CLM-P685

UKAEA

Preprint

OLYMPUS CONVENTIONS

M. H. HUGHES
K. V. ROBERTS

CULHAM LABORATORY
Abingdon Oxfordshire

1982

CLM-P 685

OLYMPUS CONVENTIONS

M.H.Hughes and K.V.Roberts

Culham Laboratory
Abingdon, Oxfordshire, OX14 3DB, UK
(EURATOM/UKAEA Fusion Association)

ABSTRACT

This article records, with examples, the detailed conventions used for notation and layout in OLYMPUS Fortran 66 programs. Many of these conventions are now maintained semi-automatically by utilities such as the COMPOSITOR [1] and the GENSIS Generator [2], described in two following papers in this ussue of Computer Physics Communications. They will however be of use to those who may wish to design similar utilities, or to construct or edit OLYMPUS programs by hand. A list of references to previous OLYMPUS publications is also included [3-22].

(Submitted for publication in Computer Physics Communications)

October 1982

CONTENTS

TABLES

FIGURES

REFERENCES

INTRODUCTION

This article records, with examples, the conventions for notation and layout used for OLYMPUS Fortran 66 programs whether written by hand or generated automatically. The general philosophy underlying the rules that we have adopted is described elsewhere [3-9]. At present, OLYMPUS programs are circulated and published as system-independent Standard Program Files (SPF) [6-10]. A subsequent development may be to distribute these programs in more compact versions which can be converted into SPF form by means of appropriate generators or preprocessors.

The general structure of the SPF, illustrated in Table 1, corresponds closely to that of OLYMPUS programs previously published in Computer Physics Communications [1,2,5,10-19],and comprises four main sections concerned with:

1. Documentation
2. Common blocks
3. Fortran subprograms
4. Test data

Each section is subdivided into modules which are given standardized names. The following paragraphs describe in detail the conventions employed in each section of the SPF. Many of the figures used to illustrate the conventions were produced automatically by the COMPOSITOR [1] or the GENSIS generator [2]. The intention is to give an OLYMPUS program listing the structure and appearance of a textbook in order to make large Fortran programs as intelligible as possible to the reader.

Although the conventions that we shall describe in this article may appear at first sight rather detailed for manual use, experience has shown that such attention to detail is essential if a neat and consistent layout is to be achieved, particularly when several people collaborate on the same program. Furthermore it is necessary to establish a precise set of SPF conventions in order to be able to specify the output format that the COMPOSITOR, GENSIS and other generators should produce. Generator input is intended to be in a progressively more relaxed style, thus eventually relieving the OLYMPUS programmer of much of the routine work.

Control Statements

It is convenient at this point to mention the 14 types of system-independent control statement that are used within the SPF itself, shown in Table 2. These all start with 'C/' in columns 1 & 2 so that they are treated as comments by a compiler and should not interfere with control characters used by any particular operating system, but may be recognized by OLYMPUS utilities which process the modules of the SPF. Other system-independent and system-dependent control statements can be used outside the SPF in order to control the type of run that is required.

Comments

Except for initial letters, highlighting etc, comments are normally in lower case to distinguish them clearly from the Fortran statements.

Standard Comment Lines

We list in Table 3 the 5 types of standard Fortran comment line that are used to improve the layout and to annotate the program.

Their definitions are as follows, (columns 73-80 being reserved for the line-numbering required by Computer Physics Communications). Examples are shown in Fig.1 and elsewhere in this note:

(a) C-blank line

Only the character "C" in column 1; columns 2-72 blank.

(b) Ruled line

Character "C" in column 1, "-" (minus) in columns 2-70, columns 71-72 blank.

(c) Heading line

Characters "CL" in columns 1-2.
Label (if any) starting in column 17 and followed by"."
Heading starting in column 28.
A heading line is always preceded by a ruled line and followed by a C-blank line.

(d) Sub-heading line

Characters "CL" in columns 1-2.
Decimal label (if any) starting in column 21.
Sub-heading starting in column 30.
A sub-heading line is always preceded by a C-blank line, (except in a COMMON block where it is preceded by a ruled line - see §2.1 and Fig.9).
(N.B. if a heading line is immediately followed by a sub-heading line, only one intermediate C-blank line is used).

(e) Version line

Character "C" in column 1.
Blank in column 2.
"Version" in columns 3-9
Blank in column 10
The following information in columns 11-60 with intermediate blanks but otherwise free format:

(i) Version number and code characters
(ii) Date
(iii) Authorship
(iv) Laboratory

A version line defines the version of the corresponding module. An example is:

C VERSION 3A 12/Nov/76 MHH/KVR/JPC/ANO Culham

This style for the date avoids conflict between British and American usage. The version number ("3" in this instance) is followed either by "*" which indicates a universal module that precisely conforms to ANSI Fortran 66 and can be used on any computer, or by a code letter indicating the type of computer for which it is intended. Because of the very large number of Fortran 66 dialects now in existence it has proved impracticable to allocate these code letters in a standard way and their significance should be indicated

in the INTRO module.

The remaining sections 1-4 of this article are correlated with the sections 1-4 of the Standard Program File.

1. DOCUMENTATION

All OLYMPUS programs contain at least seven documentation modules in the order indicated in Table 1, each preceded by the corresponding control statement. The purpose of each module is as follows:

1.1 Title page

Examples of title pages are shown in Fig.2. The title page is always given the name

C/ MODULE TITLE

and contains the following information in standardized form, with a layout constructed by GENSIS [2].

(a) Title
(b) Authorship
(c) Address of author
(d) Notice to users
(e) Version line

Fig.2a contains the notice to users for a program that is intended for publication in Computer Physics Communications, while Fig.2b contains the version for a program that has already been published.

1.2 Introduction to Program

This module is intended to contain a brief introduction to the program together with references or 'pointers' to program commentaries etc. Fig.3 shows an example taken from the ATHENE 1A code [10]. Another recommended technique which may be mentioned here is to cross-reference the listing and a published write-up so that the two can be read together. A good example appears in the ATHENE 1A code where the listing refers to equation numbers in the write-up while the write-up refers to sub-program numbers in the code as illustrated in Fig.4. This technique enables the INTRO module itself to be kept quite short.

Note that modules INTRO, INDSUB, INDCOM, INDVAR are all headed by the standard combination:

(i) C/ MODULE statement
(ii) Ruled line
(iii) Heading line
(iv) C-blank line
(v) Version line
(vi) C-blank line

1.3 Index of subprograms

Module INDSUB is an index of subprograms used in the code excluding CYCLOPS utility routines contained in the OLYMPUS library [4,5], e.g. MESAGE, IVAR etc). An example produced by GENSIS appears in Fig.5.

The subprograms of an OLYMPUS code are divided into classes according to their function: the seven possible classes are summarised in Table 4. Each subprogram of a particular class is assigned a decimal identification number whose first digit indicates that class, e.g. Ø.5, 2.11, A16 etc. In module INDSUB the subprograms of each class appear in groups, in the order indicated in Table 4, in the following format:

(i) A ruled line followed by a heading line

CL INDEX OF SUBPROGRAMS

starting in column 28 with the characters CL in columns 1 and 2; in INDSUB as in all other documentation modules, headings are not labelled.

(ii) Each group of subprograms is preceded by a sub-heading line and ends with a C-blank line. The sub-heading line takes the form

CL CALCULATION CLASS 2

As mentioned in the Introduction the sub-heading starts in column 30 with the characters CL in columns 1 and 2; sub-headings are not labelled here but the class is indicated and starts in column 50.

(iii) Each subprogram is described by a single identification line in the form

C <entryname (<parameters>)> <subprogram label> <decimal classification>

The entryname starts in column 4 and is followed, in parenthesis, by the number of formal parameters (if any); the subprogram label starts in column 20 while the decimal classification number starts in column 69. The entries are arranged in decimal order.

1.4 Index of COMMON Blocks

INDCOM is an index of all COMMON blocks used in a program, excluding the standard OLYMPUS blocks. An example is shown in Fig.6.

Labelled COMMON is classified into groups as indicated in Table 5 and then further into blocks. Each block is allocated a decimal identification number [Cr.s] where r denotes the group to which the block belongs ($1 \leq r \leq 8$) and s is the serial number within the group ($1 \leq s \leq 9$). Group 1 is reserved for general OLYMPUS use while group 7 can be used for arbitrary purposes by an individual programmer; group 8 is reserved for general-purpose blocks introduced by a local installation and group 9 is reserved for blank COMMON. Thus, for example, the OLYMPUS blocks COMBAS and COMDDP are labelled C1.1 and C1.9 respectively. Blank COMMON is C9.0 and has the name COMBLA.

The index of COMMON blocks is arranged in decimal order and is constructed in a similar format to that of §1.3 INDSUB, viz:

(i) A ruled line followed immediately by a heading line (§1)

```
CL                                  INDEX OF COMMON BLOCKS
```

(ii) Each group of COMMON blocks is preceded by a sub-heading line and ends with a C-blank line. The sub-heading line takes the form

```
CL                    C2.1      TOKAMAK PARAMETERS
```

(iii) Each COMMON block is described by a line in the format

C <block name> <block subheading> <decimal identification>

e.g.

```
C COMBAS              BASIC SYSTEM PARAMETERS                    C1.1
```

The block name starts in column 4, the sub-heading in column 20 and the decimal identification in column 68.

1.5. Alphabetic Index of COMMON Variables

Every COMMON variable used in an OLYMPUS program is described in an alphabetic index of variables contained in a module called INDVAR. This index is intended to help users interpret any COMMON variable very easily and to locate the particular block in which it resides. An example is show in Fig.7; the format is as follows:

(i) A ruled line followed by a heading

```
CL                              ALPHABETIC INDEX OF COMMON VARIABLES
```

(which itself is followed by a C-blank line)

(ii) Each variable is described by a line of the form

```
C  <identifier><dimension> <purpose>  <type>  <COMMON block identifica-
↑↑                         ↑          ↑       ↑                    tion>
1 4                        20         66      69
```

e.g.

```
C  RATIOZ(NEQ,NEQ)         Ratio of atomic numbers squared        RA 2.1
```

The identifier and dimension(s) (if any) start in column 4: if this field extends beyond column 18, (e.g. if the array has 2 symbolic dimensions each of 6 characters) then the remaining fields appear on the next line. The type is indicated by a code letter "C,D,R,I,L" as described below in §2.1 and an array by "A". Where appropriate the dimensions of arrays are given symbolic names.

1.6 Index of COMMON Variables

In addition to the alphabetic index, OLYMPUS programs also contain an index of COMMON variables by blocks. This index is similar to that described in §1.5 but here the variables appear in the same order as in the COMMON blocks (§2.1). An example is shown in Fig.8.

INDBLK is divided into sub-sections each of which is preceded by a sub-heading line. This line corresponds identically with that used in the COMMON block itself, i.e. the sub-heading starts in column 30 and the label in column 21. The sub-heading is followed by a version line which again must correspond identically with that in the COMMON block. The version line is followed by the COMMON block name in the format (e.g.):

```
C      COMMON/COMBAS/
```

with the character string COMMON/COMBAS/ starting in column 8. This line is followed by a C-blank line and the variable index in the same format as that of §1.5.

1.7 Master Index for GENSIS Generator

Module MINDEX is a Master Index which contains enough information about the program documentation and the data structure to allow the GENSIS Generator [2] to construct those sections of the program that depend on this information. An example is shown in Fig. 9, and its structure is explained in detail in a following article [2]. It is included in the SPF so that the user can carry out certain modifications in a consistent way by editing MINDEX and then re-running GENSIS, rather than by editing all the individual modules themselves.

2. COMMON BLOCKS

The division of COMMON blocks into groups with decimal identification numbers is already defined in §1.4. The following paragraphs deal with the conventions for constructing COMMON blocks,which are implemented by GENSIS.

Note that, in any OLYMPUS program only one copy of each block is normally used throughout the code. A particular block is inserted where required by means of a control statement of the form (e.g.):

```
C/ INSERT COMPHY
```

which is intercepted by a preprocessor and results in substitution of the named module. This makes the source code shorter and the data structure easier to understand. Moreover, any updating of COMMON is greatly simplified since only one file or set of files need be changed. This facility is not confined solely to COMMON block substitution; it is used to substitute any section of code common to several subprograms. e.g. EQUIVALENCE blocks etc.

Many computer systems now have a file substitution facility available, and at the Culham Laboratory the $INSERT facility of the PRIME computer system [23] is normally employed for this purpose. A preprocessor has however been published in Computer Physics Communications as part of the IBM 360/370 OLYMPUS package [14] and this could be adapted for SPF use by changing the previous format

```
// SUBSTITUTE <name>
```

to

```
C/ INSERT <name>
```

2.1 Structure of COMMON blocks

The variables and arrays contained in a particular block must appear in the following order of types:

	Type	Code Letter
1)	complex	C
2)	double precision	D
3)	real	R
4)	integer	I
5)	logical	L

One reason for adopting this convention is that the process of clearing the store becomes particularly simple, and is easily automated and independent of the arithmetic on a particular machine. Fig.10 shows an example of the automatically-generated subprogram <1.2>CLEAR to illustrate the usefulness of this rule; the structure of the COMMON declarations which appear in this example is discussed further in §2.2. This is one example where block substitution is not used.

Variables of a particular type are grouped in alphabetic order; the characters C,D,R,I,L are used as continuation symbols (in column 6) corresponding to the type.

For the sake of clarity, type declarations and array dimensions appear separately from the COMMON list; Fig.11 shows a typical example. Note that all declarations start in column 8. The order of declarations is as follows:

1) COMMON
2) TYPE
3) DIMENSION

The layout conventions for COMMON blocks are:

1) COMMON module names normally start COM----.

2) Each block starts with a ruled line.

3) The ruled line is followed immediately by a sub-heading.
   The sub-heading label in this case starts with the character C, e.g.
   CL C1.1 BASIC SYSTEM PARAMETERS

4) The sub-heading is followed by a version line (see introduction)

5) The COMMON declaration.

6) The variable identifiers appear on continuation lines (with the continuation symbol indicating the variable type) with each name starting in a multiple of column 10.

7) Only variables of the same type appear on any one line.

8) Variables follow DIMENSION/TYPE declarations on continuation cards in the same format as the COMMON list (i.e. each identifier starts in a multiple of column 10).

2.2 Notation for variables and arrays

OLYMPUS defines the initial letter of variable and array names of different types; the convention is summarised in Table 6. Careful adherence to these rules for notation minimises the risk of error and, at the same time, allows the reader to see at a glance which variables are in COMMON and which are local. Further, the methodical use of symbolic rather than explicit arithmetic notation enables parameters to be readily updated if necessary.

For example, it is preferable to code a WRITE statement as

```
WRITE (NOUT,9100)———
```

rather than

```
WRITE (6,9100)———
```

2.3 Alternative COMMON Block Versions

It is sometimes convenient to use alternative versions of a particular block. This is permissable provided that the block length is always the same. For example, the standard OLYMPUS block COMBAS contains 25 variables comprising 113 computer words; thus in subprogram RECORD (used for reading or writing restart records) it is convenient to declare

```
COMMON/COMBAS/R11(3),I11(108),L11(2)
```

The same principle is involved in subprogram CLEAR already illustrated in Fig.10.

If an alternative version is to be used in several places in the program it should be included in the COMMON section of the SPF and distinguished by some convenient variation of the module name, e.g.

```
C/ MODULE COMVAR      (standard version)
C/ MODULE COMVRA      (alternative version)
```

The Fortran identifier must of course remain the same.

2.4 EQUIVALENCE BLOCKS

If a group of EQUIVALENCE statements appears in several subprograms, it can conveniently be treated like a COMMON block and inserted where needed by means of a C/ INSERT <name> card. No specific rules are laid down for the names, layout and order of these modules but they should conform to the general OLYMPUS pattern.

2.5 Miscellaneous Blocks

The C/ INSERT facility can be used to insert other modules within the program as required, if these are given appropriate names and placed within the COMMON section.

3. FORTRAN SUBPROGRAMS

Fortran modules are always given names of types indicated in Table 7, for example subprogram <2.6> has the module name C2S6 (which of course is distinct from its Fortran identifier). This is analogous to the numbering of chapters in a book and allows them to be located quickly in a listing.

Alternative versions of a particular subprogram append a letter A-Z at the end of the name, e.g.

```
C/ MODULE C2S6A
```

Each module is divided into the main parts indicated in Table 8 with a typical example illustrated in Fig.12. Parts marked* are optional.

3.1 Heading Part

Referring to the example of Fig.12 the layout conventions for the heading part are as follows:

1. C/ MODULE <name> statement

2. C-blank line

3. SUBROUTINE/FUNCTION declaration which itself is followed by another C-blank card: note that the SUBROUTINE/FUNCTION declaration starts in column 10.

4. Subprogram identification label and title, e.g.

```
C 2.11 MOVE NODES ON VORTEX BOUNDARIES
↑↑ ↑
1 3  8
```

5. C-blank line

6. Version line

7. C-blank line

8. Brief description of the subprogram and an index of dummy parameters followed by a C-blank line. (Not illustrated in Fig.12). The conventions are the same as for module INDVAR - but with the code letters I(input), O(output) or IO(input/output) starting in col.68. A '.' in col.2 indicates a comment line that is left unchanged by the COMPOSITOR; see ref.[1] for an example.

3.2 Non-Executable Part

The structure of the non-executable part is shown in Table 9. The order is mandatory for the standard version of OLYMPUS (although minor variations may be needed for specific types of machine). All items in Table 9 are optional. All declaration statements begin in column 8 and have the structure described in §2.1. Statement function definitions begin in column 10.

A frequently-used DATA statement (e.g. in subprogram <2.11>) is:

```
DATA      ICLASS,   ISUB    /2,11/
↑         ↑         ↑       ↑
8         20        30      40
```

This enables the actual parameters of EXPERT calls to be referred to symbolically, so that they do not have to be altered if the serial number of the subprogram has to be changed.

3.3 Executable Part

Unless it is very short, the executable part of a subprogram should be divided into sections and sub-sections. This is illustrated by the example

of Fig.12: the rules for constructing headings/sub-headings are already explained in the Introduction. Within each section (or sub-section) the following rules are applicable:

1. Comments always start in column 7 and are normally in lower case as explained in the Introduction. On average, each executable statement or group of related statements will be preceded by a suitable comment, unless (as in Fig.12) the sub-headings are already adequate.

2. Executable statements begin in column 10. Blanks within statements are optional but should be consistent.

3. Continuation lines are usually labelled sequentially 1,2,3--- in column 6, although alternative conventions may be used. (Note that "Ø" (zero) is not a permissable continuation symbol, that not more than 19 continuation lines are allowed, and that comments are not allowed within statements).

4. Statement labels are right-adjusted on column 5 and should not exceed 4 digits. Further, labels should correspond with the section/sub-section number in which they appear and be in numerically ascending order. For example, in section 3 labels are numbered 300-309, while in sub-section 3.2 label numbers are in the range 320-329. FORMAT statements are considered separately below.

Section 1 is frequently preceded by a preamble, containing a statement of the form (e.g):

```
IF (NLOMT2(ISUB)) RETURN
↑
10
```

which enables an immediate return to be made if the subprogram ISUB in class 2 is "switched-off" by setting the diagnostic variable NLOMT2(ISUB) = .TRUE. [5]. The preamble may also contain any necessary initialization, GO TO statements etc.

3.4. FORMAT Statements

FORMAT statements can, of course, appear anywhere in the source code. The convention preferred by OLYMPUS is that they should be collected together in section 9 at the end of the subprogram. Moreover, FORMAT statements are always labelled in the range 9000-9999. The statement numbers should be in ascending order.

3.5. Index of Local Variables and Arrays

If there are many local variables and arrays it may be convenient to include an alphanumeric index. The conventions are the same as for module INDVAR but with the COMMON block identification number and type omitted and '.' in col.2. Ref.[1] shows an example.

4. TEST DATA

This section contains a number of input data modules, one for each test run. The data for Test Run 3 (for example) is preceded by the control statement

An SPF processor then constructs a job file from each module, suitable for the particular operating system.

OLYMPUS programs use the NAMELIST facility for much of their input data This facility is, in fact, available on most of the major computer systems but is implemented in slightly different ways. NAMELIST data is therefore preceded and followed by the system-independent control statements

C/ NAMELIST<name>

C/ END NAMELIST

which are intercepted by the SPF Preprocessor and used to construct the format that is required.

SUPPORT

Any supporting modules that are not part of the program proper,but are required to make it run, e.g. standard subprograms, COMMON blocks or data files, may be included at the end of the package in one or more subpackages each preceded by

C/ SUPPORT <name>

and terminated by

C/ END SUPPORT

Examples occur in the packages for ATHENE 1 [18] and ATHENE 1A [10].

REFERENCES

[1] The OLYMPUS FORTRAN COMPOSITOR, M.H.Hughes and K.V.Roberts, CLM-P690, submitted for publication in Comput. Phys. Commun.

[2] The OLYMPUS FORTRAN GENERATOR, M.H.Hughes and K.V.Roberts, CLM-P689, submitted for publication in Comput. Phys. Commun.

[3] The Publication of Scientific Fortran Programs, K.V.Roberts, Comput. Phys. Comm. 1 (1969) 1.

[4] An introduction to the OLYMPUS System, K.V.Roberts, Comput. Phys. Comm. 7 (1974) 237.

[5] OLYMPUS - A Standard Control and Utility Package for Initial-value Fortran Programs, J.P.Christiansen and K.V.Roberts, Comput. Phys. Commun. 7 (1974) 245.

[6] Fortran, K.V.Roberts, in Software Engineering, Ch.5, ed. R.H.Perrott, Academic Press (London 1977).

[7] Experience with the OLYMPUS System, K.V.Roberts, in The Relationship between Numerical Computation and Programming Languages, ed. J.K.Reid, (North-Holland 1982).

[8] Computer Simulation using Particles, R.W.Hockney and J.W.Eastwood, McGraw-Hill, New York (1981), Chapter 3.

[9] Construction and Documentation of Software, K.V.Roberts NATO-ARI Microelectronics Conference, Les deux Alpes, France (March 1982).

[10] ATHENE 1A - A One-Dimensional Fusion Code, J.P.Christiansen, K.V.Roberts, V.A.Piotrowicz and J.W.Long, Comput. Phys. Commun. 23 (1981) 63.

[11] MEDUSA - A One-Dimensional Laser Fusion Code, J.P.Christiansen, D.E.T.F.Ashby and K.V.Roberts, Comput. Phys. Commun. 7 (1974) 271.

[12] TIMER - A Software Instrumentation Routine for Making Timing Measurements, M.H.Hughes and A.P.V.Roberts, Comput. Phys. Commun. 8 (1974) 118.

[13] OLYMPUS Restart Facilities, M.H.Hughes and K.V.Roberts, Comput. Phys. Commun. 8 (1974) 123.

[14] OLYMPUS and Preprocessor Package for an IBM 370/165, M.H.Hughes and K.V.Roberts, Comput. Phys. Commun. 9 (1975) 51.

[15] THALIA - A One-Dimensional Magnetohydrodynamic Stability Program using the Method of Finite Elements, K.Appert, D.Berger, R.Gruber, F.Troyon and K.V.Roberts, Comput.Phys. Commun. 10 (1975) 11.

[16] RAMSES - A Two-Dimensional, PIC Type, Laser Pulse Propagation Code, H.D.Dudder and D.B.Henderson, Comput. Phys. Commun. 10 (1975) 155.

[17] OLYMPUS Control and Utility Package for the CDC 6500, M.H.Hughes, K.V.Roberts and G.G.Lister, Comput.Phys. Commun. 10 (1975) 167.

[18] ATHENE 1 - A One-Dimensional Equilibrium-Diffusion Code, J.P.Christiansen, K.V.Roberts and J.W.Long, Comput. Phys. Commun. 14 (1978) 423.

[19] CASTOR 2: A Two-Dimensional Laser Target Code, J.P.Christiansen and N.K.Winsor, Comput.Phys.Commun.17 (1979)397.

[20] OLYMPUS for a BESM-6-The Development of Medium-Scale Programs within the OLYMPUS Framework: System OLYDES, P.Carl, Comput. Phys. Commun. 20 (1980) 165.

[21] OLYMPUS System and Development of its Pre-processor, M.Okamoto, T.Takeda,M.Tanaka,K.Asai,K.Nakano and I.Kawakami, Japan Atomic Energy Research Institute Report JAERE-M 7228, (1977). (In Japanese with program listing in English).

[22] The Regeneration of Fortran Software, K.V.Roberts, CLM-P683, submitted for publication Comp. Phys. Commun.

[23] The Fortran Programmer's Guide, PDR3057, PRIME Computer Inc. (November 1977), page 16-11.

TABLE 1. Structure of SPF

(Standard file names are indicated. All names comprise ≤ 6 characters beginning with a letter)

Section		
	C/ PACKAGE <name>	
	C/ PROGRAM <name>	
1.	C/ DOCUMENTATION	
	C/ MODULE TITLE (Title page)	
	C/ MODULE INTRO (Introduction to program)	
	C/ MODULE INDSUB (Index of subprograms)	
	C/ MODULE INDCOM (Index of COMMON blocks)	
	C/ MODULE INDVAR (Alphabetic Index of COMMON variables)	
	C/ MODULE INDBLK (Index of COMMON variables by block)	
	C/ MODULE MINDEX (Master Index for Generator)	
2.	C/ COMMON	
	C/ MODULE <name> COMMON or other module to be inserted in subprograms by means of C/ INSERT statement	*
3.	C/ FORTRAN	
	C/ MODULE <name> Fortran subprogram module	*
4.	C/ TEST DATA	
	C/ MODULE <name> Test data module	*
	C/ END PACKAGE	

* Any number of modules arranged in sequential order.

TABLE 2. SPF Control Statements

C/ PACKAGE <name>	Name of package
C/ PROGRAM <name>	Name of program
C/ DOCUMENTATION	Section 1
C/ COMMON	Section 2
C/ FORTRAN	Section 3
C/ TEST DATA	Section 4
C/ END PROGRAM	End of program
C/ SUPPORT <name>	Supporting modules
C/ END SUPPORT	End of support
C/ END PACKAGE	End of package
C/ MODULE <name>	Name of module
C/ INSERT <name>	Insert of common module
C/ NAMELIST <name>	Begin NAMELIST data
C/ END NAMELIST	End NAMELIST data

Here <name> is a Fortran-type identifier of 1-6 characters (A-Z,Ø-9), starting with a letter, and follows conventions which are discussed elsewhere in this article

TABLE 3. Standard Comment Lines

a.	C-blank line	Used to improve the layout
b.	Ruled line	Separates and emphasises the sections
c.	Heading line	Heading for sections
d.	Sub-heading line	Heading for sub-sections
e.	Version line	Defines the version of a module

TABLE 4. Classification of Subprograms

Class	Function
Ø	Control
1	Prologue(initialization)
2	Calculation
3	Output
4	Epilogue
5	Diagnostics
A-F*	Utilities

*Utility routines labelled A-F are reserved for individual programs while classes 6,7,N-T,V-Z are reserved for future expansion of OLYMPUS.

TABLE 5. Labelled COMMON groups

1	General OLYMPUS data
2	Physical problem
3	Numerical scheme
4	Housekeeping
5	Input-output and diagnostics
6	Text manipulation

TABLE 6. Initial Letters for Variable and Array Names

	REAL or COMPLEX	INTEGER (or HOLLERITH)	LOGICAL
Subprogram dummy arguments	P	K	KL
Common variable and array names	A-H,O,Q-Y	L,M,N	LL,ML,NL
Local variable and array names	Z	I	IL
Loop indexes		J	

TABLE 7. Fortran Module Names

Class	Name	Example	Control Statement
0-9	$C_m S_n$	C2S6	C/ MODULE C2S6
A-Z	U_n	U12	C/ MODULE U12

TABLE 8. Structure of a Fortran Module

1.	Heading part	
2.	Non-executable part	*
3.	Executable part	
4.	FORMAT statements	*
5.	Index of local variables and arrays	*
6.	END statement	

Parts marked * are optional.

TABLE 9. Structure of Non-Executable Part of Fortran Module

1.	IMPLICIT statement	
2.	Specifications of dummy arguments in the order DIMENSION, TYPE	
3.	EXTERNAL statements	
*4.	C/ INSERT statements for labelled COMMON blocks	
*5.	C/ INSERT statement for blank COMMON	
6.	Specifications for local variables in the order DIMENSION, TYPE	
7.	NAMELIST statements	
8.	Specifications for variables and arrays equivalenced to COMMON variables and arrays	in the order DIMENSION,TYPE
9.	Specifications for variables and arrays equivalenced to local variables and arrays	
*10.	C/ INSERT statements for EQUIVALENCE (COMMON)	
11.	EQUIVALENCE statements (local)	
12.	DATA statements	
13.	Statement function definitions	

* Versions of items 4,5 and 10 that are specific to individual subroutines are sometimes expanded in full rather than being inserted via C/ INSERT statements, see for example Fig.10.

```
        1              17  21     28 30          50              70

(a)  C-blank line

     C

(b)  Ruled line

     C-----------------------------------------------------------------

(c)  Heading lines

     CL                                INTRODUCTION
     CL                                INDEX OF SUBPROGRAMS
     CL                                INDEX OF COMMON BLOCKS
     CL                                ALPHABETIC INDEX OF COMMON VARIABLES
     CL                                INDEX OF COMMON VARIABLES
     CL                1.              Plasma pressure at half-integral points
     CL                4.              Parallel and perpendicular currents

(d)  Sub-heading lines

     CL                                  CONTROL              CLASS 0
     CL                                  OUTPUT               CLASS 3
     CL                                  UTILITIES            CLASS A
     CL                    1.            GENERAL OLYMPUS DATA
     CL                    3.            NUMERICAL SCHEME
     CL                    5.            INPUT-OUTPUT AND DIAGNOSTICS
     CL                    C2.1          Principal subscripted variables
     CL                    C4.1          Calculation control
     CL                    C5.5          Output variables
     CL                    2.1           Pitch at half-integral points
     CL                    4.3           Special treatment at origin

(e)  Version lines

     C VERSION 3A 12/Nov/76 MHH/KVR/JPC/ANO Culham
     C VERSION 1* 1/Jun/76 MLW Culham
```

Fig.1 Examples of Standard Comment Lines

Entries should begin in columns indicated, except for the date and subsequent entries on a version line. The format of the date is chosen to avoid conflict between British and American usage. The version line should not extend beyond column 60.

```
C/ MODULE TITLE
C
C                 20
C
C
C                 ********************
C                 *                  *
C                 *  A T H E N E  1 A *
C                 *                  *
C                 ********************
C
C
C                 A 1-D Pinch Diffusion Code
C
C                 by
C
C                 J.P. Christiansen, K.V. Roberts, J.W. Long,
C                 V.A. Piotrowicz, J.W. Johnston and A.A. Newton
C                                       40
C
C                                       Computational Physics Group
C     7                                 Culham Laboratory
C                                       Abingdon
C                                       Oxon OX14 3DB, England
C
C
C     This is a preliminary version of the program
C     The final version will be submitted to
C
C                 Computer Physics Communications
C
C     for inclusion in
C
C                 CPC Program Library
C                 School of Physics and Applied Mathematics
C                 Queens University
C                 Belfast BT7 1NN, N.Ireland
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
```

Fig.2a Title Page for Preliminary Version

The title page is produced automatically by the GENSIS generator and may then be edited as required. It is recommended that a disclaimer similar to that indicated should be included on all privately circulated programs. This and following examples relate to ref.[10], with some editions, except where indicated.

```
C/ MODULE TITLE
C
C
C
C
C                 ********************
C                 *                  *
C                 *  A T H E N E  1 A *
C                 *                  *
C                 ********************
C
C
C                 A 1-D Pinch Diffusion Code
C
C                 by
C
C                 J.P. Christiansen, K.V. Roberts, J.W. Long,
C                 V.A. Piotrowicz, J.W. Johnston and A.A. Newton
C
C
C                                       Computational Physics Group
C                                       Culham Laboratory
C                                       Abingdon
C                                       Oxon OX14 3DB, England
C
C
C     Further copies of the published version of this program
C     including any subsequent corrections or amendments notified
C     by the author, should be obtained on magnetic tape directly
C     from
C
C                 CPC Program Library
C                 School of Physics and Applied Mathematics
C                 Queens University
C                 Belfast BT7 1NN, N.Ireland
C
C     An application form appears in each issue of
C
C                 Computer Physics Communications
C
C     Catalogue number - ABQI
C     Writeup reference - Comput. Phys. Commun. 23 (1981) 63-80
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
```

Fig.2b Title Page for Published Version

The title page is produced automatically by the GENSIS generator and may then be edited as required. It is recommended that users be encouraged to obtain 'clean' copies of published programs from the CPC Library and not directly from the author.

```
C/ MODULE INTRO
C-------------------------------------------------------------------------
CL                          INTRODUCTION
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
CL              1.          Initial Version 1, CPC 14 (1978) 423
C
C       For details of how to run the ATHENE 1 code on any computer,
C     and in particular the use of auxiliary components or support
C     modules, the conversion of C/ control statements and the
C     conversion to double length, see the Computer Physics
C     Communications paper.
C
C       With an empty namelist NEWRUN the code will perform Standard
C     Test 1 using the default parameters set in subroutine PRESET and
C     shown in Table 5 of the writeup. This Table should be consulted
C     when alternative parameter settings are used.
C
C       The ATHENE 1 code can be run starting from experimental data,
C     if the switch NLIXPT is set to .TRUE. in namelist NEWRUN. This
C     causes subroutine EXPROF (CISIO) to be called from INITAL and to
C     read in two namelists MESH and EXPT which the user must then
C     supply.
C
C       Namelist MESH should contain the radial coordinates at which
C     the experimentally determined values (see below) have been
C     measured. Those radii not specified in namelist MESH take the
C     values given by INITAL and correspond to an equidistant mesh.
C     Notice that the values supplied in namelist are stored in the
C     array RI, e.g. the statements
C
C       RI(1) = 0.0, 0.12, 0.25, 0.37, 0.5,
C       RI(9) = 1.1, 1.14, 1.16,
C
C     will leave RI(6)-RI(8) at the values defined in INITAL. If a user
C     accidentally specifies RI(J).GT.RI(J+1) the program stops.
C
C       The coordinates RI specified in namelist MESH are now used to
C     define the half-points RH and the spacings DR. Subprogram INITAL
C     then calculates the radially dependent profiles of AM, AZ, XNE,
C     TE, TI, BZ and BT according to the analytic functions.
C
C       Any of the above mentioned 7 main variables which have been
C     measured experimentally at the coordinates RI can now be supplied
C     to ATHENE 1 through the namelist EXPT. Again those variables not
C     specified in namelist EXPT will take the values defined by INITAL.
C     Linear interpolation is used to obtain values at half-points such
C     that upon exit from EXPROF all 7 main variables are defined at the
C     points RH.
C
CL              2.          Extensions made in Version 1A
C
C     Equation numbers followed by an A in the listing of ATHENE 1A
C   refer to the second write-up of the extensions to ATHENE 1,
C   published in CPC 23 (1981) 63. Otherwise equations 1-74 refer to
C   the first writeup, and 75-139 to the second.
C
C     The array FACTOR allows classical transport coefficients to be
C   scaled. The factors apply as follows
C
C   FACTOR (1)   Parallel resistivity
C          2     Perpendicular resistivity
C          3     Electron thermal conductivity
C          4     Ion thermal conductivity
C          5     Equipartition time
C          6     Bremsstrahlung rate
C          7     Dynamo coefficients alpha and beta
C          8     Wall values of alpha and beta
C          9     Thermoelectric coefficient
C          10
C          11    Control 1-e deposition of DD reaction product
C          12    Control 1-e deposition of DT reaction product
C          13    Class. diff. coeff. for fusion reaction products
C          14    Vdrift/Vthe value causing streaming losses
C          15    Scales par. resistivity when streaming
C          16    Scales par. resistivity when streaming
C
C     Scaling factors for anomalous transport are stored in CCA.
C   Scaling factors for turbulent transport are stored in CCT.
C   Empirical scaling laws for TAUE can be used to define DIFA. These
C   are selected via NEMPIR as follows
C
C   NEMPIR = 1        General scaling with const. in CCEMP
C          = 2,3,4    Hugill-Sheffield scaling law
C          = 5,6      Alcator scaling laws
C          = 7        Pfeiffer-Waltz scaling laws
C
C     The impurity package included calculates the atomic physics,
C   ionisation, recombination, line radiation, with a full time
C   dependent treatment. All levels of ionization are calculated and
C   so far Oxygen and Iron have been taken into account. Other
C   impurity elements, e.g. carbon or nitrogen, can also be included.
C
C     The dynamo calculation is based on specific assumptions for the
C   alpha and beta coefficients. If it is used in conjunction with
C   the circuit package then it may be necessary to allow for more
C   mesh points to be added at the boundary if a pinch configuration
C   is being set up.
C
```

Fig.3 Program Introduction

This has been produced with the aid of a word-processor utility. The use of lower-case makes it more readable. In principle it might be expanded considerably, thus reducing the necessary length of the published program writeup.

```
C/ MODULE C2S19
C
      SUBROUTINE EQUIPT
C
C 2.19 Equipartition rate
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
C/ INSERT COMBAS
C/ INSERT COMFUN
C/ INSERT COMDDP
C/ INSERT COMMHD
C/ INSERT COMNC
C/ INSERT COMESH
C-------------------------------------------------------------------------
      DATA        ICLASS,   ISUB     /2,19/
C
        IF(NLOMT2(ISUB)) RETURN
C-------------------------------------------------------------------------
CL              1.
C
C     The method used and the equation numbers with * refer to
C     J. Comp. Phys. vol 17, p 332. No iterations are made and
C     we use here Te-Ti rather than Ti-Te as in the reference.
C     This means that the quantities Beta, CV, Phi are changed
C     to conform with the calculation of Te-Ti.
        ZK = FCK * 10.0 ** FXK
C     At t=0 Te=Ti. We estimate the previous difference in the
C     ion and electron source terms from the ohmic heating rate
C     since iterations are not being used
        IF(NSTEP.GT.1) GO TO 102
        DO 101 J=1,N
        ZBETA = 1.0 + AZ(J)
  101   EQUIP(J) = QE(J) / ZBETA
  102   CONTINUE
C
        DO 141 J=1,N
C
C     The quantity Beta (Eq.21*) is 1+Ne/Ni = 1 + Z
        ZBETA = 1.0 + AZ(J)
C
C     Epsilon is Dt/Tauequ  (Eq.20*)
        ZEPS = DT / TAUEQU(J)
C
C     Exponent  (Eq.17*)
        ZEXP = ZEPS * ZBETA
C
C     The expression for CV
        ZCV = XNE(J) * ZK / GAMIN1
C     The present temperature difference
        ZTEI2 = TE(J) - TI(J)
C     The term Phi (Eq.22*). Notice that the difference (Te-Ti)
C     is assumed to be the same as (Te2-Ti2)
        ZPHI = ZBETA * EQUIP(J) / ZCV
C     Now examine exponent
        IF(ZEXP.LT.EXPMIN) GO TO 110
        IF(ZEXP.GT.EXPMAX) GO TO 130
        GO TO 120
C
CL                  1.1     Small rate  (Taylor expand eq.18*)
  110   CONTINUE
        ZTEI = ZTEI2
        GO TO 140
C
CL                  1.2     Medium rate  (Eq.18*)
  120   CONTINUE
        ZS = ZPHI * DT / ZEXP
        ZRELAX = (1.0 - EXP(-ZEXP)) / ZEXP
        ZTEI = (ZTEI2 - ZS) * ZRELAX + ZS
        GO TO 140
C
CL                  1.3     High rate  (limit of eq.18*)
  130   CONTINUE
        ZS = ZPHI * DT / ZEXP
        ZRELAX = 1.0 / ZEXP
        ZTEI = (ZTEI2 - ZS) * ZRELAX + ZS
C
CL                  1.4     Energy exchange rate (Eq.32)
  140   CONTINUE
        EQUIP(J) = ZCV * ZTEI / TAUEQU(J)
  141   CONTINUE
C
        RETURN
        END
```

Fig.4 Cross-Referencing by Equation Numbers

A documentation technique of great practical value is to include equation numbers of a published write-up as "pointers" in the listing. The decimal subprogram, section and subsection numbers of the code can be similarly included in the write-up. This cross referencing enables the reader to find his way easily when reading the listing and the write-up together, and reduces the number of comments that need to be included in the code.

```
C/ MODULE INOSUB
C-------------------------------------------------------------------
CL                          INDEX OF SUBPROGRAMS
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
CL                            PROLOGUE            CLASS 1
C  LABRUN          Label the run                                  1.1
C  CLEAR           Clear variables and arrays                     1.2
C  PRESET          Set default values                             1.3
C  DATA            Define data specific to run                    1.4
C  AUXVAL          Set auxiliary values                           1.5
C  INITAL          Define physical initial conditions             1.6
C  START           Start or restart the run                       1.8
C  FUNCT(2)        Define initial dependent physics arrays        1.9
C  EXPROF(1)       Obtain initial profiles from experiment        1.10
C  AUXMSH          Derive auxiliary mesh variables                1.11
C
CL                          CALCULATION         CLASS 2
C  STEPON          Step on the calculation                        2.1
C  RZERO(1)        Reset energy rates, coeffs. and sources        2.2
C  SWAP(1)         Copy current values to old positions           2.3
C  FIELDS          Field diffusion and circuit control            2.4
C  JOULE           Joule and turbulent heating                    2.5
C  PLASMA          Plasma processes                               2.6
C  DIFUSE          Thermal diffusion                              2.7
C  ENERGY(1)       Energy calculation                             2.8
C  EQUIL           Relaxation to pressure equilibrium             2.9
C  BPROG           Programming of vacuum fields                   2.10
C  TIMSTP          Most restrictive timestep                      2.11
C  AUXQ            Auxiliary quantities (pitch etc.)              2.12
C  MHDCFF          Transport coefficients                         2.13
C  MISCEL          Miscellaneous diagnostics                      2.14
C  DYNAMO          Dynamo effects                                 2.15
C  FLDIFF          Magnetic field diffusion                       2.16
C  JZT(4)          Calculate currents (Curl B)                    2.17
C  EZT(3)          Calculate electric fields                      2.18
C  EQUIPT          Equipartition rate                             2.19
C  RADIAT          Radiation losses                               2.20
C  FUSION          Fusion processes                               2.21
C  GAUSS(6)        Gauss elimination                              2.22
C  PRESS           Evaluate pressure                              2.23
C  GETXSI          Calculate displacement vector Xsi              2.24
C  MESH            Reset mesh                                     2.25
C  ADBAT           Adiabatic changes to main variables            2.26
C  SUYDAM(1)       Diffusion from Suydam instabilities            2.27
C  DANOML(1)       Anomalous diffusion                            2.28
C  LINER           Include liner resistivity                      2.29
C  GASPUF(1)       Gas puffing rates for all ions                 2.30
C  DIVF(3)         Flux divergences                               2.31
C  FTHERM(1)       Thermo-electric field and flux                 2.32
C  EXAM(1)         Examine principal variables                    2.33
C  NEWPNT          Add or remove meshpoints                       2.34
C  IMP(3)          Impurity control program                       2.35
C  TIMEDP          Record time-dependence                         2.36
C  PPARAM(1)       Plasma composition parameters                  2.37
C
CL                            OUTPUT              CLASS 3
C  SELECT(2)       Select variables for output                    3.2
C  SCALE(2)        Scale factors for graphical output             3.3
C  MADHOC(1)       Ad hoc output for graphical output             3.11
C  PILINE(5)       Print 1 line of diagnostics                    3.21
C
CL                            EPILOGUE            CLASS 4
C  TESEND          Test for completion of run                     4.1
C  ENDRUN          Terminate the run                              4.2
C
CL                            DIAGNOSTICS         CLASS 5
C  CLIST(2)        Print COMMON variables                         5.2
C  ARRAYS(2)       Print COMMON arrays                            5.3
C
```

Fig.5 Index of Subprograms

This index is produced by the GENSIS generator and the correct columns are defined in §1.3. The number of arguments is indicated in parentheses. (Extract).

```
C/ MODULE INOCOM
C-------------------------------------------------------------------
CL                          INDEX OF COMMON BLOCKS
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
CL                    1.      GENERAL OLYMPUS DATA
C  COMBAS          Basic system parameters                        C1.1
C  COMDDP          Development and diagnostic parameters          C1.9
C
CL                    2.      PHYSICAL PROBLEM
C  COMCON          Physics control, Switch-on = .TRUE.            C2.1
C  COMMHD          MHD variables                                  C2.2
C  COMPHY          Physics and diagnostics                        C2.3
C  COMANO          Anomalous effects on transport                 C2.4
C  COMRUN          Parameters defining a run                      C2.5
C  COMPOS          Composition of plasma                          C2.6
C  COMIMP          Impurity control                               C2.7
C  COMWAL          Wall physics                                   C2.8
C  COMFUS          Thermonuclear fusion                           C2.9
C
CL                    3.      NUMERICAL SCHEME
C  COMNC           Numerical control parameters                   C3.1
C  COMESH          Mesh and auxiliary variables                   C3.2
C
CL                    4.      HOUSEKEEPING
C  COMWRK          Working space and spare buffers                C4.1
C  COMDIM          Maximum dimensions                             C4.2
C  COMMOD          Modifications to code                          C4.3
C
CL                    5.      INPUT-OUTPUT AND DIAGNOSTICS
C  COMPUT          Time-dependent output variables                C5.1
C
C                          ADDITIONAL COMMON BLOCKS
C  COMFUN          Fundamental constants
C  COMOUT          Standard output parameters
C  COMCFF          MHD-coefficients and plasma parameters
C  COMOX           Data for oxygen impurities
C  COMFE           Data for iron impurities
C
```

Fig.6 Index of COMMON Blocks

This index is produced by the GENSIS generator and the correct columns are defined in §1.4.

```
C/ MODULE INDVAR
C-------------------------------------------------------------------------
CL                        ALPHABETIC INDEX OF COMMON VARIABLES
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
C  AK(MAXVAR)          ** Control Dt by variations of variables      RA 3.1
C  AKG                 ** Control Dt by variations of Dt             R  3.1
C  ALPHAT(MAXDIM)      Dynamo coefficient                            RA 2.4
C  AM(MAXDIM)          Atomic mass number                            RA 2.2
C  AM2(MAXDIM)         AM at previous time level                     RA 3.2
C  AMX1                ** Atomic mass of X1                          R  2.6
C  AMX2                ** Atomic mass of X2                          R  2.6
C  AMX3                ** Atomic mass of X3                          R  2.6
C  AMX4                ** Atomic mass of X4                          R  2.6
C  AMX5                ** Atomic mass of X5                          R  2.6
C  AZ(MAXDIM)          Atomic charge number                          RA 2.2
C  AZ2(MAXDIM)         AZ at previous time level                     RA 3.2
C  AZEFF(MAXDIM)       Atomic Zeff                                   RA 2.2
C  AZERO(MAXDIM)       Array of zeroes                               RA 3.1
C  AZSQ(MAXDIM)        Mean value of Z**2                            RA 2.2
C  AZSQ2(MAXDIM)       Previous value of AZSQ                        RA 3.2
C  AZX1                ** Atomic Z of X1                             R  2.6
C  AZX2                ** Atomic Z of X2                             R  2.6
C  AZX3                ** Atomic Z of X3                             R  2.6
C  AZX4                ** Atomic Z of X4                             R  2.6
C  AZX5                ** Atomic Z of X5                             R  2.6
C  BBT(MAXCRC)         ** Control constants (Bt wall)                RA 2.5
C  BBZ(MAXCRC)         ** Control constants (Bz wall)                RA 2.5

C  DBT                 Bt flux increment in vacuum interspace        R  2.2
C  DBTVEL              Bt-flux increment during relaxation           R  2.8
C  DBZ                 Bz flux increment in vacuum interspace        R  2.2
C  DBZVEL              Bz-flux increment during relaxation           R  2.8
C  DIFA(MAXDIM)        Anomalous diffusion coefficients              RA 2.4

C  MAXCON              Initialisation of physics                     I  4.2
C  MAXCRC              Circuit variables                             I  4.2
C  MAXDIM              Physics and mesh variables                    I  4.2
C  MAXGPA              Dimension of GPA                              I  4.2
C  MAXV2               MAXVAR-10                                     I  4.2
C  MAXVAR              No. of physics variables                      I  4.2
C  MR(MAXVAR)          Location of fastest changes                   IA 3.1
C  N                   * Number of cells or mesh intervals           I  3.2
C  NAMES(MAXVAR)       Names of principal variables                  IA 3.1
C  NBTPRG              ** Bt (wall) programme selector               I  2.5
C  NBZPRG              ** Bz (wall) programme selector               I  2.5
C  NCELDT              Cell no. restricting Dt                       I  3.1
C  NCONDT              Condition restricting Dt                      I  3.1
C  NDEVIC(2)           Device identifier (optional)                  IA 2.5
C  NEMPIR              ** Select empirical scaling law               I  2.4
C  NIT                 Current iteration cycle                       I  3.1

C  WRAD                Total energy lost by radiation                R  2.3
C  WRADC               Bremsstrahlung loss                           R  2.3
C  WRADI               Recombination energy                          R  2.3
C  WRADL               Impurity radiation loss                       R  2.3
C  WRADR               Recombination radiation loss                  R  2.3
C  WRELAX              Total energy consumed by relaxation           R  2.3
C  WTOTAL              Total energy                                  R  2.3
C  WTRUNC              Energy truncated                              R  2.3
C  WTURB               Total energy generated by turbulence          R  2.3
C  WTVAC               Vacuum energy of Bt                           R  2.3
C  WWALFA              Alpha wall loss                               R  2.9
C  WWE                 Total energy (electron wall loss)             R  2.3
C  WWI                 Total energy (ion wall loss)                  R  2.3
C  WZERO               Total energy in system at t=0                 R  2.3
C  WZVAC               Vacuum energy of Bz                           R  2.3
C  XBETAO              ** Maximum permitted beta poloidal            R  2.5
C  XBETAT              ** Maximum permitted beta axial               R  2.5
C  XELINE              Electron line density                         R  2.3
C  XILT                Liner current (theta)                         R  2.3
C  XILZ                Liner current (z)                             R  2.3
C  XIONIZ(MAXDIM)      Total impurity ionization rate                RA 2.2
C  XIT                 Poloidal current                              R  2.3
C  XIZ                 Total current Iz                              R  2.3

C  YPIN(MAXDIM)        Power input                                   RA 5.1
C  YPOUT(MAXDIM)       Power output                                  RA 5.1
C  YTHETA(MAXDIM)      Pinch parameter                               RA 5.1
C  YTIME(MAXDIM)       Time values                                   RA 5.1
C  ZERO                Value of zero                                 R  3.1
C
```

Fig.7 Alphabetic Index of COMMON Variables
This index is produced by the GENSIS generator and the correct columns are defined in §1.5. (Extract)

```
C/ MODULE INDBLK
C-------------------------------------------------------------------------
CL                          INDEX OF COMMON VARIABLES
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
CL                  C2.1     Physics control. Switch-on = .TRUE.
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C      COMMON/COMCON/
C
C  NLANOM              ** Anomalous diffusion                        L  2.1
C  NLBURN              ** Burn-up of fuel                            L  2.1
C  NLCIRC              ** External circuit                           L  2.1
C  NLCLAS              ** Classical diffusion                        L  2.1
C  NLDRFT              ** Drift wave effects                         L  2.1
C  NLDYNA              ** Dynamo effects                             L  2.1
C  NLECON              ** Electron thermal conduction                L  2.1
C  NLEQUI              ** Ion-electron energy equipartition          L  2.1
C  NLFUSE              ** Thermonuclear fusion                       L  2.1
C  NLICON              ** Ion thermal conduction                     L  2.1
C  NLIMP               ** Impurity calculation                       L  2.1
C  NLINER              ** Include liner resistance                   L  2.1
C  NLJXB               ** Pressure equilibrium                       L  2.1
C  NLNPT               ** Allow for new points on mesh               L  2.1
C  NLNTRL              ** Calculation of neutrals                    L  2.1
C  NLOHM               ** Ohmic heating                              L  2.1
C  NLRAD               ** Radiation (bremsstrahlung)                 L  2.1
C  NLSUYD              ** Suydam stability calculation               L  2.1
C  NLTHET              ** Thermoelectric field and flux              L  2.1
C
CL                   C2.2     MHD variables
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C      COMMON/COMMHD/
C
C  AM(MAXDIM)          Atomic mass number                            RA 2.2
C  AZ(MAXDIM)          Atomic charge number                          RA 2.2
C  AZEFF(MAXDIM)       Atomic zeff                                   RA 2.2
C  AZSQ(MAXDIM)        Mean value of Z**2                            RA 2.2
C  BT(MAXDIM)          Magnetic field (theta-component)              RA 2.2
C  BZ(MAXDIM)          Magnetic field (z-component)                  RA 2.2
C  DBT                 Bt flux increment in vacuum interspace        R  2.2
C  DBZ                 Bz flux increment in vacuum interspace        R  2.2
C  DIVFE(MAXDIM)       Electron thermal loss rate                    RA 2.2
C  DIVFI(MAXDIM)       Ion thermal loss rate                         RA 2.2
C  EFDR                Effective Dr of vacuum interspace             R  2.2
C  EFRDR               Effective r*Dr of vacuum interspace           R  2.2
C  EQUIP(MAXDIM)       Energy equipartition rate (e-i)               RA 2.2
C  ET(MAXDIM)          Electric field (Theta-component)              RA 2.2
C  ETAII(MAXDIM)       Parallel Eta                                  RA 2.2
C  ETAL(MAXDIM)        Perpendicular Eta                             RA 2.2
C  ETATT(MAXDIM)       Electrical resistivity                        RA 2.2
C  ETATZ(MAXDIM)       Electrical resistivity                        RA 2.2
C  ETAZZ(MAXDIM)       Electrical resistivity                        RA 2.2
C  EZ(MAXDIM)          Electric field (z-component)                  RA 2.2
C  GAMIN1              Gamma - 1.0                                   R  2.2
C  GAMMA               Ratio of specific heats                       R  2.2
C  QE(MAXDIM)          Ohmic and turbulent heating (electrons)       RA 2.2
C  QI(MAXDIM)          Turbulent heating (ions)                      RA 2.2
C  RAD(MAXDIM)         Total radiation loss rate (e-i)               RA 2.2
C  RADC(MAXDIM)        Total bremsstrahlung rate                     RA 2.2
C  RADI(MAXDIM)        Recombination energy rate                     RA 2.2
C  RADL(MAXDIM)        Total impurity radiation rate                 RA 2.2
C  RADR(MAXDIM)        Recombination radiation rate                  RA 2.2
C  RH(MAXDIM)          Coordinates of cell centres                   RA 2.2
C  RHDR(MAXDIM)        Volume between integer points                 RA 2.2
C  RHO(MAXDIM)         Mass density                                  RA 2.2
C  RI(MAXDIM)          Coordinates of cell boundaries                RA 2.2
C  RIDR(MAXDIM)        Volume between half-points                    RA 2.2
C  SE(MAXDIM)          Source term in eq. for Te                     RA 2.2
C  SI(MAXDIM)          Source term in eq. for Ti                     RA 2.2
C  TAUEQU(MAXDIM)      Energy equipartition time                     RA 2.2
C  TE(MAXDIM)          Electron temperature                          RA 2.2
C  TI(MAXDIM)          Ion temperature                               RA 2.2
C  VFAC                Volume factor                                 R  2.2
C  VPT(MAXDIM)         Vector potential (Theta)                      RA 2.2
C  VPZ(MAXDIM)         Vector potential (z)                          RA 2.2
C  XIONIZ(MAXDIM)      Total impurity ionization rate                RA 2.2
C  XJPAR(MAXDIM)       Parallel current density                      RA 2.2
C  XJPERP(MAXDIM)      Perpendicular current density                 RA 2.2
C  XJT(MAXDIM)         Current density (Theta-component)             RA 2.2
C  XJZ(MAXDIM)         Current density (z-component)                 RA 2.2
C  XKE(MAXDIM)         Electron thermal conductivity                 RA 2.2
C  XKI(MAXDIM)         Ion thermal conductivity                      RA 2.2
C  XNE(MAXDIM)         Electron number density                       RA 2.2
C  XNI(MAXDIM)         Ion number density                            RA 2.2
C  XP(MAXDIM)          Total plasma pressure                         RA 2.2
C
```

Fig.8 Index of COMMON Variables by Blocks
This index is produced by the GENSIS generator and the correct columns are defined in §1.6.
A separate version line is used for each block.

```
PROGRAM HERMEZ

TITLE A 1-D Tokamak Transport Code

AUTHOR M H Hughes

VERSION 1* 01/Feb/79  MHH  Culham

ARRAY DIMENSIONS
MATSIZ=5544
MAXU=312
MAXEQN=6
MAXMSH=52,NMESH2
NEQ=6,6
NSPLIT=3,3
MAXSRF=52,NSURF
MAXIMP=2,NIMP

INDEX OF SUBPROGRAMS

2.1 STEPON Advance one timestep
2.2 COEFFS Calculate classical coefficients
2.3 IONDEN Set up ion density equation

2.10 NEUTRL(1) Recycling neutrals
2.11 ANOMLY(7) Anomalous diffusion

3.1 OUTPUT(1) Control the output
3.2 DPRINT(1) Diagnostic printing

INDEX OF COMMON VARIABLES

S 2.1 Tokamak Parameters

COMMON/COMTOK/

R RMAJOR *Major radius
R RAWALL *Wall radius
R RALIM *Limiter radius
R BZ *Toroidal field
R CURENT **Tokomak current
I NCOIL **Number of toroidal field coils
R RIPMAX **Maximum ripple amplitude

S 3.3 Tridiagonal Solver

COMMON/COMTRI/

DA BMATRX(MATSIZ) Block matrix
DA U(MAXU) Solution vector
DA OLDU(MAXU) Old values of *U*
DA RHS(MAXU) R.h.s. of equations
I MAXU Size of solution vector
I MATSIZ Size of block matrix
I NMATSZ (3*NMESH-2)*NEQ*NEQ
R THETA *Implicitness parameter
RA WEIGHT(MAXMSH) Interpolation weights
DA SCALEF(MAXEQN) Scale factors
DA CHI(MAXEQN,MAXMSH) Flux coefficient
DA RPSI(MAXEQN,MAXMSH) Flux coefficient
RA SF(MAXEQN,MAXMSH) Source term
RA S(MAXMSH) Source term
I NEQ Equation number
I MEQ Number of physical equations
RA CHANGE(NEQ) Maximum change in variable
LA NLFLUX(NEQ) *.T. if prescribed flux b/c
RA BFLUX(NEQ) **Boundary flux
IA MORDER(MAXEQN) Order of physical equations
IA MSOLVE(MAXEQN) Order of solution
IA NAME(MAXEQN) Variable names
I LHYD Index - hydrogen
I LIMP Index - impurities
I LHE Index - helium
I LPE Index - electron pressure
I LPI Index - ion pressure
I LBTH Index - Btheta

S 3.4 Work areas for SETBLK

COMMON/COMW1/

DA ZSUMA(MAXEQN) Sum of coefficients at j+1/2
DA ZDIFFA(MAXEQN) Difference of coefficients at j+1/2
DA ZSUMB(MAXEQN) Sum of coefficients at j-1/2
DA ZDIFFB(MAXEQN) Difference of coefficients at j-1/2
```

Fig.9 Master Index

Free-format input file MINDEX used by GENSIS [2] to construct documentation, COMMON blocks and standard subroutines. (Edited extract from program HERMEZ by MHH).

```
C/ MODULE C1S2
C
      SUBROUTINE CLEAR
C
C 1.2  Clear variables and arrays
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
      COMMON/COMCON/ L21(21)
      COMMON/COMMHD/ R22(2347)
      COMMON/COMPHY/ R23(250)
      COMMON/COMANO/ R24(738),I24(1)
      COMMON/COMRUN/ R25(256),I25(25),L25(1)
      COMMON/COMPOS/ R26(1160)
      COMMON/COMIMP/ R27(4),L27(8)
      COMMON/COMWAL/ R28(14)
      COMMON/COMFUS/ R29(113),L29(3)
      COMMON/COMNC / R31(109),I31(45),L31(1)
      COMMON/COMESH/ R32(1458),I32(4)
      COMMON/COMWRK/ R41(728)
      COMMON/COMOIM/ I42(6)
      COMMON/COMMOD/ R43(196),L43(1)
      COMMON/COMPUT/ R51(729),I51(54),L51(1)
C-----------------------------------------------------------------------
C
      ZERO=0.0
C
C     BLOCK COMCON
        CALL RESETL(L21,21,.FALSE.)
C     BLOCK COMMHD
        CALL RESETR(R22,2347,ZERO)
C     BLOCK COMPHY
        CALL RESETR(R23,250,ZERO)
C     BLOCK COMANO
        CALL RESETR(R24,738,ZERO)
        CALL RESETI(I24,1,0)
C     BLOCK COMRUN
        CALL RESETR(R25,256,ZERO)
        CALL RESETI(I25,25,0)
        CALL RESETL(L25,1,.FALSE.)
C     BLOCK COMPOS
        CALL RESETR(R26,1160,ZERO)
C     BLOCK COMIMP
        CALL RESETR(R27,4,ZERO)
        CALL RESETL(L27,8,.FALSE.)
C     BLOCK COMWAL
        CALL RESETR(R28,14,ZERO)
C     BLOCK COMFUS
        CALL RESETR(R29,113,ZERO)
        CALL RESETL(L29,3,.FALSE.)
C     BLOCK COMNC
        CALL RESETR(R31,109,ZERO)
        CALL RESETI(I31,45,0)
        CALL RESETL(L31,1,.FALSE.)
C     BLOCK COMESH
        CALL RESETR(R32,1458,ZERO)
        CALL RESETI(I32,4,0)
C     BLOCK COMWRK
        CALL RESETR(R41,728,ZERO)
C     BLOCK COMOIM
        CALL RESETI(I42,6,0)
C     BLOCK COMMOD
        CALL RESETR(R43,196,ZERO)
        CALL RESETL(L43,1,.FALSE.)
C     BLOCK COMPUT
        CALL RESETR(R51,729,ZERO)
        CALL RESETI(I51,54,0)
        CALL RESETL(L51,1,.FALSE.)
C
        RETURN
        END
```

Fig.10 Subprogram ⟨1.2⟩ CLEAR

Automatic construction of subprogram ⟨1.2⟩ CLEAR which clears common variables and arrays to zero or .FALSE. as appropriate. This is an example where the COMMON blocks are declared directly rather than by a C/ INSERT statement.

```
C/ COMMON
C/ MODULE COMCON
C-----------------------------------------------------------------------
CL                  C2.1     Physics control. Switch-on = .TRUE.
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
      COMMON/COMCON/
     L   NLANOM,   NLBURN,   NLCIRC,   NLCLAS,   NLDRFT,   NLDYNA,
     L   NLECON,   NLEQUI,   NLFUSE,   NLICON,   NLIMP ,   NLINER,
     L   NLJXB ,   NLNPT ,   NLNTRL,   NLOHM ,   NLRAD ,   NLSUYD,
     L   NLTHET
       LOGICAL
     L   NLANOM,   NLBURN,   NLCIRC,   NLCLAS,   NLDRFT,   NLDYNA,
     L   NLECON,   NLEQUI,   NLFUSE,   NLICON,   NLIMP ,   NLINER,
     L   NLJXB ,   NLNPT ,   NLNTRL,   NLOHM ,   NLRAD ,   NLSUYD,
     L   NLTHET
C/ MODULE COMMHD
C-----------------------------------------------------------------------
CL                  C2.2     MHD variables
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
      COMMON/COMMHD/
     R   AM    ,   AZ    ,   AZEFF ,   AZSQ  ,   BT    ,   BZ    ,
     R   DBT   ,   DBZ   ,   DIVFE ,   DIVFI ,   EFOR  ,   EFROR ,
     R   EQUIP ,   ET    ,   ETAII ,   ETAL  ,   ETATT ,   ETATZ ,
     R   ETAZZ ,   EZ    ,   GAMIN1,   GAMMA ,   QE    ,   QI    ,
     R   RAD   ,   RADC  ,   RADI  ,   RADL  ,   RADR  ,   RH    ,
     R   RHOR  ,   RHO   ,   RI    ,   RIDR  ,   SE    ,   SI    ,
     R   TAUEQU,   TE    ,   TI    ,   VFAC  ,   VPT   ,   VPZ   ,
     R   XIONIZ,   XJPAR ,   XJPERP,   XJT   ,   XJZ   ,   XKE   ,
     R   XKI   ,   XNE   ,   XNI   ,   XP
       DIMENSION
     R   AM(52),   AZ(52),   AZEFF(52),          AZSQ(52),
     R   BT(52),   BZ(52),   DIVFE(52),          DIVFI(52),
     R   EQUIP(52),          ET(52),   ETAII(52),          ETAL(52),
     R   ETATT(52),          ETATZ(52),          ETAZZ(52),
     R   EZ(52),   QE(52),   QI(52),   RAD(52),  RADC(52),
     R   RADI(52), RADL(52), RADR(52), RH(52),   RHOR(52),
     R   RHO(52),  RI(52),   RIDR(52), SE(52),   SI(52),
     R   TAUEQU(52),         TE(52),   TI(52),   VPT(52),
     R   VPZ(52),  XIONIZ(52),          XJPAR(52),          XJPERP(52),
     R   XJT(52),  XJZ(52),  XKE(52),  XKI(52),  XNE(52),
     R   XNI(52),  XP(52)
```

Fig.11 Structure of the COMMON Section
COMMON blocks are constructed automatically by the GENSIS generator to a format defined in §2.1. (Extract)

```
C/ MODULE C1S12
C
      SUBROUTINE AUXQ
C
C 2.12 Auxiliary quantities (pitch etc.)
C
C VERSION 2* 01/Aug/80 JPC/KVR/JWL/VAP Culham
C
C/ INSERT COMFUN
C/ INSERT COMODP
C/ INSERT COMCON
C/ INSERT COMMHD
C/ INSERT COMPHY
C/ INSERT COMANO
C/ INSERT COMRUN
C/ INSERT COMWAL
C/ INSERT COMNC
C/ INSERT COMESH
C/ INSERT COMWRK
C-----------------------------------------------------------------------
      DIMENSION
     R   ZPD(1),    ZS1(1),    ZS2(1)
      EQUIVALENCE
     R   (ZPD(1),WORK1(1)),  (ZS1(1),WORK2(1)),  (ZS2(1),WORK3(1))
      DATA        ICLASS,    ISUB      /2,12/
C
      IF(NLOMT2(ISUB)) RETURN
C-----------------------------------------------------------------------
CL              1.          Plasma pressure at half-integral points
C
      ZE = FCE * 10.0 ** FXE
      ZMUO = FCMUO * 10.0 ** FXMUO
      ZK1ME = FCK / FCME * 10.0 ** (FXK - FXME)
C
      CALL PRESS
C
C-----------------------------------------------------------------------
CL              2.          Pitch (Eq.26)
C
CL                  2.1       Pitch at half-integral points
      DO 212 J=1,NP1
C     The denominator is not allowed to be zero
        Z=BT(J)
        IF(ABS(Z).GT.SMALL)GO TO 211
        IF(Z.GE.ZERO) Z=SMALL
        IF(Z.LT.ZERO) Z=-SMALL
  211   XPITCH(J)=RH(J)*BZ(J)/Z
  212   CONTINUE
C
CL                  2.2       Pitch derivative at integral points
      DO 221 J=2,N
      ZPD(J)=(XPITCH(J)-XPITCH(J-1))/(RH(J)-RH(J-1))
C     De-stabilize positive gradient regions
        ZPD(J) = AMIN1(ZPD(J),SMALL)
  221   CONTINUE
C
C-----------------------------------------------------------------------
CL              3.          Suydam parameter (Eq.25)
C
CL                  3.1       First term at integral points
      DO 311 J=2,N
      ZS1(J) = (XP(J) - XP(J-1)) / (RH(J) - RH(J-1))
  311   CONTINUE
C
CL                  3.2       Second term at integral points
      DO 321 J=2,N
      ZBZ = FWM(J) * BZ(J-1) + FWP(J) * BZ(J)
      ZBZSQ = AMAX1(ZBZ * ZBZ,SMALL)
      Z1 = FWM(J) * XPITCH(J-1) + FWP(J) * XPITCH(J)
      Z1SQ = AMAX1(Z1*Z1,SMALL)
      Z2 = ZPD(J) * ZPD(J)
      ZS2(J) = RI(J) * ZBZSQ * Z2 / Z1SQ
  321   CONTINUE
C
CL                  3.3       Suydam ratio at integral points
      DO 331 J=2,N
      Z = AMAX1(ZS2(J),SMALL)
      SIGMA(J) = -8.0 * ZMUO * ZS1(J) / Z - 1.0
  331   CONTINUE
      SIGMA(1) = SIGMA(2)
      SIGMA(NP1) = SIGMA(N)
C
C-----------------------------------------------------------------------
CL              4.          Parallel and perpendicular currents
C
CL                  4.1       Jz and Jt at integral points
      CALL JZT(BT,BZ,XJT,XJZ)
C
CL                  4.2       Resolved components at integral points
      DO 421 J=2,NP1
      ZZJT = XJT(J)
      ZZJZ = XJZ(J)
      ZBZ = FWM(J) * BZ(J-1) + FWP(J) * BZ(J)
      ZBT = FWM(J) * BT(J-1) + FWP(J) * BT(J)
      ZB=SQRT(ZBZ*ZBZ+ZBT*ZBT+SMALL)
      XJPAR(J)=(ZZJZ*ZBZ+ZZJT*ZBT)/ZB
      XJPERP(J)=(ZZJT*ZBZ-ZZJZ*ZBT)/ZB
      XMU(J) = XJPAR(J) / ZB * ZMUO
  421   CONTINUE
C
CL                  4.3       Special treatment at origin
      ZB = ABS(BZ(1))
      IF(ZB.LT.SMALL) GO TO 431
C     Non-zero Bz-field at origin
        XJPAR(1)=XJZ(1)*BZ(1)/ZB
        XJPERP(1)=ZERO
        XMU(1) = XJPAR(1) / ZB * ZMUO
        GO TO 432
C     Zero Bz-field at origin
  431   XJPAR(1)=ZERO
        XMU(1) = ZERO
        XJPERP(1) = - XJZ(1)
  432   CONTINUE
C
CL                  4.4       Average Xmu (volume weighted)
      ZMUBAR = ZERO
      DO 441 J=1,NP1
      ZMUBAR = ZMUBAR + XMU(J) * RIDR(J)
  441   CONTINUE
      XMUBAR = ZMUBAR * 2.0 / (RMINOR*RMINOR)
      ZMUSQD = ZERO
      DO 442 J=1,NP1
      ZMUDEV = XMU(J) - XMUBAR
      ZMUSQD = ZMUSQD + ZMUDEV * ZMUDEV * RIDR(J)
  442   CONTINUE
      ZMUSQD = ZMUSQD * 2.0 / (RMINOR*RMINOR)
      XMUSTD = SQRT(ZMUSQD)
C
CL                  4.5       Vector potential at half-points
      VPT(NP1) = ZERO
      VPZ(NP1) = ZERO
      DO 451 J=1,N
      I = NP1 - J
C     B at integer points
        ZBT = BT(I)*FWM(I+1) + BT(I+1)*FWP(I+1)
        ZBZ = BZ(I)*FWM(I+1) + BZ(I+1)*FWP(I+1)
        VPT(I) = (RH(I+1)*VPT(I+1)+ZBZ*RIDR(I+1)) / RH(I)
        VPZ(I) = VPZ(I+1) - ZBT * (RH(I+1)-RH(I))
  451   CONTINUE
C     Integral (A * B). (Woltjer invariant)
        ZWOLTJ = ZERO
        DO 452 J=1,N
        ZWOLTJ = ZWOLTJ + (VPT(J)*BT(J)+VPZ(J)*BZ(J))*RHOR(J)
  452   CONTINUE
        WOLTJ1 = ZWOLTJ * VFAC
C-----------------------------------------------------------------------
CL              5.          Vdrift/Vthermal
C
CL                  5.1       Ratio at integral points
      DO 501 J=2,N
      ZNE = FWM(J) * XNE(J-1) + FWP(J) * XNE(J)
      ZNE = AMAX1(ZNE,SMALL)
      ZVD = XJPAR(J) / (ZE * ZNE)
      ZTE = FWM(J) * TE(J-1) + FWP(J) * TE(J)
      ZVESQ = ZK1ME * ZTE
      ZVESQ = AMAX1(ZVESQ,SMALL)
      VDVTHE(J) = ZVD / SQRT(ZVESQ)
  501   CONTINUE
C
CL                  5.2       Treatment at r=0
      ZVD = XJPAR(1) / (ZE * XNE(1))
      XNE(1) = AMAX1(XNE(1),SMALL)
      ZVESQ = ZK1ME * TE(1)
      ZVESQ = AMAX1(ZVESQ,SMALL)
      VDVTHE(1) = ZVD / SQRT(ZVESQ)
C
CL                  5.3       Treatment at wall
      XNE(N) = AMAX1(XNE(N),SMALL)
      ZVD = XJPAR(NP1) / (ZE * XNE(N))
      ZVESQ = ZK1ME * TE(N)
      ZVESQ = AMAX1(ZVESQ,SMALL)
      VDVTHE(NP1) = ZVD / SQRT(ZVESQ)
C
C-----------------------------------------------------------------------
CL              6.          Rate of displacement of mesh
C
      IF(.NOT.NLJXB) RETURN
C
      XSIDOT(1) = ZERO
      DO 601 J=2,N
      XSIDOT(J) = (RI(J) - RI2(J)) / DTLAST
  601   CONTINUE
C     Boundary normally does not move
      XSIDOT(NP1) = ZERO
      IF(NLCIRC.AND.NLDYNA) XSIDOT(NP1) = (RMNNEW-RI2(NP1))/DTLAST
      RETURN
      END
```

Fig.12 Structure of a Fortran Subprogram
This example illustrates some of the structural conventions defined in §3.