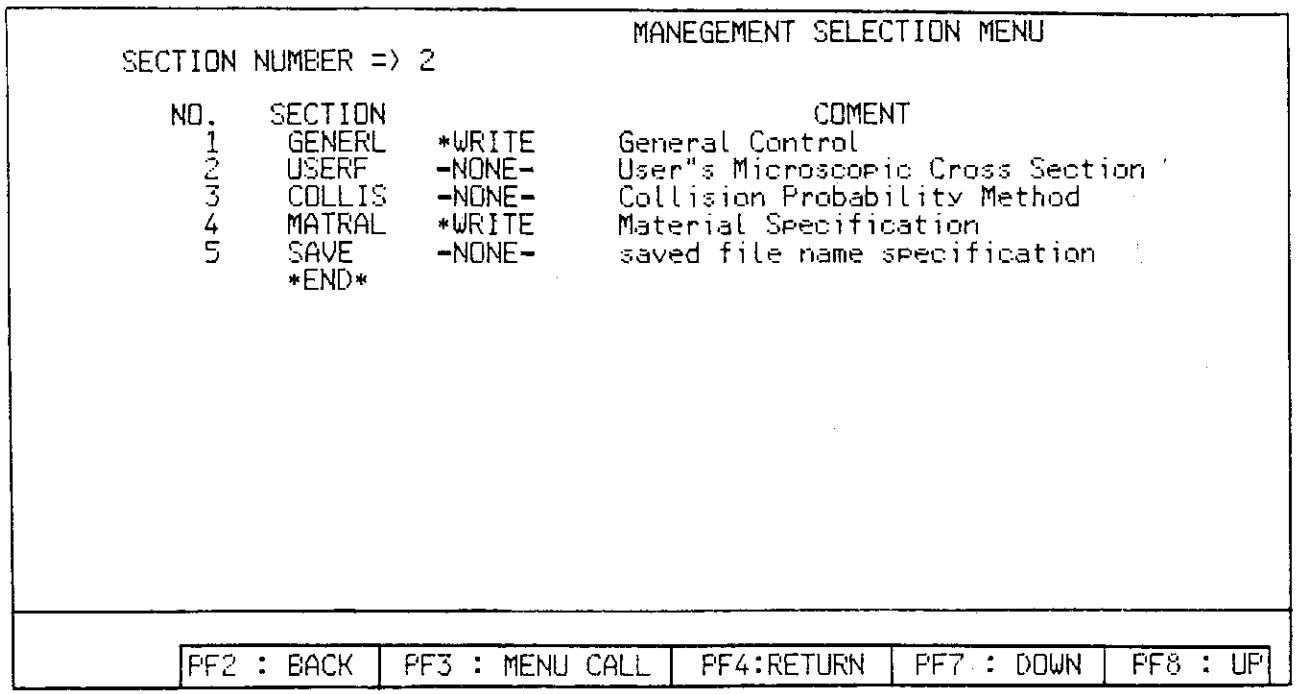


Fig. 6.2 Example of Section Selection Menu of SRAC Code

セクション呼び出しの構造

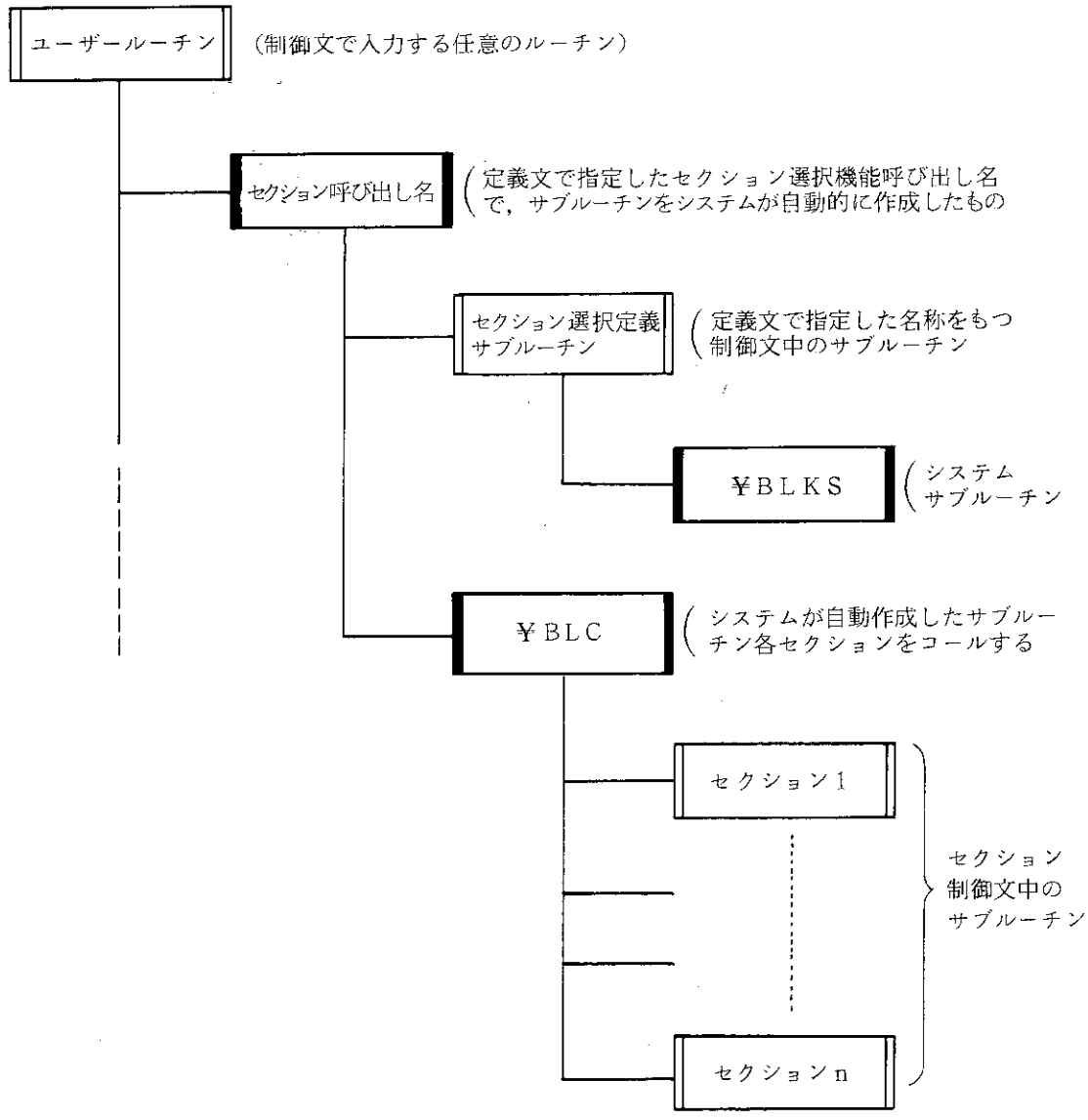


Fig. 6.3 Program Flow for the Description of Section Selection

定義文

```
 &FILES
   MNUFIL      = 'J9055.SRACTRNS.MENUS',
   OUTFIL      = 'J7858.SRACJDIS.FORT77(SRACMAIN)',
   OUTMNU      = 'J7858.SRACJDIS.FORT77(SRACMENU)',
   INCFIL      = 'J7858.SRACINC.FORT77',
   IOPT(1)     = 1
   IOPT(2)     = 0
   MENUE(1)    = '*'
   INCMEM      = '¥INC¥',
 &END
*
******  SRAC INPUT FILE NAME
*
1
J7858.SRINT.DATA(SRAC00)
*
******  SECTION DEFINED
*    機能呼出し名  各組で異なる名前をつける
1
BLOCK       BLK001      13
GENERL      定義名
USERF
COLLIS
ANISN
TWOTRN
TUD                          セクションの数
CITATN
MATRAL
REACT
BURNUP
MCROSX
PEACO
¥END
```

組の数

注)

1) BLOCKは，機能呼出し名である。

2) BLK001は，セクション選択定義名である。

3) GENERL～PEACOは制御文でのサブルーチン名である。

4) ¥ENDは，中間ファイルデータを格納するシステムサブルーチン名である。

Fig. 6.4 Control Description for Section Selection

```
C
C***** MAIN     1989 09/19
C
*INCLUDE ¥INC¥,INSOURCE
 3000 CONTINUE
C--------
      CALL ¥READX(IRT)
C--------
     CALL BLOCK (IRT)          ← 呼び出し名をＣＡＬＬするとセクション選択機能が働く。
C--------                        （この名のサブルーチンは自動的に作られる）
      IF ( IRT .EQ. 2  .OR. IRT .EQ. 14 ) GO TO 3000
      END
C **********                                    ↓
C * BLK001 *                                    セクション定義サブルーチン
C **********
C
C---------- SUBROUTINE BLK001
      SUBROUTINE BLK001(INO,¥IRT )
C
*INCLUDE ¥INC¥
      CHARACTER *  6 BNAM(13)
セクション名
      DATA  (BNAM(I),I=01,03) /'GENERL' ,'USERF ' ,'COLLIS' /
      DATA  (BNAM(I),I=04,06) /'ANISN'  ,'TWOTRN' ,'TUD'    /
      DATA  (BNAM(I),I=07,09) /'CITATN' ,'MATRAL' ,'REACT'  /
      DATA  (BNAM(I),I=10,13) /'BURNUP' ,'MCROSS' ,'PEACO' ,'SAVE' /
      CHARACTER * 40 BCOM(13)
そのコメント
      DATA  BCOM(01) / 'General Control                                ' /
      DATA  BCOM(02) / 'User"s Microscopic Cross Section               ' /
      DATA  BCOM(03) / 'Collision Probability Method                   ' /
      DATA  BCOM(04) / 'ANISN    : One Dimensional SN Transport        ' /
      DATA  BCOM(05) / 'TWOTRAN  : Two Dimensional SN Transport        ' /
      DATA  BCOM(06) / 'TUD      : One Dimensional Diffusion           ' /
      DATA  BCOM(07) / 'CITATION : Two Dimensional Diffusion           ' /
      DATA  BCOM(08) / 'Material Specification                         ' /
      DATA  BCOM(09) / 'Reaction Rate Calculation                      ' /
      DATA  BCOM(10) / 'Cell Burn-up Calculation                       ' /
      DATA  BCOM(11) / 'MCROSS                                         ' /
      DATA  BCOM(12) / 'PEACO                                          ' /
      DATA  BCOM(13) / 'saved file name specification                  ' /
      DIMENSION IU (13)
C
      N     = 0
C-----  GENERAL CONTROL
C
C
      IF  (     (  IABS(IC2) .EQ. 1  )  .OR.
     #          (  IC12      .EQ. 1  )  .OR.
     #          (  IC3       .GT. 0  )    )  THEN
        IC1 = 1
      ELSE
        IC1 = 0
      ENDIF
      IF  (  IC2  .NE.  0  )  THEN
        IC6 = 1
        IC7 = 4
      ELSE
        IC6 = 0
        IC7 = 0
```

Fig. 6.5a Coding Example for Section Selection in SRAC Code

```
        ENDIF
        IF     (  NERT .EQ.  1  )  THEN
          IC15 = -1
        ELSEIF(  NERT .GE.  2  )  THEN
          IC15 =  1
        ELSE
          IC15 = 0
        ENDIF
C-----
        N     = N + 1
        IU(N) = 1
C-----  USER MICROSCOPIC CROSS SECTION
        IF ( (FLAG1.EQ.'Y') .AND. (FLAG11.EQ.'N') ) THEN
          N     = N + 1
          IU(N) = 2        } 表示するセクションナンバーを格納する
        ENDIF
C-----  COLLISION PROBALLITY CROSS SECTION
        IF ( (IC11.EQ.0) .AND. (IC1.EQ.1) ) THEN
          N     = N + 1
          IU(N) = 3
        ENDIF
C-----  ANISN
        IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.2).OR.(ABS(IC12).EQ.2)) ) THEN
          N     = N + 1
          IU(N) = 4
        ENDIF
C-----  TWOTRAN
        IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.3).OR.(ABS(IC12).EQ.3)) ) THEN
          N     = N + 1
          IU(N) = 5
        ENDIF
C-----  TUD
        IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.4).OR.(ABS(IC12).EQ.4)) ) THEN
          N     = N + 1
          IU(N) = 6
        ENDIF
C-----  CITATION
        IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.5).OR.(ABS(IC12).EQ.5)) ) THEN
          N     = N + 1
          IU(N) = 7
        ENDIF
C------  MATELIAL SPACIFICATION
        N     = N + 1
        IU(N) = 8
C------  reaction rate
        IF ( IC18 .EQ. 1 ) THEN
          N     = N + 1
          IU(N) = 9
        ENDIF
C------  CELL BURNUP
        IF (  IC20 .EQ. 1 ) THEN
          N     = N + 1
          IU(N) =  10
        ENDIF
C------  MCROSS
C       IF (  IC8  .EQ. 1 ) THEN
C         N     = N + 1
C         IU(N) = 11
```

Fig. 6.5a (Continued)

```
C      ENDIF
C------  PEACO
       IF (  IC5  .EQ. 1 ) THEN
         N      = N + 1
         IU(N) = 12
       ENDIF
C------  DATA SAVE
       N      = N + 1
       IU(N) = 13
C------------- BLOCK MANU
       CALL ¥BLKS(INO , N ,BNAM ,BCOM ,IU , ¥IRT )
C
       RETURN
       END
```

```
C **********
C * TUD    *
C **********
       SUBROUTINE TUD(¥IRT )
*INCLUDE ¥INC¥
C
       CALL SO6NO1
       CALL SO6NO2
       CALL SO6NO3
       IF (NXRO .NE. 0) CALL SO6NOO
       CALL SO6NO4
       CALL SO6NO5
C
       RETURN
       END
```

選択すると実行されるセクション

Fig. 6.5a (Continued)

自動作成されたＢＬＯＣＫと¥ＢＬＣ

下記の２つのサブルーチンが入力データより自動的に作られる

```
      SUBROUTINE BLOCK ( ¥IRT                                          )
C*********************************************************************
C*
C*   BLOCK  :  BLOCK CONTROL
C*
C*    THIS SUBROIUTINE WAS CREATED BY JDISS SYSTEM(1989 JAERI).
C*                                  CREATE DATE : 89-10-23
C*
C*   <ARGUMENT>
C*
C*   IRT    : RETURN CODE
C*
C*********************************************************************
C
      INTEGER * 4 ¥IRT ,¥IB ,¥INO
      ¥INO = 0
      ¥IB =  1
      DO 4000 WHILE(.TRUE.)
        ¥IRT = 0
        CALL BLK001( ¥INO ,¥IRT )
        IF ( (¥IRT.EQ.4) .OR. (¥IRT.EQ.16)  ) GO TO 9999
        IF ( (¥IRT.EQ.2) .OR. (¥IRT.EQ.14)  ) GO TO 9999
        CALL ¥BLC   ( ¥IB ,¥INO ,¥IRT )
 4000 CONTINUE
C
C>>>>>  RETURN END
C
 9999 CONTINUE
      RETURN
      END

C   ***********
C   * ¥BLC    *
C   ***********
      SUBROUTINE ¥BLC  ( IB       ,ISUB   ,IRT                         )
C*********************************************************************
C*
C*   ¥BLC   :  JUNP TO SUBROUTINE BY INPUT
C*
C*    THIS SUBROIUTINE WAS CREATED BY JDISS SYSTEM(1989 NEDAC).
C*                                  CREATE DATE : 89-09-21
C*
C*   <ARGUMENT>
C*
C*   IB     : SELECT BLOCK NUMBER
C*   ISUB   : SELECT SUBROUTINE NUMBER
C*   IRT    : RETURN CODE
C*
C*********************************************************************
C
C
C>>>>>  BLOCK NAME IS BLOCK
C
      IF     ( IB .EQ.   1 )  THEN
        IF ( ISUB .EQ.   1 ) CALL GENERL( IRT )
        IF ( ISUB .EQ.   2 ) CALL USERF ( IRT )
        IF ( ISUB .EQ.   3 ) CALL COLLIS( IRT )
        IF ( ISUB .EQ.   4 ) CALL ANISN ( IRT )
        IF ( ISUB .EQ.   5 ) CALL TWOTNN( IRT )
        IF ( ISUB .EQ.   6 ) CALL TUD   ( IRT )
        IF ( ISUB .EQ.   7 ) CALL CITATN( IRT )
        IF ( ISUB .EQ.   8 ) CALL MATRAL( IRT )
        IF ( ISUB .EQ.   9 ) CALL REACT ( IRT )
        IF ( ISUB .EQ.  10 ) CALL BURNUP( IRT )
        IF ( ISUB .EQ.  11 ) CALL MCROSX( IRT )
        IF ( ISUB .EQ.  12 ) CALL PEACO ( IRT )
        IF ( ISUB .EQ.  13 ) CALL ¥END  ( IRT )
      ELSE
      WRITE(*,*) ' *** ERROR AT BLC *** INVALID BLOCK NUMBER =>', IB
      ENDIF
C
C>>>>>  RETURN END
C
      RETURN
      END
```

Fig. 6.5b Obtained Subroutines BLOCK and ¥BLK

ＰＦ２キィーの処理機能の追加

～入力定義文～

```
C **********
C * GENERL *
C **********
      SUBROUTINE   GENERL( ¥IRT )
C  GENERAL CONTROL
*INCLUDE  ¥INC¥
      CHARACTER * 80 CEM ,CEM2 ,CEMX
      DATA  CEM  /' INPUT "Y" OR "N" ' /
      DATA  CEM2 /' INPUT "E" OR "F" ' /
C
C       M E N U    P A T H
C
      CALL  S01N00
 3001 CALL  S01N01
C
C>>>>> FIRST CASE OR NOT .
C
      IF    ( FLAG1 .EQ. 'Y' )  THEN
 3002     CALL  S01N02
C--------------- ADD NEDAC 1989 08/08
          IF ( INDEX('YN',FLAG11) .EQ. 0 )  THEN
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO 3002
          ENDIF
C------ *** DEFINE IC11 ****
          IC11 = 0
C-----------------------------------
      ELSEIF( FLAG1 .EQ. 'N' )  THEN
 3003     CALL  S01N03
          IF    ( FLAG12 .EQ. 'Y' )  THEN
              IC11 = 1
          ELSEIF( FLAG12 .EQ. 'N' )  THEN
              IC11 = 0
          ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO  3003
          ENDIF
      ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
          GO TO  3001
      ENDIF
```

Fig. 6.6a Process for PF2-key

加工されて得られた結果

```
C **********
C * GENERL *
C **********
      SUBROUTINE   GENERL( ¥IRT )
C  GENERAL CONTROL
*INCLUDE  ¥INC¥
      CHARACTER * 80 CEM ,CEM2 ,CEMX
      DATA  CEM  /' INPUT "Y" OR "N" ' /
      DATA  CEM2 /' INPUT "E" OR "F" ' /
C
C      M E N U    P A T H
C
C-----  ADD JDISS SYSYTEM ----------------------------------------
*INCLUDE ¥IPFDAT
      INTEGER * 4   ¥IRT
C
      IF ( (¥IRT.EQ.2).OR.(¥IRT.EQ.14) ) GO TO 7999
      IF ( (¥IRT.EQ.4).OR.(¥IRT.EQ.16) ) GO TO 7999
C-----------------------
C----------- ADD JDISS SYSTEM ----
 7000 CONTINUE
 3001 CALL  S01N01 ( ¥IRT )
      IF ( (¥IRT.EQ.2) .OR. (¥IRT.EQ.14) )  THEN
        GO TO 7999
      ENDIF
      ¥CLFLG(0001) = 1
      IF ( (¥IRT.EQ.4) .OR. (¥IRT.EQ.16) )  GO TO 7999
C-----------
C
C>>>>> FIRST CASE OR NOT .
C
C---- < JDISS >
 7010 CONTINUE
      IF    ( FLAG1 .EQ. 'Y' )  THEN
C----------- ADD JDISS SYSTEM ----
 3002     CALL  S01N02 ( ¥IRT )
          IF ( (¥IRT.EQ.2) .OR. (¥IRT.EQ.14) )  THEN
            IF ( ¥CLFLG(0001) .EQ. 1 ) GO TO 7000
            GO TO 7999
          ENDIF
          ¥CLFLG(0002) = 1
          IF ( (¥IRT.EQ.4) .OR. (¥IRT.EQ.16) )  GO TO 7999
C-----------
C--------------- ADD NEDAC 1989 08/08
C---- < JDISS >
 7020 CONTINUE
          IF ( INDEX('YN',FLAG11) .EQ. 0 )  THEN
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO 3002
          ENDIF
C------ *** DEFINE IC11 ****
          IC11 = 0
C-----------------------------------
      ELSEIF( FLAG1 .EQ. 'N' )  THEN
C----------- ADD JDISS SYSTEM ----
 3003     CALL  S01N03 ( ¥IRT )
          IF ( (¥IRT.EQ.2) .OR. (¥IRT.EQ.14) )  THEN
            IF ( ¥CLFLG(0002) .EQ. 1 ) GO TO 7010
            IF ( ¥CLFLG(0001) .EQ. 1 ) GO TO 7000
            GO TO 7999
          ENDIF
          ¥CLFLG(0003) = 1
          IF ( (¥IRT.EQ.4) .OR. (¥IRT.EQ.16) )  GO TO 7999
C-----------
C---- < JDISS >
 7030 CONTINUE
          IF    ( FLAG12 .EQ. 'Y' )  THEN
              IC11 = 1
          ELSEIF( FLAG12 .EQ. 'N' )  THEN
              IC11 = 0
          ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO  3003
          ENDIF
      ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
          GO TO  3001
      ENDIF
```

システムが自動的に付加した部分

←メニューが呼ばれると１を入れている

前のメニューが呼ばれていたらもどる

Fig. 6.6b Obtained Subroutine

7. ＪＤＩＳＳＰＨ１プログラムの実行

7.1 コマンドによるＪＤＩＳＳＰＨ１の起動

Fig.7.1に処理の流れを示した。使用するファイルと機番の関係は以下の通りである。

ＦＴ７７Ｆ００１ ： デフォルトのファイル名を入力

ＦＴ１０Ｆ００１ ： メニューファイル名入力

以上のファイルを結合して実行するコマンドリストをFig.7.2に示す。

7.2 オペレーション

Fig.7.1に示すとおり処理の流れとしては，新規にメニュー画面を作成する場合と，以前に作成した画面を参照・修正する場合の大きく２つに分かれる。そして，各々の画面は，一般画面，テーブル画面－１，テーブル画面－２の３つのタイプのいずれかであり，新規作成の場合はそのタイプを直接指定し，既成のメニュー画面を利用する場合はもとのタイプが自動的に選択される。

実行はすべてディスプレー上で会話的に行われる。基本的にＰＦキィーを押すことで次の処理へ移り，誤った指定に対してはメッセージを出力するので訂正を行う。

Ⅰ） ＪＤＩＳＳＰＨ１の起動

Fig.7.2で示したコマンドを，ディスプレー上で実行することによってＪＤＩＳＳＰＨ１が起動する。

Ⅱ） 初期画面（ Fig.7.3 ）

まず初期画面が表示される。

Ａには，一連の処理で作成するメニュー画面を格納するファイル名を指定する（必須）。また既存のメニュー画面を参照するのにも使用する。Ｂには，既成のメニュー画面を参照するのに使用する参照専用のファイル名を指定する，このファイルに書き込むことはできない。参照ファイルを使用しない場合は指定しなくてもよい。

ＰＦ３キィーを押すとⅢ）の画面に移る。その際，ＡとＢのファイルのチェックが行われ，もしそれらのファイルが存在しなかったり，ファイル名として正しく指定されていない場合にはエラーメッセージが出力され，正しい名前が入力されるまで再入力を促す。

ＰＦ４キィーを押すとプログラムは終了する。

Ⅲ） メニューデザイン初期画面（ Fig.7.4 ａ）

後述の画面が表示される。この画面は一般画面のデザイン入力画面であるが，メニュー名及

びＰＦキィーによっては他のデザイン画面となる。

Ａの部分は，メニュー名を指定する部分である。メニュー名は，特殊文字を除いた英字で始まる６文字以内の英数字ないし，@ＴＢＬ１，@ＴＢＬ２が許される。許されないメニュー名を指定すると，正しいメニュー名を入力するまで再入力を促す。それらの指定の方法によって以下のような意味を持つ。格納メニューファイル中にはこのメニュー名がメンバー名となって格納される。

i) @ＴＢＬ１と入力したのちＰＦ１キィーを押すと，テーブル画面－１デザイン入力画面が表示される。→ Ⅵ)に行く。

ii) @ＴＢＬ２と入力したのちＰＦ１キィーを押すと，テーブル画面－２デザイン入力画面が表示される。→ Ⅷ) に行く。

iii) @ＴＢＬ１と@ＴＢＬ２以外のメニュー名を指定したのち，ＰＦ１キィーを押すとⅡ)で指定したファイルより同じメニュー名の画面が表示される。ファイルの検索順序はＡ，Ｂの順であり，両方のファイルになければその旨出力する。検索されたメニュー画面のタイプが，そのまま引き継がれる。

Ⅳ) i)～iii)以外のメンバー名の入力では，画面は一般画面とみなされる。→Ⅳ) へ行く。

ＰＦ２キィーを押すと，表示されている画面は総てクリアーされて，本画面の初期状態に戻る。間違って入力した場合等に使用する。

ＰＦ４キィーを押すとプログラムは終了する。

Ⅳ) 一般画面のデザイン（ Fig.7.4 a ）

一般画面のデザインを行う画面である。

１行目

Ａ：格納するメニュー名を指定する。このとき同じ名前のメニュー画面が格納ファイルに存在すればそれを置き換える。なければ新たにメニューをここで指定したメニュー名と同じメンバー名で格納する。

Ｂ：メニューのコメントである。

２行目　自由設定領域。

ここでの指定方法は，テーブル画面－１やテーブル画面－２の２～６行目の自由設定領域と共通したルールである。

画面には，(1)原子力コードの利用者が会話型入力データ作成システムを利用して値を入力したりシステムより何らかの値を出力したりする入出力領域と，(2)入力すべきデータの説明などが画面に表示されている表示データ，を記述する。

(1) 入出力領域

入出力領域はプログラムと画面とのデータの受け渡しを行うものであり，記述方法は，@シンボル名@のように@と@によって挟むことで定義する。シンボル名は，この画面における位置を認識するためのものであり実際の変数名と一致させる必要はない。また同一画面において同じシンボル名を他の領域のシンボル名として使用したり，２行にわたって入出力領域を指定することはできない。その場合はエラーメッセージを出力するので変更する。また

シンボル名は，特殊文字を除いた英字で始まる６文字以内の英数字で指定する。

この部分が入出力領域か，出力専用域かの指定は →V)で行うが，実際に会話型入力データ作成システムによって表示される際には，@は消え，アトリビュート記号になる。入出力領域の場合にはアンダーラインが引かれディスプレーから入力可能となるが，出力専用領域の場合は特別アンダーラインは引かれず，ディスプレーから入力できず出力のみ行われる。

(2) 表示データ

表示データは，画面に打ち込んだ文字がそのままディスプレーより表示されるものである。入力データの説明等を記述する文字として，”！｜＆＄￣＿＃￥¬@を除いた英数字が使用できる。

23行目　メッセージ行

各種メッセージを出力する領域である。利用者はここのメッセージに従って入力作業を行う。また，会話型入力データ作成システムにおいても本行はメッセージ出力行となる。

24行目　ＰＦキィー記述欄

使用できるＰＦキィーの説明が書いてある。ただし，ＰＦｎのｎは12を加えたものと同じと見なしてもよい。例えばＰＦ１キィーとＰＦ13キィーの持つ意味は同じである。

〔ＰＦキィーの説明〕

ＰＦ１キィー　：　指定されたメニューをファイルから入力して表示する。

ＰＦ２キィー　：　画面をクリアーする。

ＰＦ３キィー　：　次の変数定義画面へ行き，処理を継続する。

このキィーを押すと，同じ名前のメニュー名があると

ＭＥＭＢＥＲ　ＯＶＥＲ　ＷＲＩＴＥ　ＯＫ？

とメッセージが出力されるので，置き換えて処理を継続する場合には再度ＰＦ３キィーを押すことによって，次の画面　→V)へ行く。

同様に，同じ名前のメニュー名がない場合には，

ＮＥＷ　ＭＥＭＢＥＲ　ＣＲＥＡＴＥ　ＯＫ？

とメッセージが出力されるので，メニュー名を確認して再度ＰＦ３キィーを押すと次の画面　→V)へ行く。

いずれの場合もＰＦ３キィーを続けて押すことによって初めて次の処理を行うことができる。

間違った場合には，ＥＮＴＥＲキィーを押すと再度入力モードになるので，訂正を行う。

ＰＦ４キィー　：　プログラムを終了する。

V)　変数定義画面（Fig.7.4 b）

Ⅳ)で定義した，シンボル名と会話型入力データ作成システムで使用する変数を結合する。

Ａ：メニュー名

Ｂ：コメント

Ｃ：Ⅳ)で定義されたシンボル名の総数

Ｄ：シンボルナンバー　Ⅳ)で定義したシンボルは，左上より順番に番号がつけられる。

Ｅ：シンボル名

Ａ～Ｅまでは自動的に表示される。

Ｆ：シンボル名に対応する変数名か@を指定する。この変数名は，会話型入力データ作成システムで使用する変数名である。

@の時は，Ｍで指定した値を常に出力したいときに指定する。そのとき，ＨからＫの部分に指定することはできない。そして，Ｌに“Ｏ”を指定して，Ｍに出力したい値を指定する。

Ｇ：タイプ名

Ｆのタイプを，Ｉ（整数タイプ），Ｒ（実数タイプ），Ｃ（文字タイプ）で指定する。

Ｈ～Ｋ：変数の範囲指定

ＪＤＩＳＳＰＨ２で自動的に作成される会話処理プログラムで，入力されたデータに関して範囲のチェックを行う場合に指定する。指定がなければ，会話処理プログラムでは特別に範囲のチェックは行わない。

指定の方法はＨとＩで最小値，ＪとＫで最大値を指定する。Ｉ，Ｋに値を入力し，Ｈ，Ｊには，空白ないし＝を指定する。＝のときはそれぞれ下限値，上限値を含むことを意味し（≦，≧），空白のときは含まない（＜，＞）。指定は，片方だけでもよい。

Ｌ：入出力の指定

領域の入出力の指定を行う。ＩＯでは入出力，Ｏでは出力専用領域であることを現している。Ｆが@の時は，必ずＯでなければならない。

Ｍ：初期値

初期値を指定する。このとき，Ｆは必ず@でなければならない。

変数定義画面には，一度にＤ～Ｍまでのデータは15行しか表示されないので，それ以上のデータがある場合にはＰＦキィーで上下にスクロールする。

〔ＰＦキィーの説明〕

ＰＦ３キィー　：　データを格納ファイルに格納する。
１回押すと，確認メッセージが出るので，間違いのないことを確認したのち，もう一度ＰＦ３キィーを押すと，ファイルに出力される。
ファイルに格納されたのち，再びⅡ)に戻る。

ＰＦ５キィー　：　Ⅳ)の画面を再表示する。しかし，その画面に再入力はできない。入力するときには，再度ＰＦ５キィーを押すとⅣ)へ戻る。その他のキィーでは，この画面に戻る。

ＰＦ７キィー　：　上にスクロールする。

ＰＦ８キィー　：　下にスクロールする。

ＰＦ７・ＰＦ８キィーはシンボル数が15以上のとき有効となる。

Ⅵ)　テーブル画面－１のデザイン（Fig.7.5 a）

テーブル画面－１のデザインを行う部分である。

Ａ：メニュー名の表示

Ｂ：コメント

Ｃ：自由設定領域

一般画面Ⅳ)自由設定領域と全く同じ入力方法をとる。

定義したシンボルの中の１つはテーブルデータの入力個数を定義するものでなければならない。

Ｄ：会話型入力データ作成システムから呼び出されると，テーブルデータ入力域となるが，ここでは入力禁止領域。

Ｅ，Ｆ：Ⅳ)と同様に，それぞれメッセージ出力領域，ＰＦキィー表示領域である。

〔ＰＦキィーの説明〕

Ⅳ)と同じである。

Ⅶ) テーブル画面－１の変数定義画面（Fig.7.5 b）

Ａ：メニュー名の表示

Ｂ：コメント

Ｃ：テーブルデータに入力する数値の個数を格納しているデータが，どれであるかをＧのシンボル・ナンバーで指定する。

Ｄ：テーブルデータを格納する配列のサイズ

Ｅ：テーブルデータを格納する配列の名前

会話型入力データ作成システム中では，本領域は72×（Ｄで指定されたサイズ）の大きさを持つ１バイトの文字型配列として定義される。

Ｆ：Ⅵ)の自由設定領域で入力されたシンボルの個数

Ｇ～Ｐ：Ⅴ)と同じ

〔ＰＦキィーの説明〕

Ⅴ)と同じである。

Ⅷ) テーブル画面－２デザイン画面（Fig.7.6 a）

テーブル画面－２のデザインを行う部分である。

Ａ：メニュー名の表示

Ｂ：コメント

Ｃ：自由設定領域

一般画面Ⅳ)の自由設定領域と全く同じ入力方法をとる。

Ｄ：テーブルデータの第１レコードのフォーマットを指定する。指定方法はＣと同じであるが，そのなかの１つは必ず第２レコードの繰り返し回数を指定するものである。

Ｅ：コメント行

第２レコードに対するコメントを入力する。@シンボル@によって，入出力領域を定義することはできない。

Ｆ：第２レコード行

一般画面Ⅳ)の自由設定領域と同じ入力方法をとるが，@と@の間は，すなわち入出領域の間

隔は３カラム以上でなければならない。

会話型入力データ作成システムによって表示するときには，この第２レコードがあと12行繰り返され，合計13行にわたって表示される。

〔ＰＦキィーの説明〕

Ⅳ)と同じである。

Ⅸ) テーブル画面－２の変数定義画面（Fig.7.6）

Ａ：メニュー名の表示

Ｂ：コメント

Ｃ：第２レコードの繰り返し回数を格納している，第１レコード中の変数をＨのシンボルナンバーで指定する。

Ｄ：テーブルデータを格納する配列のサイズ

Ｅ：テーブルデータを格納する配列の名前

会話型入力データ作成システム中では，ＣＨＡＲＡＣＴＥＲ＊１で指定した配列名（72，Ｄで指定したサイズ）と定義される。

Ｆ：本テーブル画面－２の繰り返し回数を，変数名，数字，ＦＵＮＣＴＩＯＮ名で指定する。繰り返し回数を特別限定しないときには空白にしておく。しかし会話型入力データ作成システム利用者が繰り返し回数をはっきり認識して誤操作を防ぎ，明確な入力とするためにも指定する方が望ましい。

Ｇ：シンボル総数

Ｈ～Ｑ：Ⅴ)のＦ～Ｍまでと同じ。

〔ＰＦキィーの説明〕

Ⅴ)と同じである。

処理の流れ

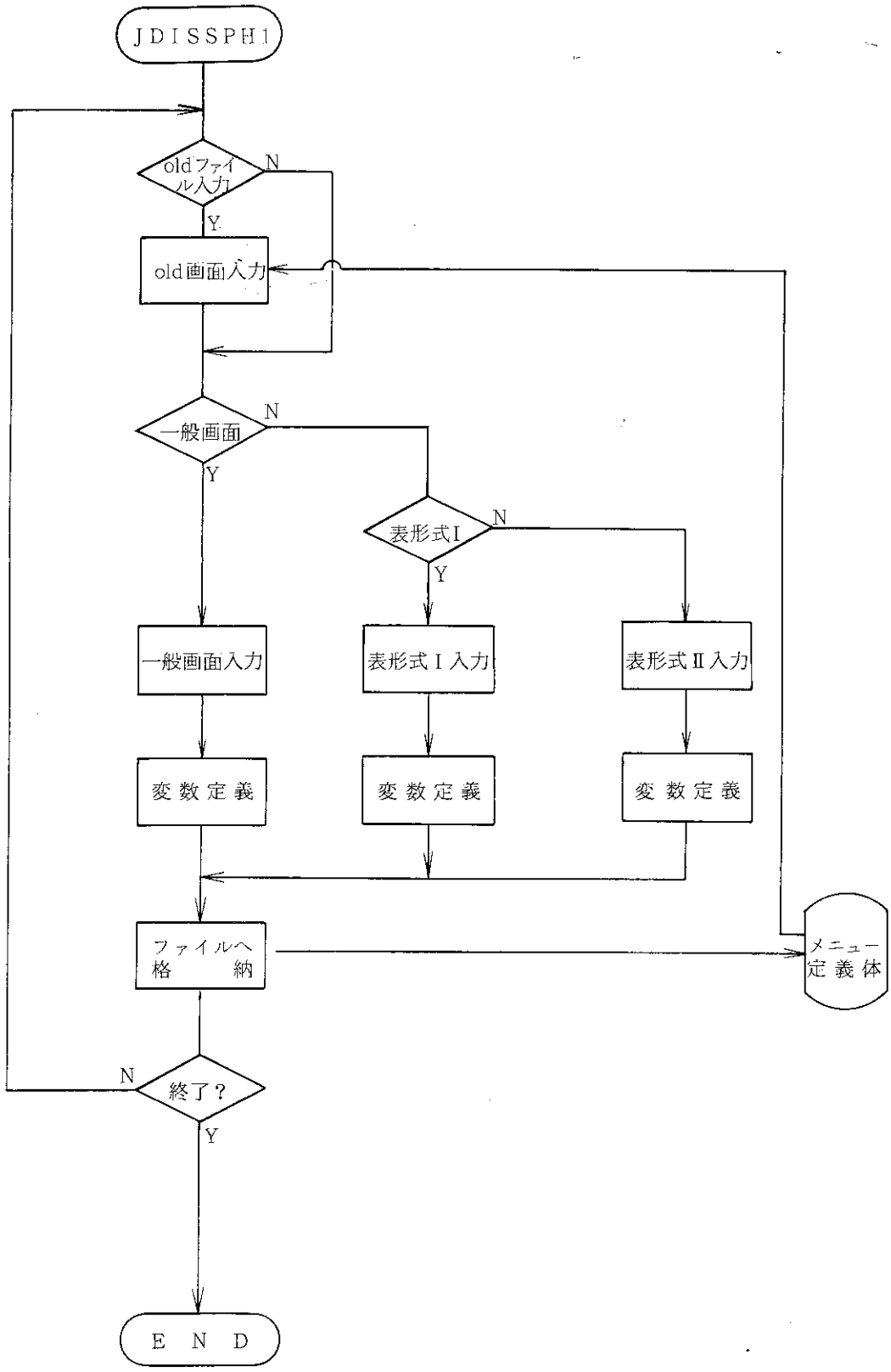


Fig. 7.1 Flow Diagram for JDISSPH1

```
PROC 1 SS  SOR(JDISSPH1) INCLUDE(INC)  OP(OPT(3))  OBJ(OBJ01)    +
       DAT() LOA(JDISS.LOAD)  LMEM(JDISSPH1) ELM(*)
IF &SS = 1 THEN GOTO IKOU
GLOBAL NN
CONTROL      MSG NOLIST NOSYMLIST
WRITE  SELECT MEMBER IS => &ELM
DEL &OBJ
ALLOC DA(&OBJ..OBJ) NEW CAT SP(100 50) T UNIT(TSSWK)          F(FF)
FREE F(FF)
FORT77 'J7858.&SOR..FORT77' ELM(&ELM) &OP INC(&INCLUDE) OBJ(&OBJ) NONUM
/*   DEBUG(ARGCHK,SUBCHK,UNDEF)
/*
  DEL @TESTRUN
  LINK (&OBJ) LOAD(@TESTRUN) +
     LIB( 'SYS2.FORTLIB'       'SYS9.IGL.LOAD'     'SYS9.GGS.LOAD'      +
          'J7858.ATJ.LOAD'     'SYS2.IPFLIB'  'SYS2.ADJUST'             +
          'J7858.TOOLLO.LOAD' 'SYS9.SSL2.LOAD' 'SYS9.SSL.LOAD' )
  MCOND &LOA
  COPY  @TESTRUN.LOAD(TEMPNAME) &LOA(&LMEM)
/*  FREE F(FT05F001)
IKOU: FREE F(FT50F001)
      FREE F(FT70F001)
      FREE F(FT31F001)
      FREE F(FT32F001)
CONTROL LIST
      FREE F(FT77F001)
       INDATA    FT77F001
DATA      PROMPT
J9055.SRACTRNS.MENUS
ENDDATA
    FREE F(MENULIB)
    FREE  DA('J7858.JDISMENU.DATA')
    ALLOC DA('J7858.JDISMENU.DATA') SHR F(MENULIB)
    CALL 'J7858.&LOA(&LMEM)' 'FLIB(DFB=YES)'
    FREE F(MENULIB)
    FREE  DA('J7858.JDISMENU.DATA')
EXIT
```

Fig. 7.2 Command Procedure of JDISSPH1

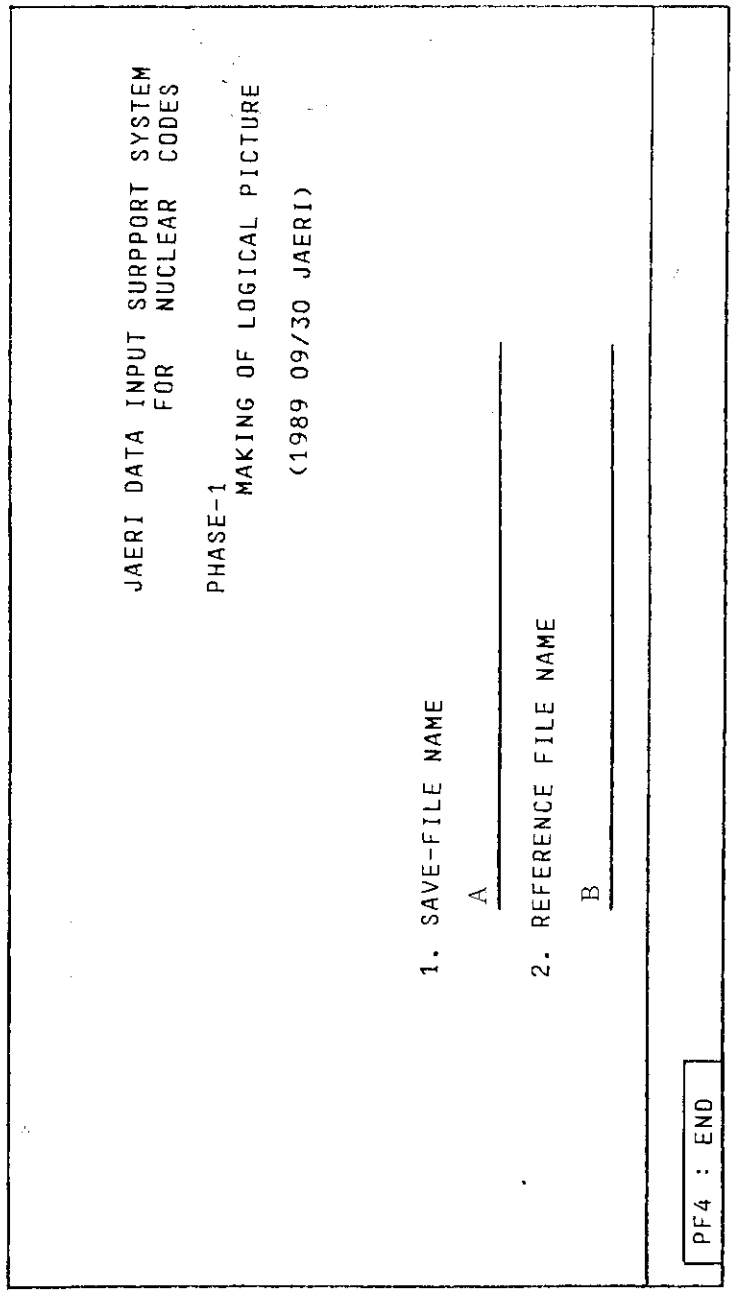


Fig. 7.3 JDISS Initial Menue

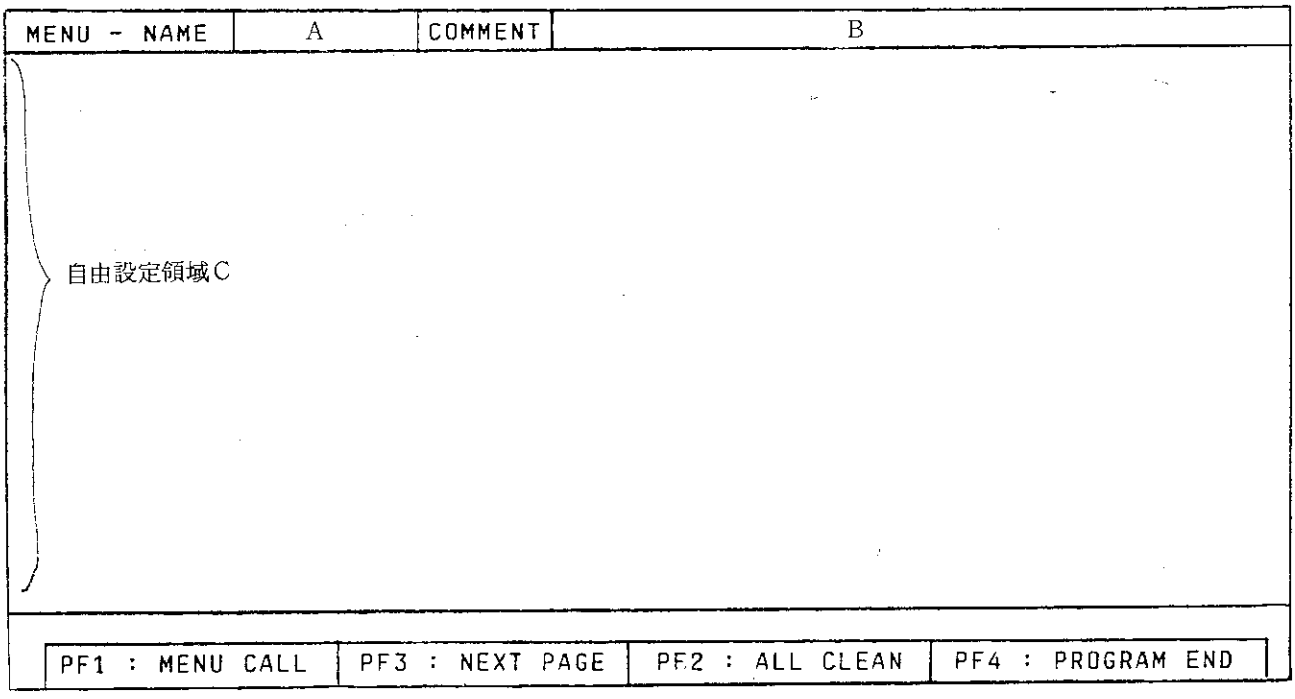


Fig. 7.4a Format of Usual Menu

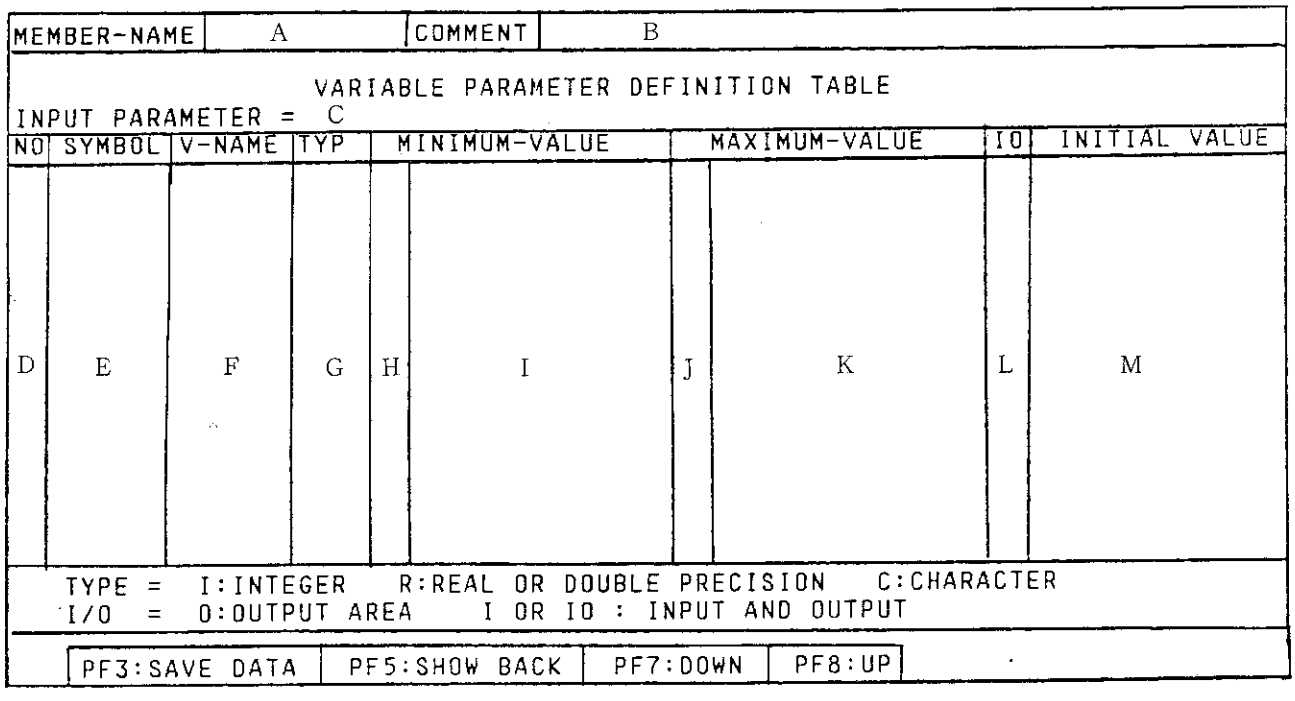


Fig. 7.4b Variable Definition Menu for Usual Menu

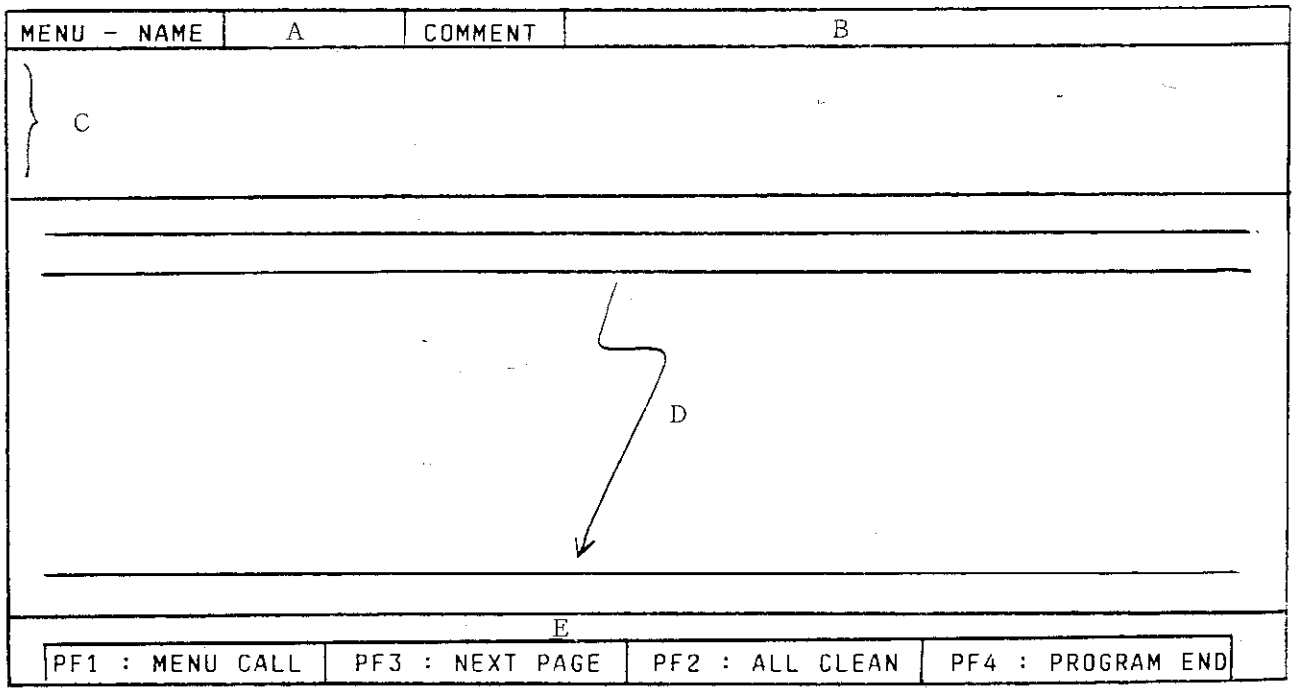


Fig. 7.5a Format of T1-Table

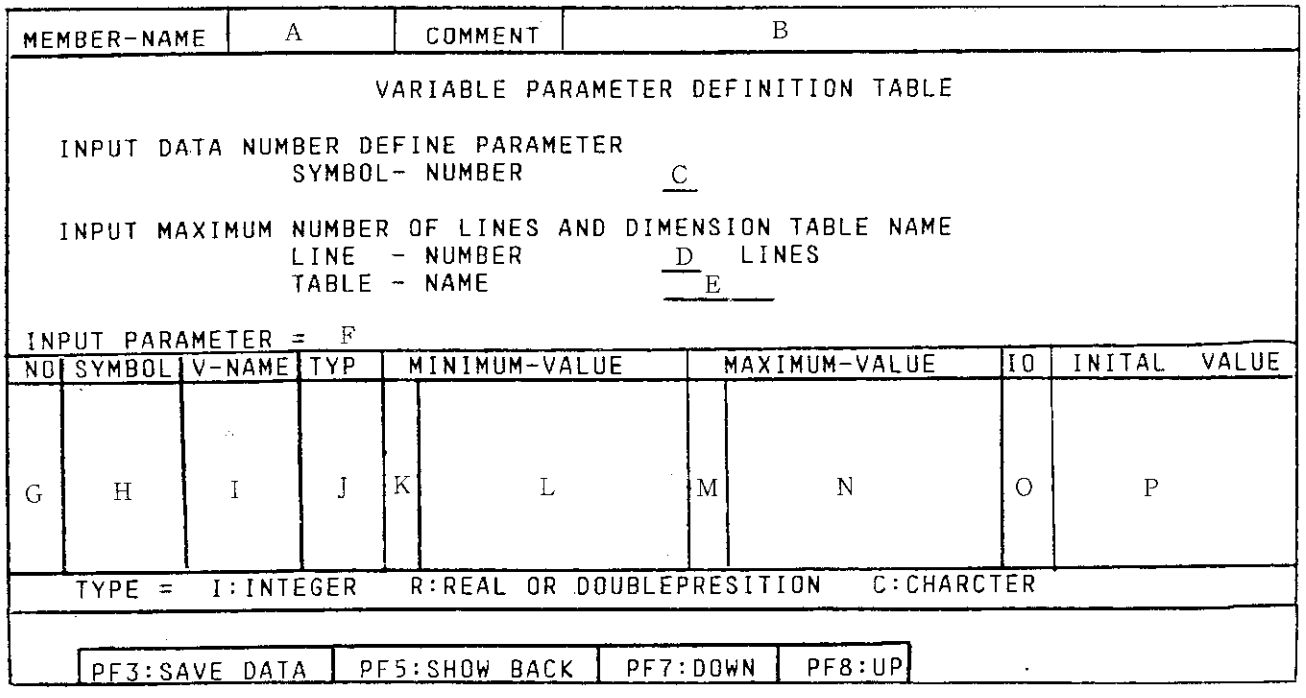


Fig. 7.5b Variable Definition Menu for T1-Table

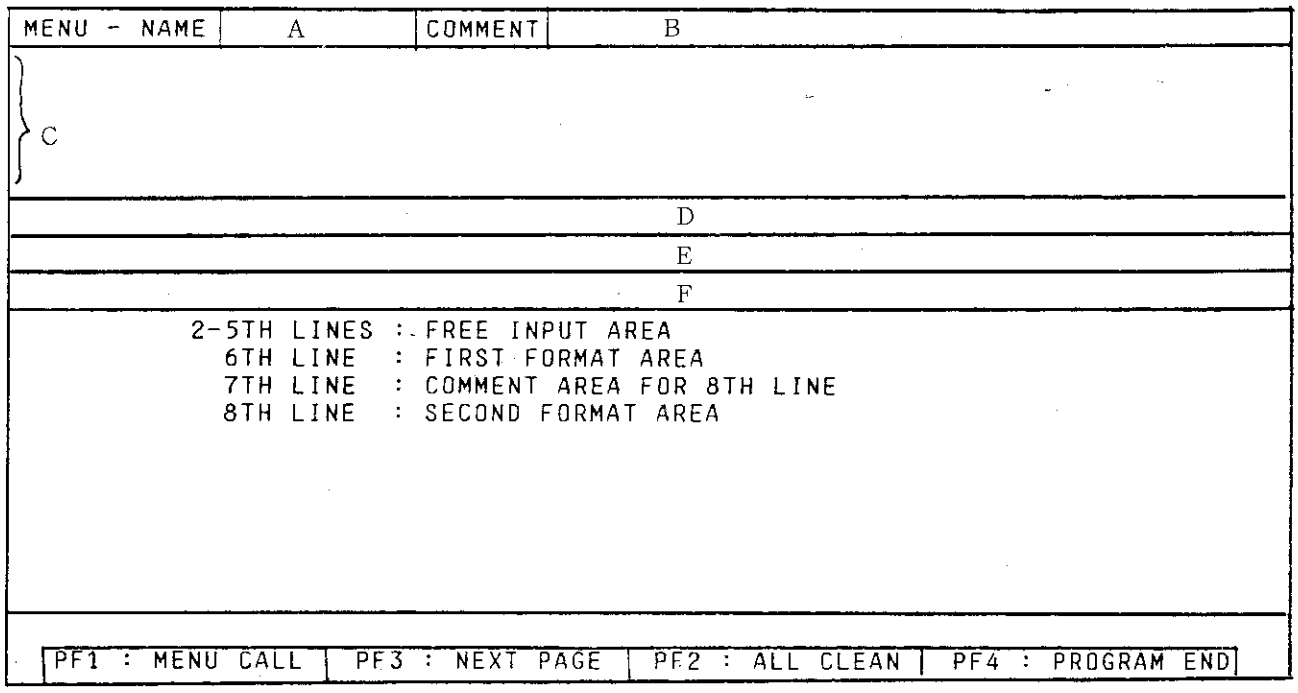


Fig. 7.6a Format of T2-Table

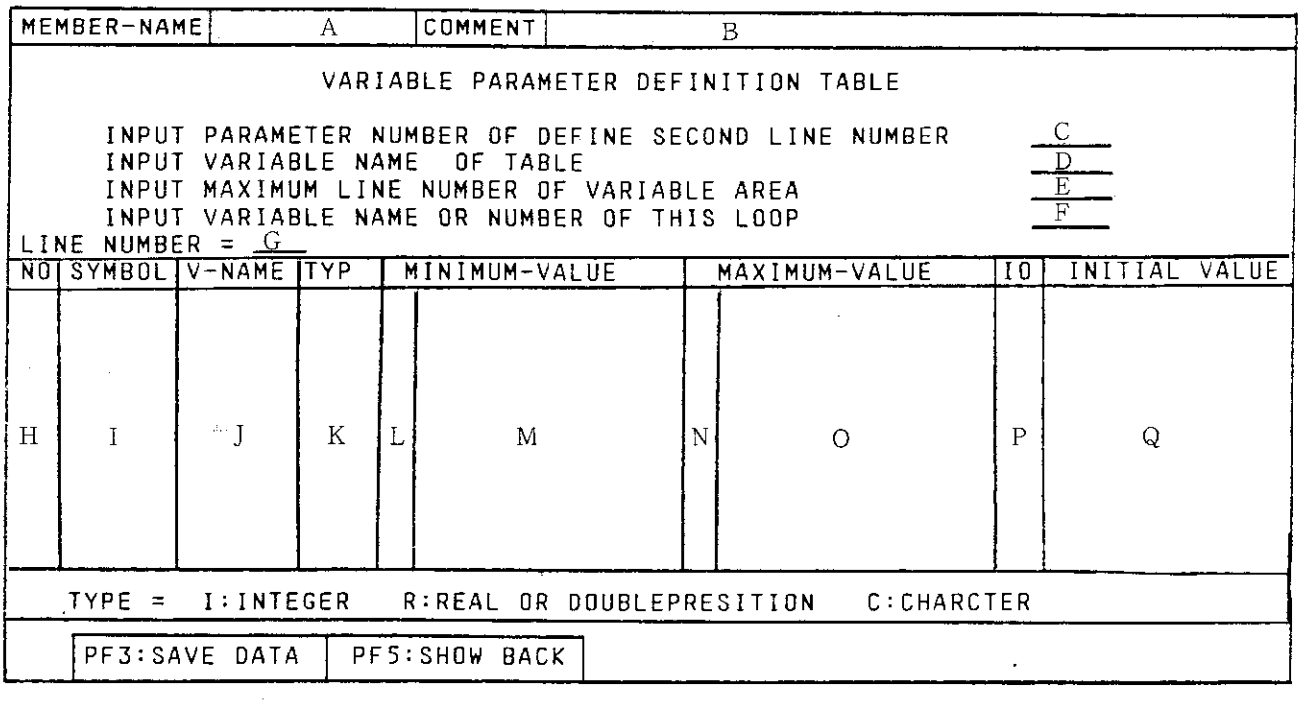


Fig. 7.6b Variable Definition Menu for T2-Table

8. ＪＤＩＳＳＰＨ２プログラムの実行

8.1 実行コマンド

使用する機番とファイルの関係はFig.8.1の通りである。Fig.8.2にコマンドリストを示す。

8.2 定 義 文

定義文は，ネームリスト入力部分とフォーマット入力部分に分かれている。先頭にネームリスト文を，次にフォーマット入力部を示す。

(1) ネームリスト部

ネームリスト名：＆ＦＩＬＥＳ

リ ス ト 名		説　　明
ＭＮＵＦＩＬ	Ｃ＊４５	メニューファイル名 ＰＳファイルである。ＰＯファイルに格納する時はメンバー名まで指定する。
ＭＥＮＵＥ(n)	Ｃ＊　６	ＭＥＮＵＦＩＬ中のメニュー名を指定する。 総てのメニューを使用する時には， ＭＥＮＵＥ(1)＝“＊”　とする。
ＯＵＴＦＩＬ	Ｃ＊４５	作成した会話処理プログラム格納ファイル名 ＰＳファイルである。ＰＯファイルに格納する時はメンバー名まで指定する。
ＩＮＣＦＩＬ	Ｃ＊４５	ＩＮＣＬＵＤＥファイル名 ＰＯファイルのファイル名を指定する。
ＩＮＣＭＥＭ	Ｃ＊　８	ＩＮＣＬＵＤＥメンバー名
ＩＯＰＴ(1)	Ｉ	作成する会話処理プログラムのうちで，メニュー制御サブルーチンだけを別ファイルに格納するか否かのフラグ。 ＝０　：　総てのファイルをＯＵＴＦＩＬで指定されたファイルに格納する。 ＝１　：　メニュー制御サブルーチンだけをＯＵＴＭＮＵで指定されたファイルに格納する。 ＝－１：　メニュー制御サブルーチンのみを作成してＯＵＴＦＩＬに格納する（ＩＮＣＬＵＤＥも作らない）。
ＩＯＰＴ(2)	Ｉ	中間ファイルへの出力に関するフラグ。 ＝０　：　中間ファイルへのデータの出力を制御するサブルーチンを￥ＳＡＶＥＦという名前で用意してある。 ＝１　：　用意してなく，システムにまかせる（この時は，総てのデータを使用，未使用にかかわらず格納する）。
ＯＵＴＭＮＵ	Ｃ＊４５	ＩＯＰＴ(1)＝１の時に，メニュー制御サブルーチンを格納するファイル名 ＭＡＩＮやその他の制御サブルーチンはＯＵＴＦＩＬに格納される。

(2) フォーマット入力部

① コメント行

1カラム目が＊のものである。各々の区切りであればどこにあってもよい。

② 中間ファイル・フォーマット指定行

・入力方法指定フラグ（ＩＲＴＹＰ）

カラム　：　1

＝0　　：　以降にフォーマットを書きくだす。

＝1　　：　中間ファイルを解析してフォーマットを決定。

・中間ファイル・フォーマット

ＩＲＴＹＰ＝0のときは，5.2の中間ファイルの実データ部を除いたフォーマットに従って入力を行う。

ＩＲＴＹＰ＝1のときは，先頭1カラム目から，使用する中間ファイルの変数が総て格納されている中間ファイル名を記述する。システムはそのファイルのフォーマットを解析する。

(3) セクション選択定義部

・セクション数

1～5カラムに，セクション選択定義数を記述する。

変数名	カラム	タイプ	説明
ＮＢＬＫ(0)	1～6	I	セクション選択機能の数

上述したセクション数だけ，下記のデータを繰り返す。

・呼び出し名定義

変数名	カラム	タイプ	説明
ＢＬＫＮＡＭ(1,n)	1～6	Ｃ＊6	セクション選択機能呼び出し名。ここで定義した名前を制御文中でＣＡＬＬすると，その時にセクション選択機能が働く。
ＢＬＫＮＡＭ(2,n)	10～15	Ｃ＊6	セクション選択定義サブルーチン名。この名前で，制御文中にサブルーチンを作り，条件によってセクションをどのように選ぶかを定める。
ＮＢＬＫ(n)	20～25	I	このセクション中の，セクションの数

・セクション名

セクション名を上記ＮＢＬＫ(n)個並べて入力する。

変数名	カラム	タイプ	説明
ＢＬＮＡＭ(I,J)	1～6	Ｃ＊6	セクション名。この名前のサブルーチンが，セクションにおけるメニューの制御を行っている。

Fig.8.3に入力データ例を示す。

8.3 制御文

制御文は，ＭＡＩＮから始まって条件に応じてメニュー画面を呼び出す制御をFORTRAN77の文法に従って行うものである。ＦＯＲＴＲＡＮ77で使用できる機能は殆どそのまま使用できるので，ＪＤＩＳＳＰＨ２の利用者は，作成しようとする会話型入力データ作成システムの構造に応じて自由にサブルーチンに分割したり，計算等を行うことができる。以下に，制御文の制限及び，セクション呼出し機能や下敷ファイルの選択機能，を実行させるための規則を示す。

(1) 変数の制限

変数に，¥マークを使用することはできない。ＪＤＩＳＳＰＨ２で自動作成したプログラムの変数の先頭に¥を使用しているからである。そのため変数に¥マークを使用した場合はその結果は保証できない。

(2) 文関数の禁止

文関数は使用できない。

(3) メニューで定義した変数の引用

各サブルーチンで，メニュー画面より定義した変数を使用するときには，それらの変数は定義文のＩＮＣＮＡＭで指定したＩＮＣＬＵＤＥ中にＣＯＭＭＯＮ変数として定義されているので，そのサブルーチン内でＩＮＣＬＵＤＥ文を指定するだけで使用できる。

逆に，ＩＮＣＬＵＤＥを指定したサブルーチンで，メニュー画面より入力する変数に値を代入するとその値は置き換わることになる。

(4) ＲＥＴＵＲＮ文

ＲＥＴＵＲＮ文は，制御文中のサブルーチンの最後に１つしか存在してはならない。サブルーチン途中でのＲＥＴＵＲＮ文の出現は認めない。

(5) ラ ベ ル

7000番代のラベルを使用してはならない。

(6) Ｉ Ｆ 文

メニュー制御サブルーチンをＣＡＬＬする際に，その文がＢＬＯＣＫ－ＩＦ文中にある場合は，そのＣＡＬＬ文は単純ＩＦ文であってはならない。

例

許されない

```
IF(……)THEN
   IF(……)CALL  Menu
ENDIF
```

⇒ 許される

```
IF(……)THEN
   IF(……)THEN
      CALL  Menu
   ENDIF
ENDIF
```

(7) メニュー画面の呼び出し

ＪＤＩＳＳＰＨ１で作成したメニュー画面を表示させ，ディスプレーとデータのやりとりを行いたい場合には，メニュー画面の制御を行うサブルーチンがメニュー名に対応して自動作成

されているので，ＣＡＬＬするだけでよい。すると，ＪＤＩＳＳＰＨ２システムが自動的にそのＣＡＬＬステートメントに引数（リターンコード）を付加し，リターンコード（ＰＦ２キィー・ＰＦ４キィー）の処理を加える。

例　　ＣＡＬＬ　メニュー名

(8)　下敷ファイルの選択機能の実行

任意の中間ファイル名を指定してそれらを下敷ファイルとする機能は，サブルーチン￥ＲＥＡＤＸをＣＡＬＬすることによって実行される。

ＣＡＬＬ　￥ＲＥＡＤＸ（ＩＲＴ）

ＩＲＴ　：リターンコード

＝８　：サブルーチン異常終了

リターンコードの処理は，必要に応じて付ける。

(9)　データの格納

中間ファイル指定メニューを表示して，全データを中間ファイルに格納するには￥ＥＮＤサブルーチンをＣＡＬＬする。

ＣＡＬＬ　￥ＥＮＤ（ＩＲＴ）

ＩＲＴ　：リターンコード

＝８　：サブルーチン異常終了

リターンコードの処理は，必要に応じて付ける。

(10)　セクション呼出し機能

セクション呼出し機能は，定義文で定義したセクション選択定義サブルーチンを作成し，制御文でセクション選択機能呼び出し名をＣＡＬＬすることによって実行される。

(a)　セクション選択定義サブルーチンのＣＡＬＬ

定義文で指定したセクション選択機能呼び出し名で，セクション選択機能を実行させるサブルーチンを作成しているので，その名前をＣＡＬＬすることによって，実行される。

ＣＡＬＬ　セクション選択機能呼び出し名（ＩＲＴ）

ＩＲＴ　：リターンコード

＝８　：サブルーチン異常終了

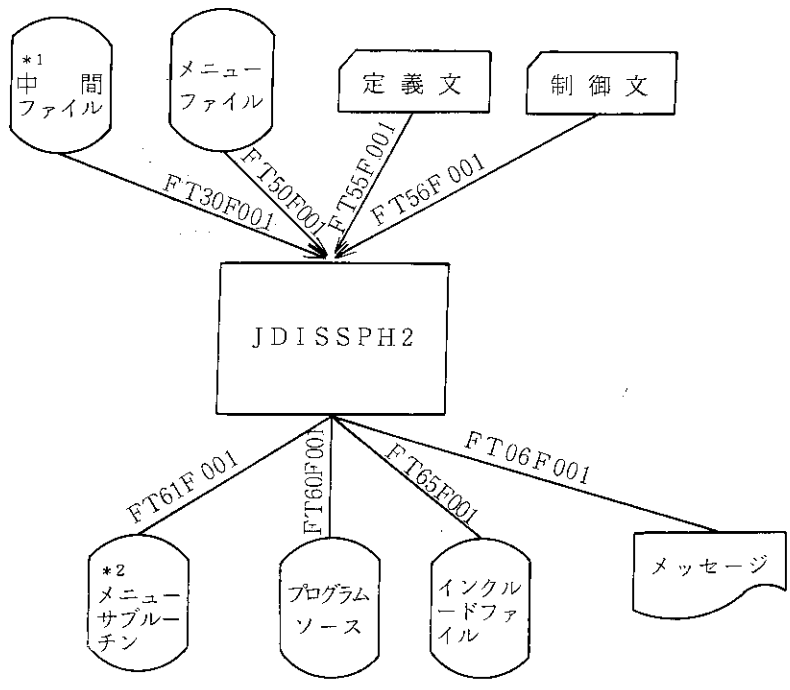
リターンコードの処理は，必要に応じて付ける。

(b)　セクション選択定義サブルーチンの作成

定義文で定めるセクション選択定義サブルーチン名で，条件によるセクションの呼び出しを定義するものである。

セクション選択定義画面を表示させるのは￥ＢＬＫＳサブルーチンであるので，ＪＤＩＳＳＰＨ２利用者はこのセクション選択定義サブルーチンでは，￥ＢＬＫＳの引数をセットして，ＣＡＬＬすることになる。

制御文の例，及びセクション選択機能の実際の定義方法を示すために，ＳＲＡＣコードにおけるプログラムフローをFig.8.4に，そのＦＯＲＴＲＡＮによるコーディングを付録で示す。



*1　ＩＲＴＹＰ＝１のとき必要

*2　ＩＯＰＴ(1)＝±１のとき必要

Fig. 8.1　Input/Output Files of JDISSPH2

```
PROC 1 GO  DSN(JDISSPH2)  DMAIN(SRACCNT) DSUB(SRACFRT2) +
           LOA(JDISS.LOAD)     LMEM(JDISSPH2) INC(INC) MEM(*)
CONTROL   LIST MSG PROMPT
FREE  F(FT55F001 FT50F001 FT60F001 ) ATTR(XX)
FREE  DA(TESMENU1.DATA)
FREE  DA(TESTSUB1.FORT77)
ATTR XX  INPUT
/* ALLOC DA(@SAME.DATA) REUSE F(FT25F001)
/* ALLOC DA(@SAME.DATA) NEW CAT SP(5 5) T  F(FT25F001)
ALLOC DA(JDISSPH2.DATA(&DMAIN))   F(FT55F001)  US(XX) REUSE
ALLOC DA(JDISSPH2.DATA(&DSUB))    F(FT56F001)  US(XX) REUSE
/*MCOND SRACJDIS.FORT77
  IF &GO EQ 1 THEN GOTO GOGO
  DEL @OBJX
  FORT77  &DSN         OBJ(@OBJX)       ELM(*) INC(&INC)
  DEL @RUNX
  LINK  @OBJX    LOAD(@RUNX)   F  LIB('J7858.SEAK.LOAD')
  MCOND &LOA
  COPY @RUNX.LOAD(TEMPNAME) &LOA(&LMEM)
GOGO:  LIB ('J7858.SEAK.LOAD')
       WRITE ***** GO GO GO **** &SYSTIME
       CALL &LOA(&LMEM)
       WRITE *****  OWARI   **** &SYSTIME
FREE  DA(TESMENU1.DATA)
FREE  F(FT55F001 FT50F001 FT60F001 ) ATTR(XX)
FREE  DA(TESMENU1.DATA)
FREE  DA(TESTSUB1.FORT77)
 EXIT
```

Fig. 8.2 Command Procedure of JDISSPH2

```
 &FILES
   MNUFIL      = 'J9055.SRACTRNS.MENUS',
   OUTFIL      = 'J7858.SRACJDIS.FORT77(SRACMAIN)',
   OUTMNU      = 'J7858.SRACJDIS.FORT77(SRACMENU)',
   INCFIL      = 'J7858.SRACINC.FORT77',
   IOPT(1)     = 1
   IOPT(2)     = 0
   MENUE(1)    = '*'
   INCMEM      = '¥INC¥',
 &END
*
******  SRAC INPUT FILE NAME
*
1
J7858.SRINT.DATA(SRAC00)
*
******  SECTION DEFINED
*
1
BLOCK      BLK001     13
GENERL
USERF
COLLIS
ANISN
TWOTRN
TUD
CITATN
MATRAL
REACT
BURNUP
MCROSX
PEACO
¥END
```

Fig. 8.3 Definition Description for SRAC Code

ＳＲＡＣの処理の流れ

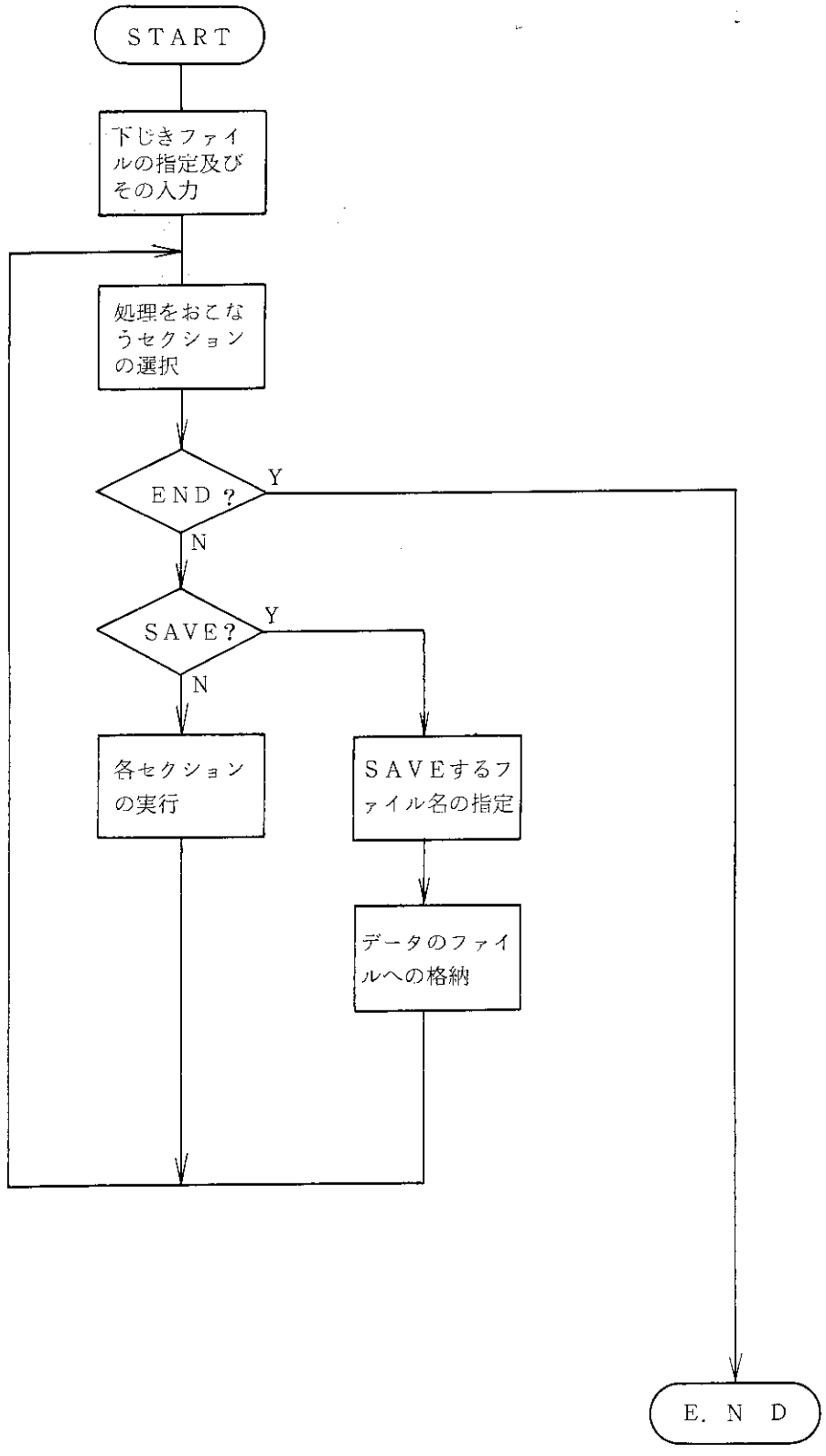


Fig. 8.4 Flow Diagram of Contorol Description for SRAC Code

9. ユーティリィティーサブルーチン群

いくつかのユーティリィティー・サブルーチンを開発したので，それらを以下に示す。

★ ¥ＳＡＶＥＤ（ＩＯＸ，ＣＮＡＭ，ＩＲＴ）

データを中間ファイルに格納する。

ＩＯＸ　：　入力出力機番

ＣＮＡＭ　：　格納するデータの変数名ないしテーブル名

ＩＲＴ　：　リターンコード

★ ¥ＭＩＮＵＳ（ＣＮＡＭ，ＩＡＮＳ，ＮＲＲ，ＩＷ，ＩＲＴ）

テーブル２データ中に負の値が有るか無いかを調べる。

ＣＮＡＭ　：　テーブル２の配列名

ＩＡＮＳ　：　答。負の値があるときは“１”，なければ“０”を格納。

ＮＲＲ　：　テーブル２のデータ数

ＩＷ　：　作業領域

ＩＲＴ　：　リターンコード

★ ¥ＥＲＯＵＴ（ＥＲＭ，ＮＬ，ＩＲＴ）

現在使用されている画面にエラーメッセージを表示する。

ＥＲＭ　：　エラーメッセージ　（Ｃ＊４０）

ＮＬ　：　メッセージを出力する行（普通は１を入力する）

ＩＲＴ　：　リターンコード

注）メニュー画面をＣＡＬＬしたあとに本サブルーチンをＣＡＬＬすると直前の画面に引数ＥＲＭの内容が表示される。本サブルーチンをＣＡＬＬしたのちは，¥ＩＲＴに－１を代入した後に再びメニュー画面をＣＡＬＬしなくてはならない。

本ソースは，ＪＤＩＳＳＵＴＹに格納されている。Fig.9.1に¥ＥＲＯＵＴの例を示す。

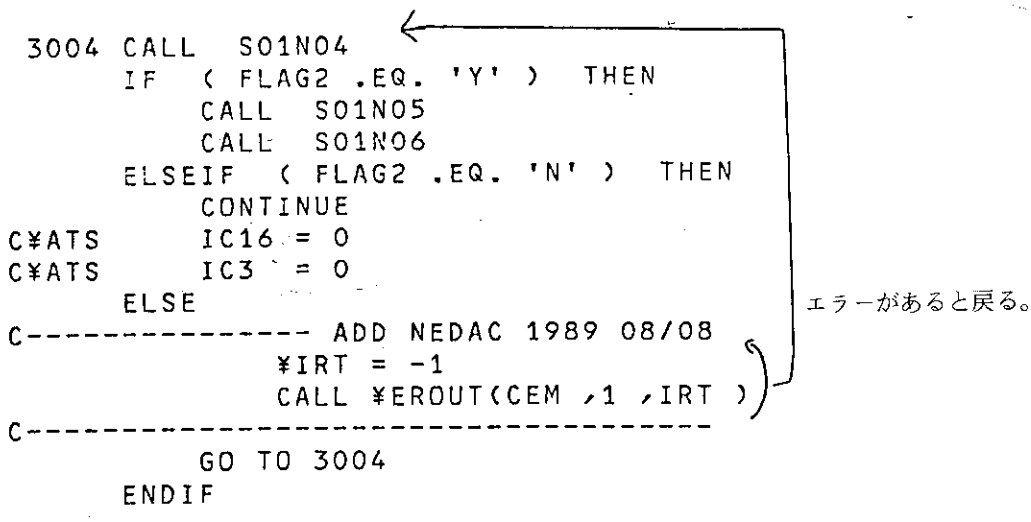


Fig. 9.1 Example of Error Processing Using ¥EROUT

10. ＪＤＩＳＳの簡便な利用

ＪＤＩＳＳは，原子力コードの会話型入力データ作成システムの構築を目的としたシステムである。その処理を大きく２つに分けるならば(1)メニュー画面の作成や，メニュー画面とプログラムのインターフェースを行う画面制御サブルーチンの作成といったＩＰＦ関連の制御部の自動作成部分と，(2)セクション呼出し機能，下敷ファイルの設定や中間ファイルへのデータ格納機能といった会話型入力データ作成システム全体の構成を決定するための制御部の自動作成部分に分けることができる。そして，それらの部分を制御文と定義文の入力によって有機的に結合してシステムの構築を図っている。

現在，メニュー画面を使って会話的にデータの入出力を行いながら処理を進めて行くことは，単に原子力コードの入力データ作成のみにとどまらず，ＴＳＳコマンドの実行や各種ユーティリィティープログラムの実行等，幅広く行われており今後さらに増えて行くものと思われる。

しかし，現状ではやはりＩＰＦ関連のサブルーチン作成，すなわちメニュー画面の制御にかなり煩わされているようである。そこで，それらに対してもＪＤＩＳＳでは，先に述べた機能のうち(1)の部分だけを取り出して，メニュー画面と，その制御サブルーチンを単独で使用することによって様々な利用にも対処できるようになっている。

具体的には定義文中でＩＯＰＴ(1)＝１と指定すると，メニュー画面を制御して，画面とデータの入出力を行うサブルーチンのみを１つのファイルに作成する。そこで，利用者は様々な処理を行っていく上で画面を表示したい所で，あたかもライブラリィーをアクセスするかのようにメニュー名をＣＡＬＬするだけで画面の制御が行えるようになる。そして，ＣＯＭＭＯＮ中に格納されたデータをもとに，ＴＳＳコマンドや各種ユーティリィティープログラムを実行することになる。

このようにＪＤＩＳＳから，メニュー画面を会話的に作成する機能（ＪＤＩＳＳＰＨ１）及びそのメニュー画面を制御して，データの画面からの入出力を制御するサブルーチンを作成する機能（ＪＤＩＳＳＰＨ２）を単独に取り出すことによって，ＩＰＦを利用しての様々な処理の負荷軽減にも大きく貢献することができる。

Fig.10.1にその様子を示す。

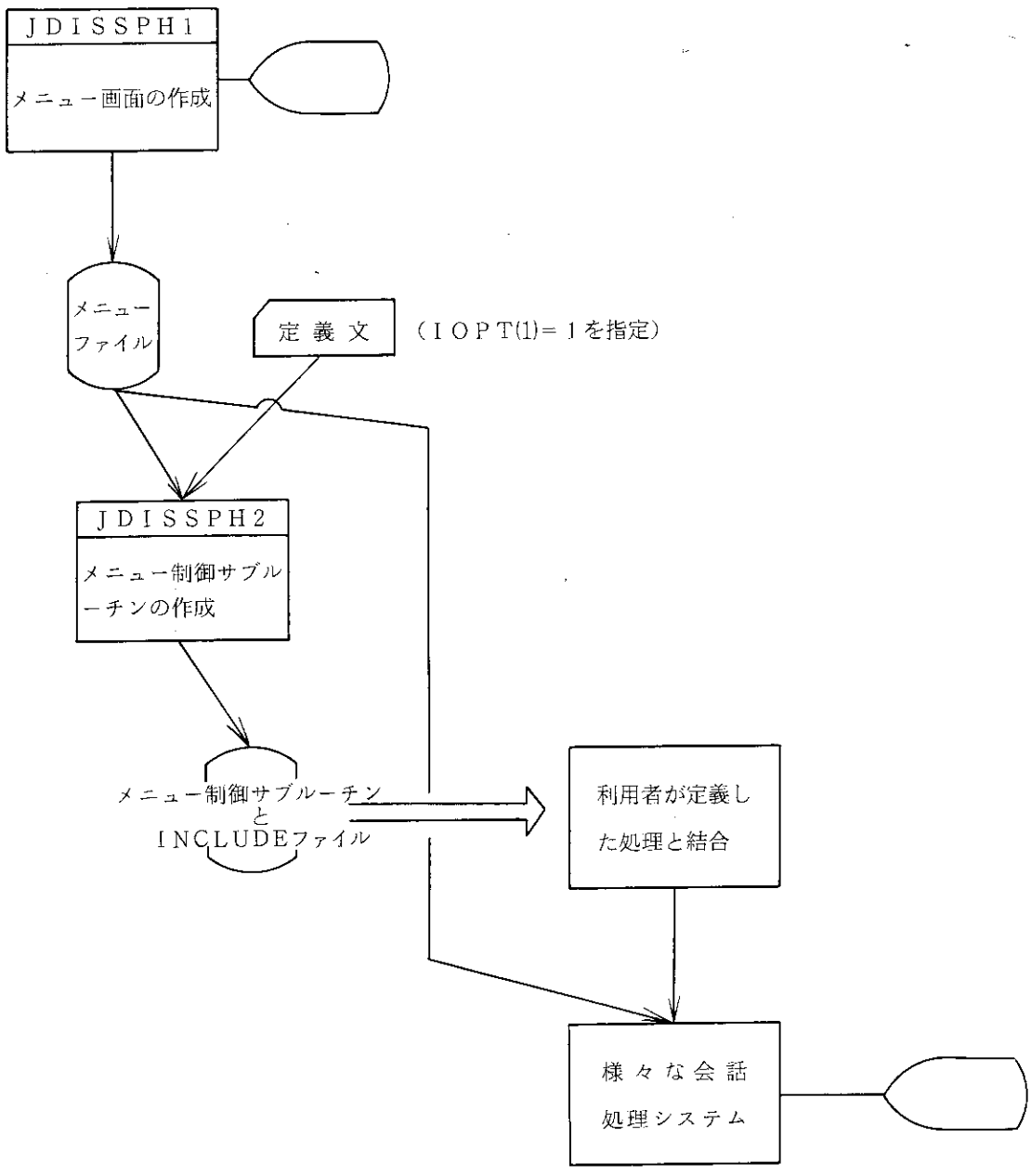


Fig. 10.1 Simplified Use of JDISS

11. ＪＤＩＳＳシステムのＳＲＡＣへの適用

今回の作業では，ＪＤＩＳＳシステムを用いて，ＳＲＡＣ入力データ作成システムの開発を行い，その過程で出てきた問題点をＪＤＩＳＳシステムにフィードバックしながらＪＤＩＳＳシステムの改良，ＳＲＡＣ入力データ作成システムの開発を並行的に行ってきた。ＪＤＩＳＳシステムに加えられた改良は，既に９章までに述べられている。そこで，本章では，実際にＳＲＡＣにどのように適用していったのかを述べる。

11.1 ＳＲＡＣの入力構造

ＳＲＡＣの入力は，Fig.8.3に示したようにＧＥＮＥＲＡＬ～ＰＥＡＣＯの12のセクション（ＭＣＲＯＳＳは使用しないので実際には11のセクション）に分かれており，１番目のＧＥＮＥＲＡＬで入力された条件によって，11の項目のうち必要なセクションが選択されるようになっている。そして，各々のセクションがまとまった意味を持っている。

ＣＩＴＡＴＩＯＮなどの一部の例外を除いてはフリーフォーマット入力であり，特別な書式を必要とするものではない。

11.2 ブロック分割

ＪＤＩＳＳにおけるブロック分割は，入力を意味のあるいくつかの塊に分けて処理をしようとするものである。それは必ずしも入力データの構造によって規定されるものではないが，本例では，入力のセクションがそれぞれ独立した意味を持つことから，11.1で示した入力におけるセクション分割にしたがって，中間ファイルのデータ分割及びブロック分割を行った。

11.2.1 ＧＥＮＥＲＬ

ＧＥＮＥＲＬ（General Control and Energy Structure Specification）は，ＳＲＡＣの各種計算の制御データの入力を行う部分であり，個別の画面の構成は簡単であるが，入力された値のそれぞれの関係が複雑であり，その制御には細心の注意を必要とする。（Fig.11.1参照）

11.2.2 ＵＳＥＲＦ

ＵＳＥＲＦ（User's Microscopic Cross Section Libraries）では，８バイトの領域を持つデータを，指定した数だけ（最高91個）入力する。そこで，テーブル画面－２を利用して全体の数を入力した後，その値から入力に必要な行数をファンクションによって計算させることとした。（Fig.11.2参照）

11.2.3　ＣＯＬＬＩＳ

ＣＯＬＬＩＳ（Collion Probability Method）では，テーブル画面－１より入力された配列ＭＡＲ中に負の値を取るものがあるかどうかのチェックを行い，負のデータがあった場合のみブロック14の画面を呼ぶという処理が問題となる。そこで，データを読み込み，負のデータがあるか否かをチェックするサブルーチン¥ＭＩＮＵＳの開発を行い，その判定を行うこととした。¥ＭＩＮＵＳはＪＤＩＳＳＵＴＹに追加されている。（Fig.11.3参照）

11.2.4　ＡＮＩＳＮ

ＡＮＩＳＮ（One Dimensional SN Transport）の入力は比較的単純であり，一般画面とテーブル画面の組み合わせで行った。（Fig.11.4参照）

11.2.5　ＴＷＯＴＲＮ

ＴＷＯＴＲＮ（Two Dimensional SN Transport）の入力も比較的単純であり，一般画面とテーブル画面の組み合わせで行った。（Fig.11.5参照）

11.2.6　ＴＵＤ

ＴＵＤ（One Dimensional Diffusion）の入力も比較的単純であり，一般画面とテーブル画面の組み合わせである。（Fig.11.6参照）

11.2.7　ＣＩＴＡＴＩＯＮ

ＣＩＴＡＴＩＯＮ（Multi－Dimensional Diffusion）では，ブロック４からブロック８までの処理が相互に関連しており，各々の入力データ量も多くその制御には工夫を要するが，最終的にテーブル画面－２で対処した。（Fig.11.7参照）

11.2.8　ＭＡＴＲＡＬ

ＭＡＴＲＡＬ（Material Specification）では，テーブル画面－２を使ってデータの入力を行っている。（Fig.11.8参照）

11.2.9　ＲＥＡＣＴ，ＢＵＲＮＵＰ，ＲＥＡＣＯ

ＲＥＡＣＴ（Reaction Rate Calculation），ＢＵＲＮＵＰ（Cell Burn-up Calculation），ＲＥＡＣＯ（Ultra-fine Resonance Absorption Calculation）の３つは，比較的単純な構造であった。（Fig.11.9～11.11参照）

11.3　中間ファイルへの書き込みサブルーチン群（ＳＲＡＣＷＲＴ）

データの中間ファイルへの出力は，実行したい計算に応じて必要なデータのみを取り出してファイルへ出力するようにした。そのため，中間ファイルにデータを出力する時点でそれらのデータを取捨選択する特別のサブルーチンを介在させることが必要である。

これらのロジックを持つサブルーチン群は，ＳＲＡＣコードでは，ＳＥＣ００１～ＳＥＣ０１０として作成されている。ＪＤＩＳＳシステムでは，サブルーチン¥ＳＡＶＥＦで，これらをＣＡＬＬすることにより中間ファイルへのデータの格納を制御する。これらのサブルーチン群に関する情報は，通常，ＪＤＩＳＳＰＨ２の入力データである制御文に記述される。

ＳＲＡＣコードの中間ファイルの書込み制御を行うサブルーチン群をＳＲＡＣＷＲＴと呼ぶ。

今回の開発では，それらサブルーチン群のステップ数も多くなったので，ＪＤＩＳＳＰＨ２で作成されたプログラムＳＲＡＣＪＤＩＳに，後から結合する形態を取ったが，先に述べたように制御文に加えてもよい。

中間ファイルへの出力は，すべてユーティリィティーサブルーチン¥ＳＡＶＥＤを使用して出力を行う。¥ＳＡＶＥＤは，引数に変数名と，機番，リターンコードを送ると，中間ファイルのフォーマットで出力するものである。

ＳＲＡＣＷＲＴはいくつかのサブルーチンに分割されているが，基本的には制御文でメニュー画面を呼び出すのと同じロジックを持ったものであり，画面を出力する代わりにそれらデータを中間ファイルに書き出すものである。ただし，対象原子力コードの入力によっては，画面を呼び出すのと異なる方法でもよい。

ＳＲＡＣＷＲＴの木構造をFig. 11. 12に示す。

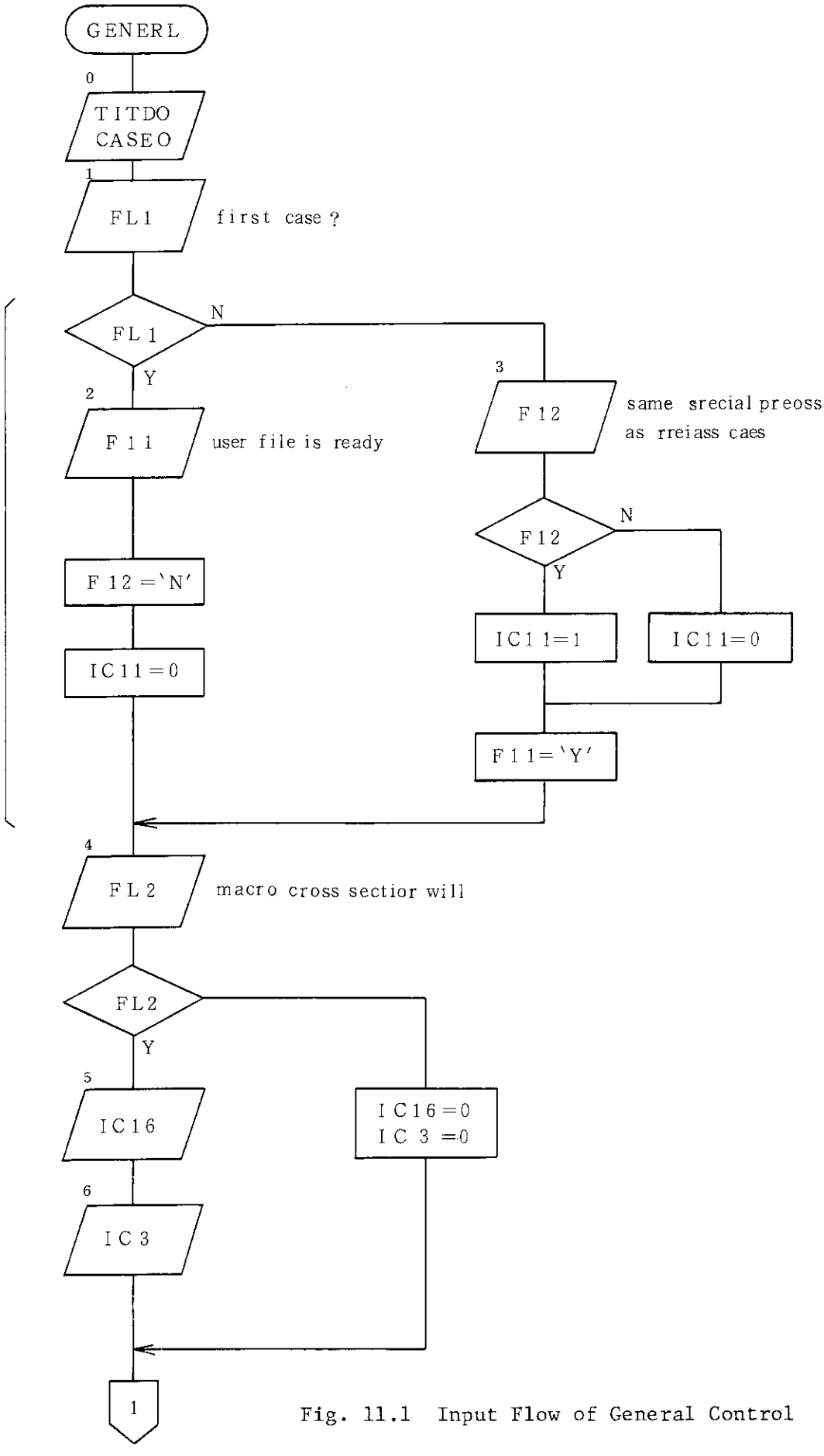


Fig. 11.1 Input Flow of General Control

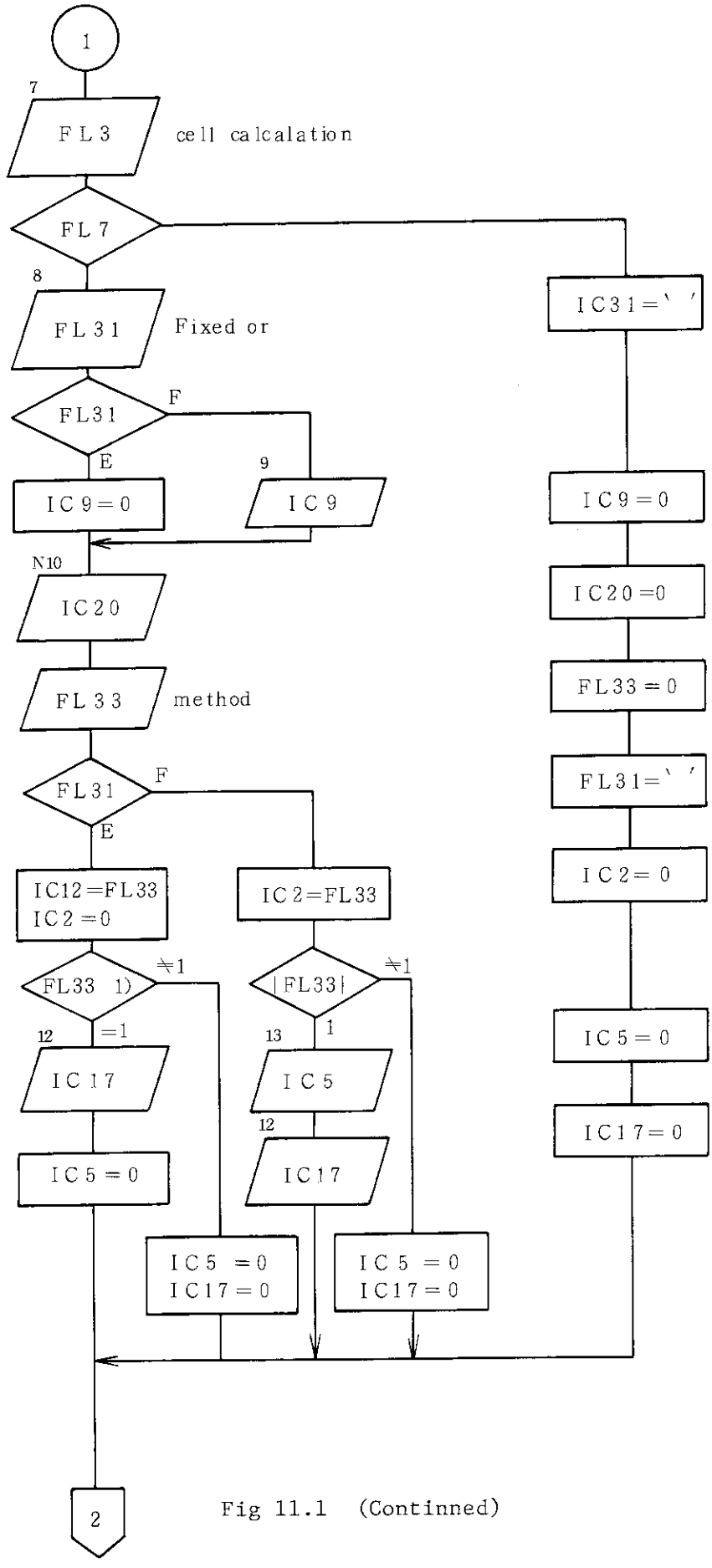


Fig 11.1 (Continned)

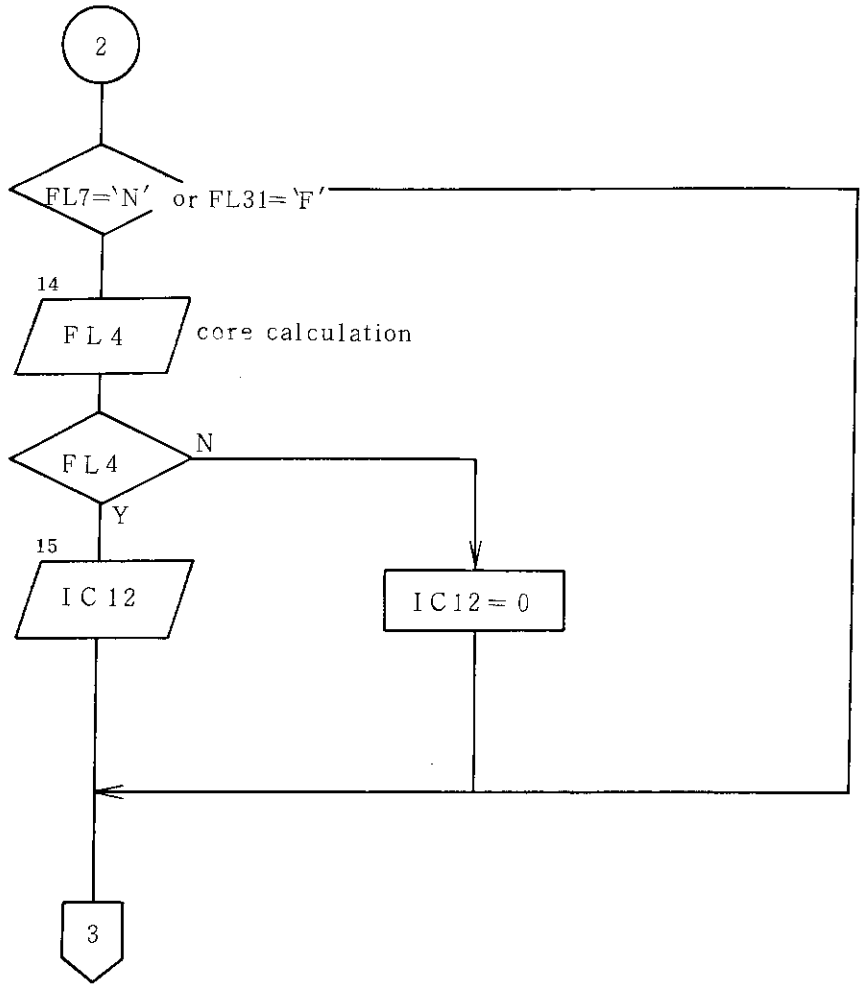


Fig 11.1 (Continned)

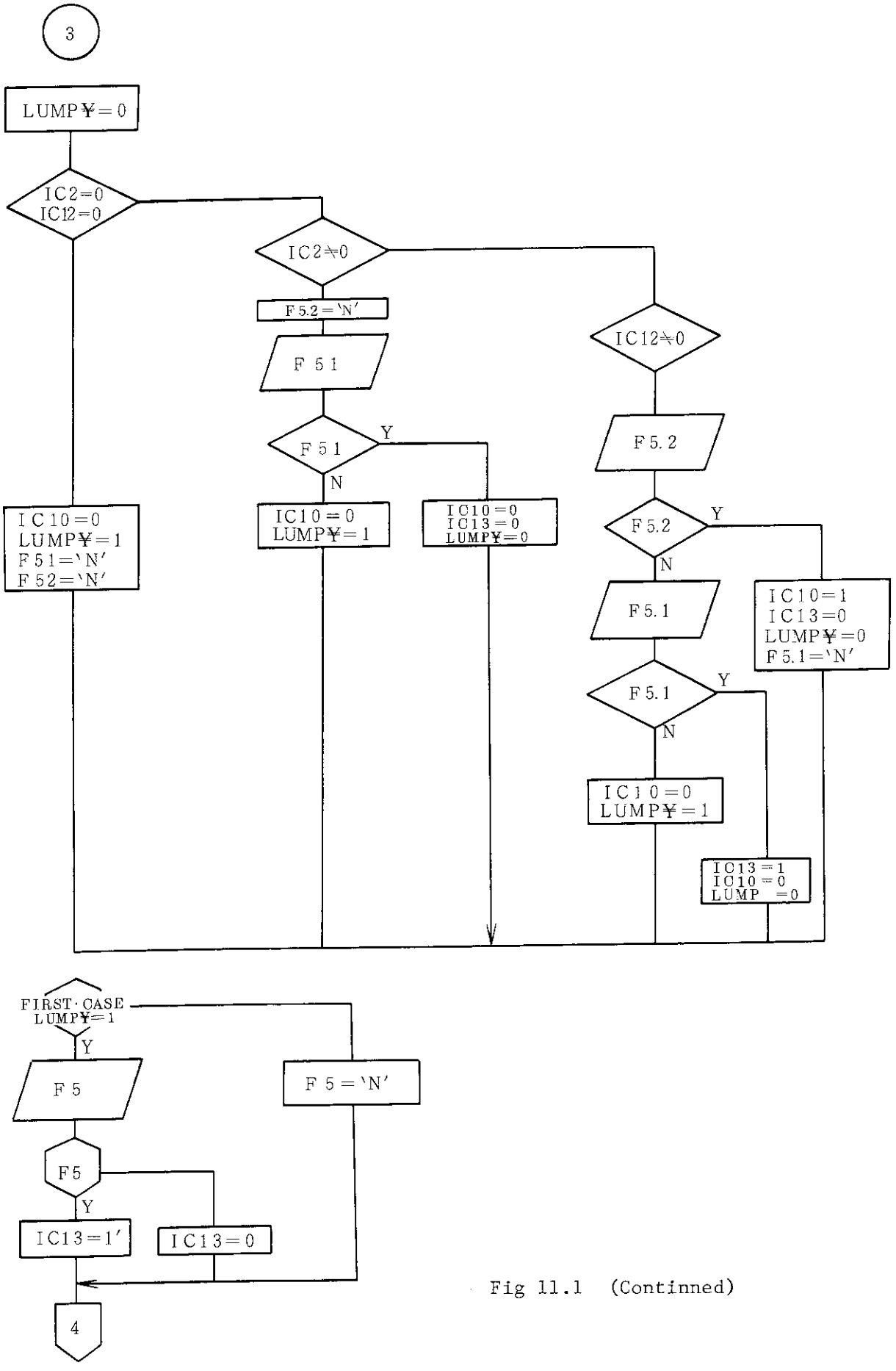


Fig 11.1 (Continned)

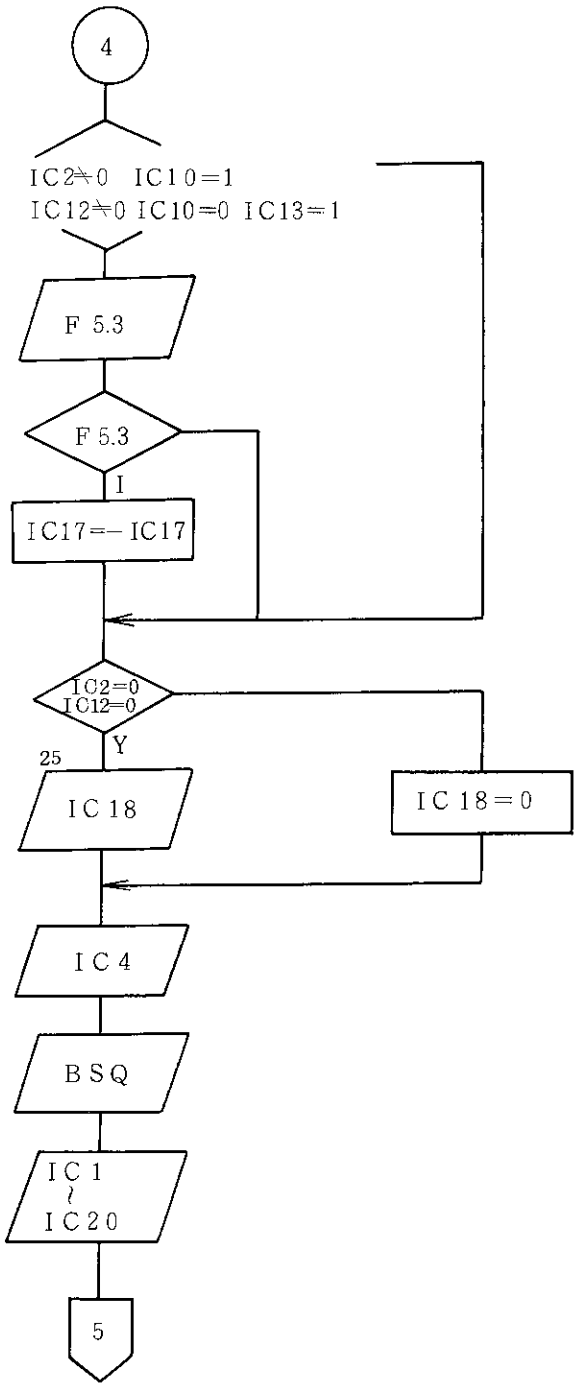


Fig 11.1 (Continned)

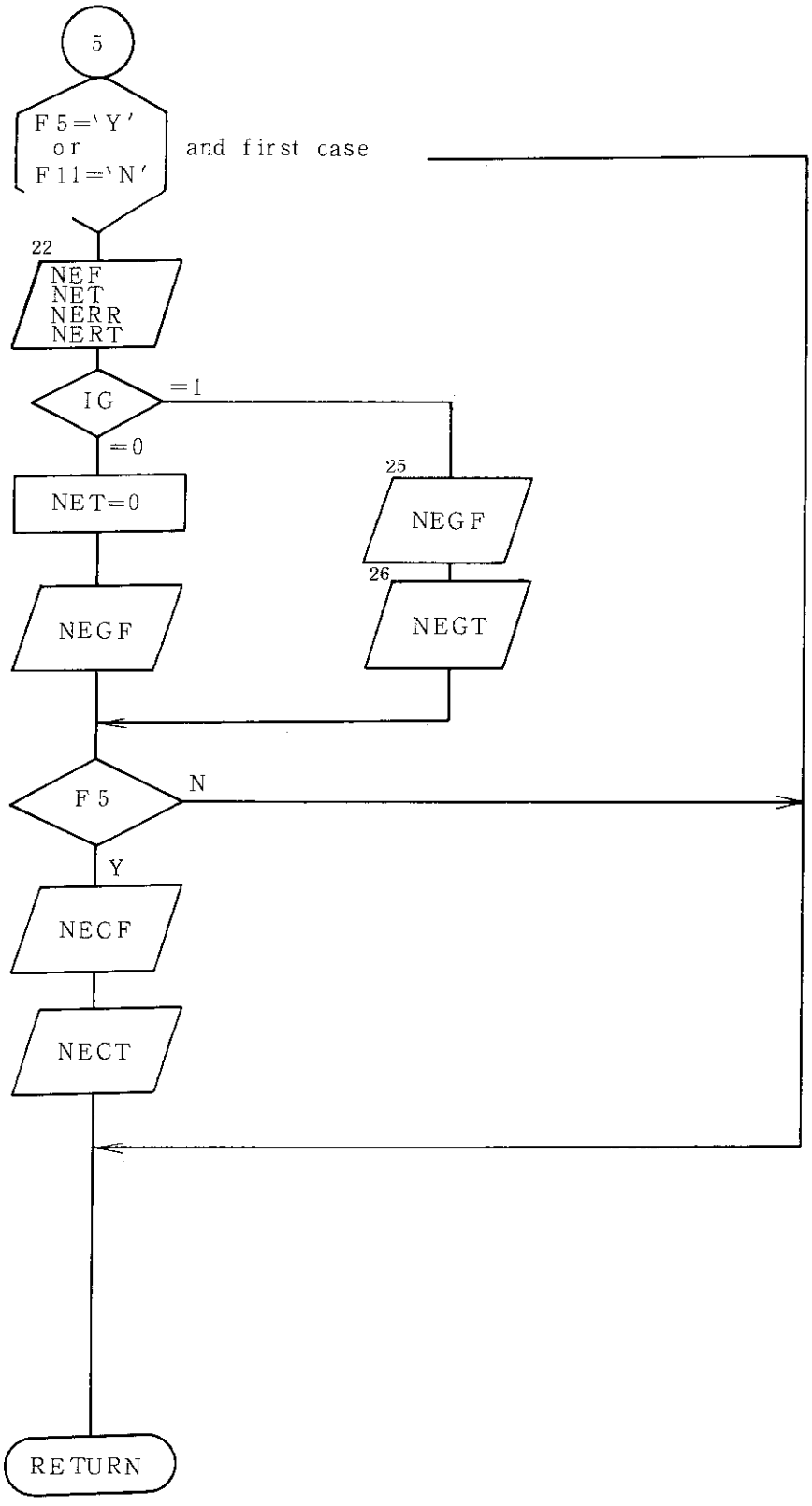


Fig 11.1 (Continned)

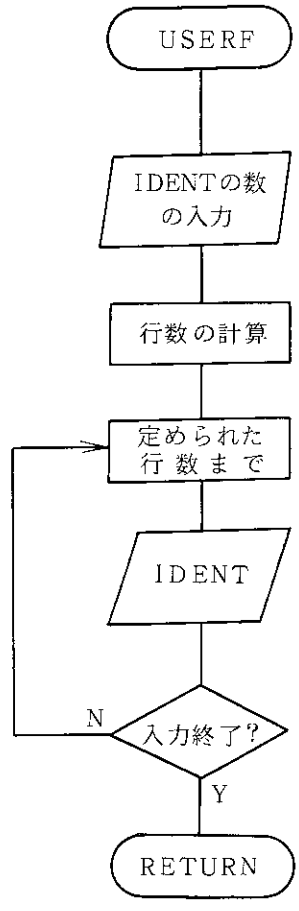


Fig. 11.2 Input Flow of User's Microscopic Cross Section Library

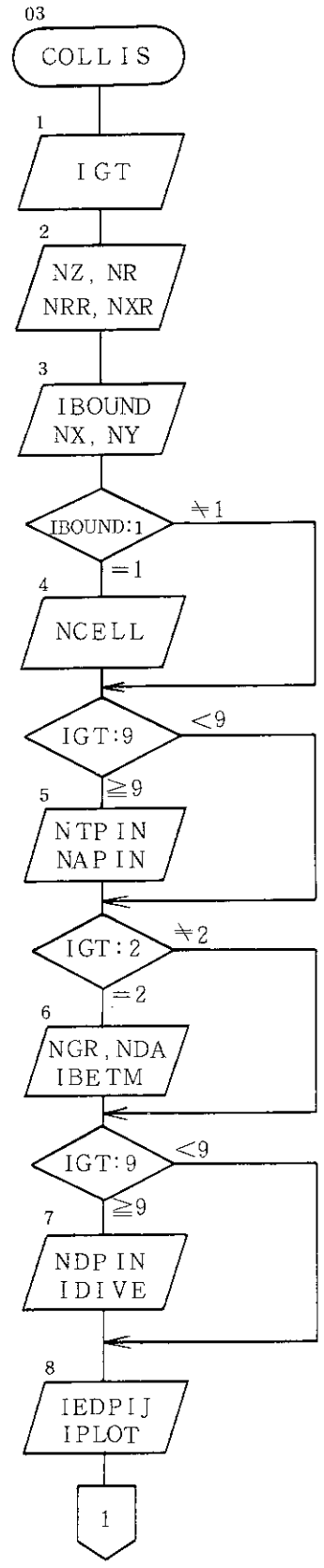


Fig. 11.3 Input Flow of Collision Probability Method

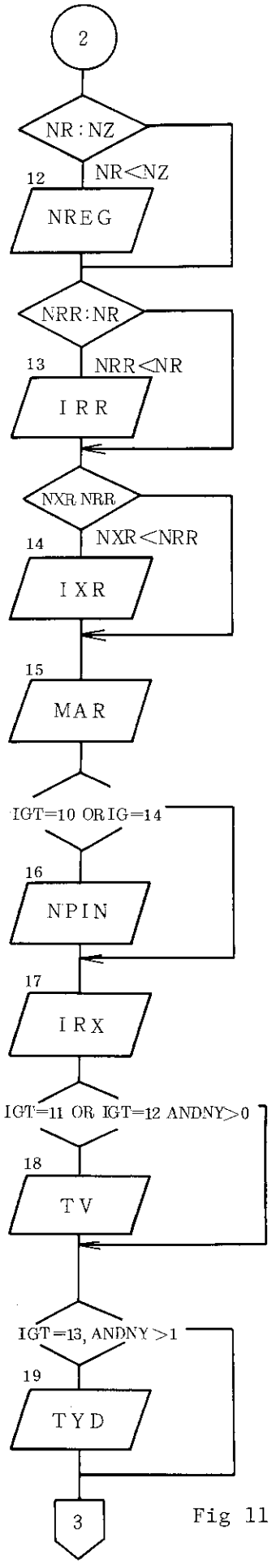


Fig 11.3 (Continned)

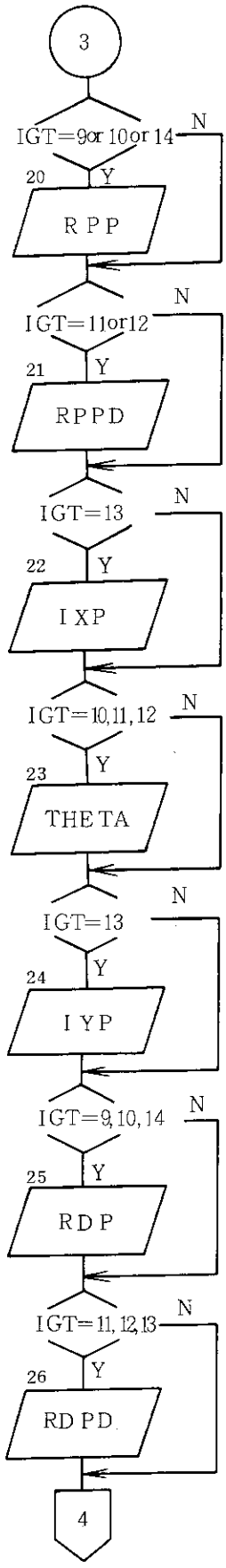


Fig 11.3 (Continned)

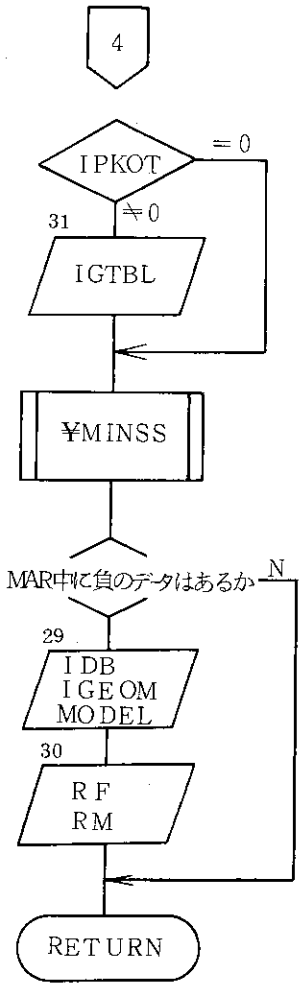


Fig 11.3 (Continned)

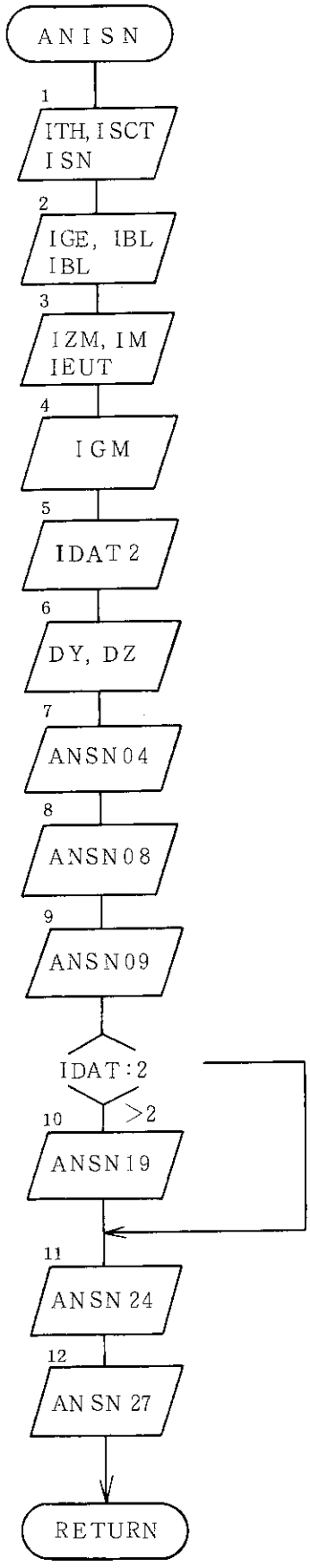


Fig. 11.4 Input Flow of ANISN Code

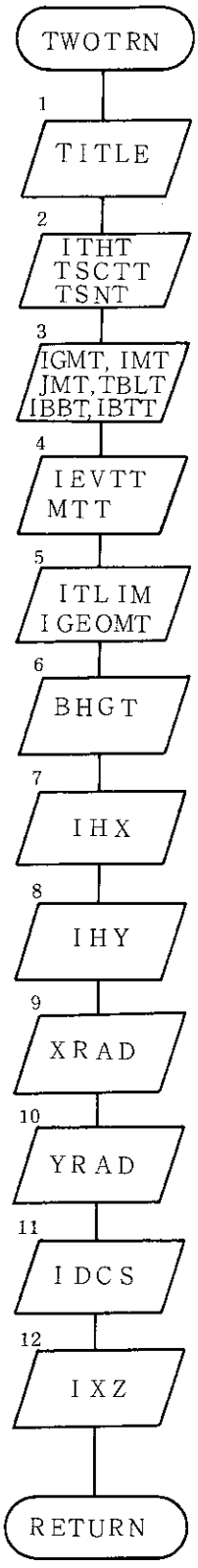


Fig. 11.5 Input Flow of TWOTRAN Code

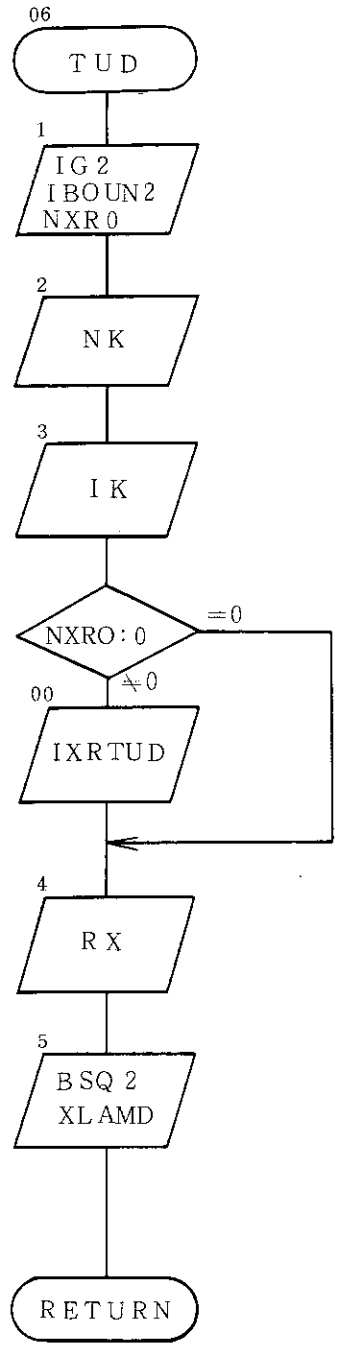


Fig. 11.6 Input Flow of TUD Code

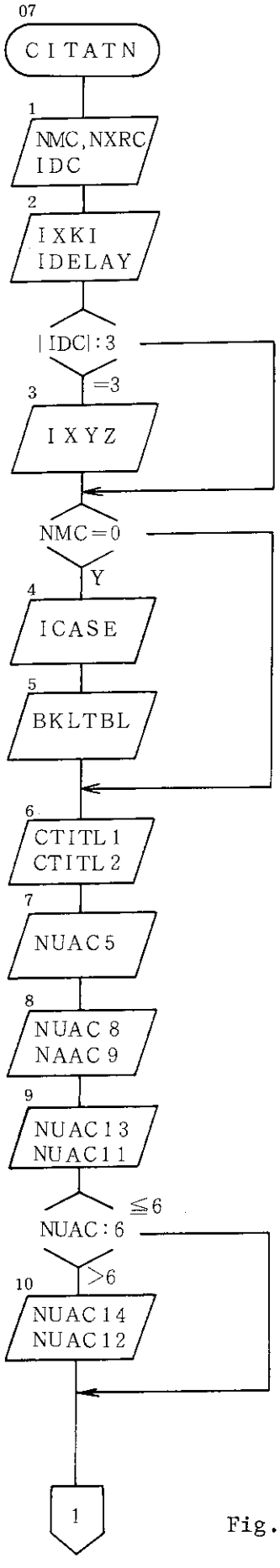


Fig. 11.7 Input Flow of CITATION Code

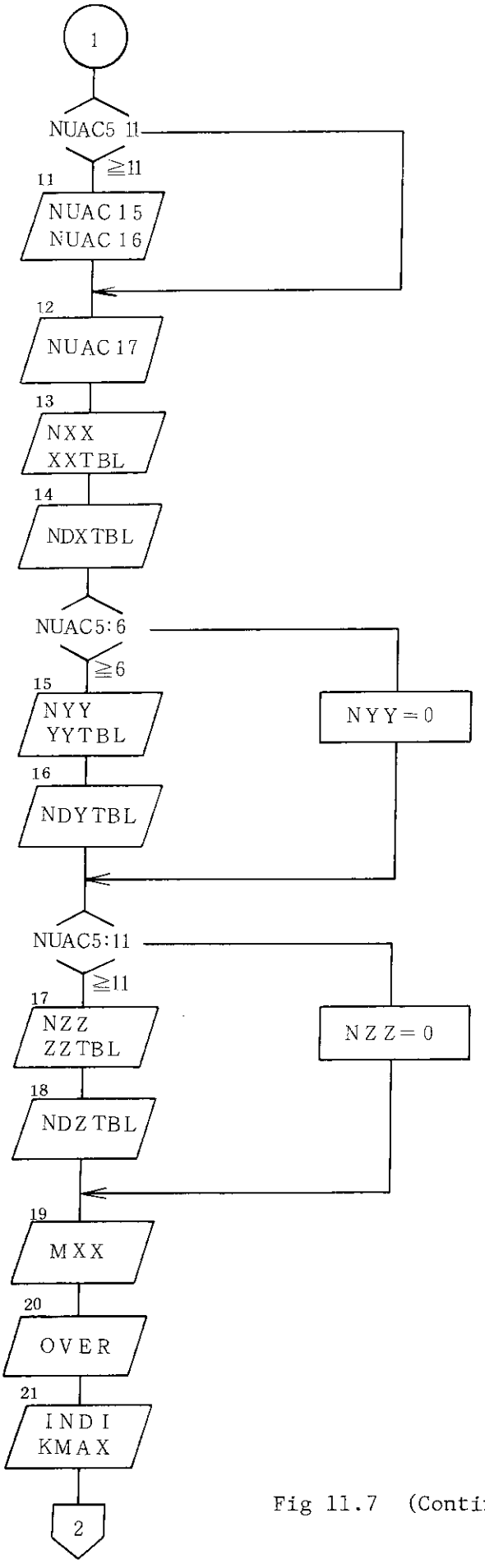


Fig 11.7 (Continned)

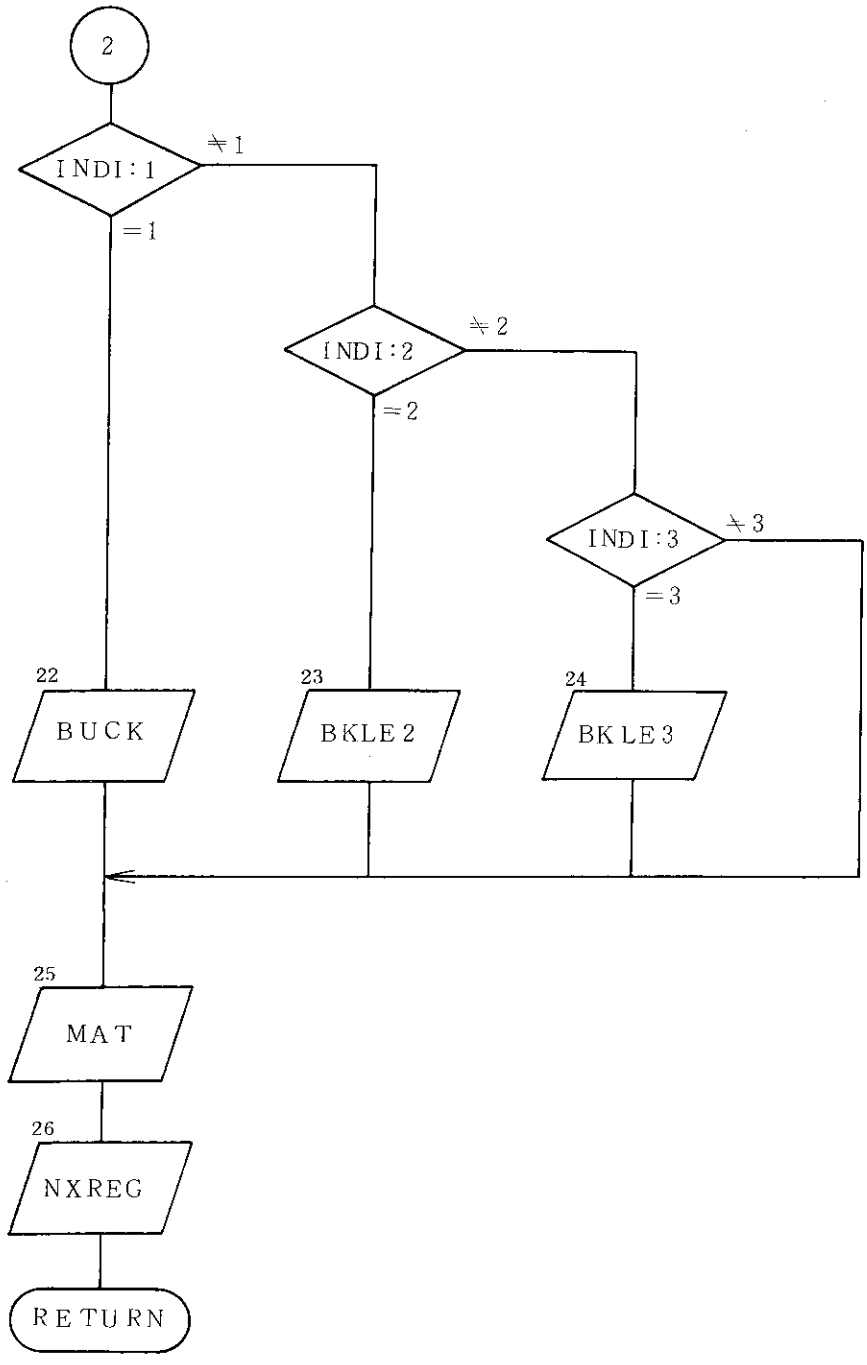


Fig 11.7 (Continned)

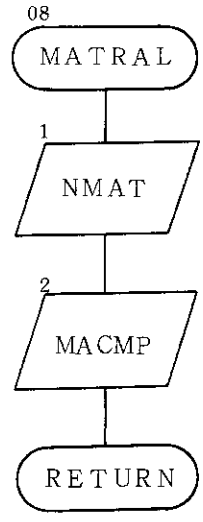


Fig. 11.8 Input Flow of Material Specification

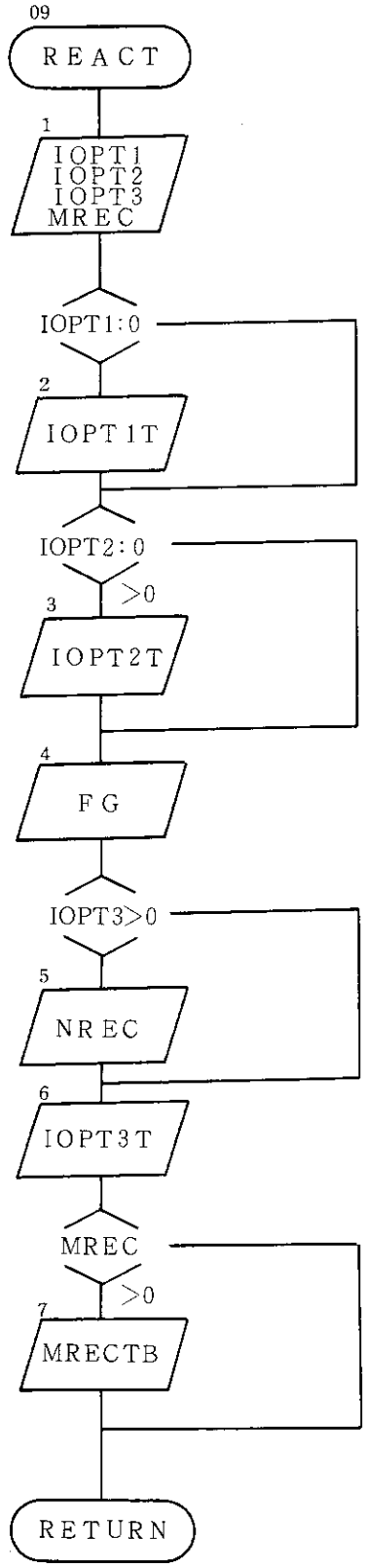


Fig. 11.9 Input Flow of Reaction Rate Calculation

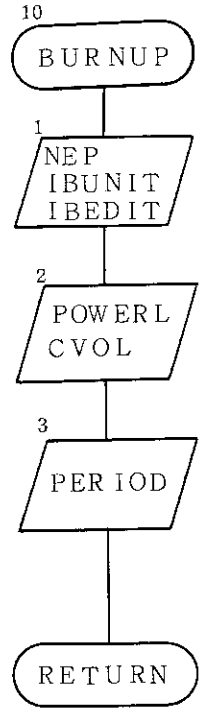


Fig. 11.10 Input Flow of Burn-up Calculation

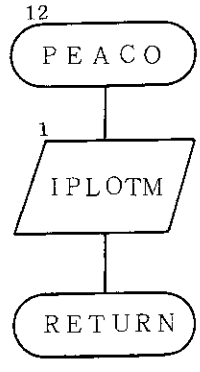


Fig. 11.11 Input Flow of Ultra-fine Resonance Absorption Calculation

```
=ANALYSIS/77= ** TREE STRUCTURE **  ENTRY POINT = YSAVEF          DATE  1989/10/09(MONDAY)     TIME 13:34:57  PAGE 0021

YSAVEF-----*DATE
         +--YSCNTL-----SEC001-----S01XXX----*YSAVED
                    |            +-%S01X00----*YSAVED
                    |            +-%S01X01----*YSAVED
                    |            +-%S01X5P----*YSAVED
                    |            +-%S01X02----*YSAVED
                    |            +-%S01X03----*YSAVED
                    |            +-%S01X04----*YSAVED
                    |            +-%S01X05----*YSAVED
                    |            +-%S01X06----*YSAVED
                    |            +-%S01X07----*YSAVED
                    |            +-%S01X08----*YSAVED
                    |            +-%S01X09----*YSAVED
                    |            +-%S01X10----*YSAVED
                    |            +-%S01X11----*YSAVED
                    |            +-*IABS
                    |            +-%S01X12----*YSAVED
                    |            +-%S01X13----*YSAVED
                    |            +-%S01X14----*YSAVED
                    |            +-%S01X15----*YSAVED
                    |            +-%S01X18----*YSAVED
                    |            +-%S01X19----*YSAVED
                    |            +-%S01X17----*YSAVED
                    |            +-%S01X16----*YSAVED
                    |            +-%S01X20----*YSAVED
                    |            +-%S01X21----*YSAVED
                    |            +-%S01X27----*YSAVED
                    |            +-%S01X22----*YSAVED
                    |            +-%S01X25----*YSAVED
                    |            +-%S01X26----*YSAVED
                    |            +-%S01X23----*YSAVED
                    |            +-%S01X24----*YSAVED
                    +--SEC002-----S02X00----*YSAVED
                    |            +-%S02X01----*YSAVED
                    +--SEC003-----S03X00----*YSAVED
                    |            +-%S03X01----*YSAVED
                    |            +-%S03X02----*YSAVED
                    |            +-%S03X03----*YSAVED
                    |            +-%S03X04----*YSAVED
                    |            +-%S03X05----*YSAVED
                    |            +-%S03X06----*YSAVED
                    |            +-%S03X07----*YSAVED
                    |            +-%S03X08----*YSAVED
                    |            +-%S03X12----*YSAVED
                    |            +-%S03X13----*YSAVED
                    |            +-%S03X14----*YSAVED
                    |            +-%S03X15----*YSAVED
                    |            +-%S03X16----*YSAVED
                    |            +-%S03X17----*YSAVED
                    |            +-%S03X18----*YSAVED
                    |            +-%S03X19----*YSAVED
                    |            +-%S03X20----*YSAVED
                    |            +-%S03X21----*YSAVED
                    |            +-%S03X22----*YSAVED
                    |            +-%S03X29----*YSAVED
                    |            +-%S03X24----*YSAVED
                    |            +-%S03X25----*YSAVED
                    |            +-%S03X26----*YSAVED

=ANALYSIS/77= ** TREE STRUCTURE **  ENTRY POINT = YSAVEF          DATE  1989/10/09(MONDAY)     TIME 13:34:57  PAGE 0022

YSAVEF-----YSCNTL-----SEC003----%S03X31----*YSAVED
                    |            +-*YMINUS
                    |            +-%S03X29----*YSAVED
                    |            +-%S03X30----*YSAVED
                    +-*ABS
                    +--SEC004-----S04X00----*YSAVED
                    |            +-%S04X01----*YSAVED
                    |            +-%S04X02----*YSAVED
                    |            +-%S04X03----*YSAVED
                    |            +-%S04X04----*YSAVED
                    |            +-%S04X05----*YSAVED
                    |            +-%S04X06----*YSAVED
                    |            +-%S04X07----*YSAVED
                    |            +-%S04X08----*YSAVED
                    |            +-%S04X09----*YSAVED
                    |            +-%S04X10----*YSAVED
                    |            +-%S04X11----*YSAVED
                    |            +-%S04X12----*YSAVED
                    +--SEC005-----S05X00----*YSAVED
                    |            +-%S05X01----*YSAVED
                    |            +-%S05X02----*YSAVED
                    |            +-%S05X03----*YSAVED
                    |            +-%S05X04----*YSAVED
                    |            +-%S05X05----*YSAVED
                    |            +-%S05X06----*YSAVED
                    |            +-%S05X07----*YSAVED
                    |            +-%S05X08----*YSAVED
                    |            +-%S05X09----*YSAVED
                    |            +-%S05X10----*YSAVED
                    |            +-%S05X11----*YSAVED
                    |            +-%S05X12----*YSAVED
                    +--SEC006-----S06XXX----*YSAVED
                    |            +-%S06X01----*YSAVED
                    |            +-%S06X02----*YSAVED
                    |            +-%S06X03----*YSAVED
                    |            +-%S06X04----*YSAVED
                    |            +-%S06X00----*YSAVED
                    |            +-%S06X05----*YSAVED
                    +--SEC007-----S07X00----*YSAVED
                    |            +-%S07X01----*YSAVED
                    |            +-%S07X02----*YSAVED
                    |            +-%S07X03----*YSAVED
                    |            +-*ABS
                    |            +-%S07X04----*YSAVED
                    |            +-%S07X05----*YSAVED
                    |            +-%S07X06----*YSAVED
                    |            +-%S07X07----*YSAVED
                    |            +-%S07X08----*YSAVED
                    |            +-%S07X09----*YSAVED
                    |            +-%S07X10----*YSAVED
                    |            +-%S07X11----*YSAVED
                    |            +-%S07X12----*YSAVED
                    |            +-%S07X13----*YSAVED
                    |            +-%S07X14----*YSAVED
                    |            +-%S07X15----*YSAVED
                    |            +-%S07X16----*YSAVED
                    |            +-%S07X17----*YSAVED
                    |            +-%S07X18----*YSAVED
```

Fig. 11.12 Tree Structure of SRACWRT

```
=ANALYSIS/77= ** TREE STRUCTURE **  ENTRY POINT = %SAVEF          DATE  1989/10/09(MONDAY)     TIME 13:34:57  PAGE 0023

%SAVEF-----%SCNTL-----SEC007----%S07X19----*%SAVED
                  |           +-%S07X20----*%SAVED
                  |           +-%S07X21----*%SAVED
                  |           +-%S07X22----*%SAVED
                  |           +-%S07X23----*%SAVED
                  |           +-%S07X24----*%SAVED
                  |           +-%S07X25----*%SAVED
                  |           +-%S07X26----*%SAVED
                  +--SEC008-----S08X00----*%SAVED
                  |           +-%S08X01----*%SAVED
                  |           +-%S08X02----*%SAVED
                  +--SEC009-----S09X00----*%SAVED
                  |           +-%S09X01----*%SAVED
                  |           +-%S09X02----*%SAVED
                  |           +-%S09X03----*%SAVED
                  |           +-%S09X04----*%SAVED
                  |           +-%S09X05----*%SAVED
                  |           +-%S09X06----*%SAVED
                  |           +-%S09X07----*%SAVED
                  +--SEC010-----S10X00----*%SAVED
                  |           +-%S10X01----*%SAVED
                  |           +-%S10X02----*%SAVED
                  |           +-%S10X03----*%SAVED
                  +--SEC012-----S12X00----*%SAVED
                              +-%S12X01----*%SAVED
```

```
=ANALYSIS/77=     THE CLASSIFICATION OF FORTRAN STATEMENTS          DATE  1989/10/09(MONDAY)     TIME 13:34:57  PAGE 0024

=============================================================================================================
|          T H E   C L A S S I F I C A T I O N   O F   F O R T R A N   S T A T E M E N T S                 |
=============================================================================================================
|     COMMENT              1001( 10.34 %)                    X     WAIT                    0(  0.0  %)      |
|     CONTINUATION         1398( 14.44 %)                    X     INQUIRE                 0(  0.0  %)      |
|     ASSIGNMENT             20(  0.21 %)                    X     COMPLEX                 0(  0.0  %)      |
|     ASSIGN                  0(  0.0  %)                    X     LOGICAL                 0(  0.0  %)      |
|     GO TO                   0(  0.0  %)                    X     INTEGER               213(  2.20 %)      |
|     ASSIGNED GO TO          0(  0.0  %)                    X     CHARACTER             345(  3.56 %)      |
|     COMPUTED GO TO          0(  0.0  %)                    X     DOUBLE PRECISION        0(  0.0  %)      |
|     ARITHMETIC IF           0(  0.0  %)                    X     REAL                   52(  0.54 %)      |
|     LOGICAL IF            228(  2.36 %)                    X     EQUIVALENCE           5226( 53.99 %)      |
|          ASSIGNMENT         0   PRINT              0      X     DATA                    0(  0.0  %)      |
|          ASSIGN             0   PUNCH              0      X     NAMELIST                0(  0.0  %)      |
|          GO TO             20   REWIND             0      X     IMPLICIT                0(  0.0  %)      |
|          ASSIGNED GO TO     0   FIND               0      X     PARAMETER              52(  0.54 %)      |
|          COMPUTED GO TO     0   WAIT               0      X     COMMON                342(  3.53 %)      |
|          ARITHMETIC IF      0   CALL              27      X     SAVE                    0(  0.0  %)      |
|          OPEN               0   RETURN             0      X     DIMENSION               1(  0.01 %)      |
|          CLOSE              0   STOP               0      X     EXTERNAL                0(  0.0  %)      |
|          READ               0   PAUSE              0      X     INTRINSIC               0(  0.0  %)      |
|          WRITE              0   DECODE             0      X     DEFINE FILE             0(  0.0  %)      |
|          BACKSPACE          0   ENCODE             0      X     PROGRAM                 0(  0.0  %)      |
|          ENDFILE            0   INQUIRE            0      X     BLOCKDATA               0(  0.0  %)      |
|     IF (...) THEN          32(  0.33 %)                    X     FUNCTION                0(  0.0  %)      |
|     DO                      0(  0.0  %)                    X     SUBROUTINE             26(  0.27 %)      |
|     DO   UNTIL(...)         0(  0.0  %)                    X     ENTRY                 130(  1.34 %)      |
|     DO   WHILE(...)         0(  0.0  %)                    X     CALL                  331(  3.42 %)      |
|     ELSE                    2(  0.02 %)                    X     RETURN                168(  1.74 %)      |
|     ELSE IF                 9(  0.09 %)                    X     STOP                    0(  0.0  %)      |
|     END IF                 32(  0.33 %)                    X     PAUSE                   0(  0.0  %)      |
|     FORMAT                  0(  0.0  %)                    X     END                    26(  0.27 %)      |
|     OPEN                    1(  0.01 %)                    X     CONTINUE               17(  0.18 %)      |
|     CLOSE                   1(  0.01 %)                    X     DECODE                  0(  0.0  %)      |
|     READ                    0(  0.0  %)                    X     ENCODE                  0(  0.0  %)      |
|     WRITE                  27(  0.28 %)                    X     DEBUG                   0(  0.0  %)      |
|     BACKSPACE               0(  0.0  %)                    X     AT                      0(  0.0  %)      |
|     ENDFILE                 0(  0.0  %)                    X     DISPLAY                 0(  0.0  %)      |
|     PRINT                   0(  0.0  %)                    X     INIT                    0(  0.0  %)      |
|     PUNCH                   0(  0.0  %)                    X     TRACE                   0(  0.0  %)      |
|     REWIND                  0(  0.0  %)                    X     -- ROQUE --             0(  0.0  %)      |
|     FIND                    0(  0.0  %)                    X     NCHARACTER (JEF)        0(  0.0  %)      |
=============================================================================================================

                                TOTAL STATEMENTS =  9680
```

Fig. 11.12 (Continued)

12. おわりに

ＪＤＩＳＳシステムの開発目的は，誰でも簡単に会話型入力データ作成システムを作成することが出来るようにすることであった。簡単な入力方法で自動的に作成するという目的を達成するためには，ある程度型にはまった処理でなければならない。例えば，画面の反転やアンダーラインを任意に引くことなど，本来ＩＰＦが持つ機能であっても使用できないものもある。

汎用化のディメリットとして，対象とする原子力コードの特徴に合った細かい要求を100％満足することが難しいことがあげられる。実際ＪＤＩＳＳによって作成された会話型入力データ作成システムのみを作る場合において，まわりくどいと感じさせる部分もいくつか見受けられる。例えば，セクション選択機能や中間ファイル出力機能等の諸機能を実行する場合の制御文や定義文といったＪＤＩＳＳＰＨ２の入力データに当たる部分としてＦＯＲＴＲＡＮプログラムを含むかなりの情報を必要とする。

しかし，ＪＤＩＳＳを使用することなく原子力コード開発整備者がＩＰＦを使用して総てをコーディングすることと比較してみると，その負荷は大きく軽減されている。そういった面からも，ＪＤＩＳＳによって簡単に会話型入力データ作成システムが作成できることの意義は大きい。

さらに，本システムのうち必要な機能，例えばディスプレーと会話的にメニュー画面を作成してメニュー定義体を自動作成する機能，或いはその制御サブルーチンを自動的に作成する機能，等を利用してもう少し単純な方法で会話型入力データ作成システムを作ることも可能である。

ＪＤＩＳＳをＳＲＡＣに適用した経験から評価すれば，ＪＤＩＳＳＰＨ１についてはほぼ当初の目的を達しているが，ＪＤＩＳＳＰＨ２では画面の制御，エラーチェック，中間ファイルへの格納部分など汎用化が難しい部分に関しては，作成した補助ツールを駆使しても煩雑な定義が必要になってしまい，改良の余地を残している。しかし，コードの開発者がこれらの点を慎重に処理して画面制御をうまく定義できれば，それによって原子力コードの利用者にとって使い易い会話型入力データ作成システムが提供できる。今後，ＳＲＡＣコード以外にも適用し，不都合部分の改善を図って行けば，幅広く使われるシステムとなることが期待できる。

また，ＳＵＮ－４などワークステーションのマルチウインド機能を使用した場合の処理も今後進める予定である。この場合，メニュー画面作成を目的とするＪＤＩＳＳＰＨ１の部分は軽くなるが，画面制御ＪＤＩＳＳＰＨ２の部分は，今回と同様のものが必要となる。

謝　　辞

本報告書を投稿するに当たり，計算センター・秋元正幸氏に貴重な助言をいただきましたことを感謝します。

参　考　文　献

(1) 富士通：ＩＰＦ使用手引書　Ｖ10用，ＦＡＣＯＭ　ＯＳⅣ　マニュアル（1986).

(2) K. Tsuchihashi, et. al ：Revised ＳRAC Code System. JAERI 1302 (1986).

(3) 福岡久雄，他：ユーザインターフェース設計ツール，文書処理とヒューマンインターフェース　17－3（情報処理学会研究会）（1988).

(4) 井上修二，他(編)：ＪＳＳＬ（原研版・科学用サブルーチン・ライブラリ）マニュアル第3版，ＪＡＥＲＩ－Ｍ 82－095（1982).

謝　　辞

本報告書を投稿するに当たり，計算センター・秋元正幸氏に貴重な助言をいただきましたことを感謝します。

参　考　文　献

(1) 富士通：ＩＰＦ使用手引書　Ｖ10用，ＦＡＣＯＭ　ＯＳⅣ　マニュアル（1986).

(2) K. Tsuchihashi, et. al ：Revised ＳＲＡＣ Code System. ＪAERI 1302（1986).

(3) 福岡久雄，他：ユーザインターフェース設計ツール，文書処理とヒューマンインターフェース　17－3（情報処理学会研究会）（1988).

(4) 井上修二，他(編)：ＪＳＳＬ（原研版・科学用サブルーチン・ライブラリ）マニュアル第3版，ＪＡＥＲＩ－Ｍ 82－095（1982).

付録　ＳＲＡＣコードの場合の ＪＤＩＳＳＰＨ２の制御文例

（Example of Control Description for JDISSPH2 in SRAC Code）

```
C
C***** MAIN    1989 09/19
C
*INCLUDE ¥INC¥,INSOURCE
 3000 CONTINUE
C--------
      CALL ¥READX(IRT)
C--------
     CALL BLOCK (IRT)
C--------
      IF ( IRT .EQ. 2  .OR. IRT .EQ. 14 ) GO TO 3000
      END
C **********
C * BLK001 *
C **********
C
C---------- SUBROUTINE BLK001
      SUBROUTINE BLK001(INO,¥IRT )
C
*INCLUDE ¥INC¥
      CHARACTER *  6 BNAM(13)
      DATA  (BNAM(I),I=01,03) /'GENERL' ,'USERF ' ,'COLLIS' /
      DATA  (BNAM(I),I=04,06) /'ANISN'  ,'TWOTRN' ,'TUD'    /
      DATA  (BNAM(I),I=07,09) /'CITATN' ,'MATRAL' ,'REACT'  /
      DATA  (BNAM(I),I=10,13) /'BURNUP' ,'MCROSS' ,'PEACO' ,'SAVE' /
      CHARACTER * 40 BCOM(13)
      DATA  BCOM(01) / 'General Control                                ' /
      DATA  BCOM(02) / 'User"s Microscopic Cross Section               ' /
      DATA  BCOM(03) / 'Collision Probability Method                   ' /
      DATA  BCOM(04) / 'ANISN    : One Dimensional SN Transport        ' /
      DATA  BCOM(05) / 'TWOTRAN  : Two Dimensional SN Transport        ' /
      DATA  BCOM(06) / 'TUD      : One Dimensional Diffusion           ' /
      DATA  BCOM(07) / 'CITATION : Two Dimensional Diffusion           ' /
      DATA  BCOM(08) / 'Material Specification                         ' /
      DATA  BCOM(09) / 'Reaction Rate Calculation                      ' /
      DATA  BCOM(10) / 'Cell Burn-up Calculation                       ' /
      DATA  BCOM(11) / 'MCROSS                                         ' /
      DATA  BCOM(12) / 'PEACO                                          ' /
      DATA  BCOM(13) / 'saved file name specification                  ' /
      DIMENSION IU (13)
C
      N     = 0
C-----  GENERAL CONTROL
C
C
      IF  (    (  IABS(IC2) .EQ. 1  )  .OR.
     #         (  IC12      .EQ. 1  )  .OR.
     #         (  IC3       .GT. 0  )    )  THEN
        IC1 = 1
      ELSE
        IC1 = 0
      ENDIF
      IF  (  IC2  .NE.  0  )  THEN
        IC6 = 1
        IC7 = 4
      ELSE
        IC6 = 0
        IC7 = 0
```

```
      ENDIF
      IF    (  NERT .EQ.  1  )  THEN
        IC15 = -1
      ELSEIF(  NERT .GE.  2  )  THEN
        IC15 =  1
      ELSE
        IC15 = 0
      ENDIF
C-----
      N     = N + 1
      IU(N) = 1
C-----  USER MICROSCOPIC CROSS SECTION
      IF ( (FLAG1.EQ.'Y') .AND. (FLAG11.EQ.'N') ) THEN
        N     = N + 1
        IU(N) = 2
      ENDIF
C-----  COLLISION PROBALLITY CROSS SECTION
      IF ( (IC11.EQ.0) .AND. (IC1.EQ.1) ) THEN
        N     = N + 1
        IU(N) = 3
      ENDIF
C-----  ANISN
      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.2).OR.(ABS(IC12).EQ.2)) ) THEN
        N     = N + 1
        IU(N) = 4
      ENDIF
C-----  TWOTRAN
      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.3).OR.(ABS(IC12).EQ.3)) ) THEN
        N     = N + 1
        IU(N) = 5
      ENDIF
C-----  TUD
      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.4).OR.(ABS(IC12).EQ.4)) ) THEN
        N     = N + 1
        IU(N) = 6
      ENDIF
C-----  CITATION
      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.5).OR.(ABS(IC12).EQ.5)) ) THEN
        N     = N + 1
        IU(N) = 7
      ENDIF
C------  MATELIAL SPACIFICATION
      N     = N + 1
      IU(N) = 8
C------  reaction rate
      IF ( IC18 .EQ. 1 ) THEN
        N     = N + 1
        IU(N) = 9
      ENDIF
C------  CELL BURNUP
      IF (  IC20 .EQ. 1 ) THEN
        N     = N + 1
        IU(N) =  10
      ENDIF
C------  MCROSS
C     IF (  IC8  .EQ. 1 ) THEN
C       N     = N + 1
C       IU(N) = 11
```

```
C      ENDIF
C------  PEACO
       IF (  IC5  .EQ. 1 ) THEN
         N      = N + 1
         IU(N) = 12
       ENDIF
C------  DATA SAVE
       N      = N + 1
       IU(N) = 13
C------------- BLOCK MANU
       CALL ¥BLKS(INO , N ,BNAM ,BCOM ,IU , ¥IRT )
C
       RETURN
       END
C **********
C * GENERL *
C **********
       SUBROUTINE   GENERL( ¥IRT )
C  GENERAL CONTROL
*INCLUDE  ¥INC¥
       CHARACTER * 80 CEM ,CEM2 ,CEMX
       DATA  CEM  /' INPUT "Y" OR "N" ' /
       DATA  CEM2 /' INPUT "E" OR "F" ' /
C
C       M E N U    P A T H
C
       CALL  SO1NOO
 3001  CALL  SO1NO1
C
C>>>>> FIRST CASE OR NOT .
C
       IF    ( FLAG1 .EQ. 'Y' )  THEN
 3002      CALL  SO1NO2
C--------------- ADD NEDAC 1989 08/08
           IF ( INDEX('YN',FLAG11) .EQ. 0 )  THEN
               ¥IRT = -1
               CALL ¥EROUT(CEM ,1 ,IRT )
               GO TO 3002
           ENDIF
C------ *** DEFINE IC11 ****
           IC11 = 0
C-----------------------------------
       ELSEIF( FLAG1 .EQ. 'N' )  THEN
 3003      CALL  SO1NO3
           IF    ( FLAG12 .EQ. 'Y' )  THEN
               IC11 = 1
           ELSEIF( FLAG12 .EQ. 'N' )  THEN
               IC11 = 0
           ELSE
               ¥IRT = -1
               CALL ¥EROUT(CEM ,1 ,IRT )
               GO TO  3003
           ENDIF
       ELSE
               ¥IRT = -1
               CALL ¥EROUT(CEM ,1 ,IRT )
           GO TO  3001
       ENDIF
```

```
C
C>>>>> MACRO CROSS SECTION WILL ?
C
 3004 CONTINUE
      CALL  S01N04
      IF       ( FLAG2 .EQ. 'Y' )  THEN
          CALL  S01N05
          CALL  S01N06
      ELSEIF  ( FLAG2 .EQ. 'N' )  THEN
          IC16 = 0
          IC3  = 0
      ELSE
          ¥IRT = -1
          CALL ¥EROUT(CEM ,1 ,IRT )
          GO TO 3004
      ENDIF
C
C>>>>> CELL CALUCURATION ? SET IC2 IC5 IC9 IC17 IC20
C
 3007 CONTINUE
      CALL  S01N07
C
      IF  ( FLAG3 .EQ. 'Y' )  THEN
 3008     CONTINUE
          CALL  S01N08
          IF    ( FLAG31 .EQ. 'F' )  THEN
              CALL  S01N09
          ELSEIF( FLAG31 .EQ. 'E' )  THEN
              IC9 = 0
          ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO  3008
C-------------------------------------
          ENDIF
C--------- IC20
          CALL  S01N10
C--------- FLAG31 : SOLV METHOD
 3011     CONTINUE
          CALL  S01N11
          IF  ( FLAG31 .EQ. 'F' )  THEN
              IC2  = IFLG33
              IC12¥= IC12
              IC12 =  0
          ELSEIF ( FLAG31 .EQ. 'E' ) THEN
              IC2  = 0
              IC12 = IFLG33
          ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM2 ,1 ,IRT )
              GO TO 3011
          ENDIF
C==============
          IF  ( IABS(IFLG33)  .EQ.  1  )  THEN
              IF  ( FLAG31 .EQ.  'F' )  THEN
                  CALL  S01N12
                  CALL  S01N13
              ELSE
```

```
                IC5 = 0
                IF  ( IFLG33 .GE. 1 )  THEN
                  CALL  SO1N12
                ELSE
                  IC17 = 1
                ENDIF
              ENDIF
            ENDIF
          ELSEIF  ( FLAG3 .EQ. 'N' )  THEN
            IC2  = 0
            IC5  = 0
            IC9  = 0
            IC12¥= IC12
            IC12 = 0
            IC17 = 1
          ELSE
C--------------- ADD NEDAC 1989 08/08
                ¥IRT = -1
                CALL ¥EROUT(CEM ,1 ,IRT )
C-------------------------------------
            GO TO  3007
          ENDIF
C
C>>>>> CORE CALUCURATION  ? < IC12 >
C
          IF ( (FLAG3.NE.'Y') .OR. (FLAG31.NE.'E')  ) THEN
 3014       CONTINUE
            CALL  SO1N14
            IF    ( FLAG4  .EQ. 'Y' )  THEN
              IC12 = IC12¥
 3015         CONTINUE
              CALL  SO1N15
              IF ( IC12 .EQ. IC2) THEN
                CEMX = ' INVALID SELECTION. SAME NUMBER OF IC2 '
                ¥IRT = -1
                CALL ¥EROUT(CEMX ,1 ,IRT )
                GO TO  3015
              ENDIF
            ELSEIF( FLAG4  .EQ. 'N' )  THEN
            ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO  3014
            ENDIF
          ENDIF
C
C>>>>>  FEW GROUP CALCULATION IC10 IC13
C
          LUMP¥ = 0
          IF    (    (  IC2    .EQ.   0  ) .AND.
         &           (  IC12   .EQ.   0  )      )    THEN
              LUMP¥ = 1
              IC10  = 0
              IC13  = 0
          ELSEIF(    (  IC2    .NE.   0  ) .AND.
         &           (  IC12   .EQ.   0  )      )    THEN
 3018         CALL  SO1N18
              IF    (  FLAG51  .EQ.  'Y'  )   THEN
```

```
            LUMP¥ = 0
            IC10  = 1
            IC13  = 0
          ELSEIF(  FLAG51  .EQ.  'N'  )   THEN
            LUMP¥ = 1
            IC10  = 0
            IC13  = 0
          ELSE
            ¥IRT = -1
            CALL ¥EROUT(CEM ,1 ,IRT )
            GO TO  3018
          ENDIF
      ELSEIF(    (  IC2    .EQ.   0  ) .AND.
     &           (  IC12   .NE.   0  )      )    THEN
          LUMP¥ = 0
 3019     CALL  S01N19
          IF    (  FLAG52  .EQ.  'Y'  )   THEN
            IC10 = 1
            IC13 = 0
          ELSEIF(  FLAG52  .EQ.  'N'  )   THEN
30182       CALL  S01N18
            IF    (  FLAG51  .EQ.  'Y'  )   THEN
              IC10 = 0
              IC13 = 1
            ELSEIF(  FLAG51  .EQ.  'N'  )   THEN
              IC10 = 0
              IC13 = 0
            ELSE
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
              GO TO  30182
            ENDIF
          ELSE
C--------------- ADD NEDAC 1989 08/08
            ¥IRT = -1
            CALL ¥EROUT(CEM ,1 ,IRT )
C-----------------------------------
            GO TO  3019
          ENDIF
      ENDIF
C
C>>>>> TRANSPORT CROSS SECTION
C
      IF  (  ( FLAG51 .EQ. 'Y' )  .OR.
     #       ( FLAG52 .EQ. 'N' )    )  THEN
 3017     CALL  S01N17
          IF    ( FLAG53 .EQ. 'I' )  THEN
          ELSEIF( FLAG53 .EQ. 'C' )  THEN
            IC17 =  - IC17
          ELSE
C--------------- ADD NEDAC 1989 08/08
            ¥IRT = -1
            CALL ¥EROUT(CEM ,1 ,IRT )
C-----------------------------------
            GO TO  3017
          ENDIF
      ENDIF
C
```

```
C>>>>> FLAG5
C
      IF  (   (  FLAG1  .EQ.  'Y'  )  .AND.
     &        (  LUMP¥  .EQ.   1   )    )  THEN
 3016     CALL  SO1N16
          IF   (  FLAG5  .EQ.  'Y'  )  THEN
              IC13 = 1
          ELSEIF(  FLAG5  .EQ.  'N'  )  THEN
              IC13 = 0
          ELSE
C--------------- ADD NEDAC 1989 08/08
              ¥IRT = -1
              CALL ¥EROUT(CEM ,1 ,IRT )
C-------------------------------------
              GO TO  3016
          ENDIF
       ELSE
          FLAG5  =  'N'
       ENDIF
C
C>>>>> IC18 IC4 BSQ
C
       IF  (  (  IC2  .EQ.  0  )  .AND.
      &       (  IC12 .EQ.  0  )   )  THEN
         CALL  SO1N20
       ELSE
         IC18 = 0
       ENDIF
C
       CALL  SO1N21
       CALL  SO1N27
C
C>>> EFL ETC....
C
       CALL  SO1N22
       IF ( (FLAG1.EQ.'Y').AND.(FLAG11.EQ.'N') .AND. (IC4.EQ.0)) THEN
         CALL SO1N25
         NET = 0
       ELSEIF ( (FLAG1.EQ.'Y').AND.(FLAG11.EQ.'N').AND.(IC4.EQ.1) ) THEN
         CALL SO1N25
         CALL SO1N26
       ELSE
         NEF = 0
         NET = 0
       ENDIF
C------  NERF
       IF  ( (FLAG1.EQ.'Y') .AND. ( (IC10.NE.0).OR.(IC13.NE.0))
      1      .AND.  ( NERF  .NE.  0 )  ) THEN
           CALL SO1N23
       ELSE
           NERF  =  0
       ENDIF
C------ NERT
       IF  (         (FLAG1.EQ.'Y')
      1    .AND.   ( (IC10 .NE.0) .OR. (IC13.NE.0) )
      2    .AND.     (IC4  .NE.0) .AND. (NERT.NE.0)      )   THEN
          CALL SO1N24
       ELSE
```

```
        NERT  =  0
       ENDIF
C
       IF  (    (  IABS(IC2) .EQ. 1  )  .OR.
      #         (  IC12      .EQ. 1  )  .OR.
      #         (  IC3       .GT. 0  )    )  THEN
         IC1 = 1
       ELSE
         IC1 = 0
       ENDIF
       IF  (  IC2  .NE.  0  )  THEN
         IC6 = 1
         IC7 = 4
       ELSE
         IC6 = 0
         IC7 = 0
       ENDIF
       IF    (  NERT .EQ.  1  )  THEN
         IC15 = -1
       ELSEIF(  NERT .GE.  2  )  THEN
         IC15 =  1
       ELSE
         IC15 = 0
       ENDIF
C
       CALL  SO1DSP
       RETURN
       END
C **********
C * USERF  *
C **********
      SUBROUTINE USERF  ( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL SO2NO1
C
      RETURN
      END
C **********
C * COLLIS *
C **********
      SUBROUTINE COLLIS ( ¥IRT )
*INCLUDE ¥INC¥
      DIMENSION IW (100)
C*
C*     P.I.J.
C*
      CALL  SO3NO1
      CALL  SO3NO2
      CALL  SO3NO3
      IF  ( IBOUND .EQ. 1 )          CALL  SO3NO4
      IF  ( IGT    .GE. 9 )          CALL  SO3NO5
      IF  ( IGT    .NE. 2 )          CALL  SO3NO6
      IF  ( IGT    .GE. 9 )          CALL  SO3NO7
      CALL  SO3NO8
      IF ( NR  .LT. NZ  ) CALL  SO3N12
      IF ( NRR .LT. NR  ) CALL  SO3N13
      IF ( NXR .LT. NRR ) CALL  SO3N14
```

```
      CALL  S03N15
      IF  ( (IGT.EQ.10) .OR.  (IGT.EQ.14)  )      CALL  S03N16
C
      CALL  S03N17
C
      IF  ( ((IGT.EQ.11).OR.(IGT.EQ.12)) .AND. (NY.GT.0) ) CALL S03N18
C
      IF  (  (IGT.EQ.13)                  .AND. (NY.GT.1) ) CALL S03N19
C
      IF  ( (  IGT  .EQ.  9   ) .OR.
     &      (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 14   )           )  CALL  S03N20
C
      IF  ( (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   )       )  CALL  S03N21
C
      IF  (    IGT  .EQ. 13           )  CALL  S03N22
C
      IF  ( (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   )       )  CALL  S03N23
C
      IF  (    IGT  .EQ. 13           )  CALL  S03N24
C
      IF  ( (  IGT  .EQ.  9   ) .OR.
     &      (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 14   )       )  CALL  S03N25
C
      IF  ( (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   ) .OR.
     &      (  IGT  .EQ. 13   )       )  CALL  S03N26
C
      IF ( IPLOT .NE. 0 ) CALL S03N31
C
      CALL ¥MINUS( 'MAR   ',IANS ,NRR ,IW ,IRT )
      IF ( IANS .EQ. 0 )  THEN
        CALL  S03N29
        CALL  S03N30
      ENDIF
C
      RETURN
      END
C **********
C * ANISN  *
C **********
      SUBROUTINE ANISN( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL S04N01
      CALL S04N02
      CALL S04N03
      CALL S04N04
      CALL S04N05
      CALL S04N06
      CALL S04N07
      CALL S04N08
      CALL S04N09
      IF ( IDAT2 .GT. 0 ) CALL S04N10
```

```
      CALL S04N11
      CALL S04N12
C
      RETURN
      END
C **********
C * TWOTRN *
C **********
      SUBROUTINE TWOTRN( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL  S05N01
      CALL  S05N02
      CALL  S05N03
      CALL  S05N04
      CALL  S05N05
      CALL  S05N06
      CALL  S05N07
      CALL  S05N08
      CALL  S05N09
      CALL  S05N10
      CALL  S05N11
      CALL  S05N12
      RETURN
      END
C **********
C * TUD    *
C **********
      SUBROUTINE TUD(¥IRT )
*INCLUDE ¥INC¥
C
      CALL S06N01
      CALL S06N02
      CALL S06N03
      IF (NXRO .NE. 0) CALL S06N00
      CALL S06N04
      CALL S06N05
C
      RETURN
      END
C **********
C * CITATN *
C **********
      SUBROUTINE CITATN( ¥IRT )
*INCLUDE ¥INC¥
C
C  CITATION
      CALL  S07N01
      CALL  S07N02
      IF  ( ABS(IDC) .EQ.  3 )    CALL  S07N03
      IF  ( NMC      .LT.  0 )  THEN
          CALL  S07N04
          CALL  S07N05
      ENDIF
      CALL  S07N06
      CALL  S07N07
      CALL  S07N08
C
```

```
      CALL  S07N09
      IF  ( NUAC5  .GE.  6 )    CALL  S07N10
      IF  ( NUAC5  .GE. 11 )    CALL  S07N11
C
      CALL  S07N12
C
      CALL  S07N13
      CALL  S07N14
      IF  ( NUAC5  .GE.  6 )    THEN
          CALL  S07N15
          CALL  S07N16
      ELSE
          NYY = 1
      ENDIF
      IF  ( NUAC5  .GE. 11 )    THEN
          CALL  S07N17
          CALL  S07N18
      ELSE
          NZZ = 1
      ENDIF
C
      CALL  S07N19
      CALL  S07N20
      CALL  S07N21
      IF    ( INDI  .EQ.  1 )  THEN
          CALL  S07N22
      ELSEIF( INDI  .EQ.  2 )  THEN
          CALL  S07N23
      ELSEIF( INDI  .EQ.  3 )  THEN
          CALL  S07N24
      ENDIF
      CALL  S07N25
      CALL  S07N26
C
      RETURN
      END
C **********
C * MATRAL *
C **********
      SUBROUTINE MATRAL( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL S08N01
      CALL S08N02
C
      RETURN
      END
C **********
C * REACT  *
C **********
      SUBROUTINE REACT( ¥IRT )
*INCLUDE ¥INC¥
C  REACTION RATE CALCULATION
      CALL  S09N01
      IF  ( IOPT1 .GT. 0 )    CALL  S09N02
      IF  ( IOPT2 .GT. 0 )    CALL  S09N03
      CALL  S09N04
      IF  ( IOPT3 .GT. 0 )    CALL  S09N05
```

```
      CALL  S09N06
      IF  ( MREC  .LT. 0 )    CALL  S09N07
C
      RETURN
      END
C **********
C * BURNUP *
C **********
      SUBROUTINE BURNUP( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL S10N01
      CALL S10N02
      CALL S10N03
C
      RETURN
      END
C **********
C * MCROSX *
C **********
      SUBROUTINE MCROSX( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL S11N01
      CALL S11N02
      CALL S11N03
C
      RETURN
      END
C **********
C * PEACO  *
C **********
      SUBROUTINE PEACO( ¥IRT )
*INCLUDE ¥INC¥
C
      CALL S12N01
C
      RETURN
      END
```

中間ファイルへデータを格納する¥ＳＡＶＥＦ

```
      C   ***********
      C   * ¥SAVEF  *
      C   ***********
000001        SUBROUTINE ¥SAVEF( ¥C1       ,¥C2                                )
      C*********************************************************************
      C*
      C*   ¥SAVEF :  DATA SAVE TO MIDWAY FILE
      C*
      C*    THIS SUBROIUTINE WAS CREATED BY JOISS SYSTEM(1989 NEDAC).
      C*                                  CREATE DATE : 89-08-08
      C*
      C*   <ARGUMENT>
      C*
      C*   ¥C1     : FILE NAME
      C*   ¥C2     : MEMBER NAME
      C*
      C*********************************************************************
      C
      *INCLUDE ¥INC¥
      *INCLUDE ¥IPFDAT
      C
000252        CHARACTER * 45  ¥C1
000253        CHARACTER * 45  ¥OPFIL
000254        CHARACTER *  8  ¥C2  ,¥C3
      C
      C>>>>>  DATA SAVE TO MIDWAY FILE
      C
000255        IOX = 66
000256        OPEN (66 ,FILE=¥OPFIL(¥C1 ,¥C2) )
000257        CALL  DATE (¥C3)
000258        WRITE(66,'(A)') '*  '
000259        WRITE(66,'(A,A)') '* MIDWAY FILE  CREATED DATE : ', ¥C3
000260        WRITE(66,'(A)') '*  '
000261        CALL ¥SCNTL ( IOX )
000262        CLOSE (66)
      C
      C>>>>>  RETURN
      C
000263   9999 CONTINUE
000264        RETURN
000265        END
```

```
      C **********
      C * ¥SCNTL * CONTROL
      C **********
000001      SUBROUTINE  ¥SCNTL(IOX)
      *INCLUDE ¥INC¥
      C
      C     CONTROL ROUTINE OF SRAC CALLED BY MAIN (DIMENSION CONTROLLER)
      C
000240   1000 CONTINUE
      C-----  GENERAL CONTROL
      C-    CALL INPUT
000241      CALL SEC001 ( IOX )
      C-----  USER MICROSCOPIC CROSS SECTION
000242      IF ( FLAG11 .EQ. 'N' ) THEN
      C-      CALL USERF
000243        CALL         SEC002 ( IOX )
000244      ENDIF
      C-----  COLLISION PROBALLITY CROSS SECTION
000245      IF ( (IC11.EQ.0) .AND. (IC1.EQ.1) ) THEN
      C----   CALL PIJIN
000246        CALL         SEC003 ( IOX )
000247      ENDIF
      C-----  ANISN
000248      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.2).OR.(ABS(IC12).EQ.2)) ) THEN
      C       CALL ANISN
000249        CALL         SEC004 ( IOX )
000250      ENDIF
      C-----  TWOTRAN
000251      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.3).OR.(ABS(IC12).EQ.3)) ) THEN
      C       CALL TWOTRN
000252        CALL         SEC005 ( IOX )
000253      ENDIF
      C-----  TUD
000254      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.4).OR.(ABS(IC12).EQ.4)) ) THEN
      C       CALL TUD
000255        CALL         SEC006 ( IOX )
000256      ENDIF
      C-----  CITATION
000257      IF ( (IC11.EQ.0).AND.((ABS(IC2).EQ.5).OR.(ABS(IC12).EQ.5)) ) THEN
      C       CALL CITATN
000258        CALL         SEC007 ( IOX )
000259      ENDIF
      C------  MATELIAL SPACIFICATION
      C     CALL INPUT2
000260      CALL         SEC008 ( IOX )
      C------  REACTION RATE
000261      IF ( IC18 .EQ. 1 ) THEN
      C       CALL REACIN
000262        CALL         SEC009 ( IOX )
000263      ENDIF
      C------  CELL BURNUP
000264      IF (  IC20 .EQ. 1 ) THEN
      C       CALL BURNIN
000265        CALL         SEC010 ( IOX )
000266      ENDIF
      C------  MCROSS
      C     IF (  IC8  .EQ. 1 ) THEN
      C       CALL PEACO
      C       CALL         SEC011 ( IOX )
      C     ENDIF
      C------  PEACO
000267      IF (  IC5  .EQ. 1 ) THEN
      C       CALL PEACO
000268        CALL         SEC012 ( IOX )
000269      ENDIF
      C------  DATA SAVE
      C-----------------
000270      RETURN
000271      END
```

```
      SUBROUTINE SEC001( IOX )
*INCLUDE  ¥INC¥
C
C----  SET THE INITIAL VALUE
000240      CALL  S01X00 ( IOX )
C-----------------------------
000241  3001 CALL  S01X01
C
000242      IF    ( FLAG1 .EQ. 'Y' )  THEN
000243          CALL  S01X02
000244      ELSEIF( FLAG1 .EQ. 'N' )  THEN
000245          CALL  S01X03
000246      ENDIF
C
000247      CALL  S01X04
000248      IF  ( FLAG2 .EQ. 'Y' )  THEN
000249          CALL  S01X05
000250          CALL  S01X06
000251      ENDIF
C
000252  3007 CALL  S01X07
C
000253      IF  ( FLAG3 .EQ. 'Y' )  THEN
000254  3008    CALL  S01X08
000255          IF    ( FLAG31 .EQ. 'F' )  THEN
000256              CALL  S01X09
000257          ENDIF
000258          CALL  S01X10
000259          CALL  S01X11
000260          IF  ( IABS(IFLG33)  .EQ.  1  )  THEN
000261              IF  ( FLAG31 .EQ.  'F' )  THEN
000262                  CALL  S01X12
000263                  CALL  S01X13
000264              ELSE
000265                  IF  ( IFLG33 .GE. 1 )  THEN
000266                      CALL  S01X12
000267                  ENDIF
000268              ENDIF
000269          ENDIF
000270      ELSEIF( FLAG3 .EQ. 'N' )  THEN
000271  3014    CALL  S01X14
000272          IF    ( FLAG4  .EQ. 'Y' )  THEN
000273              CALL  S01X15
000274          ENDIF
000275      ENDIF
C
C
000276      LUMP¥ = 0
000277      IF    (    ( IC2    .EQ.   0  ) .AND.
     &           ( IC12   .EQ.   0  )      )    THEN
000278          LUMP¥ = 1
000279          IC10  = 0
000280      ELSEIF(    ( IC2    .NE.   0  ) .AND.
     &           ( IC12   .EQ.   0  )      )    THEN
000281  3018    CALL  S01X18
000282      ELSEIF(    ( IC2    .EQ.   0  ) .AND.
     &           ( IC12   .NE.   0  )      )    THEN
000283  3019    CALL  S01X19
000284          IF(  FLAG52  .EQ.  'N'  )   THEN
000285 30182         CALL  S01X18
000286          ENDIF
000287      ENDIF
C
000288      IF  (  ( FLAG51 .EQ. 'Y' )  .OR.
     #       ( FLAG52 .EQ. 'N' )    )  THEN
000289          CALL  S01X17
000290      ENDIF
C
000291      IF  (   ( FLAG1  .EQ.  'Y'  )  .AND.
     &         ( LUMP¥  .EQ.   1   )    )  THEN
000292  3016    CALL  S01X16
      ENDIF
```

```
000294          IF  (  (  IC2  .EQ.  0  )  .AND.
                &      (  IC12 .EQ.  0  )  )  THEN
000295              CALL  S01X20
000296          ENDIF
          C
000297          CALL  S01X21
          C
          C
000298          CALL  S01X27
          C
000299          IF  (  FLAGS  .EQ.  'Y'  )  THEN
000300              CALL  S01X22
000301              IF ( NEF   .NE.  0 )   CALL S01X25
000302              IF ( NET   .NE.  0 )   CALL S01X26
000303              IF ( NERF  .NE.  0 )   CALL S01X23
000304              IF ( NERT  .NE.  0 )   CALL S01X24
000305          ENDIF
          C
000306          CALL  S01XSP
000307          RETURN
000308          END
```

```
      C ***************
      C * S01X00        *
      C ***************
000001        SUBROUTINE S01X00 ( IOXX)
      *INCLUDE ¥INC¥
000240        IOX = IOXX
000241        WRITE(IOX,'(''$GENERL'')')
000242        RETURN
      C
      C======   < S01XSP   >
000243        ENTRY S01XSP
      C-------------------    IC1
000244        CALL ¥SAVED ( IOX ,'IC1    ' ,IRT )
000245          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC8
000246        CALL ¥SAVED ( IOX ,'IC8    ' ,IRT )
000247          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC15
000248        CALL ¥SAVED ( IOX ,'IC15   ' ,IRT )
000249          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC2
000250        CALL ¥SAVED ( IOX ,'IC2    ' ,IRT )
000251          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC9
000252        CALL ¥SAVED ( IOX ,'IC9    ' ,IRT )
000253          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC16
000254        CALL ¥SAVED ( IOX ,'IC16   ' ,IRT )
000255          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC3
000256        CALL ¥SAVED ( IOX ,'IC3    ' ,IRT )
000257          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC10
000258        CALL ¥SAVED ( IOX ,'IC10   ' ,IRT )
000259          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC17
000260        CALL ¥SAVED ( IOX ,'IC17   ' ,IRT )
000261          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC4
000262        CALL ¥SAVED ( IOX ,'IC4    ' ,IRT )
000263          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC11
000264        CALL ¥SAVED ( IOX ,'IC11   ' ,IRT )
000265          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC18
000266        CALL ¥SAVED ( IOX ,'IC18   ' ,IRT )
000267          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC5
000268        CALL ¥SAVED ( IOX ,'IC5    ' ,IRT )
000269          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC12
000270        CALL ¥SAVED ( IOX ,'IC12   ' ,IRT )
000271          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC19
000272        CALL ¥SAVED ( IOX ,'IC19   ' ,IRT )
000273          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC6
000274        CALL ¥SAVED ( IOX ,'IC6    ' ,IRT )
000275          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC13
000276        CALL ¥SAVED ( IOX ,'IC13   ' ,IRT )
000277          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC20
000278        CALL ¥SAVED ( IOX ,'IC20   ' ,IRT )
000279          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC7
000280        CALL ¥SAVED ( IOX ,'IC7    ' ,IRT )
000281          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IC14
000282        CALL ¥SAVED ( IOX ,'IC14   ' ,IRT )
000283          IF ( IRT .GT. 4 ) GO TO 8000
000284        RETURN
      C======   < S01X01   >
000285        ENTRY S01X01
      C-------------------    FLAG1
000286        CALL ¥SAVED ( IOX ,'FLAG1  ' ,IRT )
000287          IF ( IRT .GT. 4 ) GO TO 8000
000288        RETURN
```

```
C******   < S01X02   >
      ENTRY S01X02
C------------------    FLAG11
      CALL ¥SAVED ( IOX ,'FLAG11' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X03   >
      ENTRY S01X03
C------------------    FLAG12
      CALL ¥SAVED ( IOX ,'FLAG12' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X04   >
      ENTRY S01X04
C------------------    FLAG2
      CALL ¥SAVED ( IOX ,'FLAG2 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X05   >
      ENTRY S01X05
C VARIABLE NAME : IC16  *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X06   >
      ENTRY S01X06
C VARIABLE NAME : IC3   *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X07   >
      ENTRY S01X07
C------------------    FLAG3
      CALL ¥SAVED ( IOX ,'FLAG3 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X08   >
      ENTRY S01X08
C------------------    FLAG31
      CALL ¥SAVED ( IOX ,'FLAG31' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X09   >
      ENTRY S01X09
C VARIABLE NAME : IC9   *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X10   >
      ENTRY S01X10
C VARIABLE NAME : IC20  *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X11   >
      ENTRY S01X11
C------------------    IFLG33
      CALL ¥SAVED ( IOX ,'IFLG33' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X12   >
      ENTRY S01X12
C VARIABLE NAME : IC17  *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X13   >
      ENTRY S01X13
C VARIABLE NAME : IC5   *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X14   >
      ENTRY S01X14
C------------------    FLAG4
      CALL ¥SAVED ( IOX ,'FLAG4 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X15   >
      ENTRY S01X15
C VARIABLE NAME : IC12  *** A SAME VARIABLE NAME ***
      RETURN
C******   < S01X16   >
      ENTRY S01X16
C------------------    FLAG5
      CALL ¥SAVED ( IOX ,'FLAG5 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S01X17   >
      ENTRY S01X17
```

```
      C------------------    FLAG53
000336       CALL ¥SAVED ( IOX ,'FLAG53' ,IRT )
000337         IF ( IRT .GT. 4 ) GO TO 8000
000338       RETURN
      C======   < S01X18   >
000339       ENTRY S01X18
      C------------------    FLAG51
000340       CALL ¥SAVED ( IOX ,'FLAG51' ,IRT )
000341         IF ( IRT .GT. 4 ) GO TO 8000
000342       RETURN
      C======   < S01X19   >
000343       ENTRY S01X19
      C------------------    FLAG52
000344       CALL ¥SAVED ( IOX ,'FLAG52' ,IRT )
000345         IF ( IRT .GT. 4 ) GO TO 8000
000346       RETURN
      C======   < S01X20   >
000347       ENTRY S01X20
      C VARIABLE NAME : IC18  *** A SAME VARIABLE NAME ***
000348       RETURN
      C======   < S01X21   >
000349       ENTRY S01X21
      C VARIABLE NAME : IC4   *** A SAME VARIABLE NAME ***
000350       RETURN
      C======   < S01X22   >
000351       ENTRY S01X22
      C------------------    NEF
000352       CALL ¥SAVED ( IOX ,'NEF   ' ,IRT )
000353         IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NET
000354       CALL ¥SAVED ( IOX ,'NET   ' ,IRT )
000355         IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NERF
000356       CALL ¥SAVED ( IOX ,'NERF  ' ,IRT )
000357         IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NERT
000358       CALL ¥SAVED ( IOX ,'NERT  ' ,IRT )
000359         IF ( IRT .GT. 4 ) GO TO 8000
000360       RETURN
      C======   < S01X23   >
000361       ENTRY S01X23
      C VARIABLE NAME : NERF  *** A SAME VARIABLE NAME ***
      C------------------   NECF   < TBL-1 >
000362       CALL ¥SAVED ( IOX ,'NECF  ' ,IRT )
000363         IF ( IRT .GT. 4 ) GO TO 8000
000364       RETURN
      C======   < S01X24   >
000365       ENTRY S01X24
      C VARIABLE NAME : NERT  *** A SAME VARIABLE NAME ***
      C------------------   NECT   < TBL-1 >
000366       CALL ¥SAVED ( IOX ,'NECT  ' ,IRT )
000367         IF ( IRT .GT. 4 ) GO TO 8000
000368       RETURN
      C======   < S01X25   >
000369       ENTRY S01X25
      C VARIABLE NAME : NEF   *** A SAME VARIABLE NAME ***
      C------------------   NEGF   < TBL-1 >
000370       CALL ¥SAVED ( IOX ,'NEGF  ' ,IRT )
000371         IF ( IRT .GT. 4 ) GO TO 8000
000372       RETURN
      C======   < S01X26   >
000373       ENTRY S01X26
      C VARIABLE NAME : NET   *** A SAME VARIABLE NAME ***
      C------------------   NEGT   < TBL-1 >
000374       CALL ¥SAVED ( IOX ,'NEGT  ' ,IRT )
000375         IF ( IRT .GT. 4 ) GO TO 8000
000376       RETURN
      C======   < S01X27   >
000377       ENTRY S01X27
      C------------------    BSQ
000378       CALL ¥SAVED ( IOX ,'BSQ   ' ,IRT )
000379         IF ( IRT .GT. 4 ) GO TO 8000
000380       RETURN
      C
      C>>>> ERROR RETURN
      C
000381  8000 CONTINUE
000382       WRITE(*,*) '*** SEC001 *** INVALID VARIABLE NAMES '

          RETURN
          END
```

```
        C **********
        C * SEC002 * USERF
        C **********
000001          SUBROUTINE SEC002 ( IOX  )
        *INCLUDE ¥INC¥
        C
000240          CALL S02X00(IOX)
        C---------
000241          CALL S02X01
        C
000242          RETURN
000243          END
```

```
        C ****************
        C * S02X00       *  USERF
        C ****************
000001          SUBROUTINE S02X00 ( IOXX )
        *INCLUDE ¥INC¥
000240          IOX =IOXX
000241          WRITE( IOX ,'(''$USERF'')')
000242          RETURN
        C******   < S02X01   >
000243          ENTRY S02X01
        C **** TABLE-2 *****
        C------------------    IDNUM
000244          CALL ¥SAVED ( IOX ,'IDNUM ' ,IRT )
000245            IF ( IRT .GT. 4 ) GO TO 8000
        C------------------    IDENT *** TABLE 2****
000246          CALL ¥SAVED ( IOX ,'IDENT ' ,IRT )
000247            IF ( IRT .GT. 4 ) GO TO 8000
000248          RETURN
        C
        C>>>> ERROR RETURN
        C
000249   8000 CONTINUE
000250          WRITE(*,*) '*** SEC002 *** INVALID VARIABLE NAME  '
        C
000251          RETURN
000252          END
```

```
      C **********
      C * SEC003 *  .... COLLISION PROBABILITY METHOD
      C **********
000001      SUBROUTINE SEC003 ( IOX )
      *INCLUDE ¥INC¥
      C*
      C*     P.I.J.
      C*
      C-------------  < OUTPUT REFERENCE  NUMBER SET >
000240      CALL  S03X00( IOX )
      C----------
000241      CALL  S03X01
000242      CALL  S03X02
000243      CALL  S03X03
000244      IF  ( IBOUND .EQ. 1 )          CALL  S03X04
000245      IF  ( IGT    .GE. 9 )          CALL  S03X05
000246      IF  ( IGT    .NE. 2 )          CALL  S03X06
000247      IF  ( IGT    .GE. 9 )          CALL  S03X07
000248      CALL  S03X08
000249      CALL  S03X12
000250      CALL  S03X13
000251      CALL  S03X14
000252      CALL  S03X15
000253      IF  ( ( IGT    .EQ.  10 ) .OR.
     &      ( IGT    .EQ.  14 )  )       CALL  S03X16
000254      CALL  S03X17
000255      IF  ( ( ( IGT .EQ. 10 ).OR.
     &        ( IGT .EQ. 14 )    )  .AND.
     &      ( NY   .GT.  0       )   )  CALL  S03X18
000256      IF  ( (  IGT  .EQ. 13   ) .AND.
     &      (  NY   .GT.  1   )       )  CALL  S03X18
000257      IF  ( (  IGT  .EQ.  9   ) .OR.
     &      (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 14   )       )  CALL  S03X20
000258      IF  ( (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   )       )  CALL  S03X21
000259      IF  (    IGT  .EQ. 13           )  CALL  S03X22
000260      IF  ( (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   )       )  CALL  S03X23
000261      IF  (    IGT  .EQ. 13           )  CALL  S03X24
000262      IF  ( (  IGT  .EQ.  9   ) .OR.
     &      (  IGT  .EQ. 10   ) .OR.
     &      (  IGT  .EQ. 14   )       )  CALL  S03X25
000263      IF  ( (  IGT  .EQ. 11   ) .OR.
     &      (  IGT  .EQ. 12   ) .OR.
     &      (  IGT  .EQ. 13   )       )  CALL  S03X26
000264      CALL  S03X31
000265      CALL  S03X29
000266      CALL  S03X30
      C
000267      RETURN
000268      END


      C ***************
      C * S03X00      *
      C ***************
000001      SUBROUTINE S03X00 ( IOXX)
      *INCLUDE ¥INC¥
000240      IOX = IOXX
000241      WRITE(IOX,'(''@COLLIS'')')
000242      RETURN
      C------   < S03X01   >
000243      ENTRY S03X01
      C-------------------     IGT
000244      CALL ¥SAVED ( IOX ,'IGT    ' ,IRT )
000245        IF ( IRT .GT. 4 ) GO TO 8000
000246      RETURN
      C------   < S03X02   >
000247      ENTRY S03X02
      C-------------------     NZ
000248      CALL ¥SAVED ( IOX ,'NZ     ' ,IRT )
000249        IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------     NR
000250      CALL ¥SAVED ( IOX ,'NR     ' ,IRT )
000251        IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------     NRR
000252      CALL ¥SAVED ( IOX ,'NRR    ' ,IRT )
000253        IF ( IRT .GT. 4 ) GO TO 8000
```

```
      C------------------    NXR
000254      CALL ¥SAVED ( IOX ,'NXR    ' ,IRT )
000255        IF ( IRT .GT. 4 ) GO TO 8000
000256      RETURN
      C======   < S03X03   >
000257      ENTRY S03X03
      C------------------    IBOUND
000258      CALL ¥SAVED ( IOX ,'IBOUND' ,IRT )
000259        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NX
000260      CALL ¥SAVED ( IOX ,'NX     ' ,IRT )
000261        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NY
000262      CALL ¥SAVED ( IOX ,'NY     ' ,IRT )
000263        IF ( IRT .GT. 4 ) GO TO 8000
000264      RETURN
      C======   < S03X04   >
000265      ENTRY S03X04
      C------------------    NCELL
000266      CALL ¥SAVED ( IOX ,'NCELL  ' ,IRT )
000267        IF ( IRT .GT. 4 ) GO TO 8000
000268      RETURN
      C======   < S03X05   >
000269      ENTRY S03X05
      C------------------    NTPIN
000270      CALL ¥SAVED ( IOX ,'NTPIN  ' ,IRT )
000271        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NAPIN
000272      CALL ¥SAVED ( IOX ,'NAPIN  ' ,IRT )
000273        IF ( IRT .GT. 4 ) GO TO 8000
000274      RETURN
      C======   < S03X06   >
000275      ENTRY S03X06
      C------------------    NGR
000276      CALL ¥SAVED ( IOX ,'NGR    ' ,IRT )
000277        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    NDA
000278      CALL ¥SAVED ( IOX ,'NDA    ' ,IRT )
000279        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    IBETM
000280      CALL ¥SAVED ( IOX ,'IBETM  ' ,IRT )
000281        IF ( IRT .GT. 4 ) GO TO 8000
000282      RETURN
      C======   < S03X07   >
000283      ENTRY S03X07
      C------------------    NDPIN
000284      CALL ¥SAVED ( IOX ,'NDPIN  ' ,IRT )
000285        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    IDIVP
000286      CALL ¥SAVED ( IOX ,'IDIVP  ' ,IRT )
000287        IF ( IRT .GT. 4 ) GO TO 8000
000288      RETURN
      C======   < S03X08   >


000289      ENTRY S03X08
      C------------------    IEDPIJ
000290      CALL ¥SAVED ( IOX ,'IEDPIJ' ,IRT )
000291        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    IPLOT
000292      CALL ¥SAVED ( IOX ,'IPLOT  ' ,IRT )
000293        IF ( IRT .GT. 4 ) GO TO 8000
000294      RETURN
      C======   < S03X12   >
000295      ENTRY S03X12
      C VARIABLE NAME : NZ    *** A SAME VARIABLE NAME ***
      C------------------   NREG   < TBL-1 >
000296      CALL ¥SAVED ( IOX ,'NREG   ' ,IRT )
000297        IF ( IRT .GT. 4 ) GO TO 8000
000298      RETURN
      C======   < S03X13   >
000299      ENTRY S03X13
      C VARIABLE NAME : NR    *** A SAME VARIABLE NAME ***
      C------------------   IRR    < TBL-1 >
000300      CALL ¥SAVED ( IOX ,'IRR    ' ,IRT )
000301        IF ( IRT .GT. 4 ) GO TO 8000
000302      RETURN
```

```
      C======   < S03X14   >
000303        ENTRY S03X14
      C VARIABLE NAME : NRR   *** A SAME VARIABLE NAME ***
      C-------------------   IXR    < TBL-1 >
000304        CALL ¥SAVED ( IOX ,'IXR   ' ,IRT )
000305          IF ( IRT .GT. 4 ) GO TO 8000
000306        RETURN
      C======   < S03X15   >
000307        ENTRY S03X15
      C VARIABLE NAME : NRR   *** A SAME VARIABLE NAME ***
      C-------------------   MAR    < TBL-1 >
000308        CALL ¥SAVED ( IOX ,'MAR   ' ,IRT )
000309          IF ( IRT .GT. 4 ) GO TO 8000
000310        RETURN
      C======   < S03X16   >
000311        ENTRY S03X16
      C VARIABLE NAME : NAPIN *** A SAME VARIABLE NAME ***
      C-------------------   NPIN   < TBL-1 >
000312        CALL ¥SAVED ( IOX ,'NPIN  ' ,IRT )
000313          IF ( IRT .GT. 4 ) GO TO 8000
000314        RETURN
      C======   < S03X17   >
000315        ENTRY S03X17
      C-------------------   IRX    < TBL-1 >
000316        CALL ¥SAVED ( IOX ,'IRX   ' ,IRT )
000317          IF ( IRT .GT. 4 ) GO TO 8000
000318        RETURN
      C======   < S03X18   >
000319        ENTRY S03X18
      C VARIABLE NAME : NY    *** A SAME VARIABLE NAME ***
      C-------------------   TY     < TBL-1 >
000320        CALL ¥SAVED ( IOX ,'TY    ' ,IRT )
000321          IF ( IRT .GT. 4 ) GO TO 8000
000322        RETURN
      C======   < S03X19   >
000323        ENTRY S03X19
      C-------------------   TYD    < TBL-1 >
000324        CALL ¥SAVED ( IOX ,'TYD   ' ,IRT )
000325          IF ( IRT .GT. 4 ) GO TO 8000
000326        RETURN
      C======   < S03X20   >
000327        ENTRY S03X20
      C VARIABLE NAME : NAPIN *** A SAME VARIABLE NAME ***
      C-------------------   RPP    < TBL-1 >
000328        CALL ¥SAVED ( IOX ,'RPP   ' ,IRT )
000329          IF ( IRT .GT. 4 ) GO TO 8000
000330        RETURN
      C======   < S03X21   >
000331        ENTRY S03X21
      C VARIABLE NAME : NTPIN *** A SAME VARIABLE NAME ***
      C-------------------   RPPD   < TBL-1 >
000332        CALL ¥SAVED ( IOX ,'RPPD  ' ,IRT )
000333          IF ( IRT .GT. 4 ) GO TO 8000
000334        RETURN
      C======   < S03X22   >
000335        ENTRY S03X22


      C VARIABLE NAME : NTPIN *** A SAME VARIABLE NAME ***
      C-------------------   IXP    < TBL-1 >
000336        CALL ¥SAVED ( IOX ,'IXP   ' ,IRT )
000337          IF ( IRT .GT. 4 ) GO TO 8000
000338        RETURN
      C======   < S03X23   >
000339        ENTRY S03X23
      C VARIABLE NAME : NTPIN *** A SAME VARIABLE NAME ***
      C-------------------   THETA  < TBL-1 >
000340        CALL ¥SAVED ( IOX ,'THETA ' ,IRT )
000341          IF ( IRT .GT. 4 ) GO TO 8000
000342        RETURN
      C======   < S03X24   >
000343        ENTRY S03X24
      C VARIABLE NAME : NTPIN *** A SAME VARIABLE NAME ***
      C-------------------   IYP    < TBL-1 >
000344        CALL ¥SAVED ( IOX ,'IYP   ' ,IRT )
000345          IF ( IRT .GT. 4 ) GO TO 8000
000346        RETURN
```

```
      C======   < S03X25   >
000347        ENTRY S03X25
      C-------------------   RDP    < TBL-1 >
000348        CALL ¥SAVED ( IOX ,'RDP    ' ,IRT )
000349          IF ( IRT .GT. 4 ) GO TO 8000
000350        RETURN
      C======   < S03X26   >
000351        ENTRY S03X26
      C VARIABLE NAME : NTPIN *** A SAME VARIABLE NAME ***
      C-------------------   RDPD   < TBL-1 >
000352        CALL ¥SAVED ( IOX ,'RDPD   ' ,IRT )
000353          IF ( IRT .GT. 4 ) GO TO 8000
000354        RETURN
      C======   < S03X27   >
000355        ENTRY S03X27
      C-------------------    IG
000356        CALL ¥SAVED ( IOX ,'IG     ' ,IRT )
000357          IF ( IRT .GT. 4 ) GO TO 8000
000358        RETURN
      C======   < S03X28   >
000359        ENTRY S03X28
      C-------------------    ISCAL
000360        CALL ¥SAVED ( IOX ,'ISCAL  ' ,IRT )
000361          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    ICONT
000362        CALL ¥SAVED ( IOX ,'ICONT  ' ,IRT )
000363          IF ( IRT .GT. 4 ) GO TO 8000
000364        RETURN
      C======   < S03X29   >
000365        ENTRY S03X29
      C-------------------    IDB
000366        CALL ¥SAVED ( IOX ,'IDB    ' ,IRT )
000367          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IGEOM
000368        CALL ¥SAVED ( IOX ,'IGEOM  ' ,IRT )
000369          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    MODEL
000370        CALL ¥SAVED ( IOX ,'MODEL  ' ,IRT )
000371          IF ( IRT .GT. 4 ) GO TO 8000
000372        RETURN
      C======   < S03X30   >
000373        ENTRY S03X30
      C-------------------    RF
000374        CALL ¥SAVED ( IOX ,'RF     ' ,IRT )
000375          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    RM
000376        CALL ¥SAVED ( IOX ,'RM     ' ,IRT )
000377          IF ( IRT .GT. 4 ) GO TO 8000
000378        RETURN
      C======   < S03X31   >
000379        ENTRY S03X31
      C **** TABLE-2  ==> S03X31
000380        CALL ¥SAVED( IOX ,'IGTBL   ' ,IRT )
000381          IF ( IRT .GT. 4 ) GO TO 8000
000382        RETURN
      C
      C>>>> ERROR RETURN
      C
000383  8000 CONTINUE

000384        WRITE(*,*) '*** SEC003 *** INVALID VARIABLE  IRT : ',IRT
000385        RETURN
000386        END
```

```
      C **********
      C * SEC004 *  .... COLLISION PROBABILITY METHOD
      C **********
000001      SUBROUTINE SEC004 ( IOX )
      *INCLUDE ¥INC¥
      C======== <<<  INITIAL SET  >>>
000240      CALL S04X00(IOX)
      C------------------------------
000241      CALL S04X01
000242      CALL S04X02
000243      CALL S04X03
000244      CALL S04X04
000245      CALL S04X05
000246      CALL S04X06
000247      CALL S04X07
000248      CALL S04X08
000249      CALL S04X09
000250      IF ( IDAT2 .GT. 0 ) CALL S04X10
000251      CALL S04X11
000252      CALL S04X12
      C
000253      RETURN
000254      END
```

```
      C ***************
      C * S04X00      *  ANISN
      C ***************
000001        SUBROUTINE S04X00 ( IOXX )
      *INCLUDE ¥INC¥
000240        IOX = IOXX
000241        WRITE(IOX,'(''@ANISN'')')
000242        RETURN
      C======   < S04X01   >
000243        ENTRY S04X01
      C-------------------    ITH
000244        CALL ¥SAVED ( IOX ,'ITH    ' ,IRT )
000245          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    ISCT
000246        CALL ¥SAVED ( IOX ,'ISCT   ' ,IRT )
000247          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    ISN
000248        CALL ¥SAVED ( IOX ,'ISN    ' ,IRT )
000249          IF ( IRT .GT. 4 ) GO TO 8000
000250        RETURN
      C======   < S04X02   >
000251        ENTRY S04X02
      C-------------------    IGE
000252        CALL ¥SAVED ( IOX ,'IGE    ' ,IRT )
000253          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IBL
000254        CALL ¥SAVED ( IOX ,'IBL    ' ,IRT )
000255          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IBR
000256        CALL ¥SAVED ( IOX ,'IBR    ' ,IRT )
000257          IF ( IRT .GT. 4 ) GO TO 8000
000258        RETURN
      C======   < S04X03   >
000259        ENTRY S04X03
      C-------------------    IZM
000260        CALL ¥SAVED ( IOX ,'IZM    ' ,IRT )
000261          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IM
000262        CALL ¥SAVED ( IOX ,'IM     ' ,IRT )
000263          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IEVT
000264        CALL ¥SAVED ( IOX ,'IEVT   ' ,IRT )
000265          IF ( IRT .GT. 4 ) GO TO 8000
000266        RETURN
      C======   < S04X04   >
000267        ENTRY S04X04
      C-------------------    IGM
000268        CALL ¥SAVED ( IOX ,'IGM    ' ,IRT )
000269          IF ( IRT .GT. 4 ) GO TO 8000
000270        RETURN
      C======   < S04X05   >
000271        ENTRY S04X05
      C-------------------    IDAT2
000272        CALL ¥SAVED ( IOX ,'IDAT2  ' ,IRT )
000273          IF ( IRT .GT. 4 ) GO TO 8000
000274        RETURN
      C======   < S04X06   >
000275        ENTRY S04X06
      C-------------------    DY
000276        CALL ¥SAVED ( IOX ,'DY     ' ,IRT )
000277          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    DZ
000278        CALL ¥SAVED ( IOX ,'DZ     ' ,IRT )
000279          IF ( IRT .GT. 4 ) GO TO 8000
000280        RETURN
      C======   < S04X07   >
000281        ENTRY S04X07
      C-------------------   ANSN04 < TBL-1 >
000282        CALL ¥SAVED ( IOX ,'ANSN04' ,IRT )
000283          IF ( IRT .GT. 4 ) GO TO 8000
000284        RETURN
      C======   < S04X08   >
000285        ENTRY S04X08
      C VARIABLE NAME : IM    *** A SAME VARIABLE NAME ***
      C-------------------   ANSN08 < TBL-1 >
000286        CALL ¥SAVED ( IOX ,'ANSN08' ,IRT )
000287          IF ( IRT .GT. 4 ) GO TO 8000
000288        RETURN
```

```
      C======   < S04X09   >
000289        ENTRY S04X09
      C VARIABLE NAME : IZM   *** A SAME VARIABLE NAME ***
      C-------------------   ANSN09 < TBL-1 >
000290        CALL ¥SAVED ( IOX ,'ANSN09' ,IRT )
000291          IF ( IRT .GT. 4 ) GO TO 8000
000292        RETURN
      C======   < S04X10   >
000293        ENTRY S04X10
      C VARIABLE NAME : IZM   *** A SAME VARIABLE NAME ***
      C-------------------   ANSN19 < TBL-1 >
000294        CALL ¥SAVED ( IOX ,'ANSN19' ,IRT )
000295          IF ( IRT .GT. 4 ) GO TO 8000
000296        RETURN
      C======   < S04X11   >
000297        ENTRY S04X11
      C VARIABLE NAME : IGH   *** A SAME VARIABLE NAME ***
      C-------------------   ANSN24 < TBL-1 >
000298        CALL ¥SAVED ( IOX ,'ANSN24' ,IRT )
000299          IF ( IRT .GT. 4 ) GO TO 8000
000300        RETURN
      C======   < S04X12   >
000301        ENTRY S04X12
      C VARIABLE NAME : IZM   *** A SAME VARIABLE NAME ***
      C-------------------   ANSN27 < TBL-1 >
000302        CALL ¥SAVED ( IOX ,'ANSN27' ,IRT )
000303          IF ( IRT .GT. 4 ) GO TO 8000
000304        RETURN
      C
      C>>>> ERROR RETURN
      C
000305   8000 CONTINUE
000306        WRITE(*,*) '*** SEC004 *** INVALID VARIABLE NAME '
000307        RETURN
000308        END
```

```
      C **********
      C * SEC005 *
      C **********
000001        SUBROUTINE SEC005( IOX )
      *INCLUDE ¥INC¥
      C
000240        CALL  S05X00(IOX)
      C-----------
000241        CALL  S05X01
000242        CALL  S05X02
000243        CALL  S05X03
000244        CALL  S05X04
000245        CALL  S05X05
000246        CALL  S05X06
000247        CALL  S05X07
000248        CALL  S05X08
000249        CALL  S05X09
000250        CALL  S05X10
000251        CALL  S05X11
000252        CALL  S05X12
000253        RETURN
000254        END
```

```
      C ***************
      C * S05X00      *  TWOTRAN
      C ***************
000001        SUBROUTINE S05X00 ( IOXX )
      *INCLUDE ¥INC¥
000240        IOX = IOXX
000241        WRITE(IOX,'(''@TWOTRN'')')
000242        RETURN
      C======   < S05X01   >
000243      ENTRY S05X01
      C------------------     TITLE
000244      CALL ¥SAVED ( IOX ,'TITLE ' ,IRT )
000245        IF ( IRT .GT. 4 ) GO TO 8000
000246      RETURN
      C======   < S05X02   >
000247      ENTRY S05X02
      C------------------     ITHT
000248      CALL ¥SAVED ( IOX ,'ITHT  ' ,IRT )
000249        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     ISCTT
000250      CALL ¥SAVED ( IOX ,'ISCTT ' ,IRT )
000251        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     ISNT
000252      CALL ¥SAVED ( IOX ,'ISNT  ' ,IRT )
000253        IF ( IRT .GT. 4 ) GO TO 8000
000254      RETURN
      C======   < S05X03   >
000255      ENTRY S05X03
      C------------------     IGMT
000256      CALL ¥SAVED ( IOX ,'IGMT  ' ,IRT )
000257        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IMT
000258      CALL ¥SAVED ( IOX ,'IMT   ' ,IRT )
000259        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     JMT
000260      CALL ¥SAVED ( IOX ,'JMT   ' ,IRT )
000261        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IBLT
000262      CALL ¥SAVED ( IOX ,'IBLT  ' ,IRT )
000263        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IBRT
000264      CALL ¥SAVED ( IOX ,'IBRT  ' ,IRT )
000265        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IBBT
000266      CALL ¥SAVED ( IOX ,'IBBT  ' ,IRT )
000267        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IBTT
000268      CALL ¥SAVED ( IOX ,'IBTT  ' ,IRT )
000269        IF ( IRT .GT. 4 ) GO TO 8000
000270      RETURN
      C======   < S05X04   >
000271      ENTRY S05X04
      C------------------     IEVTT
000272      CALL ¥SAVED ( IOX ,'IEVTT ' ,IRT )
000273        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     MTT
000274      CALL ¥SAVED ( IOX ,'MTT   ' ,IRT )
000275        IF ( IRT .GT. 4 ) GO TO 8000
000276      RETURN
      C======   < S05X05   >
000277      ENTRY S05X05
      C------------------     ITLIM
000278      CALL ¥SAVED ( IOX ,'ITLIM ' ,IRT )
000279        IF ( IRT .GT. 4 ) GO TO 8000
      C------------------     IGEOMT
000280      CALL ¥SAVED ( IOX ,'IGEOMT' ,IRT )
000281        IF ( IRT .GT. 4 ) GO TO 8000
000282      RETURN
      C======   < S05X06   >
000283      ENTRY S05X06
      C------------------     BHGT
000284      CALL ¥SAVED ( IOX ,'BHGT  ' ,IRT )
000285        IF ( IRT .GT. 4 ) GO TO 8000
000286      RETURN
      C======   < S05X07   >
000287      ENTRY S05X07
      C VARIABLE NAME : IMT   *** A SAME VARIABLE NAME ***
      C------------------   IMX    < TBL-1 >
```

```
000288          CALL ¥SAVED ( IOX ,'IHX    ' ,IRT )
000289            IF ( IRT .GT. 4 ) GO TO 8000
000290          RETURN
        C======   < S05X08   >
000291          ENTRY S05X08
        C VARIABLE NAME : JMT   *** A SAME VARIABLE NAME ***
        C------------------   IHY    < TBL-1 >
000292          CALL ¥SAVED ( IOX ,'IHY    ' ,IRT )
000293            IF ( IRT .GT. 4 ) GO TO 8000
000294          RETURN
        C======   < S05X09   >
000295          ENTRY S05X09
        C------------------   XRAD   < TBL-1 >
000296          CALL ¥SAVED ( IOX ,'XRAD   ' ,IRT )
000297            IF ( IRT .GT. 4 ) GO TO 8000
000298          RETURN
        C======   < S05X10   >
000299          ENTRY S05X10
        C------------------   YRAD   < TBL-1 >
000300          CALL ¥SAVED ( IOX ,'YRAD   ' ,IRT )
000301            IF ( IRT .GT. 4 ) GO TO 8000
000302          RETURN
        C======   < S05X11   >
000303          ENTRY S05X11
        C------------------   IDCS   < TBL-1 >
000304          CALL ¥SAVED ( IOX ,'IDCS   ' ,IRT )
000305            IF ( IRT .GT. 4 ) GO TO 8000
000306          RETURN
        C======   < S05X12   >
000307          ENTRY S05X12
        C------------------   IXZ    < TBL-1 >
000308          CALL ¥SAVED ( IOX ,'IXZ    ' ,IRT )
000309            IF ( IRT .GT. 4 ) GO TO 8000
000310          RETURN
        C
        C>>>> ERROR RETURN
        C
000311   8000 CONTINUE
000312          WRITE(*,*) '*** SEC005 *** INVALID VARIABLE NAME '
000313          RETURN
000314          END
```

```
        C **********
        C * SEC006 *  TUD
        C **********
000001          SUBROUTINE SEC006 ( IOX )
        *INCLUDE ¥INC¥
        C
000240          CALL S06X00(IOX)
        C
000241          CALL S06X01
000242          CALL S06X02
000243          CALL S06X03
000244          CALL S06X04
000245          CALL S06X05
        C
000246          RETURN
000247          END
```

```
      C ***************
      C * S06X00       *
      C ***************
000001        SUBROUTINE S06X00 ( IOXX )
      *INCLUDE ¥INC¥
000240        IOXX= IOX
000241        WRITE(IOX,'(''@TUD'')')
000242        RETURN
      C======   < S06X01   >
000243        ENTRY S06X01
      C-------------------    NRMAX
000244        CALL ¥SAVED ( IOX ,'NRMAX ' ,IRT )
000245          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IG2
000246        CALL ¥SAVED ( IOX ,'IG2   ' ,IRT )
000247          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IBOUN2
000248        CALL ¥SAVED ( IOX ,'IBOUN2' ,IRT )
000249          IF ( IRT .GT. 4 ) GO TO 8000
      C VARIABLE NAME : NXR   *** A SAME VARIABLE NAME ***
000250        RETURN
      C======   < S06X02   >
000251        ENTRY S06X02
      C VARIABLE NAME : NRMAX *** A SAME VARIABLE NAME ***
      C-------------------   NK     < TBL-1 >
000252        CALL ¥SAVED ( IOX ,'NK    ' ,IRT )
000253          IF ( IRT .GT. 4 ) GO TO 8000
000254        RETURN
      C======   < S06X03   >
000255        ENTRY S06X03
      C VARIABLE NAME : NRMAX *** A SAME VARIABLE NAME ***
      C-------------------   IK     < TBL-1 >
000256        CALL ¥SAVED ( IOX ,'IK    ' ,IRT )
000257          IF ( IRT .GT. 4 ) GO TO 8000
000258        RETURN
      C======   < S06X04   >
000259        ENTRY S06X04
      C VARIABLE NAME : NRMAX *** A SAME VARIABLE NAME ***
      C-------------------   RX     < TBL-1 >
000260        CALL ¥SAVED ( IOX ,'RX    ' ,IRT )
000261          IF ( IRT .GT. 4 ) GO TO 8000
000262        RETURN
      C======   < S06X05   >
000263        ENTRY S06X05
      C-------------------    BSQ2
000264        CALL ¥SAVED ( IOX ,'BSQ2  ' ,IRT )
000265          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    XLAMD
000266        CALL ¥SAVED ( IOX ,'XLAMD ' ,IRT )
000267          IF ( IRT .GT. 4 ) GO TO 8000
000268        RETURN
      C
      C>>>> ERROR RETURN
      C
000269   8000 CONTINUE
000270        WRITE(*,*) '*** SEC006  *** INVALID VARIABLE NAME '
000271        RETURN
000272        END
```

```
      C **********
      C * SEC007 *  CITATN
      C **********
000001        SUBROUTINE SEC007( IOX )
      *INCLUDE ¥INC¥
      C
      C--------------
000240        CALL  S07X00(IOX)
      C  CITATION
000241        CALL  S07X01
000242        CALL  S07X02
000243        IF  ( ABS(IDC) .EQ.  3 )    CALL  S07X03
000244        IF  ( NHC      .LT.  0 )  THEN
000245            CALL  S07X04
000246            CALL  S07X05
000247        ENDIF
000248        CALL  S07X06
000249        CALL  S07X07
000250        CALL  S07X08
      C
000251        CALL  S07X09
000252        IF  ( NUAC5  .GE.  6 )    CALL  S07X10
000253        IF  ( NUAC5  .GE. 11 )    CALL  S07X11
      C
000254        CALL  S07X12
      C
000255        CALL  S07X13
000256        CALL  S07X14
000257        IF  ( NUAC5  .GE.  6 )    THEN
000258            CALL  S07X15
000259            CALL  S07X16
000260        ELSE
000261            NYY = 1
000262        ENDIF
000263        IF  ( NUAC5  .GE. 11 )    THEN
000264            CALL  S07X17
000265            CALL  S07X18
000266        ELSE
000267            NZZ = 1
000268        ENDIF
      C
000269        CALL  S07X19
000270        CALL  S07X20
000271        CALL  S07X21
000272        IF    ( INDI  .EQ.  1 )  THEN
000273            CALL  S07X22
000274        ELSEIF( INDI  .EQ.  2 )  THEN
000275            CALL  S07X23
000276        ELSEIF( INDI  .EQ.  3 )  THEN
000277            CALL  S07X24
000278        ENDIF
000279        CALL  S07X25
000280        CALL  S07X26
      C
000281        RETURN
000282        END
```

```
      C ***************
      C * S07X00      * CITATION
      C ***************
000001        SUBROUTINE S07X00 ( IOXX )
      *INCLUDE ¥INC¥
000240        IOX = IOXX
000241        WRITE(IOX ,'(''@CITATN'')')
000242        RETURN
      C======   < S07X01   >
000243        ENTRY S07X01
      C-------------------    NMC
000244        CALL ¥SAVED ( IOX ,'NMC    ' ,IRT )
000245          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    NXRC
000246        CALL ¥SAVED ( IOX ,'NXRC   ' ,IRT )
000247          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IDC
000248        CALL ¥SAVED ( IOX ,'IDC    ' ,IRT )
000249          IF ( IRT .GT. 4 ) GO TO 8000
000250        RETURN
      C======   < S07X02   >
000251        ENTRY S07X02
      C-------------------    IXKI
000252        CALL ¥SAVED ( IOX ,'IXKI   ' ,IRT )
000253          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IDELAY
000254        CALL ¥SAVED ( IOX ,'IDELAY' ,IRT )
000255          IF ( IRT .GT. 4 ) GO TO 8000
000256        RETURN
      C======   < S07X03   >
000257        ENTRY S07X03
      C-------------------   IXYZ   < TBL-1 >
000258        CALL ¥SAVED ( IOX ,'IXYZ   ' ,IRT )
000259          IF ( IRT .GT. 4 ) GO TO 8000
000260        RETURN
      C======   < S07X04   >
000261        ENTRY S07X04
      C-------------------    ICASE
000262        CALL ¥SAVED ( IOX ,'ICASE  ' ,IRT )
000263          IF ( IRT .GT. 4 ) GO TO 8000
000264        RETURN
      C======   < S07X05   >
000265        ENTRY S07X05
      C-------------------    SAMPLE
000266        CALL ¥SAVED ( IOX ,'SAMPLE' ,IRT )
000267          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IOPT
000268        CALL ¥SAVED ( IOX ,'IOPT   ' ,IRT )
000269          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    IDOPT
000270        CALL ¥SAVED ( IOX ,'IDOPT  ' ,IRT )
000271          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------   BKLTBL < TBL-2 >
000272        CALL ¥SAVED ( IOX ,'BKLTBL' , IRT )
000273          IF ( IRT .GT. 4 ) GO TO 8000
000274        RETURN
      C======   < S07X06   >
000275        ENTRY S07X06
      C-------------------    CTITL1
000276        CALL ¥SAVED ( IOX ,'CTITL1' ,IRT )
000277          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    CTITL2
000278        CALL ¥SAVED ( IOX ,'CTITL2' ,IRT )
000279          IF ( IRT .GT. 4 ) GO TO 8000
000280        RETURN
      C======   < S07X07   >
000281        ENTRY S07X07
      C-------------------    NUAC5
000282        CALL ¥SAVED ( IOX ,'NUAC5  ' ,IRT )
000283          IF ( IRT .GT. 4 ) GO TO 8000
000284        RETURN
      C======   < S07X08   >
000285        ENTRY S07X08
      C-------------------    NUAC8
000286        CALL ¥SAVED ( IOX ,'NUAC8  ' ,IRT )
000287          IF ( IRT .GT. 4 ) GO TO 8000
      C-------------------    NUAC9
000288        CALL ¥SAVED ( IOX ,'NUAC9  ' ,IRT )
```

```
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X09 >
      ENTRY S07X09
C------------------  NUAC11
      CALL ¥SAVED ( IOX ,'NUAC11' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  NUAC13
      CALL ¥SAVED ( IOX ,'NUAC13' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X10 >
      ENTRY S07X10
C------------------  NUAC12
      CALL ¥SAVED ( IOX ,'NUAC12' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  NUAC14
      CALL ¥SAVED ( IOX ,'NUAC14' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X11 >
      ENTRY S07X11
C------------------  NUAC15
      CALL ¥SAVED ( IOX ,'NUAC15' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  NUAC16
      CALL ¥SAVED ( IOX ,'NUAC16' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X12 >
      ENTRY S07X12
C------------------  NUAC17
      CALL ¥SAVED ( IOX ,'NUAC17' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X13 >
      ENTRY S07X13
C------------------  NXX
      CALL ¥SAVED ( IOX ,'NXX   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  XXTBL  < TBL-1 >
      CALL ¥SAVED ( IOX ,'XXTBL ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X14 >
      ENTRY S07X14
C VARIABLE NAME : NXX   *** A SAME VARIABLE NAME ***
C------------------  NDXTBL < TBL-1 >
      CALL ¥SAVED ( IOX ,'NDXTBL' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X15 >
      ENTRY S07X15
C------------------  NYY
      CALL ¥SAVED ( IOX ,'NYY   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  YYTBL  < TBL-1 >
      CALL ¥SAVED ( IOX ,'YYTBL ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X16 >
      ENTRY S07X16
C VARIABLE NAME : NYY   *** A SAME VARIABLE NAME ***
C------------------  NDYTBL < TBL-1 >
      CALL ¥SAVED ( IOX ,'NDYTBL' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X17 >
      ENTRY S07X17
C------------------  NZZ
      CALL ¥SAVED ( IOX ,'NZZ   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------  ZZTBL  < TBL-1 >
      CALL ¥SAVED ( IOX ,'ZZTBL ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C====== < S07X18 >
      ENTRY S07X18
```

```
C VARIABLE NAME : NZZ   *** A SAME VARIABLE NAME ***
C------------------   NDZTBL < TBL-1 >
      CALL ¥SAVED ( IOX ,'NDZTBL' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X19   >
      ENTRY S07X19
C------------------   HXX    < TBL-1 >
      CALL ¥SAVED ( IOX ,'HXX    ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X20   >
      ENTRY S07X20
C------------------   OVER   < TBL-1 >
      CALL ¥SAVED ( IOX ,'OVER   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X21   >
      ENTRY S07X21
C------------------    KMAX
      CALL ¥SAVED ( IOX ,'KMAX   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------    INDI
      CALL ¥SAVED ( IOX ,'INDI   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X22   >
      ENTRY S07X22
C------------------    BUCK
      CALL ¥SAVED ( IOX ,'BUCK   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X23   >
      ENTRY S07X23
C VARIABLE NAME : KMAX  *** A SAME VARIABLE NAME ***
C------------------   BKLE2  < TBL-1 >
      CALL ¥SAVED ( IOX ,'BKLE2  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X24   >
      ENTRY S07X24
C------------------   BKLE3  < TBL-2 >
      CALL ¥SAVED ( IOX ,'BKLE3  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X25   >
      ENTRY S07X25
C------------------   MAT    < TBL-1 >
      CALL ¥SAVED ( IOX ,'MAT    ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S07X26   >
      ENTRY S07X26
C------------------   NXREG  < TBL-1 >
      CALL ¥SAVED ( IOX ,'NXREG  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C
C>>>> ERROR RETURN
C
 8000 CONTINUE
      WRITE(*,*) '*** SEC007 *** INVALID VARIABLE NAME '
      RETURN
      END
```

```
C **********
C * SEC008 * MATRAL
C **********
      SUBROUTINE SEC008( IOX  )
*INCLUDE ¥INC¥
C
      CALL S08X00(IOX)
C----------
      CALL S08X01
      CALL S08X02
C
      RETURN
      END
```

```
C ***************
C * S08X00      * MATERIAL
C ***************
       SUBROUTINE S08X00 ( IOXX )
*INCLUDE ¥INC¥
       IOX = IOXX
       WRITE(IOX,'(''$MATRAL'')')
       RETURN
C******   < S08X01   >
      ENTRY S08X01
C------------------    NMAT
      CALL ¥SAVED ( IOX ,'NMAT  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C******   < S08X02   >
      ENTRY S08X02
C------------------   MTCMP  < TBL-1 >
      CALL ¥SAVED ( IOX ,'MTCMP ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C
C>>>> ERROR RETURN
C
 8000 CONTINUE
      WRITE(*,*) '*** SEC008 *** INVALID VARIABLE NAME'
      RETURN
      END
```

```
C **********
C * SEC009 * REACT
C **********
      SUBROUTINE SEC009(IOX)
*INCLUDE ¥INC¥
C  REACTION RATE CALCULATION
      CALL  S09X00(IOX)
C---------------
      CALL  S09X01
      IF  ( IOPT1 .GT. 0 )    CALL  S09X02
      IF  ( IOPT2 .GT. 0 )    CALL  S09X03
      CALL  S09X04
      IF  ( IOPT3 .GT. 0 )    CALL  S09X05
      CALL  S09X06
      IF  ( MREC  .LT. 0 )    CALL  S09X07
C
      RETURN
      END
```

```
C ***************
C * S09X00       *  REACT
C ***************
      SUBROUTINE S09X00 ( IOXX )
*INCLUDE ¥INC¥
      IOX =IOXX
      WRITE(IOX,'(''¤REACT'')')
      RETURN
C======   < S09X01   >
      ENTRY S09X01
C------------------    IOPT1
      CALL ¥SAVED ( IOX ,'IOPT1 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------    IOPT2
      CALL ¥SAVED ( IOX ,'IOPT2 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------    IOPT3
      CALL ¥SAVED ( IOX ,'IOPT3 ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------    MREC
      CALL ¥SAVED ( IOX ,'MREC  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X02   >
      ENTRY S09X02
C------------------   IOPT1T < TBL-2 >
      CALL ¥SAVED ( IOX ,'IOPT1T' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X03   >
      ENTRY S09X03
C------------------   IOPT2T < TBL-2 >
      CALL ¥SAVED ( IOX ,'IOPT2T' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X04   >
      ENTRY S09X04
C------------------   FG     < TBL-1 >
      CALL ¥SAVED ( IOX ,'FG    ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X05   >
      ENTRY S09X05
C------------------   NREC   < TBL-1 >
      CALL ¥SAVED ( IOX ,'NREC  ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X06   >
      ENTRY S09X06
C------------------   IOPT3T < TBL-2 >
      CALL ¥SAVED ( IOX ,'IOPT3T' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S09X07   >
      ENTRY S09X07
C------------------   MRECTB < TBL-2 >
      CALL ¥SAVED ( IOX ,'MRECTB' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C
C>>>> ERROR RETURN
C
 8000 CONTINUE
      WRITE(*,*) '*** SEC009 *** INVALID VARIABLE NAME'
      RETURN
      END
```

```
C **********
C * SECO10 * BURNUP
C **********
      SUBROUTINE SECO10( IOX  )
*INCLUDE ¥INC¥
C
      CALL S10X00(IOX)
C-------------
      CALL S10X01
      CALL S10X02
      CALL S10X03
C
      RETURN
      END



C ***************
C * S10X00      * BURNUP
C ***************
      SUBROUTINE S10X00 ( IOXX )
*INCLUDE ¥INC¥
      IOX= IOXX
      WRITE(IOX,'(''@BURNUP'')' )
      RETURN
C======   < S10X01   >
      ENTRY S10X01
C------------------     NEP
      CALL ¥SAVED ( IOX ,'NEP    ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------     IBUNIT
      CALL ¥SAVED ( IOX ,'IBUNIT' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------     IBEDIT
      CALL ¥SAVED ( IOX ,'IBEDIT' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S10X02   >
      ENTRY S10X02
C------------------     POWERL
      CALL ¥SAVED ( IOX ,'POWERL' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
C------------------     CVOL
      CALL ¥SAVED ( IOX ,'CVOL   ' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C======   < S10X03   >
      ENTRY S10X03
C VARIABLE NAME : NEP   *** A SAME VARIABLE NAME ***
C------------------    PERIOD < TBL-1 >
      CALL ¥SAVED ( IOX ,'PERIOD' ,IRT )
        IF ( IRT .GT. 4 ) GO TO 8000
      RETURN
C
C>>>> ERROR RETURN
C
 8000 CONTINUE
      WRITE(*,*) '*** SECO10 *** INVALID VARIABLE NAME '
      RETURN
      END
```

```
      C ***********
      C * SEC011 * MCROSS
      C ***********
000001        SUBROUTINE SEC011( IOX )
      *INCLUDE ¥INC¥
      C
000240        CALL S11X00(IOX)
      C--------------
000241        CALL S11X01
000242        CALL S11X02
000243        CALL S11X03
      C
000244        RETURN
000245        END
```

```
      C ****************
      C * S11X00       * MCROSS
      C ****************
000001        SUBROUTINE S11X00 ( IOXX )
      *INCLUDE ¥INC¥
000240        CHARACTER * 80 CL1 ,CL2
000241        IOX =IOXX
000242        WRITE(IOX,'(''@MCROSS'')')
000243        RETURN
      C======   < S11X01   >
000244        ENTRY S11X01
      C------------------    IDENTM
000245        CALL ¥SAVED ( IOX ,'IDENTM' ,IRT )
000246          IF ( IRT .GT. 4 ) GO TO 8000
000247        RETURN
      C======   < S11X02   >
000248        ENTRY S11X02
      C------------------    NT
000249        CALL ¥SAVED ( IOX ,'NT    ' ,IRT )
000250          IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    IOUT
000251        CALL ¥SAVED ( IOX ,'IOUT  ' ,IRT )
000252          IF ( IRT .GT. 4 ) GO TO 8000
      C------------------    IPLOTP
000253        CALL ¥SAVED ( IOX ,'IPLOTP' ,IRT )
000254          IF ( IRT .GT. 4 ) GO TO 8000
000255        RETURN
      C======   < S11X03   >
000256        ENTRY S11X03
      C------------------    INUM2
000257        CALL ¥SAVED ( IOX ,'INUM2 ' ,IRT )
000258          IF ( IRT .GT. 4 ) GO TO 8000
      C------------------   MCROSS < TBL-2 >
000259        CALL ¥SAVED ( IOX ,'MCROSS' ,IRT )
000260          IF ( IRT .GT. 4 ) GO TO 8000
000261        RETURN
      C
      C>>>> ERROR RETURN
      C
000262   8000 CONTINUE
000263        WRITE(*,*) '*** SEC011 *** INVALID VARIABLE NAME '
000264        RETURN
000265        END
```