UHSAP - A POL FOR ANALYSIS OF LINEAR AND QUASI-LINEAR SYSTEMS

A Thesis

Presented to

the Faculty of the Department of Electrical Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Srirammohan S. Beltangady

December 1971

ACKNOWLEDGMENTS

I take this opportunity to express my sincere thanks to my advisor, Dr. G. F. Paskusz, for his encouragement and guidance. I would also like to thank Dr. R. L. Motard and Dr. L. S. Shieh for their advice. It is indeed with pleasure that I recognize the valuable suggestions and help given by Mr. A. D. Alley and Mr. R. F. Miller.

The financial assistance and computer facilities provided by Project THEMIS-ONR Contract N00014-68-A-0151 are gratefully acknowledged.

UHSAP - A POL FOR ANALYSIS
OF LINEAR AND QUASI-LINEAR SYSTEMS

An Abstract of a Thesis

Presented to

the Faculty of the Department of Electrical Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

---

by

Srirammohan S. Beltangady

December 1971

ABSTRACT

This thesis discusses the addition of a language processor and the modifications made in the output routines of ECAP to extend its capability to analyze problems in other linear and quasi-linear time-invariant systems. A new language called UHSAP is the desired extension.

INTRODUCTION

ECAP is a versatile problem oriented language for analysis of electronic networks. It also has capabilities for sensitivity analysis and worst case solutions. Since the basic algorithms to analyze electronic networks can be effectively used to analyze other linear and quasi-linear time-invariant systems, ECAP algorithmic units can perform the analysis problems for other systems.

This has been achieved in UHSAP which is an extension of ECAP. The data structure of UHSAP has been designed to accept the commands and directives in other systems. This addition of a relatively small unit to ECAP has considerably extended the capability of ECAP. The approach for translating the commands and data from UHSAP to ECAP has been that of character manipulation. Valid characters are identified and replaced to generate an object code in ECAP.

The package has been implemented effectively on OS 360/Model 44 and may, with minor changes, be implemented on other comparable computer systems.

TABLE OF CONTENTS

CHAPTER I

PROBLEM ORIENTED LANGUAGES

Webster's dictionary defines a language as a systematic means of communicating ideas or feelings by the use of conventionalized signs, gestures or marks and understood by a considerable community. The number of variables in this definition itself gives a fairly good idea of the possible number of languages human beings use. Translation of ideas from one language into another has been of interest since a long time. With the advent of machines as an aid to contribute towards the efficiency of human activities, machine translation of languages has drawn considerable attention. In spite of the amount of research and money spent on machines capable of translating one language into another, the attempts have met with little success for obvious reasons such as the syntactical and semantical ambiguity of languages. Even with such simple redefinition of a language as a series of symbols representing ideas, the simplest conceivable written language having one symbol per idea, Richens and Booth have not succeeded in the objective of machine translation.

It is at this point that Artificial Languages differ from the earlier set of so called Natural Languages. Because of their inherently simple grammar and the concise, unambiguous meanings, artificial languages have led to an effective machine translation among themselves. Programming languages which facilitate our communication with computers, both concisely and precisely, fall in this category.

Higher level programming languages are characterized by ease of understanding, machine independence, similarity with a natural language and a flexible sentence structure as against more complex and unintelligible lower level languages such as machine language. Higher level languages differ among themselves not only on the level their intelligibility but also in the general areas of their applicability. Procedural languages like FORTRAN and PL/I are useful for a wide range of problems in numerical analysis and scientific computations. Nevertheless, a demand for their employment in other areas has led to modifications and subsequently to more complexity in their syntax and vocabulary.

Problem oriented languages, or POL's , as they are commonly called, are more restricted in their usage than procedural languages. They are specifically designed for solving specific types of problems in a relatively narrow field. This loss of generality is compensated by their simplicity and the ease with which they may be learned and used. A typical POL is easy to learn, has simple syntax, has a vocabulary normally conforming with that in the application area and needs a minimum programming effort on the part of the user. Some POL's allow the user to add the necessary routines in the procedural language to aid the problem description and its solution, thus increasing their flexibility. The commands and directives in a POL, which is normally written in a procedural language, are interfaced in such a way as to allow the addition of extra commands in a POL. Each command or directive causes the appropriate routines to be called. The routines of a POL can often be divided into three basic parts.

The first part is the language processor. This is normally a translator which produces an appropriate code for the commands in the POL. The language being translated to is called the object language. An object language may be either a procedural language or another POL. A translator, typically, reads a statement, checks it for validity and meaning, and then produces the corresponding object code. A good POL translator also detects the syntactical validity of a statement and points out any errors with suitable error messages. The advantage of having a translator is the saving in the core space required by the program. Once the object code for the problem is generated, this segment of the program no longer needs to reside in the core and to occupy the space, till execution is completed.

The second part consists of a set of arithmetic expressions, algorithms and analytical routines normally used for the solution of the problems in the area. Often, this unit is quite complex and it is this complexity from the user's point of view that usually justifies the need for a POL.

The third part is a set of output routines. This is responsible for printing out the requested output in specific formats. The degree of control a user has over the flexibility of the formats and matter to be printed differs from one POL to another.

In Figure 1.1 the general structure of a typical problem oriented language is shown.

STRUCTURE OF A POL

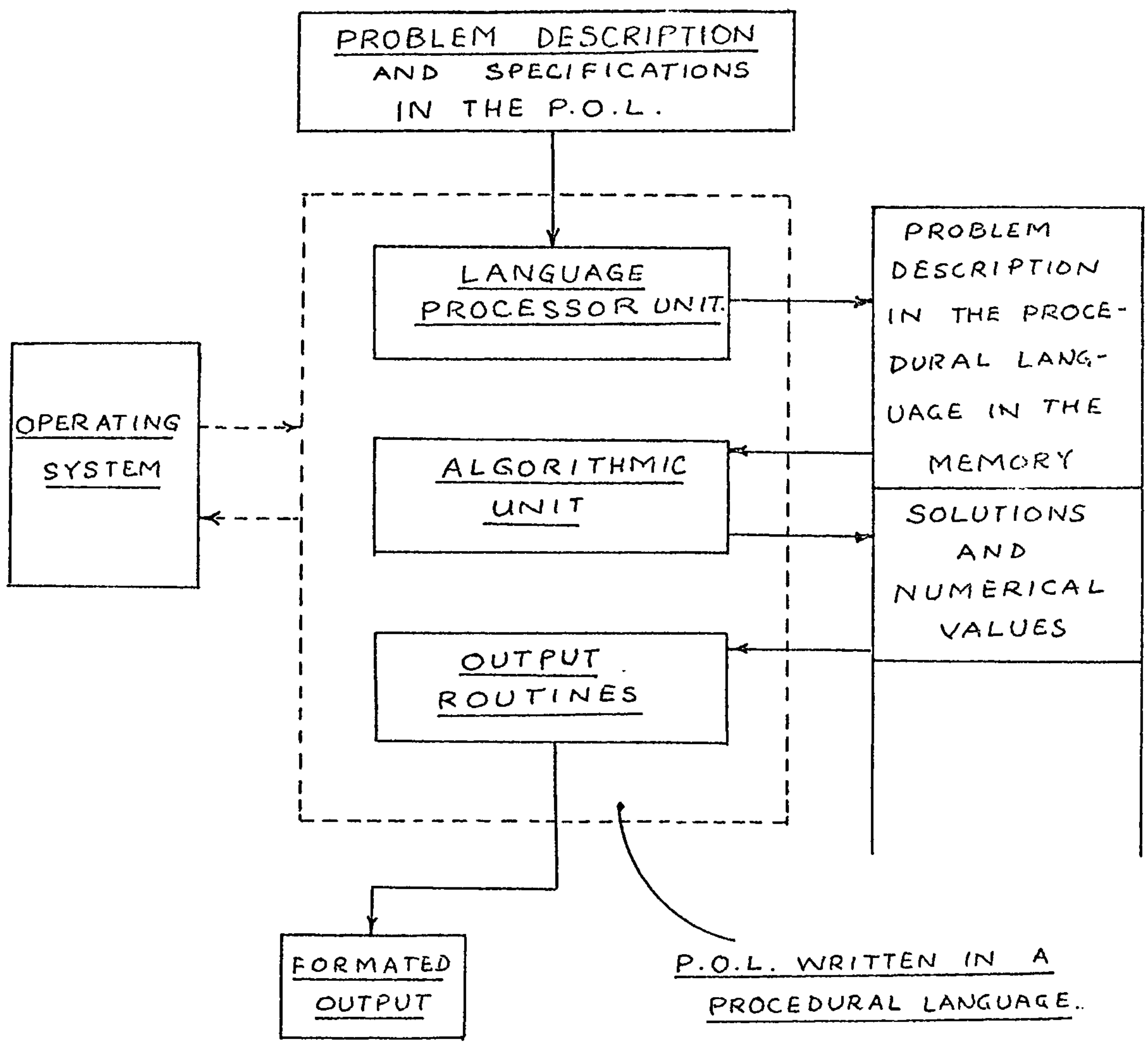


Fig. 1.1

CHAPTER II

ANALYSIS OF SYSTEMS USING ECAP

Analysis, or prediction of performance, of a physical system has been of importance to designers as well as to users. Whereas systems with distributed type parameters can be analyzed by solving partial differential equations, those with lumped parameters can be analyzed by solving ordinary differential equations. These equations, called the mathematical model of the system, have constant coefficients for each term in case of linear time-invariant systems. Quasi-linear systems are those non-linear systems whose behavior can be represented by a piecewise linear approximation. Mathematical models of simple linear time-invariant systems have the form as in Equation 2.1.

$$A \frac{dx}{dt} + Bx + C\int x \, dt = 0 \qquad (2.1)$$

Equations which have the same form are called analogous equations and the systems they represent are called analogous systems. Because of the similarity of their mathematical models, the following time-invariant linear systems form examples of analogous systems:

A. Mechanical Translational

B. Mechanical Rotational

C. Electrical

D. Acoustical

E. Thermal.

System analogs, their prototype variables and their conventionalized notations are listed in Table 2.1.

The mathematical model of a physical system is generally derived from its diagram for further analysis, but there is an additional type of model called the equivalent circuit, which resembles an electrical network. The advantage of this form of representation is that it can be easily analyzed using the already developed techniques for circuit analysis, resulting in inherent economies in the solution procedures and algorithms. A number of large programs and POL's have been written to accomplish the electronic circuit analysis. ECAP is a POL capable of performing DC, AC and transient analyses of electrical networks. The data structure of ECAP, however, allows only the direct description of electrical networks. Hence, the use of ECAP to analyze other systems would normally require drawing an equivalent circuit prior to describing it to ECAP. This drudgery has been eliminated by UHSAP.

UHSAP, an acronym for University of Houston System Analysis Program, is an extension of ECAP with the capability to analyze problems in other linear and quasi-linear analogous systems. It uses ECAP's main algorithmic and analytic routines.

TABLE 2.1*

System	Elect-rical	Mech. Transl.	Mech. Rota.	Acous-tical	Thermal
Proto-type					
w	e	v	ω	p	t
u	i	f	τ	u	q
α	L	$\frac{1}{K}$	$\frac{1}{K}$	M	–
β	R	$\frac{1}{B}$	$\frac{1}{B}$	R	R
γ	$\frac{1}{C}$	$\frac{1}{M}$	$\frac{1}{J}$	$\frac{1}{C}$	$\frac{1}{C}$

*Adapted from Linear Circuit Analysis by Paskusz and Bussel.

UHSAP is different from a conventionalized POL in the sense that the object language of its language processor is another POL, viz ECAP. The three parts of UHSAP are language processor, ECAP routines and output routines, where ECAP routines themselves can be sub-divided into three parts as shown in Chapter I.

The language processor accepts the commands and converts them into corresponding ones in ECAP. To make the programming and understanding of the language easier, the data structure of UHSAP has been maintained the same as that of ECAP. Generally, the user will have to follow the following steps to analyze a problem in any one of the analogous systems mentioned earlier:

1. Draw a topological diagram of the system.
2. Identify nodes and branches.
3. Assume positive directions of flow variables.
4. Specify the type of system, type of analysis and output desired.
5. Describe the topology of the diagram to the program in accordance with valid UHSAP syntax and variables.

Output routines have been written to print out the desired solutions in corresponding system variables and appropriate formats.

The general structure of UHSAP is shown in Figure 2.1.

STRUCTURE OF UHSAP

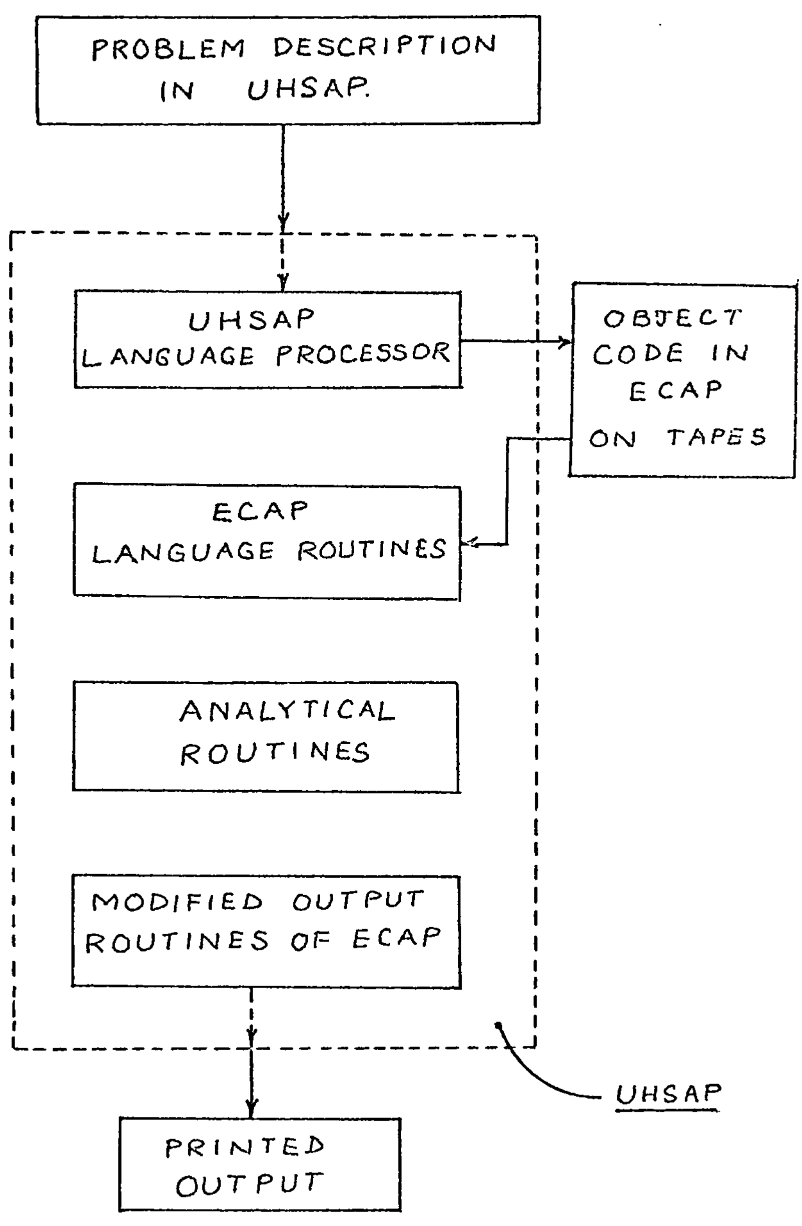


Fig. 2.1

CHAPTER III

A USER'S GUIDE TO THE LANGUAGE

The UHSAP input language is the primary communication link between the user and the analysis program, viz ECAP. It provides the user with a means of describing his topological diagram in familiar terms and notations to the program as well as a means of controlling the type of analysis to be performed. With minor restrictions, all data may appear in any order and in free form.

General Properties of Input Language

All input statements are entered on cards in columns 1 through 72. Blanks are generally ignored and may be inserted for clarity and convenience. Exceptions are noted below. Each card may have only one statement, ended with a semicolon (;). By default, the semicolon is assumed at column 72 and a diagnostic message is printed. Problems are described to the program with 6 different types of cards:

1. System identifier card
2. Data card
3. Solution control card
4. Output specification card
5. Command card
6. Comment card

Numerical values are accepted by the program in any one of the following familiar forms: signed integer, decimal and exponential notation.

The following are examples of valid numbers:

10	20.7	20 E-7
+15	+.7	+.5 E+16
-7	-100.	-7.8 E15

Description of Input Statements

A. System identifier card:

This card must be the first card of the input. It must start in the first column and specify the system. Only the first two characters are identified by the program and must not be blanks. Any text may appear in columns 3 through 72. Valid system identifier cards are as follows:

```
┌─Column 1
↓
ELECTRICAL
MECHANICAL
ROTATIONAL
ACOUSTICAL
THERMAL
```

B. Data card

Data cards contain information directly related to the problem such as the topology of the system, the parameter values, the tolerances, the phase angles, etc. They are either of branch type (*B*-Card), of dependent source type (*T*-Card), or of direction sensing type (*S*-Card). Mutual inductance type (*M*-Card) cards are also available in the electrical system.

Each card has two data fields: columns 1 through 5 and 7 through 72. A non-blank character in the sixth column indicates that the card is the continuation of the one preceding it.

1. The first field specifies the type of data card with the serial number of that type. The general format is *X*nn starting in the first column, where *X* is either *B*, *T*, *S*, or *M* and nn is the serial number.

2. The second field has one or more data subgroups, each separated by a comma (,) from the preceding one. Allowable data subgroups are as follows:

   a. Required topological description: These have different forms for different data cards.

      1. In case of *B*-Cards, this field indicates the topology of the branch and has the form:

$$N(n_1, n_2)$$

where n_1 is the initial node number and n_2 is the final node number.

2. In case of *T*-Cards, this has the form:

$$B(b_1,b_2)$$

where b_1 is the branch of the controlling branch[1] and b_2 is the branch number of the assigned branch.[1]

3. In case of *S*-Cards,[1] this has the form:

$$B = b_s,(b_1,b_2,...,b_n),x$$

where b_s is the branch number in which the sensing element lies. Parameter values of elements in $b_1,b_2,...,b_n$ are affected by the direction of flow variable in b_s. The x shows initial sense of the sensing element.

b. Required element: Each *B*-Card must contain one and only one passive element identification. Valid names for elements in different systems are tabulated in Appendix B (Table 1). On *B*-Cards, this subgroup has the form:

$$X = v$$

where *X* is the element name and v is the numerical value. For *T*-Cards, this subgroup specifies the ratio or the relations between the elements in

---

[1]For elaborate discussions about *T*-Cards and *S*-Cards see Ref. 3.

the controlling branch and the controlled branch.

c. Optional element: To specify the value of a source in a branch, this subgroup with a general form

$$X = p$$

where X is the valid source symbol listed in Appendix B (Table 2) and p is its numerical value.

d. Initial condition: For transient analysis, specification of initial condition of an element is required and is given by using valid source element name followed by either 0 or O and value specified with an equal sign and a number.

3. Another type of data cards is the time dependent source card. These are of the following general form:

```
┌──── column 1
↓
X nn     (k),p0,p1,p2,...,pn
         ↑
         └──── column 7
```

where X is the valid source symbol and nn is the branch in which it is present. The number k shows the integer number of time steps between successive entries of source magnitudes $p_0, p_1, p_2, \ldots, p_n$ which are equally spaced in time. Figure 3.2 shows image of a source card.

C. Solution control card:

There are two types of solution control cards, both types

```
C     EXAMPLES OF DATA TYPE CARDS
C
P12   N(2,5),B=1E-2,V=95.0;
P6    N(3,0),M=1.4E+3;
T3    B(2,7),BETA=20;
S9    B=10,(1,9,17),OFF;
```

Figure 3.1

```
C     EXAMPLES OF SOURCE CARDS
C
V12   (10),0.0,0.0,10.,20.,20.,10.0,0.0,0.;
V13   (10),0.,0.0,10.0,10.0,20.0,
     * 20.0,10,10,0.0,0.0
C     NON-BLANK CHARA. IN 6TH COLUMN
C     TO INDICATE CONTINUATION
```

Figure 3.2

start in column 7.

1. The first one contains information directly related to the analysis for the system such as time step, frequency, final time, etc. General format of this type of card is:

```
┌──column 7
↓
SPECIFICATION = p
```

where p is the numerical value of the valid specifications listed in Appendix B (Table 3).

2. The second type contains the specifications of calculations to be made, such as sensitivity analysis, worst case analysis, etc. and has the form:

```
┌──column 7
↓
SPECIFICATION
```

In both cases, however, only the first two characters of the specification are identified by the language processor. Therefore, blanks are not permitted in columns 7 and 8 of a valid specification card.

Figure 3.3 shows an image of solution control cards.

D. Output specification card:

This allows user to select the blocks of output which he would like to have printed. General format of this card is:

column 7

$PRINT, v_1, v_2, \ldots, v_n$

where symbols $v_1, v_2, \ldots, v_n$ are the valid output symbols the user may specify. These have been listed in Appendix B (Table 4). Each block is separated by a comma (,) from the preceding one. Here, too, only the first two characters of each block are identified, so no blanks are permitted in the first two characters of any block. Figure 3.3 shows the image of possible output specification cards.

E. Command card:

This is used to specify the type of analysis desired, or to call for a parameter modification solution, or to start the execution, or to signal the end of a job. Valid commands for different systems are listed in Appendix B (Table 5). Command cards are entered starting in column 7. Only the first two characters are identified by the language processor and hence columns 7 and 8 may contain no blanks. Any text may follow in columns 9 through 72.

F. Comment card:

Beginning with a '*C*' in the first column, any text may appear in columns 2 through 72. This is not processed by the program but appears in the listing of the problem. Figure 3.4 shows some images of command cards and comment cards.

```
C     EXAMPLES OF SOLUTION
C     CONTROL CARDS AND
C     OUTPUT SPECIFICATION CARDS
C
      FREQUENCY=1.2E3;
      TIME STEP=5E-3;
      SENSITIVITY ANALYSIS;
C
C
      PRINT,FØRCES,VELØCITIES;
      PRINT,ANGULAR VELOCITIES,TØRQUE;
```

Figure 3.3

```
C     THIS IS A COMMENT CARD SINCE
C     THE FIRST COLUMN HAS 'C'.
C     COMMAND CARD EXAMPLES
C
      DYNAMIC ANALYSIS;
      EXECUTE;
      MØDIFY;
```

Figure 3.4

Input Data Modification

The command *MODIFY* can be used for AC or DC type analysis problems and allow the user to repeat the preceding analysis with the modified values of parameters as desired. It requires the addition of only the new parameter value specification following the *MODIFY* card. The user cannot, however, change the topology of the circuit with a *MODIFY* card. The *MODIFY* feature is particularly helpful for frequency analysis, where with the card

```
┌─column 7
↓
FREQUENCY = p1(p2) p3
```

$$\text{FREQUENCY} = p_1 (p_2)\ p_3$$

the calculations can be repeated starting with frequency p_1, incremented logarithmically by a factor of p_2 till the final frequency p_3 is reached or exceeded.

Restrictions

This section is intended to point out the restrictions and minimum requirements which pertain to the input language of UHSAP. The restrictions exercised by ECAP are also effective in case of UHSAP because of the inherent similarity between the data structure of both. UHSAP can analyze problems with 50 nodes and 200 branches as maximum, in addition to other size requirements and restrictions of ECAP. As a minimum requirement, a UHSAP job must contain the following input cards:

1. System identifier card
2. Command card (specifying the analysis desired)
3. Data cards
4. Output specification card
5. Command card (signalling the start of execution)
6. Command card (end of job)

In addition, in case of AC type analysis to be performed, a solution control card specifying the frequency of the source must be entered and for transient type analysis, time step specification has to be made. In both cases, numerical values of frequency or time step cannot be zero. UHSAP assumes values for some of the input variables unless they are specified (for more details, please see Reference No. 3, pages 31 and 32). Nodes must be numbered consecutively from 1, with 0 assigned to the ground or to the frame of reference. All branches must be numbered consecutively from 1, but may be in any order.

The user is required to restrict himself to consistent system of units in specifying data. Reference No. 3 will serve as a valuable guide to the interested users of UHSAP. Valuable modeling techniques for analysis of quasi-linear systems will be found in the Reference numbers 2 and 3.

Example

Let us consider that we have to perform the dynamic analysis of the mechanical translational system shown in Figure 3.5.

Our first step will be to identify nodes and to assign some positive directions of force flow for each branch. Nodes are identified by those lumped elements of the system where the velocity can be determined. So, (1) and (2) identify two nodes with (0) at the frame of reference. Branches are indicate as [n] with n as the assigned number and the arrow pointing in the assumed direction of force flow in that branch. The UHSAP coding of the problem with desired output blocks for velocities at both the nodes and forces developed in all the elements is shown in Figure 3.6.

For the discussion about the choice of time step to be selected please see References 3 and 4.

This example shows the general structure of a typical UHSAP problem. Appendix D includes more sample problems and the analysis results from UHSAP.

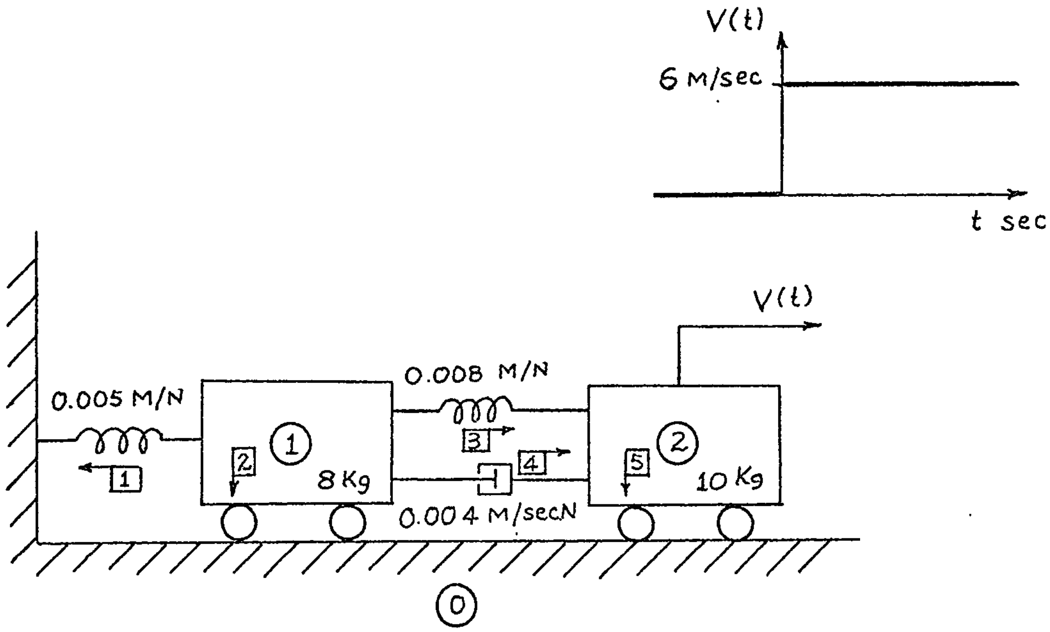


Fig. 3.5

```
MECHANICAL SYSTEM PROBLEM
C
C     EXAMPLE # 1
C
C     DYNAMIC ANALYSIS PROBLEM
C
      DYNAMIC ANALYSIS;
P1    N(1,0),K=5E-3;
P2    N(1,0),M=8;
P3    N(1,2),K=8.E-3;
P4    N(1,2),B=4.0E-3;
P5    N(2,0),M=10,V=6;
      TIME STEP=1.0E-1;
      OUTPUT INTERVAL=10;
      FINAL TIME=1.0E+2;
      PRINT,VELØCITIES,FØRCES;
      EXECUTE;
      END;
```

*A standard card form, IBM electro 888157, is available for punching statements from this form

Figure 3.6

CHAPTER IV

DESCRIPTION OF UHSAP LANGUAGE PROCESSOR AND OUTPUT ROUTINE MODIFICATIONS

This chapter is devoted to the description of routines added to accomplish the translation of UHSAP coding into valid ECAP coding. These routines, their names, brief summary of each and the flowcharts are shown in Appendix A.

Language Processor

For the purpose of simplicity and clarity, all the variables are defined to be integers in the language processor. The MAIN program puts in a call to the ULANG routine. Since the valid characters are system dependent, ULANG first calls the SYSID routine, which reads the first card and sets the value of MYSYS accordingly. In case of invalid system identifiers, the program terminates, giving an appropriate error message. If the system is identified to be the electrical system, MYSYS is set to one and control is returned to MAIN, since no translation of coding is required.

ULANG then calls subroutine READER to read the new card image into array S and to set the pointer DOL to one and the statement delimiter flag SEND to zero. The DUMP routine prints out the source image that has been entered in S.

The first step in the processing of data is to determine whether the new card is a comment card, in which case the READER is called in again to accept the next image. Function BLANK checks if the first six columns of the new image are blanks. If so, then the card belongs to either command type or solution control type or output specification type.

To process these types of cards ULANG calls routine TCOM. Valid characters for the system are initialized by the appropriate MYSYS value in the work array STAB and the work vector SCHAR. Function SELECT compares characters in columns 7 and 8 against valid two-character identifiers in STAB and sets PTR to the corresponding value. Invalid characters are identified by an error message through routine DIAG which prints out the characters in its argument vector. If the command is an end card, flag PEND is set to one which in effect transfers the control back to MAIN. Routine PUT chooses appropriate ECAP characters from the initialized array HTAB and transfers them to the output array NWORDS. In case of command cards, the control is transferred to ULANG for taking in a new card image through READER, since the text following column 8 is unimportant.

For solution control cards, characters upto the statement delimiter are important. Therefore, the function FIND is invoked to determine the column number in which the semicolon lies and all the characters upto that column are transferred as they are to the output vector NWORDS through routine RETAIN. Function FIND sets the column value to 72 in case the semicolon is not present, but gives a diagnostic message through DIAG. The control is then returned to ULANG.

For output specification cards, ULANG calls the routine TRBLOK which translates the valid output block indicators. First FIND sets the value of LAST to the column number of the statement delimiter. Then it sets COMMA by updating the scanning pointer DOL to the column number where a comma lies. The comma separates a block from the one preceding it. The updating of scanner is stopped when the value of DOL exceeds that of LAST and the control is returned to ULANG by setting SEND equal to one. The routine RETAIN transfers all the characters from the current value of DOL to COMMA as they are to output vector NWORDS. Function SELECT checks next two characters in S for their validity as block indicators by comparing them against those in STAB. PUT transfers the appropriate ones to NWORDS from array HTAB.

ULANG then calls function SOLN to process the data type cards. Function LOCATE sets the value of CTYPE by comparing the character in the first column of the card image against the valid ones, initialized in the work vector SCHAR. If invalid characters are present, proper diagnostic is given and control is returned to ULANG by making SEND value equal to one.

For *T*-cards, the procedure is different from the one described above. When a letter is found, it is checked whether the next column has a letter too and if so, whether characters equivalent to the transconductance identifier or whether the consecutive four have current gain equivalent identifier. Function ICHK compares valid characters with those in vectors BETA and GM. Subroutine PLACE transfers appropriate ones from vectors HBETA and HGM to NWORDS.

Everytime the control is transferred to ULANG from these translating routines, value of SEND is checked. If it is one, the image in NWORDS is written on the scratch tape which is used as data file 1. Value of PEND is also checked to see if it is one, in which case the tape is rewound and the control is returned to MAIN. Flag for the validity of the problem is set to zero initially. But in case of invalid characters and such fatal errors its value is changed to one. The MAIN terminates the job by giving an appropriate error message if the value of the validity flag, IABORT, is one when the control is back to MAIN.

The MAIN calls ECA, which reads the input from data file 1 on the tape if MYSYS value is different from one and from the card reader if MYSYS value is one, i.e., if the system is specified to be electrical.

This completes the description of the language processor and the part it plays in the translation. The requested analysis is performed by appropriate routines in ECAP.

Output Routine Modification

Changes were made in ECAP routines ECA21, ECA51 and PRINT2. These are the routines which print out most of the results. The changes were of the same nature in all these routines. All the formats that print system dependent information are stored for each of the systems with different format numbers. The variable MYSYS selects an appropriate format for printing.

Appendix C shows the program listing for the language processor and also the changes that need to be made in some ECAP routines to effectively implement UHSAP.

CHAPTER V

ANALYSIS AND CONCLUSIONS

The objective of extending the capabilities of ECAP to analyze other linear and quasi-linear systems is adequately met by UHSAP. The general performance of the package is satisfactory. Fatal errors in the problem coding and specification, such as illegal characters and invalid parameters are trapped successfully by the language processor. The job gets terminated if the fatal errors are detected and appropriate error messages are printed. If the errors are not fatal, the generator object code gets processed by ECAP language routines and analyzed.

The advantage of using the ECAP data structure for UHSAP, besides the economics in computer time for language processing, is the ease with which someone conversant with ECAP can use UHSAP in the analyses of other systems. The user's manual of ECAP [3] or that of UHCAP [5] (University of Houston Circuit Analysis Program) will be of help to a new user in familiarizing himself with the general usage of UHSAP.

One of the major difficulties in implementation of UHSAP was the size limitation of the operating system. IBM 360/Model 44 barely manages to hold the ECAP modules even with an extensively complicated overlay structure. Addition of routines made it increasingly difficult to accommodate the modules in the core. In addition to storing this new language processor unit in a different overlay segment, two arrays in main ECAP routines were reduced in size. This reduced the capability of ECAP to analyze circuits from a maximum of 50 nodes to 20 nodes, but made enough room for the additional routines. The package now runs efficiently on the system.

This language will also be implemented on the UNIVAC 1108 system. Because of the large core size available on the 1108, it is felt that the major hurdle will be done away with and we will be able to accommodate circuits having up to 50 nodes.

Appendix D shows UHSAP applied to a few sample problems.

REFERENCES

1. Paskusz, G. F. and B. Bussel, Linear Circuit Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1963.

2. Jensen, R. W. and M. D. Lieberman, IBM Electronic Circuit Analysis Program - Techniques and Applications, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1968.

3. , "1620 ECAP User's Manual, IBM Application Program," International Business Machines Corporation, New York, 1965.

4. McCracken, D. D., A Guide to FORTRAN IV Programming, John Wiley & Sons, New York, 1965.

5. Paskusz, G. F., UHCAP User's Manual, RE 1-70, May 1970, University of Houston, Houston, 1970.

6. Koenig, H. E., Y. Tokad, and H. K. Kesavan, Analysis of Discrete Physical Systems, McGraw-Hill Book Company, New York, 1967.

APPENDIX A

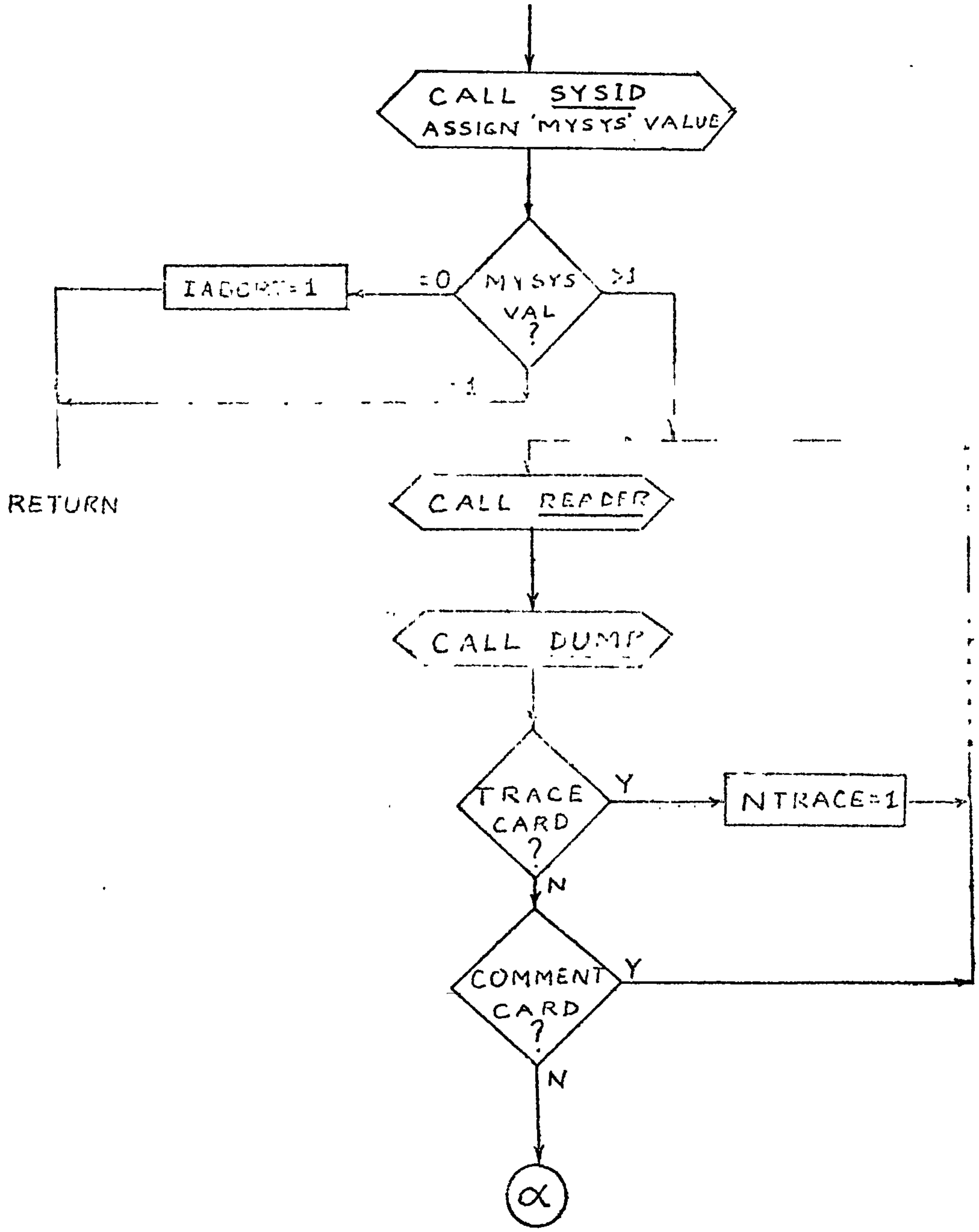
LANGUAGE PROCESSOR ROUTINES AND FLOWCHARTS

As an aid in following the processor listings and flowcharts, a brief summary of routines is given.

MAIN	Primary routine of ECAP which calls ULANG
ULANG	Controlling routine of the processor
SYSID	System identifier routine
READER	Card image reader
DUMP	Routine to write the card image
BLANK	Routine to check if the first six columns of a card image are blanks
LETTER	Function to check an alphabetic character
FIND	Function to determine the column number of a particular character
LOCATE	Routine to compare and recognize single character identifier
SELECT	Routine to compare and recognize two character identifier
PUT	Routine to substitute appropriate character from symbol tables
TCOM	Controlling routine for command type, solution control type and output specification type cards

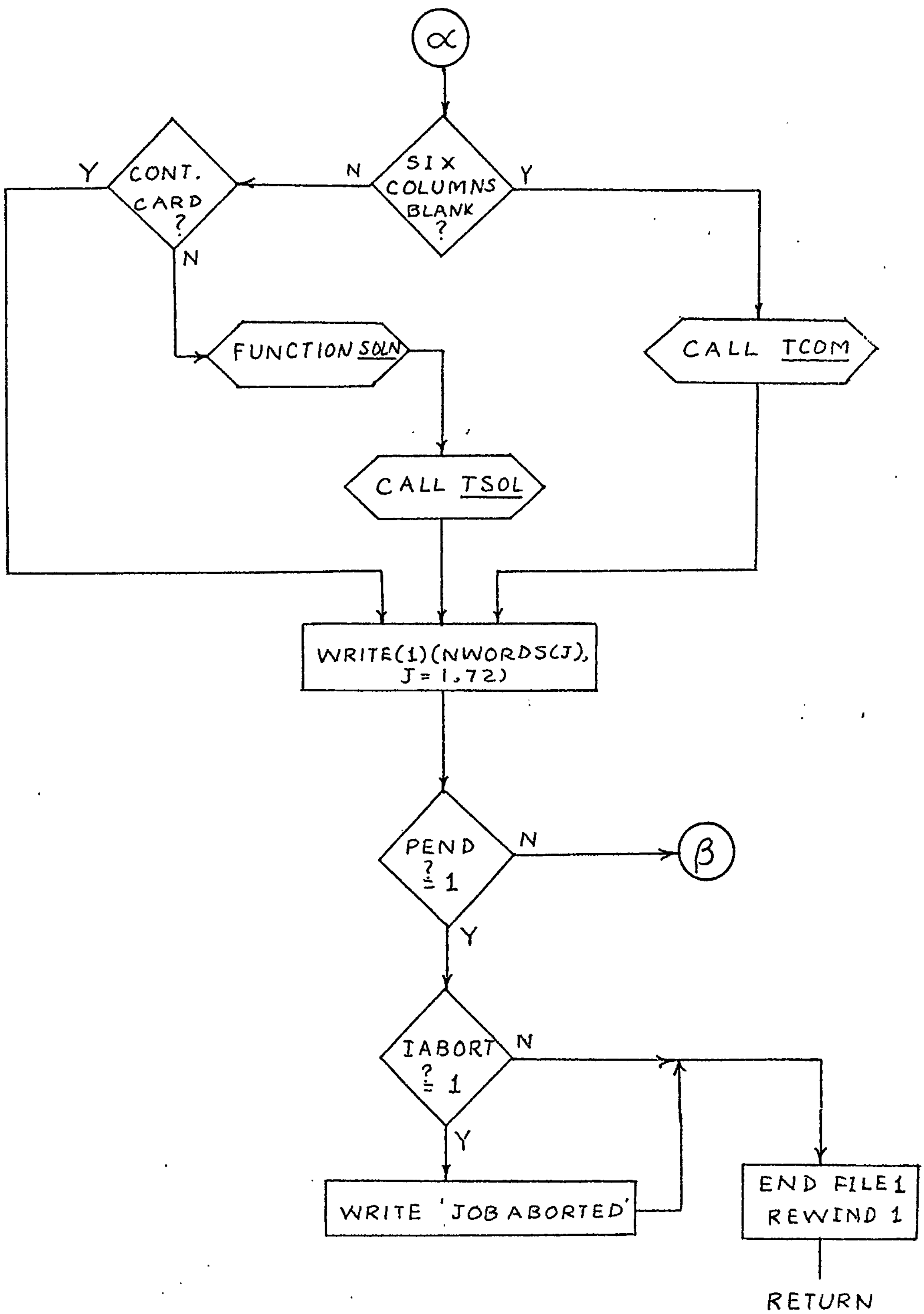
TRBLOK	Routine for translating output block indicators
TSOL	Character manipulator routine for data type cards
SOLN	Data type card recognizer routine
DIAG	Routine to print out diagnostic message
TRACE	Tracing routine
RETAIN	Routine to copy a string of characters
PLACE	Routine to replace a string of characters
INIT	Routine to initialize work arrays

ULANG

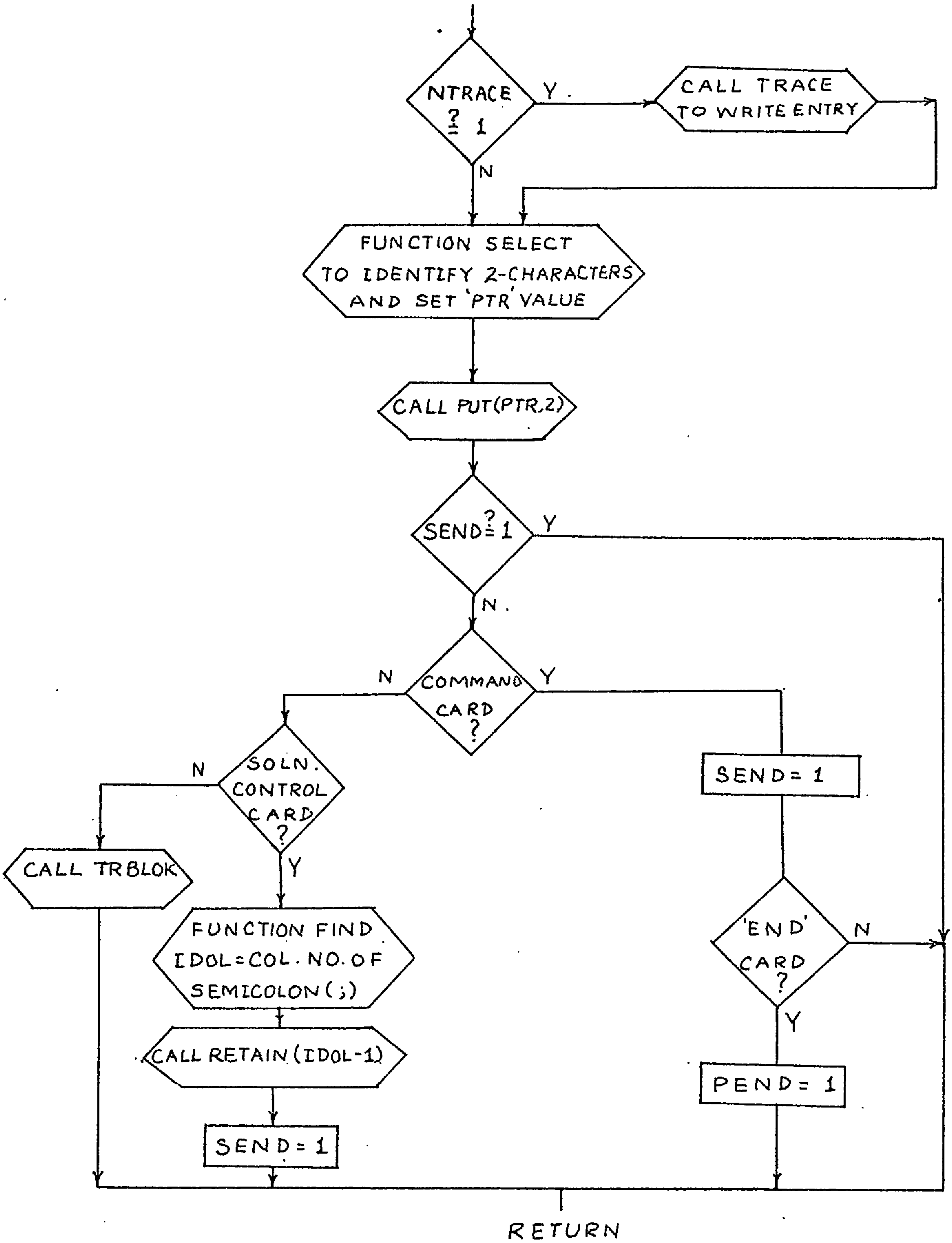


CONTD. ON NEXT PAGE

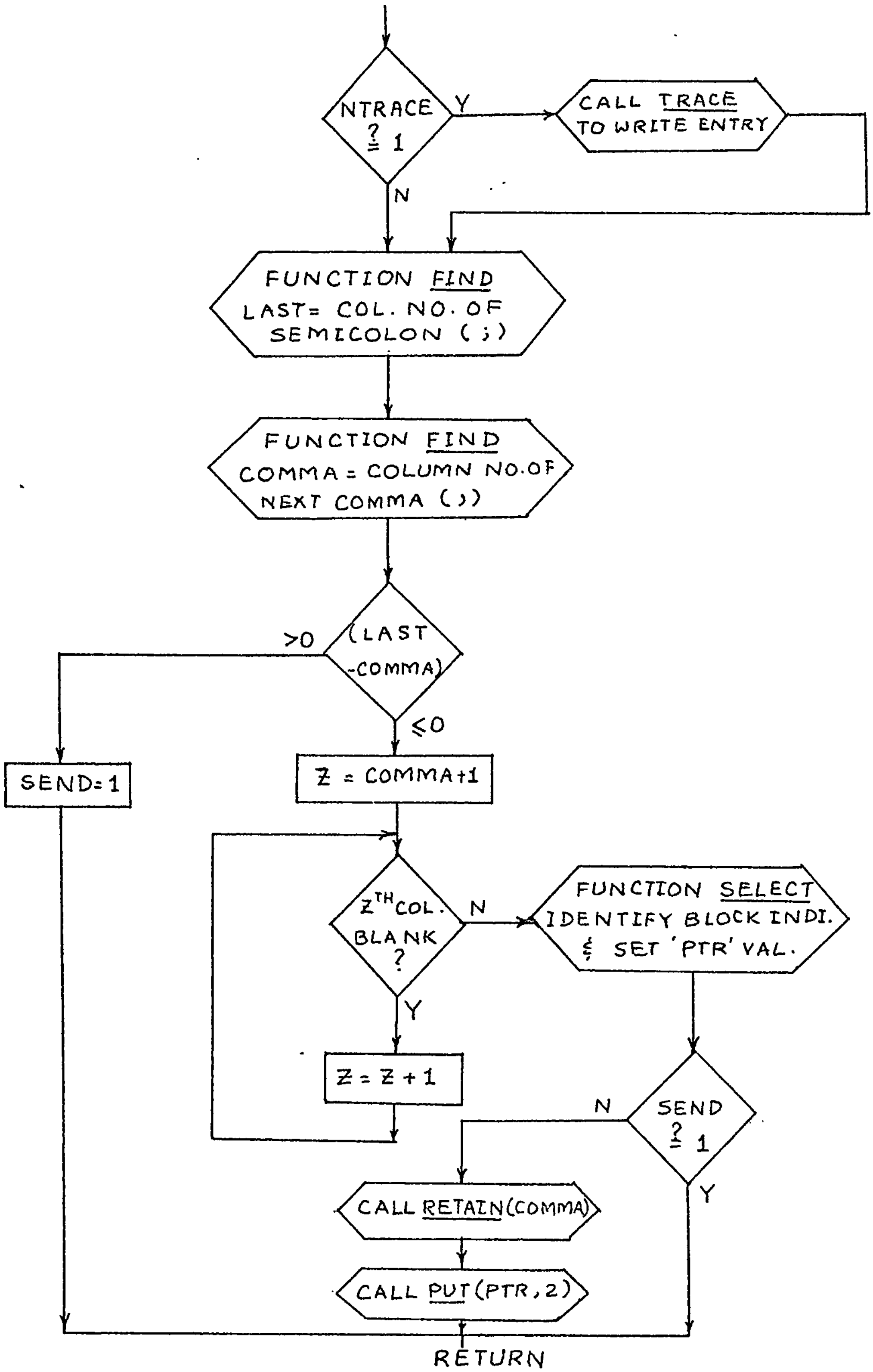
ULANG ... CONTD.


α
SIX COLUMNS BLANK ?
N
Y
CONT. CARD ?
Y
N
FUNCTION SOLN
CALL TCOM
CALL TSOL
WRITE(1)(NWORDS(J), J=1,72)
PEND ?= 1
N
β
Y
IABORT ?= 1
N
Y
WRITE 'JOB ABORTED'
END FILE1
REWIND 1
RETURN

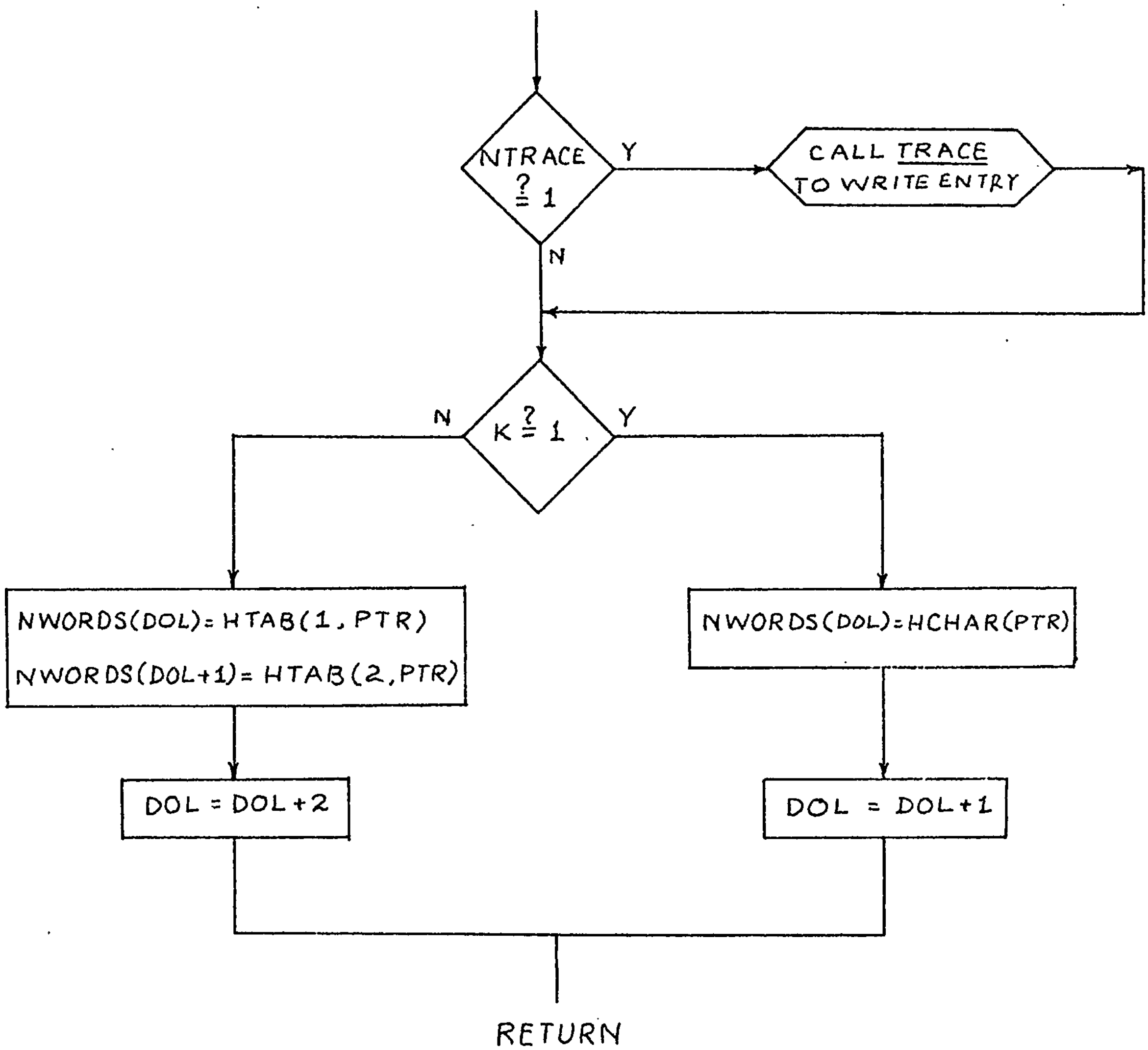
TCOM


NTRACE ?= 1
Y
N
CALL TRACE TO WRITE ENTRY
FUNCTION SELECT TO IDENTIFY 2-CHARACTERS AND SET 'PTR' VALUE
CALL PUT(PTR,2)
SEND ?= 1
Y
N
COMMAND CARD ?
N
Y
SOLN. CONTROL CARD ?
N
Y
CALL TRBLOK
FUNCTION FIND IDOL=COL. NO. OF SEMICOLON (;)
CALL RETAIN (IDOL-1)
SEND = 1
SEND = 1
'END' CARD ?
N
Y
PEND = 1
RETURN

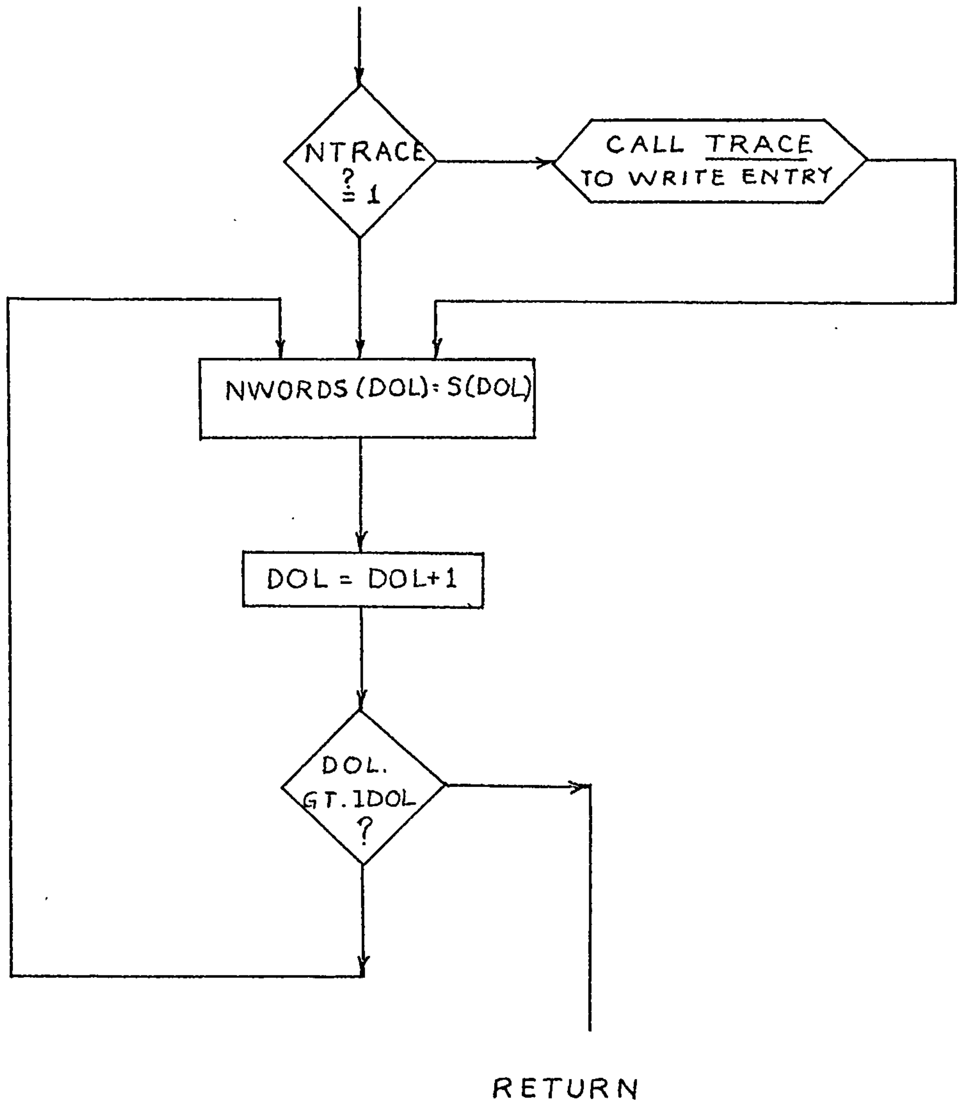
TRBLOK

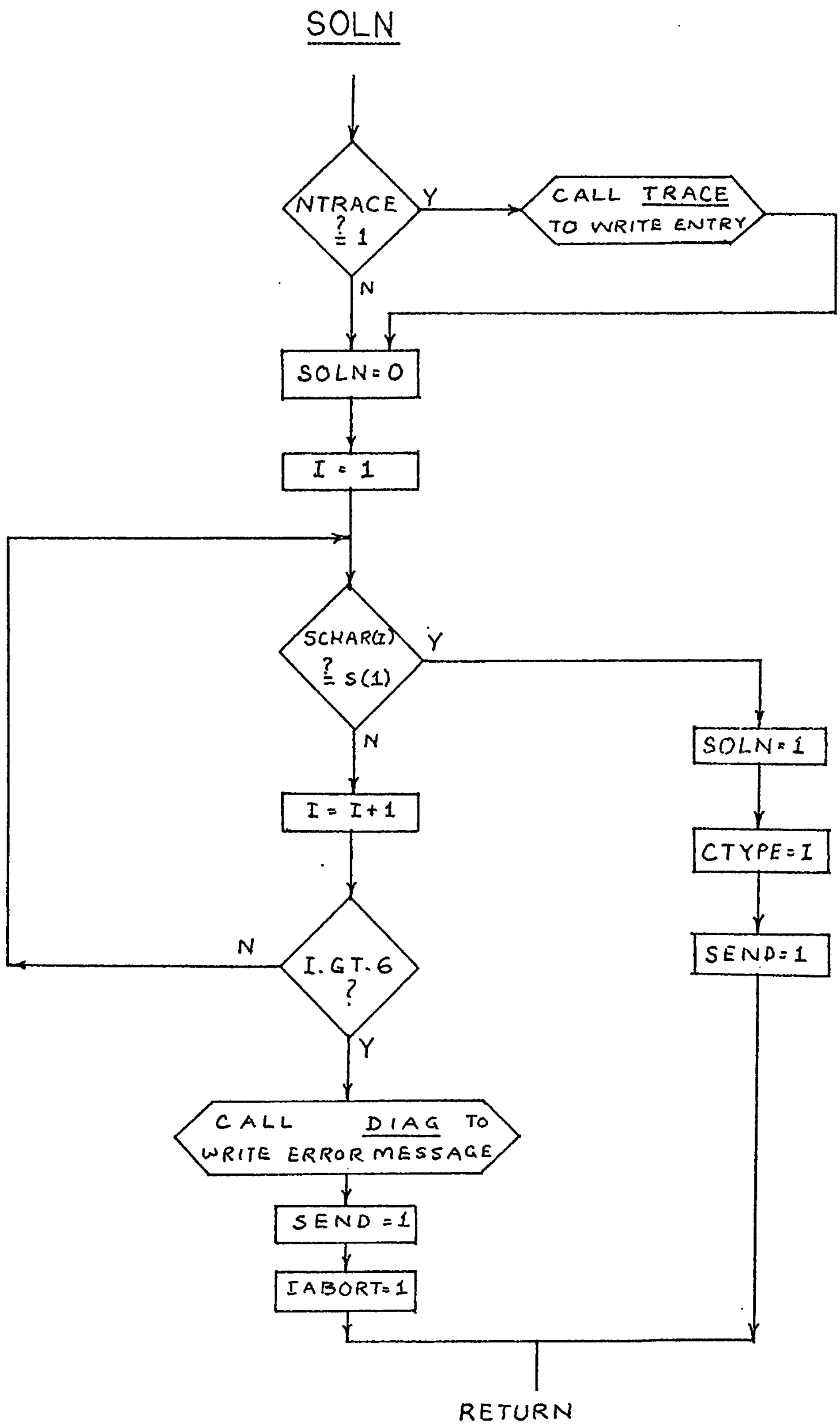

NTRACE ?= 1
Y
N
CALL TRACE TO WRITE ENTRY
FUNCTION FIND LAST= COL. NO. OF SEMICOLON (;)
FUNCTION FIND COMMA = COLUMN NO. OF NEXT COMMA (,)
(LAST -COMMA)
>0
≤0
SEND=1
Z = COMMA+1
Z^TH COL. BLANK ?
N
Y
FUNCTION SELECT IDENTIFY BLOCK INDI. & SET 'PTR' VAL.
Z = Z + 1
SEND ?= 1
N
Y
CALL RETAIN(COMMA)
CALL PUT(PTR,2)
RETURN

PUT (PTR,K)

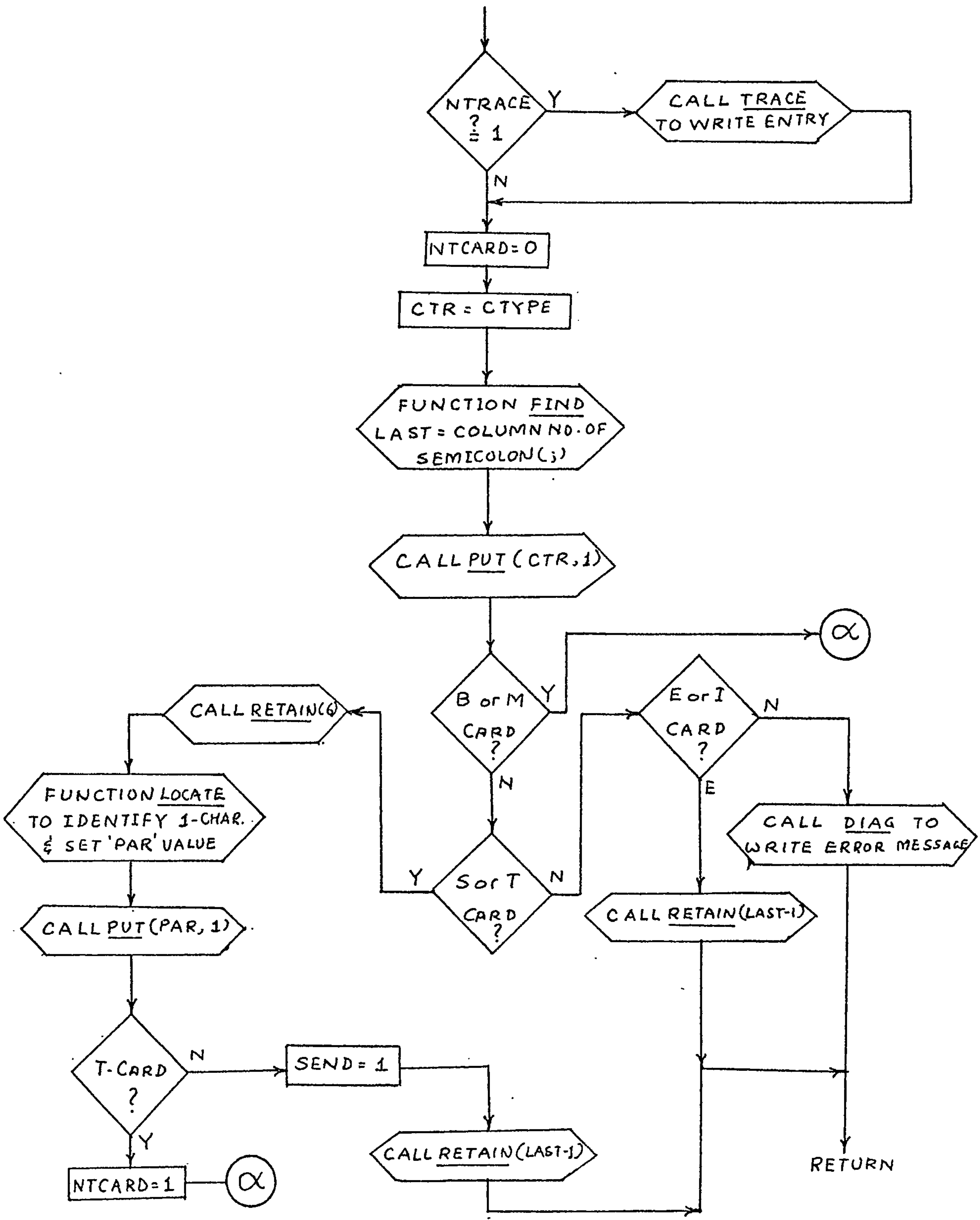

NTRACE ?= 1
Y
N
CALL TRACE TO WRITE ENTRY
K ?= 1
N
Y
NWORDS(DOL)= HTAB(1, PTR)
NWORDS(DOL+1)= HTAB(2,PTR)
NWORDS(DOL)=HCHAR(PTR)
DOL = DOL+2
DOL = DOL+1
RETURN

RETAIN (IDOL)


NTRACE ? = 1
CALL TRACE TO WRITE ENTRY
NWORDS (DOL) = S(DOL)
DOL = DOL+1
DOL. GT. IDOL ?
RETURN

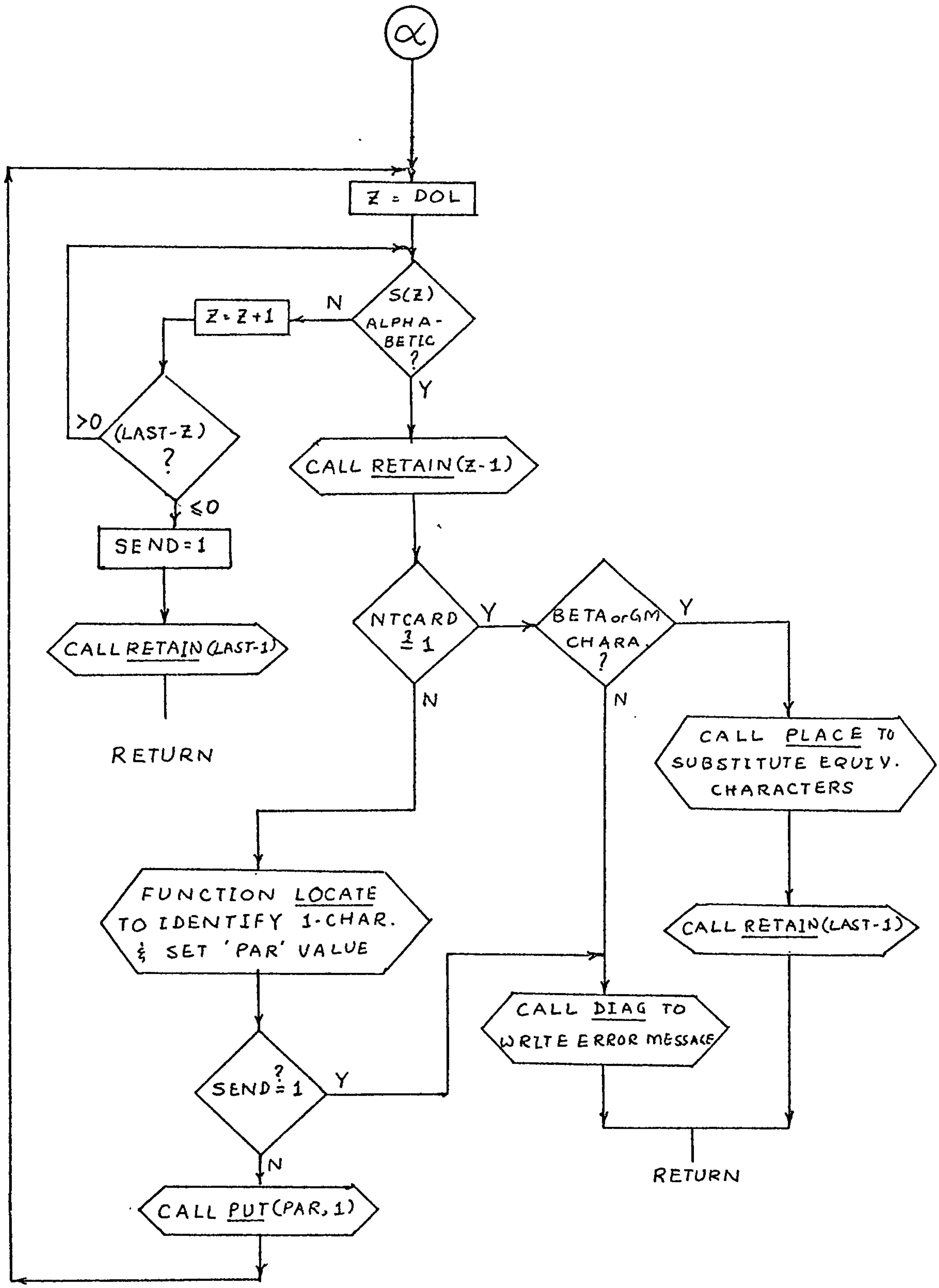

SOLN
NTRACE ?= 1
Y
N
CALL TRACE TO WRITE ENTRY
SOLN=0
I = 1
SCHAR(I) ?= S(1)
Y
N
SOLN=1
CTYPE=I
SEND=1
I = I+1
I.GT.6 ?
N
Y
CALL DIAG TO WRITE ERROR MESSAGE
SEND = 1
IABORT=1
RETURN

TSOL



CONTD. ON NEXT PAGE

TSOL ... CONTD.


Z = DOL
S(Z) ALPHA-BETIC ?
N
Y
Z = Z+1
>0
(LAST-Z) ?
≤0
SEND=1
CALL RETAIN (LAST-1)
RETURN
CALL RETAIN (Z-1)
NTCARD ?= 1
Y
N
BETA or GM CHARA. ?
Y
N
CALL PLACE TO SUBSTITUTE EQUIV. CHARACTERS
CALL RETAIN (LAST-1)
FUNCTION LOCATE TO IDENTIFY 1-CHAR. & SET 'PAR' VALUE
SEND ?= 1
Y
N
CALL DIAG TO WRITE ERROR MESSAGE
RETURN
CALL PUT (PAR, 1)

APPENDIX B

UHSAP LANGUAGE SPECIFICATIONS

TABLE 1

DATA CARD TYPES AND SPECIFICATIONS

Card contents	Electrical	Mech. Transl.	Mech. Rota.	Acoustical	Thermal
Branch (Path) Data	*Bnn*	*Pnn*	*Pnn*	*Bnn*	*Bnn*
Dep. Source Data	*Tnn*	*Dnn*	*Dnn*	*Dnn*	*Tnn*
Sensing E . Data	*Snn*	*Snn*	*Snn*	*Snn*	*Snn*
Mutual Inductance Data	*Mnn*	-	-	-	-

TABLE 2

ALLOWABLE ELEMENT IDENTIFIERS FOR *B*-CARDS

ELECTRICAL (Electrical Code)		MECHANICAL (Mechanical Code)		ROTATIONAL (Rotational Code)		THERMAL (Thermal Code)		ACOUSTICAL (Acoustical Code)	
Resistance	R	-	-	-	-	Thermal Resistance	R	Acoustic Resistance	R
Capacitance	C	Mass	M	Polar Mmt. of Inertia	J	Thermal Capacitance	C	Acoustic Capacitance	C
Inductance	L	Spring	K	Torsional Spring	K	-	-	Inertance	M
Conductance	G	Dashpot	B	Damping Factor	B	Thermal Conductivity	K	-	-
Voltage Source	E	Velocity Source	V	Angular Vel.	W	Temperature Difference	T	Pressure Difference	P
Current Source	I	Force	F	Torque	T	Heat Source	Q	Volume Vel.	U

TABLE 3*

SOLUTION CONTROL CARDS

Solution Control Card	Description of variable *q*
FREQUENCY = *q*	Frequency for AC type sources.
TIME STEP = *q*	Transient analysis time step Δt.
OUTPUT = *q*	Number of time steps per output interval.
FINAL TI = *q*	Final Time for transient solution.
SENSITIVITY	Requests sensitivity and partial derivative calculations.
WORST CASE	Requests worst case as well as sensitivity calculations.
STANDARD DEV.	Requests worst case and sensitivity.
EQUILIBRIUM	Indication of steady state solution desired.

*Adapted from Reference No. 3, page 32.

TABLE 4

VALID OUTPUT BLOCK INDICATORS

ELECTRICAL		MECHANICAL		ROTATIONAL		THERMAL		ACOUSTICAL	
Description of output	Indi- cator code	Description of output	Indi- cator code	Description of output	Indi- cator code	Description of output	Indi cator code	Description of output	Indi cator code
Node Volt- age	NV or VØ	Node Ve- locities	VE	Node Ang. Velocity	AN	Node Temp.	TE	Node Pres. Diff.	PR
El. Cur- rents	CU or CA	Forces in Element	FØ	Torque in Element	TØ	Heat in Elements	HE	Vol. Veloc- ity in El.	VØ
El. vol- tage	CV	Vel. in Element	EV	Ang. Vel. in Element	WE	Tem. diff. in Element	ET	Pressure diff. in Element	PE
Br. Curr.	BA	Forces in Branches	PF	Branch Torque	PT	Branch Heat	BQ	Branch Vol. Vel.	BU
Br. Voltage	BV	Velo. of Branch El.	PV	Branch Ang. Vel.	PW	Branch Temp. diff.	BT	Branch Pres. diff.	BP
Element Power loss	BP	Power loss in Branch	PP	Power loss in Branch	BP	Power loss in Branch	BP	Power loss in Branch	PP
Sensitivity	SE		SE		SE		SE		SE
Worst Case	WØ		WØ		WØ		WØ		WØ
Standard Dev.	ST		ST		ST		ST		ST

TABLE 5

DESCRIPTION OF COMMAND CARDS

I. ANALYSIS COMMANDS

Electrical Analysis	Code	Mechanical Analysis	Code	Rotational Analysis	Code	Acoustical Analysis	Code	Thermal Analysis	Code
DC Analy.	DC	Unidirect. Source Analysis	UD	Constant Ang. Vel. Analysis	UD	Constant Vol. Vel. Analysis	CU	Constant Heat Flow Analysis	CH
AC Analy.	AC	Alternating Force Analysis	AF	Alternating Torque Analysis	AT	Alternating Vol. Vel. Analysis	AU	Alternating Heat Flow Analysis	AH
Transient Analysis	TR	Dynamic Analysis	DY	Dynamic Analysis	DY	Transient Analysis	TR	Transient Analysis	TR

II. EXECUTIVE COMMANDS

FUNCTION	COMMAND CODE IN ALL 5 SYSTEMS
Modification calls for repeat calculations with modified data	MØ
End of input data for problem, request for solution	EX
End of job	EN

APPENDIX C

I. Language Processor Routine Listings


```
C
C      CONTROLLING ROUTINE ULANG FOR
C      THE UNSAP LANGUAGE PROCESSOR
C
C
       SUBROUTINE ULANG
       IMPLICIT INTEGER (A-Z)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
       COMMON/SYMBOL/ALPH(26),NUM(10)
       COMMON/OUT/NWORDS(80)
       COMMON/MYS/MYSYS
       COMMON/ABO/IABORT
       DIMENSION STAR(6)
       DATA STAR,BL /'*','T','R','A','C','E',' '/
       NTRACE=0
       WRITE(6,201)
201    FORMAT(1H1)
       CALL SYSID
       IF(MYSYS-1) 150,160,170
170    IABORT=0
10     CALL READER
       DO 180 I=1,6
       IF(S(I)-STAR(I)) 190,180,190
180    CONTINUE
       NTRACE=1
       CALL DUMP(S)
       GO TO 80
190    CONT=0
       IF(S(1)-ALPH(3)) 20,30,20
30     CALL DUMP(S)
       GOTO10
20     CALL DUMP(S)
       IF(BLANK(BL )) 40,40,50
50     CALL TCOM
       GOTO 80
40     IF(S(6)-BL ) 90,60,90
60     IF(SOLN(DOL))110,110,120
120    CALL TSOL
       GOTO80
90     IF(CONT.EQ.1) GOTO 80
       CONT=1
       GOTO 50
110    CALL DIAG('IMPROPER SPECIFICATION                  ')
       RETURN
80     WRITE(1)(NWORDS(I),I=1,72)
       IF(PEND)10,10,130
130    CONTINUE
```

```
      IF(IABORT.EQ.0) GOTO200
150   WRITE(6,1111)
1111  FORMAT(10X,'**** JOB NOT RUN BECAUSE OF FATAL ERRORS ****')
      IABORT=1
160   RETURN
200   CONTINUE
      END FILE 1
      REWIND 1
      RETURN
      END
C
C
C     INTEGER FUNCTION ICHK TO CHECK A PARTICULAR
C     STRING OF CHARACTERS TO MATCH WITH VECTOR V
C
C
      INTEGER FUNCTION ICHK(V,N)
      IMPLICIT INTEGER (A-Z)
      COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
      COMMON/BETAGM/BETA(4),GM(2),HBETA(4),HGM(2)
      COMMON/ABO/IABORT
      DIMENSION V(10)
      IF(NTRACE)101,102,101
101   CALL TRACE('FUNCTION ICHK                          ')
102   ICHK=0
      DO 20 I=1,N
      IF(S(DOL+I-1)-V(I)) 10,20,10
20    CONTINUE
      ICHK=1
      RETURN
10    IF(N.EQ.2) IABORT=1
      RETURN
      END
C
C
C     PLACE ROUTINE TO PUT CHARACTERS IN THE VECTOR V .
C
C
      SUBROUTINE PLACE(V,N)
      IMPLICIT INTEGER (A-Z)
      DIMENSION V(10)
      COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
      COMMON/BETAGM/BETA(4),GM(2),HBETA(4),HGM(2)
      COMMON/OUT/NWORDS(80)
      IF(NTRACE)101,102,101
101   CALL TRACE('SUBROUTINE PLACE                       ')
102   DO 10 I=1,N
10    NWORDS(DOL+I-1)=V(I)
      DOL=DOL+N
      RETURN
      END
```

```
C
C
C       SUBROUTINE TO INITIALIZE WORK ARRAYS
C       ACCORDING TO THE SYSTEM SPECIFICATION
C
C
        SUBROUTINE INIT(VNAME)
        IMPLICIT INTEGER (A-Z)
        COMMON/SYSTEM/MECH(2,70),ROTA(2,70), THER(2,70),ACOU(2,70)
        COMMON/TABLES/STAB(2,50),HTAB(2,50),SCHAR(40),HCHAR(40)
        DO 20 I=1,2
        DO 10 J=1,50
10        STAB(I,J)=VNAME(I,J)
        DO 20 J=51,70
20      SCHAR(I+J-50)=VNAME(I,J)
        RETURN
        END
C
C
C       SUBROUTINE TO IDENTIFY THE SYSTEM
C       SPECIFICATION AND SET MYSYS VALUE
C
C
        SUBROUTINE SYSID
        IMPLICIT INTEGER (A-Z)
        COMMON/SYSTEM/MECH(2,70),ROTA(2,70), THER(2,70),ACOU(2,70)
        COMMON/INTER/NTRACE,DOL,S(32)
        COMMON/MYS/MYSIS
        DIMENSION SYS(2,6)
        DATA SYS      /'E','L','M','E','R','O','T','H','A','C',2*0/
        CALL READER
        CALL DUMP(S)
        DO 30 P=1,6
        IF( S   (1)-SYS(1,P)) 30,10,30
10      IF( S   (2)-SYS(2,P))30,40,30
30      CONTINUE
        MYSIS=0
        CALL DIAG('INVALID SYSTEM IDENTIFIER CARD                 ')
        RETURN
40      MYSIS=P
        GO TO (100,200,300,400,500),MYSYS
100     RETURN
200     CALL INIT (MECH)
        RETURN
300     CALL INIT (ROTA)
        RETURN
400     CALL INIT (THER)
        RETURN
500     CALL INIT(ACOU)
        RETURN
        END
```

```
C
C
C        SUBROUTINE TCOM TO DETERMINE VALID
C        TWO-CHARACTER IDENTIFIERS
C
C
         SUBROUTINE TCOM
         IMPLICIT INTEGER (A-Z)
         COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
         DATA IHYPH/';'/
         IF(NTRACE)101,102,101
 101     CALL TRACE('SUBROUTINE TCOM                              ')
 102     PTR=SELECT(DOL,1,19)
         IF (SEND)40,40,100
 40      CALL PUT(PTR,2)
         IF(PTR-10)50,50,70
 50      SEND=1
         IF(PTR.EQ.6) PEND=1
         RETURN
 70      IF(PTR-16) 80,80,90
 80      IDOL=FIND(IHYPH)
         IDOL=IDOL-1
         CALL RETAIN(IDOL)
         SEND=1
         RETURN
 90      CALL TRBLOK
 100     RETURN
         END
C
C
C        SUBROUTINE TRBLOK TO RECOGNIZE AND
C        REPLACE OUTPUT BLOCK INDICATORS
C
C
         SUBROUTINE TRBLOK
         IMPLICIT INTEGER (A-Z)
         COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
         DATA BL/' '/
         DATA IHYPH/';'/,ICOMMA/','/
         IF(NTRACE) 101,102,101
 101     CALL TRACE('SUBROUTINE TRBLOK                            ')
 102     LAST=FIND(IHYPH)
 80      COMMA=FIND(ICOMMA)
         IF (LAST-COMMA) 100,100,10
 10      Z=COMMA+1
         IF(S(Z)-BL) 30,20,30
 20      Z=Z+1
         GO TO 10
```

```
30     PTR=SELECT(Z,20,31)
       IF(SEND) 60,60,70
70     RETURN
60     CALL RETAIN(COMMA)
       CALL PUT(PTR,2)
       GO TO 80
100    SEND=1
       RETURN
       END
C
C
C      FUNCTION SOLN TO DETERMINE IF THE CARD
C      IS OF SOLUTION CONTROL TYPE
C
C
       FUNCTION SOLN(KDOL)
C      NOTE THAT KDOL IS A DUMMY VARIABLE
       IMPLICIT INTEGER (A-Z)
       COMMON /TABLES/STAB(50,2),HTAB(50,2),SCHAR(40),HCHAR(40)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
       COMMON CTYPE
       COMMON/ABO/IABORT
       IF(NTRACE)101,102,101
101    CALL TRACE('FUNCTION SOLN                    ')
102    SOLN=0
       DO 10 I=1,6
       IF(S(1)-SCHAR(I))10,20,10
10     CONTINUE
       SEND=1
       CALL DIAG('ILLEGAL CHARACTER IN FIRST COLUMN                    ')
       IABORT=1
       RETURN
20     SOLN=1
       CTYPE=I
       RETURN
       END
C
C
C      INTEGER FUNCTION BLANK TO DETERMINE IF
C      THE FIRST SIX COLUMNS ARE BLANKS
C
C
       INTEGER FUNCTION BLANK(KDOL)
       IMPLICIT INTEGER (A-Z)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
       COMMON/OUT/WORDS(50)
       DATA BL/' '/
       IF(NTRACE)101,102,101
101    CALL TRACE('FUNCTION BLANK                   ')
```

```
102     BLANK=1
        DO 10 AUX=1,6
        IF (S(AUX)-BL) 20,10,20
10      NWORDS(AUX)=S(AUX)
        DOL=7
        RETURN
20      BLANK=0
        RETURN
        END
C
C
C       SUBROUTINE DUMP TO WRITE THE ARRAY
C
C

        SUBROUTINE DUMP(V)
        IMPLICIT INTEGER(A-Z)
        DIMENSION V(80)
        COMMON /INTER/NTRACE,RM(83)
        IF(NTRACE)101,102,101
101     CALL TRACE('SUBROUTINE DUMP                         ')
102     WRITE(6,100) (V(I),I=1,72)
100     FORMAT(5X,72A1)
        RETURN
        END
C
C
C       FUNCTION LETTER TO CHECK IF THE CHARACTER
C       IN COLUMN X IS AN ALPHABETIC CHARACTER
C
C
        INTEGER FUNCTION LETTER(X)
        IMPLICIT INTEGER (A-Z)
        COMMON/SYMBOL/ALPH(26),NUM(10)
        COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
        IF(NTRACE)101,102,101
101     CALL TRACE('FUNCTION LETTER                        ')
102     IF(S(X)-ALPH(1))3,2,1
1       IF(S(X)-ALPH(26))2,2,3
2       LETTER=1
        RETURN
3       LETTER=0
        RETURN
        END
```

```
C
C
C       SUBROUTINE PUT TO REPLACE A STRING OF CHARACTERS
C
C
        SUBROUTINE PUT (PTR,K)
        IMPLICIT INTEGER (A-Z)
        COMMON/TABLES/STAB(2,50),HTAB(2,50),SCHAR(40),HCHAR(40)
        COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
        COMMON/OUT/NWORDS(60)
        IF(NTRACE)101,102,101
101     CALL TRACE ('SUBROUTINE PUT                              ')
102     GO TO (20,10),K
10      NWORDS(DOL)=HTAB(1,PTR)
        NWORDS(DOL+1)=HTAB(2,PTR)
        DOL=DOL+2
        RETURN
20      NWORDS(DOL)=HCHAR(PTR)
        DOL=DOL+1
        RETURN
        END
C
C
C       SUBROUTINE READER TO READ A NEW CARD
C       ALSO TO SET POINTER ON ONE AND SEND=0
C
C
        SUBROUTINE READER
        IMPLICIT INTEGER (A-Z)
        COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
        COMMON/OUT/NWORDS(80)
        DATA BL/' '/
        IF(NTRACE)101,102,101
101     CALL TRACE('SUBROUTINE READER                             ')
102     SEND=0
        PEND=0
        DOL=1
        READ(5,100)(S(I),I=1,72)
100     FORMAT(72A1)
        DO 200 I=9,80
200     NWORDS(I)=BL
        RETURN
        END
```

```
C
C
C      INTEGER FUNCTION TO SELECT PARTICULAR TYPE
C      OF CHARACTERS IN THE INPUT STRING
C
C
       INTEGER FUNCTION SELECT(Z,INIT,FIN)
       IMPLICIT INTEGER (A-Z)
       COMMON/TABLES/STAB(2,50),HTAB(2,50),SCHAR(40),HCHAR(40)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,FEND
       COMMON/ABO/IABORT
       IF(NTRACE)101,102,101
101    CALL TRACE('FUNCTION SELECT                        ')
102    DO 30 VAL=INIT,FIN
       IF(S(Z)-STAB(1,VAL))30,40,30
40     IF(S(Z+1)-STAB(2,VAL))30,50,30
30     CONTINUE
       CALL DIAG(' INVALID TWO CHARACTER IDENTIFIER IN THE CARD ')
       SEND=1
       IABORT=1
       SELECT=0
       RETURN
50     SELECT=VAL
       RETURN
       END
C      INTEGER FUNCTION TO DETERMINE A PARTICULAR CHARACTER
C      IN THE STRING AND GET THE CORRESPONDING COLUMN NUMBER
C
C
       INTEGER FUNCTION FIND(V)
       IMPLICIT INTEGER (A-Z)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,FEND
       DATA ISEMI/';'/
       IF(NTRACE)101,102,101
101    CALL TRACE('FUNCTION FIND                          ')
102    DO 20 AUX=DOL,80
       IF(S(AUX)-V) 20,10,20
20     CONTINUE
       IF(V.NE.ISEMI) GOTO30
       CALL DIAG('STATEMENT DELIMITER MISSING -ASSUMED AT 72   ')
30     FIND=72
       RETURN
10     FIND=AUX
       RETURN
       END
```

```
C
C
C     SUBROUTINE RETAIN TO COPY A STRING OF CHARACTERS AS THEY ARE
C
C
      SUBROUTINE RETAIN(IDOL)
      IMPLICIT INTEGER (A-Z)
      COMMON/OUT/NWORDS(80)
      COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
      IF(NTRACE)101,102,101
101   CALL TRACE('SUBROUTINE RETAIN                ')
102   DO 10 P=DOL,IDOL
10    NWORDS(P)=S(P)
      DOL=IDOL+1
      RETURN
      END
C
C
C     SUBROUTINE TRACE TO DETERMINE THE FLOW OF CONTROL
C
C
      SUBROUTINE TRACE(V)
      IMPLICIT INTEGER (A-Z)
      DIMENSION V(10)
      WRITE(6,1)(V(I),I=1,7)
1     FORMAT(3X,'ENTRY : ',5X,7A4)
      RETURN
      END
C
C
C     ERROR SUBROUTINE TO WRITE THE ERROR MESSAGE
C
C
      SUBROUTINE DIAG(V)
      IMPLICIT INTEGER (A-Z)
      DIMENSION V(20)
      COMMON/INTER/NTRACE,REM(63)
      IF(NTRACE)101,102,101
101   CALL TRACE('SUBROUTINE DIAG                  ')
  102 WRITE(6,1) (V(I),I=1,17)
1     FORMAT(//,10X,'****DIAGNOSTIC :',17A4,//)
      RETURN
      END
```

```
C
C
C      SUBROUTINE TSOL FOR RECOGNIZING AND
C      MANIPULATING CHARACTERS IN CASE OF DATA CARDS
C
C
       SUBROUTINE TSOL
       IMPLICIT INTEGER (A-Z)
       COMMON/TABLES/STAB(50,2),HTAB(50,2),SCHAR(40),PCHAR(40)
       COMMON/INTER/NTRACE,DOL,S(80),SEND,PEND
       COMMON CTYPE
       COMMON/BETAGM/BETA(4),GM(2),HBETA(4),HGM(2)
       DATA IHYPH/';'/
       IF(NTRACE)101,102,101
101    CALL TRACE('SUBROUTINE TSOL                    ')
102    NTCARD=0
       CTR=CTYPE
       LAST=FIND(IHYPH)
       IF(SEND)10,10,20
20      CALL DIAG('INVALID CARD SPECIFICATION                                ')
       RETURN
10     CALL PUT(CTR,1)
       GO TO (30,40,40,30,70,70),CTR
70     IDOL=LAST-1
       CALL RETAIN(IDOL)
       RETURN
30     Z=DOL
80     IF(LETTER(Z)) 50,50,60
50     Z=Z+1
       IF(LAST-Z) 120,120,80
60     AUX=Z-1
       CALL RETAIN(AUX)
       IF (NTCARD) 230,230,240
230    PAR=LOCATE(DOL,5,13)
       IF(SEND)90,90,100
100    CALL DIAG('INVALID PARAMETER SPECIFICATION                           ')
       RETURN
90     CALL PUT(PAR,1)
       GOTO30
120    SEMP=1
       AUX=LAST-1
       CALL RETAIN(AUX)
       RETURN
40     CALL RETAIN(6)
       PAR=LOCATE (7,11,12)
       CALL PUT(PAR,1)
       IF(CTR.EQ.3) GO TO 140
130    VAL=LAST-1
```

```
      CALL RETAIN(VAL)
      SEND=1
      RETURN
140   NTCARD=1
      GOTO30
240   IF(ICHK(BETA,4))250,250,260
250   IF (ICHK(GM,2))270,270,290
270   CALL DIAG('INVALID CHARACTERS IN DEPENDENT SOURCE CARD          ')
      RETURN
260   CALL PLACE(HBETA,4)
      GOTO280
290   CALL PLACE(HGM,2)
280   IDOL=LAST-1
      CALL RETAIN(IDOL)
      RETURN
      END
C
C
C     FUNCTION TO LOCATE A PARTICULAR CHARACTER AND TO ASSIGN
C     THE VALUE OF THE MATCHING CHARACTER FROM THE TABLE
C
C
      INTEGER FUNCTION LOCATE(Z,F,L)
      IMPLICIT INTEGER (A-Z)
      COMMON /TABLES/STAB(50,2),HTAB(50,2),SCHAR(40),HCHAR(40)
      COMMON/INTER/NTRACE,DOL,S(80),SEND,FEND
      COMMON/ABO/IABORT
      IF(NTRACE)101,102,101
101   CALL TRACE('FUNCTION LOCATE                 ')
102   LOCATE=0
      DO20PTR=F,L
      IF(S(Z)-SCHAR(PTR))20,10,20
20    CONTINUE
      SEND=1
      CALL DIAG('ILLEGAL CHARACTER IN INPUT                           ')
      IABORT=1
      RETURN
10    LOCATE=PTR
      RETURN
      END
```

II. Listing Showing Modifications Needed in ECAP Routines

For implementation of UHSAP at other installations, a few cards may have to be removed from and a few added to the ECAP routines. The cards to be added have been shown in appropriate places with identification sequence numbers. A line has been drawn on the identification sequence numbers in case of cards that need to be removed from ECAP routines.

The following pages contain the listings for the purposes of implementation.

```
C
C   * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
      DIMENSION LIST4(200),LABEL(200),LISTE(5),LISTI(5),NUME(5),NUMI(5) ECAP0000
      DIMENSION LINK10(200),LINK1F(200)                                 ECAP0010
C                                                                       ECAP0020
      COMMON/TABLES/IDUN(280)                                           ADEC0010
      COMMON/INTER/IDUM(84)                                             ADEC0020
      COMMON/OUT/NW(60)                                                 ADEC0030
      COMMON/SYMBOL/IONE(36)                                            ADEC0040
      COMMON/MYS/MYSYS                                                  ADEC0050
      COMMON/ABO/IABORT                                                 ADEC0060
      COMMON NMAX,NNODE,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,KTO,ECAP0030
     1 MPRINT(10)                                                       ECAP0040
      COMMON E(200),EMIN(200),EMAX(200),AMP(200),AMPMIN(200),AMPMAX(200)ECAP0050
      COMMON Y(200),YMIN(200),YMAX(200),NINIT(200),NFIN(200),MODF1(200) ECAP0060
      COMMON YTERM(200),YTERMH(200),YTERML(200),IROWT(200),ICOLT(200)   ECAP0070
      COMMON ERROR1,ISEQ,MSEQ,MO,NUMMO,VFIRST(50),VSECND(50),VLAST(50)  ECAP0080
      COMMON MOBRN(50),MOPARM(50),MOSTEP(50),IWCOUT(4),NLTRMS,DELTA     ECAP0090
      COMMON IROWM(50),ICOLM(50),FLML(50),FLMH(50),FLM(50),EPHA(200),   ECAP0100
     1 AMPPHA(200),NELC,MAJOR,ERROR2,ERROR3,ETIME(5,2),ATIME(5,2)       ECAP0110
      COMMON ETR(5,126),AMPTR(5,126)                                    ECAP0120
C                                                                       ECAP0130
C     THE FOLLOWING VARIABLES ARE USED ONLY IN THE ECAP LANGUAGE PROG.  ECAP0140
C                                                                       ECAP0150
      COMMON NWORDS(72),NMCD(2,20),KLABEL(4),KPUNC(5),INDC(2,20)        ECAP0160
      COMMON INPUTR(9),NBCD(20),KTYPE(5),NBLANK,MOEXEC,ITOL,NEQUIM,IPC  ECAP0170.
      COMMON INVAL,LL,ICOL,LTYPE,KCOL,NQUIT,ITRANS,KL,KS,KELAST,NUM,M1  ECAP0180
      COMMON M2,M3,KCARD,KG,NP,NTR,MAC,NNODE,TNUM,NOEL,NOE,NOI,NOIC     ECAP0190
      COMMON EQUIVN(20),KOUT(2,10)                                      ECAP0200
C                                                                       ECAP0210
```

```
      EQUIVALENCE (YTERM(1),LIST4(1)),(ISEQ,START),(MSEQ,FINISH),       ECAP0220
     1 (NUMNO,SHORT),(VFIRST(2),LABEL(1)),(IWCOUT(2),KE),(IWCOUT(3),KI),ECAP0230
     2 (IWCOUT(4),NOSW),(DELTA,OMEGA),(FLM(1),LISTE(1)),                ECAP0240
     3 (FLM(6),LISTI(1)),(FLM(11),NUME(1)),(FLM(16),NUMI(1)),           ECAP0250
     4 (EPHA(1),LINK1D(1)),(AMPPHA(1),LINK1E(1)),(VFIRST(1),OPEN)       ECAP0260
C                                                                       ECAP0270
C                                                                       ECAP0280
      NTRACE=0                                                          ECAP0290
      KTO = 1                                                           ECAP0300
      CALL ULANG                                                        ADEC0070
      IF(IABORT.EQ.1) STOP                                              ADEC0080
 1    CALL ECA                                                          ECAP0310
      GO TO(2,3,4),NTR                                                  ECAP0320
 2    CALL ECA19                                                        ECAP0330
      GO TO 5                                                           ECAP0340
 3    CALL ECA39                                                        ECAP0350
      GO TO 5                                                           ECAP0360
 4    CALL ECA69                                                        ECAP0370
 5    KTO = 2                                                           ECAP0380
      GO TO 1                                                           ECAP0390
      END                                                               ECAP0400
      SUBROUTINE ECA                                                    LA000000
C                                                                       LA000010
C     CARD IMAGE LOADER FOR ELECTRICAL NETWORK ANALYSIS.                LA000020
C                                                                       LA000030
C                                                                       LA000040
      DIMENSION LIST4(200),LABEL(200),LISTE(5),LISTI(5),NUME(5),NUMI(5) LA000050
      DIMENSION LINK1D(200),LINK1E(200)                                 LA000060

      COMMON/MYS/MYSYS                                                  AD000010
C                                                                       LA000070
      COMMON NMAX,NNODE,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,KTO,LA000080
     1 NPRINT(10)                                                       LA000090
      COMMON E(200),EMIN(200),EMAX(200),AMP(200),AMPMIN(200),AMPMAX(200)LA000100
      COMMON Y(200),YMIN(200),YMAX(200),MINIT(200),NFIN(200),MODE1(200) LA000110
```

```
      COMMON YTERM(200),YTERMH(200),YTERML(200),IROWT(200),ICOLT(200)   LA000120
      COMMON ERROR1,ISEQ,MSEQ,MO,NUMMO,VFIRST(50),VSECND(50),VLAST(50)  LA000130
      COMMON MOPRM(50),MOPARM(50),MOSTEP(50),IWCOUT(4),NLTRMS,DELTA     LA000140
      COMMON IROWM(50),ICOLM(50),FLML(50),FLMH(50),FLM(50),EPHA(200),   LA000150
     1 AMPPHA(200),MMEC,MAJOR,ERROR2,ERROR3,ETIME(5,2),ATIME(5,2)       LA000160
      COMMON ETR(5,125),AMPTR(5,125)                                    LA000170
C                                                                       LA000180
C     THE FOLLOWING VARIABLES ARE USED ONLY IN THE ECAP LANGUAGE PROG.  LA000190
C                                                                       LA000200
      COMMON NWORDS(72),MMCD(2,20),KLABEL(4),KPUNC(5),INDC(2,20)        LA000210
      COMMON INPUTR(9),NBCD(20),KTYPE(5),NBLANK,NOEXEC,ITOL,NEQUIM,IPC  LA000220
      COMMON INVAL,LL,ICOL,LTYPE,KCOL,NQUIT,ITRANS,KO,KS,KELAST,NUM,M1  LA000230
      COMMON M2,M3,KCARD,KG,NP,NTR,MAC,HMODE,TNUM,NOEL,NOE,NOI,NOIC     LA000240
      COMMON EQUIVN(20),KOUT(2,10)                                      LA000250
C                                                                       LA000260
      EQUIVALENCE (YTERM(1),LIST4(1)),(ISEQ,START),(MSEQ,FINISH),       LA000270
     1 (NUMMO,SHORT),(VFIRST(2),LABEL(1)),(IWCOUT(2),KE),(IWCOUT(3),KI),LA000280
     2 (IWCOUT(4),NOSW),(DELTA,OMEGA),(FLM(1),LISTE(1)),                LA000290
     3 (FLM(6),LISTI(1)),(FLM(11),NUME(1)),(FLM(16),NUMI(1)),           LA000300
     4 (EPHA(1),LINK1D(1)),(AMPPHA(1),LINK1E(1)),(VFIRST(1),OPEN)       LA000310
C                                                                       LA000320
C                                                                       LA000330
C                                                                       LA000340
      IF ( NTRACE ) 3, 4, 3                                             LA000350
 2    FORMAT(' LANG MAINLINE- ECA ENTERED. KTO = ',I3 )                 LA000360
 3    WRITE (6, 2) KTO                                                  LA000370
 4    GO TO (1,44),KTO                                                  LA000380
    1 CALL ECA01                                                        LA000390
      WRITE (6, 702)                                                    LA000400
  702 FORMAT ( 1H1 )                                                    LA000410
100   READ(5,700) (NWORDS(J),J=1,72)                                    LA000420
100   IF(MYSYS.EQ.1) GOTO10                                             AD000020
      READ(1) (NWO      RDS(J),J=1,72)                                  AD000030
      GOTO20                                                            AD000040
10    READ(        5,700)(NWORDS(J),J=1,72)                             AD000050
```

```
20     IF ( NWORDS( 1 ) - NBCD( 1 )) 99, 1000, 99                    LA000430
99     IF(MYSYS.NE.1) GO TO 92                                       AD000060
       WRITE(6,701) (NWORDS(J),J=1,72)                               AD000070
99     WRITE(6,701) (JNORDS(        J),J=1,72)                       LA000440
   92 NOEXEC = NOEXEC + NQUIT                                        LA000450
      KCARD = KCARD + 1                                              LA000460
      IF ( NWORDS( 1 ) - NMCD(2, 1)) 7777, 100, 7777                 LA000470
 7777 KCOL = 6                                                       LA000480
      DO 1002 ICOL = 7, 72                                           LA000490
      IF ( NWORDS( ICOL ) - NBLANK ) 1001, 1002, 1001                LA000500
 1001 KCOL = KCOL + 1                                                LA000510
      NWORDS( KCOL ) = NWORDS( ICOL )                                LA000520
 1002 CONTINUE                                                       LA000530
      IF ( NWORDS( 6 ) - NBCD( 11)) 777, 776, 777                    LA000540
  776 IF ( NQUIT ) 500, 14, 100                                      LA000550
   14 ICOL = 6                                                       LA000560
      IF ( M1 ) 500, 109, 19                                         LA000570
  777 NQUIT = 0                                                      LA000580
      M1 = 1                                                         LA000590
      DO 24 ICOL = 1, 5                                              LA000600
      IF ( NWORDS(ICOL) - NBLANK ) 21, 24, 21                        LA000610
   21 DO 23 LTYPE = 1, 4                                             LA000620
      IF ( NWORDS( ICOL ) - KLABEL( LTYPE )) 23, 17, 23              LA000630
   23 CONTINUE                                                       LA000640
      GO TO 104                                                      LA000650
   24 CONTINUE                                                       LA000660
      IF ( KCOL - 6 ) 500, 1126, 125                                 LA000670
 1126 M3 = 4                                                         LA000680
      GO TO 805                                                      LA000690
  104 ITRANS = 1                                                     LA000700
      GO TO 126                                                      LA000710
  109 ITRANS = 2                                                     LA000720
  126 CALL ECA04                                                     LA000730
      GO TO ( 500, 500, 500, 100, 110, 110 ), ITRANS                 LA000740
  125 CALL ECA06                                                     LA000750
```

```
      GO TO ( 5, 500, 46, 100, 110, 110 ), ITRANS
   46 IF ( NOEXEC ) 500, 5004, 4667
 5004 IF ( IRTN - 1 ) 5006, 5006, 5005
 5005 GO TO ( 136, 137, 138 ), NTR
 5006 MAC = 0
   47 MAC=MAC+1
      MACFLO = KTYPE( MAC )
      GO TO ( 210, 210, 214, 216, 219, 220, 220, 238, 226, 228, 230 ),
     1 MACFLO
  210 NNODE = HNODE
      DO 3000 K=1,NNODE
      DO 2999 L = 1, NMAX
      IF ( NINIT( L ) - K ) 2998, 3000, 2998
 2998 IF ( NFIN( L ) - K ) 2999, 3000, 2999
 2999 CONTINUE
      NOEXEC=NOEXEC+1
      WRITE ( 6, 3001 ) K
 3000 CONTINUE
      DO 3205 K = 1, NMAX
      IF ( MODE1 ( K )) 3205, 3202, 3205
 3202 WRITE ( 6, 3003 ) K
      NOEXEC=NOEXEC+1
 3205 CONTINUE
      IRTN = 1
      IF( NOEXEC )500,5005,4667
C
C
 4667 WRITE ( 6, 2390 )
      WRITE ( 6, 778 ) NOEXEC
      GO TO 219
 136  RETURN
  137 IF(OMEGA)1390,1390,1370
 1370 RETURN
  138 IF(DELTA)1381,1381,1380
 1381 IF(NEQUIM)1390,1390,1382
```

```
 1382 DELTA = 1.E-6
 1380 RETURN
 1390 WRITE(6,1391)
 1391 FORMAT(//62H FREQUENCY OR TIME STEP IS IMPROPERLY DEFINED FOR THIS
     1PROBLEM//)
      NOEXEC=NOEXEC+1
      GO TO 4667
  214 IF ( NEQUIM ) 500, 210, 218
  218 TRADE = SHORT
      SHORT = OPEN
      OPEN = TRADE
      GO TO 210
  216 GO TO ( 94, 95, 500 ), NTR
   94 IRTN=3
      GO TO 136
   95 IRTN = 3
      GO TO 137
  220 WRITE ( 6, 231 )
  219 MAC=0
      MO = 0
      WRITE (6, 702)
      GO TO 100
   44 ISEQ = 0
      NTRACE = 0
      NEQUIM = 0
      WRITE (6, 702)
      GO TO 47
  226 CALL USER01
      GO TO 44
  228 CALL USER02
      GO TO 44
  230 CALL USER03
      GO TO 44
  238 CALL EXIT
C
C
```

```
C
C                                                                       LA001470
  231 FORMAT (// 25H ILLEGAL INPUT STATEMENT.//)                        LA001480
  778 FORMAT(1X,I3,25H  ERROR(S) WERE DETECTED.)                        LA001490
 2390 FORMAT(40H INPUT ERRORS MAKE EXECUTION IMPOSSIBLE.)               LA001500
 3001 FORMAT(//9H NODE NO.I4,13H  IS MISSING.//)                        LA001510
 3003 FORMAT(//11H BRANCH NO.I4,13H  IS MISSING.//)                     LA001520
  500 ITRANS = 5                                                        LA001530
      GO TO 110                                                         LA001540
  805 ITRANS = 6                                                        LA001550
      GO TO 110                                                         LA001560
 1000 ITRANS = 7                                                        LA001570
      NQUIT=0                                                           LA001580
  110 CALL ECA07                                                        LA001590
      GO TO 100                                                         LA001600
   17 ITRANS = 1                                                        LA001610
      GO TO 1713                                                        LA001620
   19 ITRANS = 2                                                        LA001630
 1713 CALL ECA02                                                        LA001640
      GO TO ( 500, 500, 1313, 100, 110, 110 ), ITRANS                   LA001650
 1313 GO TO ( 1314,1314,1315,1315,1315,1315,1315,1315,1314 ), INVAL     LA001660
 1314 CALL ECA03                                                        LA001670
      GO TO 1316                                                        LA001680
 1315 CALL ECA05                                                        LA001690
 1316 GO TO ( 500, 1713, 1713, 100, 110, 110 ), ITRANS                  LA001700
    5 CALL ECA06                                                        LA001710
      GO TO 100                                                         LA001720
  700 FORMAT( 72A1 )                                                    LA001730
  701 FORMAT ( 1H 72A1 )                                                LA001740
C                                                                       LA001750
      END                                                               LA001760
      SUBROUTINE ECA09                                                  LA001770
      DOUBLE PRECISION ACCUM                                            LA090000
C                                                                       LA090010
C     NUMERICAL EXTRACTION SUBROUTINE FOR INTERPRETING BCD NUMERALS INTOLA090020
C     F, I, OR E DECIMAL FORMATS.                                       LA090030
                                                                        LA090040
```

```
C                                                                       LA090050
      DIMENSION LIST4(200),LABEL(200),LISTE(5),LISTI(5),NUME(5),NUMI(5) LA090060
      DIMENSION LINK1D(200),LINK1E(200)                                 LA090070
C                                                                       LA090080
      COMMON/MYS/MYSYS                                                  AD090010
      COMMON NMAX,NNODE,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,KTO,LA090090
     1 MPRINT(10)                                                       LA090100
      COMMON E(200),EMIN(200),EMAX(200),AMP(200),AMPMIN(200),AMPMAX(200)LA090110
      COMMON Y(200),YMIN(200),YMAX(200),NINIT(200),NFIN(200),MODE1(200) LA090120
      COMMON YTERM(200),YTERMH(200),YTERML(200),IROWT(200),ICOLT(200)   LA090130
      COMMON ERROR1,ISEQ,MSEQ,MO,NUMMO,VFIRST(50),VSECND(50),VLAST(50)  LA090140
      COMMON MORPN(50),MOPARM(50),MOSTEP(50),IWCOUT(4),NLTRMS,DELTA     LA090150
      COMMON IROWM(50),ICOLM(50),FLML(50),FLMH(50),FLM(50),EPHA(200),   LA090160
     1 AMPPHA(200),NREC,MAJOR,ERROR2,ERROR3,ETIME(5,2),ATIME(5,2)       LA090170
      COMMON ETR(5,126),AMPTR(5,126)                                    LA090180
C                                                                       LA090190
C     THE FOLLOWING VARIABLES ARE USED ONLY IN THE ECAP LANGUAGE PROG.  LA090200
C                                                                       LA090210
      COMMON NWORDS(72),NMCD(2,20),KLABEL(4),KPUNC(5),INDC(2,20)        LA090220
      COMMON INPUTB(9),NOCD(20),KTYPE(5),NBLANK,NOEXEC,ITOL,NEQUIM,IPC  LA090230
      COMMON INVAL,LL,ICOL,LTYPE,KCOL,NOUIT,ITRANS,KC,KS,KELAST,NUM,M1  LA090240
      COMMON M2,M3,KCARD,KG,NP,NTR,MAC,HNODE,TNUM,NOEL,NOE,NOI,NOIC     LA090250
      COMMON EQUIVN(20),KOUT(2,10)                                      LA090260
C                                                                       LA090270
      EQUIVALENCE (YTERM(1),LIST4(1)),(ISEQ,START),(MSEQ,FINISH),       LA090280
     1 (NUMMO,SHORT),(VFIRST(2),LABEL(1)),(IWCOUT(2),KE),(IWCOUT(3),KI),LA090290
     2 (IWCOUT(4),NOSW),(DELTA,OMEGA),(FLM(1),LISTE(1)),                LA090300
     3 (FLM(6),LISTI(1)),(FLM(11),NUME(1)),(FLM(16),NUMI(1)),           LA090310
     4 (EPHA(1),LINK1D(1)),(AMPPHA(1),LINK1E(1)),(VFIRST(1),OPEN)       LA090320
C                                                                       LA090330
C                                                                       LA090340
    1 IF ( NTRACE ) 3, 4, 3                                             LA090350
    2 FORMAT ( 41H LANG EXTRAC SUBR-ECA-09 ENTERED.    ICOL=I4 )        LA090360
    3 WRITE(6, 2) ICOL                                                  LA090370
    4 TNUM = 0.0                                                        LA090380
      KOUNT = 0                                                         LA090390
```

```
      MINUS=0                                                        LA090400
      MINUSE=0                                                       LA090410
      NPART=0                                                        LA090420
      NE = 0                                                         LA090430
      NGOE=0                                                         LA090440
      ACCUM = 0.0D0                                                  LA090450
      GO TO 101                                                      LA090460
  100 ICOL = ICOL + 1                                                LA090470
      IF ( ICOL - KCOL ) 101, 101, 35                                LA090480
  101 DO 77 KEY = 1, 20                                              LA090490
      IF ( NWORDS( ICOL ) - NRCD( KEY )) 77, 22, 77                  LA090500
   77 CONTINUE                                                       LA090510
      M3 = 28                                                        LA090520
      GO TO 432                                                      LA090530
   22  IF(KEY-3) 100, 6, 9                                           LA090540
    6 IF(NGOE) 500, 7, 8                                             LA090550
    7 MINUS=1                                                        LA090560
      GO TO 100                                                      LA090570
    8 MINUSE=1                                                       LA090580
      GO TO 100                                                      LA090590
    9 IF(KEY-14) 12,18, 10                                           LA090600
   12 IF(NGOE) 500, 185, 33                                          LA090610
   18 NPART = 1                                                      LA090620
      GO TO 100                                                      LA090630
   10 IF(KEY-16) 21, 35, 35                                          LA090640
  185 IF(NPART) 500, 187, 186                                        LA090650
  186 KOUNT = KOUNT + 1                                              LA090660
  187 ACCUM = ACCUM *1.D1 + (KEY - 4)                                LA090670
      GO TO 100                                                      LA090680
   21 NGOE = 1                                                       LA090690
      GO TO 100                                                      LA090700
   33 NE = NE * 10 + (KEY - 4)                                       LA090710
      GO TO 100                                                      LA090720
   35 IF(MINUSE) 500, 40, 39                                         LA090730
  500 M3 = 1                                                         LA090740
  432 NQUIT = 1                                                      LA090750
```

```
      GO TO 43                                                          LA090760
   39 NE = - NE                                                         LA090770
   40 NE = NE - KOUNT                                                   LA090780
      IF ( NE ) 27, 28, 27                                              LA090790
   27 ACCUM = ACCUM * 1.D1**NE                                          LA090800
   28 TNUM = ACCUM                                                      LA090810
      IF(MINUS) 36, 43, 36                                              LA090820
   36 TNUM = - TNUM                                                     LA090830
   43 ICOL = ICOL - 1                                                   LA090840
 9996 CONTINUE                                                          LA090845
      GO TO (99962,99961,99961,99962,99962), MYSYS                      AD090020
99961 IF(INVAL.NE.7) GO TO 99962                                        AD090030
      IF(KONCE.EQ.1) GO TO 99963                                        AD090040
      AUXL=TNUM                                                         AD090050
      TNUM=1./AUXL                                                      AD090060
      KONCE=1                                                           AD090070
      GO TO 99962                                                       AD090080
99963 KONCE=0                                                           AD090090
99962 IF ( NTRACE ) 9998, 9999, 9998                                    LA090850
 9997 FORMAT(41H LANG EXTRAC SUBR-ECA-09 EXIT.        ICOL=I4,          LA090860
    1   6H   TNUM=E15.8 )                                               LA090870
 9998 WRITE(6, 9997) ICOL,TNUM                                          LA090880
 9999 RETURN                                                            LA090890
      END                                                               LA090900

      SUBROUTINE ECA25                                                  DC250000
      DOUBLE PRECISION X(200)                                           DC250010
      COMMON/MYS/MYSYS                                                  AD250010
      COMMON NMAX,NNODE,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,KTO,DC250020
     1 NPRINT(10)                                                       DC250030
      COMMON E(200),EMIN(200),EMAX(200),AMP(200),AMPMIN(200),AMPMAX(200)DC250040
      COMMON Y(200),YMIN(200),YMAX(200),NINIT(200),NFIN(200),MODE1(200) DC250050
      COMMON YTERM(200),YTERMH(200),YTERML(200),IPONT(200),ICOLT(200)   DC250060
      COMMON ERROR1,ISEQ,MSEQ,NO,NUMNO,VFIRST(50),VSECND(50),VLAST(50)  DC250070
      COMMON MOBRN(50),MOPARM(50),MOSTEP(50),IWCOUT(4)                  DC250080
```

```
C                                                                       DC250090
C     THE FOLLOWING VARIABLES ARE USED ONLY IN THE ECAP D.C. ANALYSIS   DC250100
C                                                                       DC250110
      COMMON AX1,SMLEP(50),CURR(200),SMLE(200),EQUCUR(50),EX(200)       DC250120
      COMMON EB(200),AMPX(200),AMPB(200),VNOM(50),STLSQ(50),L,M,ITOL    DC250130
      COMMON JX1,JX4,JX5,DELTA,DUM1(26)                                 DC250140
C                                                                       DC250150
      COMMON LANG(265)                                                  DC250160
C                                                                       DC250170
      COMMON MATA(200,4,3),YX(200),YB(200),YTERMX(200),YTERMB(200)      DC250180
      COMMON WCMAX(50),WCMIN(50)                                        DC250190
C                                                                       DC250200
      DOUBLE PRECISION SMLEP,CURR,SMLE,EQUCUR                           DC250210
C                                                                       DC250220
    1 IF(NTRACE)3,4,3                                                   DC250230
    2 FORMAT(6H ECA25 )                                                 DC250240
    3 WRITE(6,2)                                                        DC250250
    4 IF(JX4)8,8,101                                                    DC250260
C                                                                       DC250270
C     OUTPUT NODE VOLTAGES                                              DC250280
C                                                                       DC250290
    8 IF(NPRINT(1))101,101,99                                           DC250300
99    WRITE(6,133)                                                      DC250310
99    GO TO (1000,2000,3000,3001,3002),MYSYS                            AD250020
1000  WRITE(6,133)                                                      AD250030
      WRITE(6,134)                                                      DC250320
  133 FORMAT(// 14H NODE VOLTAGES //)                                   DC250330
  134 FORMAT( 6H NODES,15X,8HVOLTAGES /)                                DC250340
      GO TO 4000                                                        AD250040
2000  WRITE(6,1133)                                                     AD250050
1133  FORMAT(//' NODE VELOCITIES'//,' NODES',15X,'VELOCITIES'/)         AD250060
      GO TO 4000                                                        AD250070
3000  WRITE(6,2133)                                                     AD250080
      GO TO 4000                                                        AD250090
3001  WRITE(6,3113)                                                     AD250100
      GO TO 4000                                                        AD250110
```

```
6002  WRITE(6,4113)                                                      AD250120
2133  FORMAT(//' ANGULAR VELOCITIES AT NODES',//,' NODES',15X,           AD250130
     1 'VELOCITIES'/)                                                    AD250140
3113  FORMAT(//' NODE TEMPERATURES',//,' NODES',15X,'TEMPERATURES'/)     AD250150
4113  FORMAT(//' NODE PRESSURES',//,'NODES',15X,'PRESSURES'/)            AD250160
4000  CONTINUE                                                           AD250170
      DO 5 I=1,NNODE                                                     DC250350
    5 X(I)=SMLEP(I)                                                      DC250360
      KMAX=NNODE                                                         DC250370
      IND=1                                                              DC250380
      GO TO 100                                                          DC250390
C                                                                        DC250400
C     BRANCH VOLTAGES                                                    DC250410
C                                                                        DC250420
  101 DO 10 I=1,NMAX                                                     DC250430
      SMLE(I) = 0.0                                                      DC250440
      J=NINIT(I)                                                         DC250450
      IF(J)11,11,12                                                      DC250460
 12   SMLE(I) = SMLEP(J)                                                 DC250470
 11   K=NFIN(I)                                                          DC250480
      IF ( K ) 10, 10, 14                                                DC250490
 14   SMLE(I) = SMLE(I) - SMLEP(K)                                       DC250500
   10 CONTINUE                                                           DC250510
      IF(JX4)9,9,102                                                     DC250520
    9 IF(NPRINT(5))102,102,15                                            DC250530
15    WRITE(6,101)                                                       DC250540
15    GOTO(5000,6000,7000,7001,7002),MYSYS                               AD250180
5000  WRITE(6,131)                                                       AD250190
      WRITE(6,132)                                                       DC250550
  131 FORMAT(// 16H BRANCH VOLTAGES //)                                  DC250560
  132 FORMAT( 9H BRANCHES,12X,8HVOLTAGES/)                               DC250570
      GO TO 8000                                                         AD250200
6000   WRITE(6,1131)                                                     AD250210
1131  FORMAT(//,' BRANCH VELOCITIES',//,' BRANCHES',12X,'VELOCITIES'/)   AD250220
7000  WRITE(6,2131)                                                      AD250230
2131  FORMAT(//,' BRANCH ANGULAR VELOCITIES',//,' BRANCHES',12X,         AD250240
```

```
     1  'VELOCITIES'/)                                                 AD250250
        GO TO 8000                                                     AD250260
7001    WRITE(6,3131)                                                  AD250270
3131    FORMAT(//,' BRANCH TEMPERATURES',//,' BRANCHES',12X,           AD250280
     1  'TEMPERATURES'/)                                               AD250290
        GO TO 8000                                                     AD250300
7002    WRITE(6,4131)                                                  AD250310
4131    FORMAT(//,' BRANCH PRESSURES',//,' BRANCHES',12X,'PRESSURES'/) AD250320
8000    CONTINUE                                                       AD250330
        DO 16 I=1,NMAX                                                 DC250580
   16   X(I)=SMLE(I)                                                   DC250590
        KMAX=NMAX                                                      DC250600
        IND=2                                                          DC250610
        GO TO 100                                                      DC250620
C                                                                      DC250630
C       ELEMENT VOLTAGES                                               DC250640
C                                                                      DC250650
  102 DO 17 I=1,NMAX                                                   DC250660
   17 SMLE(I)=SMLE(I)+EX(I)                                            DC250670
        IF(JX4)7,7,103                                                 DC250680
    7 IF(MPRINT(3))103,103,18                                          DC250690
18      WRITE(6,135)                                                   ~~DC250700~~
18      GO TO (9000,10000,11000,11001,11002),MYSYS                     AD250340
9000     WRITE(6,135)                                                  AD250350
        WRITE(6,132)                                                   DC250710
  135 FORMAT(// 17H ELEMENT VOLTAGES//)                                DC250720
        GO TO 12000                                                    AD250360
10000 WRITE(6,1135)                                                    AD250370
1135    FORMAT(//,' ELEMENT VELOCITIES'//   ' PATHS',12X,'VELOCITIES'/) AD250380
        GO TO 12000                                                    AD250390
11000 WRITE(6,2135)                                                    AD250400
2135    FORMAT(//,' ELEMENT ANGULAR VELOCITIES'//,' BRANCHES',12X,     AD250410
     1  'VELOCITIES'/)                                                 AD250420
        GO TO  12000                                                   AD250430
11001 WRITE(6,3135)                                                    AD250440
3135    FORMAT(//,' ELEMENT TEMPERATURES',//,' BRANCHES',12X,          AD250450
```

```
     1  'TEMPERATURES'/)                                                    AD250460
        GO TO 12000                                                         AD250470
11002 WRITE(6,4135)                                                         AD250480
4135  FORMAT(//,' ELEMENT PRESSURES',//,' BRANCHES',12X,'PRSSSURES'/)       AD250490
12000  CONTINUE                                                             AD250500
      DO 19 I=1,NMAX                                                        DC250730
   19 X(I)=SMLE(I)                                                          DC250740
      KMAX=NMAX                                                             DC250750
      IAD=3                                                                 DC250760
      GO TO 100                                                             DC250770
C                                                                           DC250780
C     ELEMENT CURRENTS                                                      DC250790
C                                                                           DC250800
  103 DO 20 I=1,NMAX                                                        DC250810
   20 CURR(I)=YX(I)*SMLE(I)                                                 DC250820
      IF(NTERMS)21,22,21                                                    DC250830
   21 DO 22 I=1,NTERMS                                                      DC250840
      NR=IROWT(I)                                                           DC250850
      NC=ICOLT(I)                                                           DC250860
      IF(SMLE(NC))24,23,24                                                  DC250870
   24 CURR(NR)=CURR(NR)+YTERMX(I)*SMLE(NC)                                  DC250880
   23 CONTINUE                                                              DC250890
   22 IF(JX4)25,25,105                                                      DC250900
   25 IF(NPRINT(2))104,104,26                                               DC250910
26    WRITE(6,136)                                                          [illegible]
26    GO TO (      1300,1400,1500,1501,1502),MYSYS                          AD250510
1300  WRITE(6,136)                                                          AD250520
      WRITE(6,137)                                                          DC250930
  136 FORMAT(// 17H ELEMENT CURRENTS //)                                    DC250940
  137 FORMAT(9H BRANCHES,12X,8HCURRENTS/)                                   DC250950
      GO TO 1600                                                            AD250530
1400  WRITE(6,1136)                                                         AD250540
1136  FORMAT(//,' ELEMENT FORCES'//,' PATHS',12X,'FORCES'/)                 AD250550
      GO TO 1600                                                            AD250560
1500  WRITE(6,2136)                                                         AD250570
2136  FORMAT(//,' ELEMENT TORQUES',//,' BRANCHES',12X,' TORQUES'/)          AD250580
```

```
      GO TO 1600                                                         AD250590
 1501 WRITE(6,3136   )                                                   AD250600
 3136 FORMAT(//,' ELEMENT HEAT FLOW',//,' BRANCHES',12X,'HEAT FLOW'/)    AD250610
      GO TO 1600                                                         AD250620
 1502 WRITE(6,4136)                                                      AD250630
 4136 FORMAT(//,' ELEMENT  VOLUME VELOCITIES',//,' BRANCHES',12X,        AD250640
     1  'VOLUME VELOCITIES'/)                                            AD250650
 1600 CONTINUE                                                           AD250660
      DO 27 I=1,NMAX                                                     DC250960
   27 X(I)=CURR(I)                                                       DC250970
      KMAX=NMAX                                                          DC250980
      IND=4                                                              DC250990
      GO TO 100                                                          DC251000
C                                                                        DC251010
C     BRANCH POWER LOSSES                                                DC251020
C                                                                        DC251030
  104 IF(NPRINT(6))105,105,28                                            DC251040
   28 DO 29 I=1,NMAX                                                     DC251050
   29 X(I)=CURR(I)*SMLE(I)                                               DC251060
 1700 WRITE(6,138)                                                       DC251070
      WRITE(6,139)                                                       DC251080
  138 FORMAT( // 21H ELEMENT POWER LOSSES //)                            DC251090
  139 FORMAT( 9H BRANCHES,12X,12HPOWER LOSSES/)                          DC251100
      KMAX=NMAX                                                          DC251110
      IND=5                                                              DC251120
      GO TO 100                                                          DC251130
C                                                                        DC251140
C     BRANCH CURRENTS                                                    DC251150
C                                                                        DC251160
  105 DO 30 I=1,NMAX                                                     DC251170
   30 CURR(I)=CURR(I)-AMPX(I)                                            DC251180
C                                                                        DC251190
C     CHECK UNBALANCES                                                   DC251200
C                                                                        DC251210
      DO 33 I=1,NNODE                                                    DC251220
   33 X(I)=0.                                                            DC251230
```

```
      DO 36 I=1,NMAX                                                    DC251240
      J=NINIT(I)                                                        DC251250
      K=NFIN(I)                                                         DC251260
      IF(K)34,34,35                                                     DC251270
   35 X(K)=X(K)+CURR(I)                                                 DC251280
   34 IF(J)36,36,37                                                     DC251290
   37 X(J)=X(J)-CURR(I)                                                 DC251300
   36 CONTINUE                                                          DC251310
      SUM=0.                                                            DC251320
      DO 38 I=1,NNODE                                                   DC251330
   38 SUM = SUM + DABS( X( I ))                                         DC251340
      IF(SUM-ERROR1)106,106,40                                          DC251350
   40 WRITE(6,141)                                                      DC251360
 2190  WRITE (6,142)                                                    DC251370
  141 FORMAT( // 43H SOLUTION NOT OBTAINED TO DESIRED TOLERANCE//)      DC251380
  142 FORMAT( 6H NODES,15X,19H CURRENT UNBALANCES /)                    DC251390
      KMAX=NNODE                                                        DC251400
      IND=6                                                             DC251410
      GO TO 100                                                         DC251420
  106 IF(JX4)300,300,900                                                DC251430
  300 IF(NPRINT(4))900,900,41                                           DC251440
   41 DO 42 I=1,NMAX                                                    DC251450
   42 X(I)=CURR(I)                                                      DC251460
      GO TO(2500,2600,2700,2701,2702),MYSYS                             AD250670
 2500  WRITE(6,140)                                                     DC251470
      WRITE(6,137)                                                      DC251480
  140 FORMAT(// 16H BRANCH CURRENTS//)                                  DC251490
      GO TO 2800                                                        AD250680
 2600  WRITE(6,1140)                                                    AD250690
      GO TO 2800                                                        AD250700
 1140  FORMAT(//' BRANCH FORCES',//,' BRANCHES',12X,'FORCES'/)          AD250710
 2700  WRITE(6,2140)                                                    AD250720
 2140  FORMAT (//,' BRANCH TORQUES',//,' BRANCHES',12X,'TORQUES'/)      AD250730
      GO TO 2800                                                        AD250740
 2701  WRITE(6,3140)                                                    AD250750
 3140  FORMAT(//' BRANCH HEAT FLOW',//' BRANCHES',12X,'HEAT FLOW'/)     AD250760
```

```
      GO TO 2600                                                        AD250770
2702  WRITE(6,4140)                                                     AD250780
4140  FORMAT(//,' BRANCH VOLUME VELOCITIES',//,' BRANCHES',12X,         AD250790
     1  'VELOCITIES',/)                                                 AD250800
2800   CONTINUE                                                         AD250810
      KMAX=NMAX                                                         DC251500
      IND=7                                                             DC251510
C                                                                       DC251520
C     OUTPUT ROUTINE                                                    DC251530
C                                                                       DC251540
  100 LAST=0                                                            DC251550
  150 K=LAST+1                                                          DC251560
      LAST=LAST+4                                                       DC251570
      IF(LAST-KMAX)200,200,201                                          DC251580
  201 LAST=KMAX                                                         DC251590
  200 WRITE(6,203)K,LAST,(X(J),J=K,LAST)                                DC251600
  203 FORMAT(1X,I3,1H-,I3,2X,4(3X,E15.6))                               DC251610
      IF(KMAX-LAST)500,500,150                                          DC251620
C                                                                       DC251630
  500 GO TO (101,102,103,104,105,41,900),IND                            DC251640
  900 RETURN                                                            DC251650
      END                                                               DC251660

      SUBROUTINE ECA51(IA)                                              AC510000
      DIMENSION POWRL(200)                                              AC510010
C                                                                       AC510020
      COMMON/MYS/MYSYS                                                  AD510010
      COMMON NMAX,NNODE,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,KTO,AC510030
     1 NPRINT(10)                                                       AC510040
      COMMON E(200),EMIN(200),EMAX(200),AMP(200),AMPMIN(200),AMPMAX(200)AC510050
      COMMON Y(200),YMIN(200),YMAX(200),NINIT(200),NFIN(200),MODE1(200) AC510060
      COMMON YTERM(200),YTERMH(200),YTERML(200),IROWT(200),ICOLT(200)   AC510070
      COMMON ERROR1,ISEQ,MSEQ,MO,NUMAC,VFIRST(50),VSECND(50),VLAST(50)  AC510080
      COMMON MOWRN(50),MCPARM(50),MCITER(50),IWCOLT(4),NLTRMS,OMEGA     AC510090
      COMMON IROWM(50),ICOLM(50),FLML(50),FLMH(50),FLM(50),EPHA(200)    AC510100
      COMMON AMPPHA(200),NREC                                           AC510110
```

```
C
C     THE FOLLOWING VARIABLES ARE USED ONLY IN THE ECAP A.C. ANALYSIS
C
      COMMON EQUCRL(50),EQUCIM(50),CIM(200),ELIM(200),EPRL(50),EPIM(50)
      COMMON FLMSAV(25),IRSAV(25),ICSAV(25),HNODE,CVOLTI(200),OMGINV
      COMMON DELTA,ITOL,LL,ITRANS,DUMX(2)
C
      COMMON LANG(265)
C
      COMMON ELSAV(200),CVOLTR(200),CAMPRL(200),CAMPIM(200)
C
      DOUBLE PRECISION EQUCRL,EQUCIM,EPRL,EPIM,CVOLTI,CVOLTR,CAMPRL,
     1 CAMPIM
      DOUBLE PRECISION EBIM(200),EBRL(200),BAMPRL(200),BAMPIM(200)
      EQUIVALENCE (EQUCRL(1),POWRL(1)),(CVOLTI(1),EBIM(1)),
     1  (CVOLTR(1),EBRL(1)),(CAMPRL(1),BAMPRL(1)),
     2 (CAMPIM(1),BAMPIM(1))
 1    IF(NTRACE)2,4,2
 2    WRITE(6, 3) IA
 3    FORMAT(' ECA51 ENTERED. IA=',I3 )
 4    IF ( IA - 1 ) 1000,552,1000
  552 IF(NPRINT(1)+NPRINT(2)+NPRINT(3)+NPRINT(4)+NPRINT(5)+NPRINT(6)
     1  +NPRINT(10))511,1000,511
 511  FREQ = OMEGA/6.283185
      WRITE(6,525) FREQ
 525  FORMAT(//,' FREQ =',E16.8)
 1000 IF(IA-11)1001,30,1001
 1001 IF(NPRINT(IA))7,500,7
 7    GO TO (21,22,23,24,25,26,27,28,29),IA
 100  FORMAT( /// 7X 5HNODES, 10X 14H NODE VOLTAGES //)
 101  FORMAT( /// 4X 8HBRANCHES 10X   17H ELEMENT CURRENTS  //)
 102  FORMAT( /// 4X 8HBRANCHES 10X 17H ELEMENT VOLTAGES  //)
 103  FORMAT(/// 4X 8HBRANCHES 10X 16H BRANCH CURRENTS //)
 104  FORMAT(/// 4X 8HBRANCHES 10X 16H BRANCH VOLTAGES //)
 105  FORMAT(/// 4X 8HBRANCHES 10X 16H BRANCH POWER //)
```

```
 106   FORMAT( /// 1X 24H SENSITIVITIES NOT CALC. //)                AC510470
 107   FORMAT( /// 1X 21H WORST-CASE NOT CALC. //)                   AC510480
 108   FORMAT( /// 1X 20H STD. DEV. NOT CALC. //)                    AC510490
 109   FORMAT(///,' SOLUTION NOT OBTAINED TO DESIRED TOLERANCE',//)  AC510500
 110   FORMAT(7X,5HNODES,10X,19H CURRENT UNBALANCES,/)               AC510510
 21    WRITE (6, 100 )                                               [illegible]
       GO TO 500                                                     [illegible]
 22    WRITE (6, 101 )                                               [illegible]
       GO TO 500                                                     [illegible]
 23    WRITE (6, 102 )                                               [illegible]
       GO TO 500                                                     [illegible]
 24    WRITE (6, 103 )                                               [illegible]
       GO TO 500                                                     [illegible]
 25    WRITE (6, 104 )                                               [illegible]
       GO TO 500                                                     [illegible]
 26    WRITE ( 6, 105 )                                              [illegible]
       GO TO 500                                                     [illegible]
2100   FORMAT(///,7X,'NODES',10X,'VELOCITIES AT NODES',//)           AD510020
3100    FORMAT(///,7X,'NODES' ,10X,'NODE ANGULAR VELOCITIES'//)      AD510030
4100   FORMAT(///,7X,'NODES',10X,'NODE TEMPERATURES'//)              AD510040
5100   FORMAT(///,7X,'NODES',10X,'NODE PRESSURES'//)                 AD510050
2101   FORMAT(///,7X,'BRANCHES',7X,'FORCES DEVELOPED IN ELEMENTS',//) AD510060
3101   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT TORQUES'//)             AD510070
4101   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT HEAT FLOWS'//)          AD510080
5101   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT VOLUME VELOCITIES'//)   AD510090
2102   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT VELOCITIES'//)          AD510100
3102   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT ANGULAR VELOCITIES'//)  AD510110
4102   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT TEMPERATURES'//)        AD510120
5102   FORMAT(///,4X,'BRANCHES',10X,'ELEMENT PRESSURES'//)           AD510130
2103   FORMAT(///,4X,'BRANCHES',10X,' BRANCH FORCES'///)             AD510140
3103   FORMAT(///,4X,'BRANCHES',10X,' BRANCH TORQUES'//)             AD510150
4103   FORMAT(///,4X,'BRANCHES',10X,' BRANCH HEAT FLOWS'//)          AD510160
5103   FORMAT(///,4X,'BRANCHES',10X,' BRANCH VOLUME VELOCITIES'//)   AD510170
2104   FORMAT(///,4X,'BRANCHES',10X,'BRANCH VELOCITIES'//)           AD510180
3104   FORMAT(///,4X,'BRANCHES',10X,' BRANCH ANGULAR VELOCITIES'//)  AD510190
```

```
4104  FORMAT(///,4X,'BRANCHES',10X,' BRANCH TEMPERATURES'//)
5104  FORMAT(///,4X,'BRANCHES',10X,' BRANCH PRESSURES'//)
21    GO TO (121,221,321,421,521),MYSYS
121    WRITE(6, 100)
       GOTO500
221     WRITE(6,2100)
321    WRITE(6,3100)
       GOTO500
421    WRITE(6,4100)
       GO TO 500
521    WRITE(6,5100)
       GO TO 500
22     GO TO (122,222,322,422,522),MYSYS
122    WRITE(6,101)
       GOTO500
222    WRITE(6,2101)
       GOTO500
322    WRITE(6,3101)
       GOTO500
422    WRITE(6,4101)
       GO TO 500
522    WRITE(6,5101)
       GO TO 500
23     GO TO (123,223,323,423,523),MYSYS
123    WRITE(6,102)
       GOTO500
223    WRITE(6,2102)
       GOTO500
323    WRITE(6,3102)
       GOTO500
423    WRITE(6,4102)
       GO TO 500
523    WRITE(6,5102)
       GO TO 500
24     GO TO (124,224,324,424,524),MYSYS
```

```
124   WRITE(6,103)                                          AD510550
      GOTO500                                               AD510560
224   WRITE(6,2103)                                         AD510570
      GOTO500                                               AD510580
324   WRITE(6,3103)                                         AD510590
      GOTO500                                               AD510600
424   WRITE(6,4103)                                         AD510610
      GO TO 500                                             AD510620
      GO TO 500                                             AD51063
524   WRITE(6,5103)                                         AD510640
      GO TO 500                                             AD510645
25    GO TO (125,225,325,425,525),MYSYS                     AD510650
125   WRITE(6,104)                                          AD510660
      GOTO500                                               AD510670
225   WRITE(6,2104)                                         AD510680
      GOTO500                                               AD510690
325   WRITE(6,3104)                                         AD510700
      GOTO500                                               AD510710
425   WRITE(6,4104)                                         AD510720
      GO TO 500                                             AD510730
525   WRITE(6,5104)                                         AD510740
      GO TO 500                                             AD510750
 27   WRITE ( 6, 106 )                                      AC510640
      GO TO 500                                             AC510650
 28   WRITE ( 6, 107 )                                      AC510660
      GO TO 500                                             AC510670
 29   WRITE ( 6, 108 )                                      AC510680
      GO TO 500                                             AC510690
 30   WRITE(6,109)                                          AC510700
      WRITE(6,110)                                          AC510710
 500  IF(NTRACE)9997,9999,9997                              AC510720
 9997 WRITE(6, 9998)                                        AC510730
 9998 FORMAT(1X 11H ECA51 EXIT )                            AC510740
 9999 RETURN                                                AC510750
      END                                                   AC510760
```

```
      SUBROUTINE PRINT2                                                 TR890000
C                                                                       TR890010
      DIMENSION X(200)                                                  TR890020
C                                                                       TR890030
C                                                                       TR890040
      COMMON LINKS,NODES,NTERMS,NUMBL,NUMBR,NUMBC,IRTN,NTRACE,NSWTCH,   TR890050
     1 KTO,NPRINT(10)                                                   TR890060
      COMMON/MYS/MYSYS                                                  AD890010
      COMMON E0(200),BIAS1(200),BIAS2(200),AMPO(200),FLOW1(200)         TR890070
      COMMON FLOW2(200),Y(200),Y1(200),Y2(200),NT(200),NH(200)          TR890080
      COMMON NODE1(200),LIST4(200),YTERM2(200),YTERM1(200),IROWT(200)   TR890090
      COMMON ICOLT(200),ERROR1,START,FINISH,NSTEP,SHORT,OPEN,LABEL(200) TR890100
      COMMON B(50),KE,KI,LOCKS,NLTRMS,DELTA,IROWM(50),ICOLM(50),FLM1(50)TR890110
      COMMON FLM2(50),LISTE(5),LISTI(5),NUME(5),NUMI(5),JSTEP(10)       TR890120
      COMMON JSTEPS(10),JLINK(10),LINK1D(200),LINK1E(200),NREC,MAJOR    TR890130
      COMMON ERROR2,ERROR3,ETIME(5,2),ATIME(5,2),ETR(5,126),AMPTR(5,126)TR890140
C                                                                       TR890150
      COMMON LANG(265)                                                  TR890160
C                                                                       TR890170
C     THE FOLLOWING VARIABLES ARE USED ONLY BY TRANSIENT ANALYSIS       TR890180
C                                                                       TR890190
      COMMON V(50),V0(50),FLUX(200),VALUE(200),LEVER(200),LEVERS(200)   TR890200
      COMMON LINK1A(200),TTAU(10),ESLOPE(5),ASLOPE(5),HOLD              TR890210
      COMMON LEAST,LIST,LATCH,LOCK,LOCKA,LOCKB,LOCKD,LOCKF,LOCKG,MINOR  TR890220
      COMMON NUMBER,PARTS,SAVE,STEP,T,TEST,THIGH,TLOW,TSTAR,TZERO       TR890230
      COMMON UNIT,JX1                                                   TR890240
C                                                                       TR890250
      DOUBLE PRECISION B,V,V0,FLUX,VALUE                                TR890260
    1 IF( NTRACE ) 3, 4, 3                                              TR890270
    2 FORMAT( 7H PRINT2 )                                               TR890280
    3 WRITE(6, 2)                                                       TR890290
    4 NPR = 0                                                           TR890300
      DO 5 I=1,6                                                        TR890310
    5 NPR = NPR +NPRINT(I)                                              TR890320
      IF ( LOCKF ) 8100, 500, 500                                       TR890330
```

```
 500  IF(LOCK) 1000, 1007, 1000                                    TR890340
 1000 IF( JX1 ) 1001, 1001, 8090                                   TR890350
 1001 CALL SSWITCH ( 1,LOST)                                        TR890360
      GO TO ( 1004 , 1003 ), LOST                                  TR890370
 1003 IF( NPR ) 1005, 1005, 1004                                   TR890380
 1004 WRITE( 6, 10 ) T                                             TR890390
 10   FORMAT(///,'     T = ', E15.7, /)                            TR890400
 1005 GO TO ( 1008, 1007 ), LOST                                   TR890410
 1007 IF( NPRINT(1)) 1010, 1010, 1008                              TR890420
 1008 WRITE (6, 11 )                                               TR890430
 11   FORMAT(/,' NODES',13X,'NODE VOLTAGES',/)                     TR890440
1008  GOTO (11008,21008,31008,41008,51008),NYSYS                   AD890020
11008 WRITE(6,11)                                                  AD890030
      GO TO 61008                                                  AD890040
21008 WRITE(6,211)                                                 AD890050
      GO TO 61008                                                  AD890060
31008 WRITE(6,311)                                                 AD890070
      GO TO 61008                                                  AD890080
41008 WRITE(6,411)                                                 AD890090
      GO TO 61008                                                  AD890100
51008 WRITE(6,511)                                                 AD890110
61008 CONTINUE                                                     AD890120
      LIMIT=NODES                                                  TR890450
      JX2 = 1                                                      TR890460
      DO 1009 I = 1, NODES                                         TR890470
 1009 X(I) = V(I)                                                  TR890480
      GO TO 9010                                                   TR890490
 1010 IF( NPRINT(5)+NPRINT(6)+NPRINT(3))1023,1023,1011             TR890500
 1011 DO 1015 I = 1, LINKS                                         TR890510
      J = NT(I)                                                    TR890520
      IF(J)1013,1013,1012                                          TR890530
 1012 X(I) = V(J)                                                  TR890540
 1013 J= NH(I)                                                     TR890550
      IF( J ) 1015,1015,1014                                       TR890560
 1014 X(I)=X(I) - V(J)                                             TR890570
```

```
 1015 CONTINUE
      IF ( NPRINT( 5 )) 1018, 1018, 1016
1016  GO TO (11016,21016,31016,41016,51016),MYSYS
11016 WRITE(6,1017)
      GO TO 61016
21016   WRITE(6,21017)
      GO TO 61016
31016 WRITE(6,31017)
      GO TO 61016
41016 WRITE(6,41017)
      GO TO 61016
51016 WRITE(6,51017)
61016 CONTINUE
 1016 WRITE( 6, 1017 )
 1017 FORMAT(/,' BRANCHES',12X,'BRANCH VOLTAGES',/)
      LIMIT = LINKS
      JX2 = 2
      GO TO 9010
 1018 IF( NPRINT(6) + NPRINT(3)) 1023, 1023, 1019
 1019 DO 1020  I = 1, LINKS
 1020 X(I) =X(I)+ BIASL(I)
      IF(NPRINT(3))1023,1023,1021
 1021 JX2=3
      LIMIT=LINKS
      WRITE(6,1022)
     GO TO (190,290,390,490,590),MYSYS
190   WRITE(6,1022)
      GOTO 9010
290   WRITE(6,21022)
      GOTO 9010
390   WRITE(6,31022)
      GO TO 9010
490   WRITE(6,41022)
      GO TO 9010
590   WRITE(6,51022)
```

```
 1022 FORMAT(/,' BRANCHES',12X,'ELEMENT VOLTAGES',/)                      TR890720
      GO TO 9010                                                          TR890730
 1023 GO TO ( 1026, 1025 ), LOST                                          TR890740
 1025 IF ( NPRINT(2)) 1027, 1027, 1026                                    TR890750
 1026 WRITE(6, 30)                                                        TR890760
1026  GO TO (11026,21026,31026,41026,51026),MYSYS                         AD890340
11026 WRITE(6,30)                                                         AD890350
      GO TO 61026                                                         AD890360
21026 WRITE(6,230)                                                        AD890370
      GO TO 61026                                                         AD890380
31026 WRITE(6,330)                                                        AD890390
      GO TO 61026                                                         AD890400
41026 WRITE(6,430)                                                        AD890410
      GO TO 61026                                                         AD890420
51026 WRITE(6,530)                                                        AD890430
61026 CONTINUE                                                            AD890440
 30   FORMAT(/,' BRANCHES',12X,'ELEMENT CURRENTS', /)                     TR890770
      JX2 = 4                                                             TR890780
      LIMIT = LINKS                                                       TR890790
      GO TO 9010                                                          TR890800
 1027 IF (NPRINT(3)) 1031, 1031, 1028                                     TR890810
 1028 DO 1029  I = 1, LINKS                                               TR890820
 1029 X(I) = X(I) * FLUX(I)                                               TR890830
      WRITE(6, 1030)                                                      TR890840
 1030 FORMAT(/,' BRANCHES',12X,'INSTANTANEOUS ELEMENT POWER',/)           TR890850
      JX2 = 5                                                             TR890860
      LIMIT = LINKS                                                       TR890870
      GO TO 9010                                                          TR890880
 1031 IF( NPRINT(4)) 8000, 8000, 1032                                     TR890890
 1032 DO 1033 I = 1, LINKS                                                TR890900
 1033 X(I) = FLUY(I) - FLOW1(I)                                           TR890910
      WRITE( 6, 1034 )                                                    TR890920
      GO TO (191,291,391,491,591),MYSYS                                   AD890450
191   WRITE(6,1034)                                                       AD890460
      GO TO 691                                                           AD890470
```

```
291   WRITE(6,21034)                                        AD890480
      GO TO 691                                             AD890490
391   WRITE(6,31034)                                        AD890500
      GO TO 691                                             AD890510
491   WRITE(6,41034)                                        AD890520
      GO TO 691                                             AD890530
591   WRITE(6,51034)                                        AD890540
691   CONTINUE                                              AD890550
 1034 FORMAT(/,' BRANCHES',12X,'BRANCH CURRENTS',/)         TR890930
      JX2 = 6                                               TR890940
      LIMIT = LINKS                                         TR890950
 9010 LAST =0                                               TR890960
 9011 K = LAST + 1                                          TR890970
      LAST = LAST + 4                                       TR890980
      IF ( LAST - LIMIT ) 9013, 9013, 9012                  TR890990
 9012 LAST = LIMIT                                          TR891000
 9013 IF( JX2 - 4 ) 9014, 9015, 9014                        TR891010
 9014 WRITE(6,9016) K, LAST, ( X(J), J = K, LAST )          TR891020
      GO TO 9017                                            TR891030
 9015 WRITE(6,9016) K, LAST, (FLUX(J), J = K, LAST )        TR891040
 9017 IF (LIMIT - LAST ) 9018, 9018, 9011                   TR891050
 9018 GO TO ( 1010, 1018, 1023, 1027, 1031, 8000), JX2      TR891060
 9016 FORMAT(1X,I3,'-',I3,2X,4(5X,E15.8))                   TR891070
 8000 IF ( LOCAF ) 9000, 9000, 8100                         TR891080
 8100 DO 8400 N = 1, LIST                                   TR891090
      M = LEVER( N )                                        TR891100
      LATCH = LABEL( M )                                    TR891110
      GO TO ( 8200, 8250 ), LATCH                           TR891120
 8200 LABEL( M ) = 2                                        TR891130
      WRITE(6,90) M                                         TR891140
 90   FORMAT(//,' SWITCH',I3,' IS ON')                      TR891150
      GO TO 8400                                            TR891160
 8250 LABEL( M ) = 1                                        TR891170
      WRITE(6,91)M                                          TR891180
 91   FORMAT(//,' SWITCH',I3,' IS OFF')                     TR891190
```

```
 8400 CONTINUE                                                            TR891200
      GO TO ( 8500, 8600 ), LOCKP                                          TR891210
 8500 LOCKB = 2                                                           TR891220
      GO TO 8700                                                          TR891230
 8600 DELTA = HOLD - ( T - TSTAR )                                        TR891240
 8700 LOCK = 1                                                            TR891250
      LOCKF = 0                                                           TR891260
211   FORMAT(/,' NODES',15X,'NODE VELOCITIES',/)                          AD890560
311   FORMAT(/,' NODES',15X,'ANGULAR VELOCITIES AT NODES',/)              AD890570
411   FORMAT(/,' NODES',15X,'TEMPERATURES AT NODES',/)                    AD890580
511   FORMAT(/,' NODES',15X,'NODE PRESSURES',/)                           AD890590
230   FORMAT(/,' BRANCHES',12X,'ELEMENT FORCES',/)                        AD890600
330   FORMAT(/,' BRANCHES',12X,'ELEMENT TORQUES',/)                       AD890610
430   FORMAT(/,' BRANCHES',12X,'ELEMENT HEAT FLOWS',/)                    AD890620
530   FORMAT(/,' BRANCHES',12X,'ELEMENT VOLUME VELOCITIES',/)             AD890630
21034 FORMAT(/,' BRANCHES',12X,'FORCES IN BRANCHES',/)                    AD890640
31034 FORMAT(/,' BRANCHES',12X,'FORCES IN BRANCHES',/)                    AD890650
31034 FORMAT(/,' BRANCHES',12X,'TORQUES IN BRANCHES',/)                   AD890660
41034 FORMAT(/,' BRANCHES',12X,'HEAT FLOW IN BRANCHES',/)                 AD890670
51034 FORMAT(/,' BRANCHES',12X,'VOLUME VELOCITY IN BRANCHES'/)            AD890680
21017 FORMAT(/,' BRANCHES',12X,'BRANCH VELOCITIES',/)                     AD890690
31017 FORMAT(/,' BRANCHES',12X,'BRANCH  ANGULAR VELOCITIES',/)            AD890700
41017 FORMAT(/,' BRANCHES',12X,'BRANCH TEMPERATURES',/)                   AD890710
51017 FORMAT(/,' BRANCHES',12X,'BRANCH PRESSURES',/)                      AD890720
21022 FORMAT(/,' BRANCHES',12X,'ELEMENT VELOCITIES',/)                    AD890730
31022 FORMAT(/,' BRANCHES',12X,'ELEMENT ANGULAR VELOCITIES',/)            AD890740
41022 FORMAT(/,' BRANCHES',12X,'ELEMENT TEMPERATURES',/)                  AD890750
51022 FORMAT(/,' BRANCHES',12X,'ELEMENT PRESSURES',/)                     AD890760
 9000 RETURN                                                              TR891270
      END                                                                 TR891280
```

APPENDIX D

Sample Problems and UHSAP Solutions

```
MECHANICAL SYSTEM PROBLEM
C
C        EXAMPLE # 1
C
C        DYNAMIC ANALYSIS PROBLEM
C
        DYNAMIC ANALYSIS;
P1      N(1,0),K=5E-3;
P2      N(1,0),M=8;
P3      N(1,2),K=2.E-3;
P4      N(1,2),B=4.0E-3;
P5      N(2,0),M=10,V=6;
        TIME STEP=1.0E-1;
        OUTPUT INTERVAL=10;
        FINAL TIME=1.0E+2;
        PRINT,VELOCITIES,FORCES;
        EXECUTE;
        END;
```

```
   T =      0.0

NODES                    VELOCITIES AT NODES

  1-  2      -0.23998658E-03    -0.59975979 01

BRANCHES                 FORCES DEVELOPED IN ELEMENTS

  1-  4      -0.23998650D-10    -0.23998652D-01     0.59995180D-06     0.23998620D-01
  5-  5       0.23998620D-01
```

```
T =     .9999996  00

NODES              VELOCITIES AT NODES

 1-  2     -0.65345168E-02   -0.59947233E 01

BRANCHES           FORCES DEVELOPED IN ELEMENTS

 1-  4     -0.14466856D-04   -0.71896759D-01    0.47958407D-01    0.23952739D-01
 5-  5      0.71911236D-01


T =     .1999994  01

NODES              VELOCITIES AT NODES

 1-  2     -0.18602162E-01   -0.59849052E 01

BRANCHES           FORCES DEVELOPED IN ELEMENTS

 1-  4     -0.75350897D-04   -0.11957174D 00    0.95782606D-01    0.23864395D-01
 5-  5      0.11964709D 00


T =     .2999989  01

NODES              VELOCITIES AT NODES

 1-  2     -0.37037090E-01   -0.59707264E 01
```

BRANCHES FORCES DEVELOPED IN ELEMENTS

```
1- 4    -0.21243407D-03   -0.16690820D 00    0.14338737D 00    0.23733262D-01
5- 5     0.16712083D 00
```

```
T =   .39999983E 01
```

NODES VELOCITIES AT NODES

```
1- 2    -0.61039106E-01   -0.59510193E 01
```

BRANCHES FORCES DEVELOPED IN ELEMENTS

```
1- 4    -0.45529317D-03   -0.21379164D 00    0.19068726D 00    0.23559665D-01
5- 5     0.21424692D 00
```

```
T =   .49999977E 01
```

NODES VELOCITIES AT NODES

```
1- 2    -0.91013730E-01   -0.59270229E 01
```

BRANCHES FORCES DEVELOPED IN ELEMENTS

```
1- 4    -0.83317837D-03   -0.26010267D 00    0.23759783D 00    0.23344020D-01
5- 5     0.26094185D 00
```

```
T =     .5999971E 01

NODES                    VELOCITIES AT NODES

 1-  2     -0.12667257E 00   -0.55983841E 01

BRANCHES                 FORCES DEVELOPED IN ELEMENTS

 1-  4     -0.13750424D-02   -0.30574726D 00    0.28403547D 00    0.23086833D-01
 5-  5      0.30712230D 00


T =     .6999966E 01

NODES                    VELOCITIES AT NODES

 1-  2     -0.16793306E 00   -0.58651590E 01

BRANCHES                 FORCES DEVELOPED IN ELEMENTS

 1-  4     -0.21093705D-02   -0.35059699D 00    0.32991767D 00    0.22788690D-01
 5-  5      0.35270636D 00


T =     .79999[illegible] 01

NODES                    VELOCITIES AT NODES

 1-  2     -0.21483322E 00   -0.58274126E 01
```

```
BRANCHES             FORCES DEVELOPED IN ELEMENTS

  1-  4      -0.30641616D-02   -0.39454933D 00    0.37516321D 00    0.22450280D-01
  5-  6       0.39761349D 00


   T =    0.8999954E 01

NODES                VELOCITIES AT NODES

  1-  2      -0.26712143E 00   -0.57852163E 01

BRANCHES             FORCES DEVELOPED IN ELEMENTS

  1-  4      -0.42658505D-02   -0.43749781D 00    0.41069229D 00    0.22072366D-01
  5-  6       0.44176466D 00


   T =    0.9999949E 01

NODES                VELOCITIES AT NODES

  1-  2      -0.32469742E 00   -0.57386503E 01

BRANCHES             FORCES DEVELOPED IN ELEMENTS

  1-  4      -0.57442417D-02   -0.47533830D 00    0.46342675D 00    0.21655799E-01
  5-  6       0.49508254D 00
```

Example 2

The system shown in Figure P.2 is a mechanical translational system. It is required to determine the frequency response of the system for the applied alterating force f, by studying the forces developed in various elements for different frequencies of f.

UHSAP coding for the problem and the solution given by the program follows in next pages.

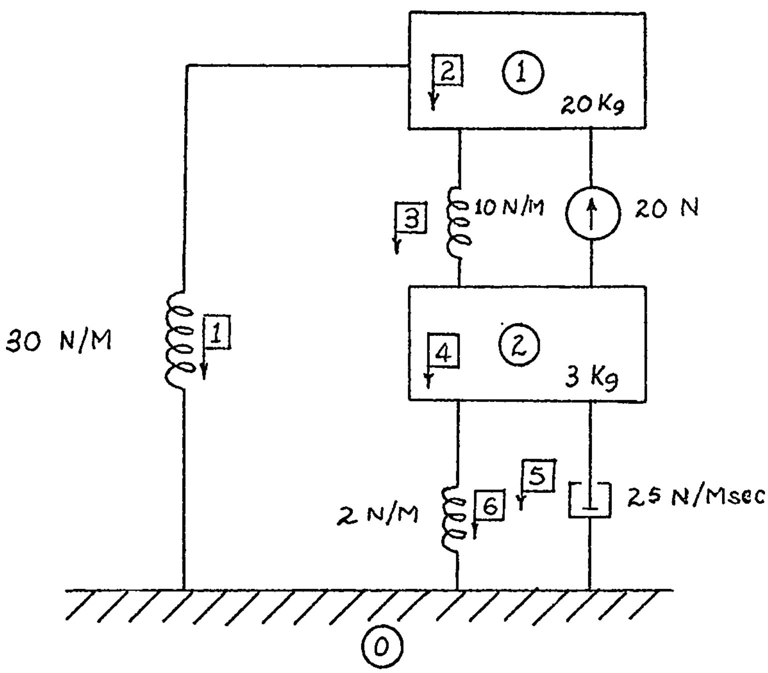


Fig. P.2

```
MECHANICAL SYSTEM PROBLEM
C
C       EXAMPLE # 2
C       ALTERNATING FORCE ANALYSIS PROBLEM
C       FOR A MECHANICAL SYSTEM
C
        AF ANALYSIS;
P1      N(1,0),K=30;
P2      N(1,0),M=20;
P6      N(2,0),K=2.0;
P3      N(1,2),K=10,F=20/0.;
P4      N(2,0),M=3.0;
P5      N(2,0),B=25.0;
        FR=0.01;
        PRINT,VELOCITIES,FORCES;
        EX;
        MODIFICATION # 1 FOR FREQUENCY RESPONSE;
        FR=0.1(10)1000.;
        EX;
        FN;
```

FREQ = 0.99999942E-02

NODES VELOCITIES AT NODES

MAG	1- 2	0.82975067E-02	0.97993122E-01
PHA		0.11890637E 03	-0.99405029E 02

BRANCHES FORCES DEVELOPED IN ELEMENTS

MAG	1- 4	0.39617691E 01	0.18425946E-01	0.16650848E 02	0.19469371E-01
PHA		0.28906372E 02	-0.15109355E 03	-0.65586254E 01	-0.94050503E 01
MAG	5- 6	0.24495773E 01	0.31169003E 01		
PHA		-0.99495029E 02	0.17599489E 03		

```
FREQ =  0.99999905E-01

      NODES              VELOCITIES AT NODES

MAG       1-  2  0.35193288E 00   0.49457574E 00
PHA              0.11314388E 03 -0.15383361E 03

      BRANCHES           FORCES DEVELOPED IN ELEMENTS

MAG       1-  4  0.16803558E 02   0.44225159E 01   0.98947048E 01   0.93225235E 00
PHA              0.23143875E 02 -0.13685605E 03 -0.29459274E 02 -0.63883621E 02

MAG       5-  6  0.12364387E 02   0.15742826E 01
PHA             -0.15338361E 03   0.11511632E 03

FREQ =  0.99999952E 00

      NODES              VELOCITIES AT NODES

MAG       1-  2  0.17242748E 00   0.67138332E 00
PHA             -0.87540059E 02   0.14591224E 03
```

```
BRANCHES        FORCES DEVELOPED IN ELEMENTS

MAG     1-  4  0.82519078E 02  0.21718124E 02  0.12519941E 01  0.12655272E 02
PHA            -0.17753999E 03  0.24599609E 01 -0.13425359E 03 -0.12408765E 03

MAG     5-  6  0.16784376E 02  0.21370810E 00
PHA             0.14591229E 03  0.55912277E 02

FLOW =  0.99799933  01

NODES           VELOCITIES AT NODES

MAG     1-  2  0.15936770E-01  0.10530049E 00
PHA            -0.89993637E 02  0.97562531E 02

BRANCHES        FORCES DEVELOPED IN ELEMENTS

MAG     1-  4  0.76092668E-02  0.20026749E 02  0.19276395E-01  0.19848648E 02
PHA            -0.17999367E 03  0.03134693E-02 -0.17342882E 03 -0.17243741E 03

MAG     5-  6  0.26325111E 01  0.3351F702--02
PHA             0.97562531E 02  0.75625725E 01
```

F W = 0.99999929E 02

NODES VELOCITIES AT NODES

MAG	1-	2	0.15915723E-02	0.10609526E-01
PHA			-0.89999969E 02	0.90759827E 02

BRANCHES FORCES DEVELOPED IN ELEMENTS

MAG	1-	4	0.75992022E-04	0.20000259E 02	0.19418474E-03	0.19998459E 02
PHA			-0.17999995E 03	0.64152055E-05	-0.17933923E 03	-0.17924011E 03
MAG	5-	6	0.26523805E 00	0.33771168E-04		
PHA			0.90759827E 02	0.75957220E 00		

FREQ = 0.99999927E 03

NODES VELOCITIES AT NODES

MAG	1-	2	0.15915514E-03	0.10610336E-02
PHA			-0.89999969E 02	0.90759745E 02

BRANCHES FORCES DEVELOPED IN ELEMENTS

```
MAG    1-  4  0.75991645E-03  0.20000000E 02  0.19419922E-05  0.19999969E 02
PHA          -0.17999995E 03  0.64162471E-08 -0.17993330E 03 -0.17992395E 03

MAG    5-  6  0.26525622E-01  0.33773756E-06
PHA           0.90075974E 02  0.75990856E-01
```

FREQ = 0.99999838E 04

NODES VELOCITIES AT NODES

```
MAG    1-  2  0.15915517E-04  0.10610346E-03
PHA          -0.89999959E 02  0.90007568E 02
```

BRANCHES FORCES DEVELOPED IN ELEMENTS

```
MAG    1-  4  0.75991018E-08  0.20000000E 02  0.19419939E-07  0.19999985E 02
PHA          -0.17999995E 03  0.64162590E-11 -0.17999335E 03 -0.17999236E 03

MAG    5-  6  0.26525850E-02  0.33772776E-08
PHA           0.90007568E 02  0.75990930E-02
```

Example 3

Figure P.3 shows a mechanical rotational system. The angular velocity applied at one end varies with time and is of triangular shape. It is required to study the transient effect of this source to this system after the system is started from a resting position.

UHSAP coding for the problem and the program solutions are in next page.

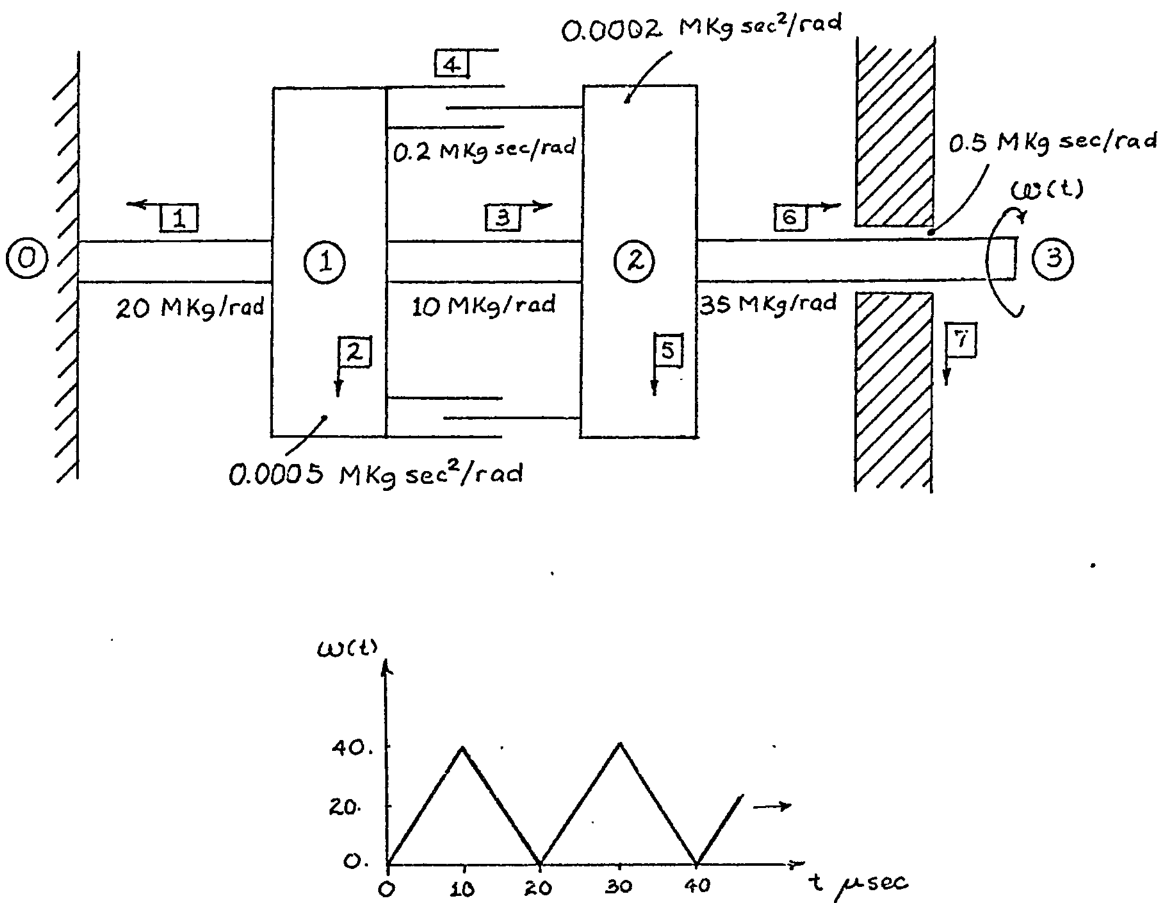

0.0002 MKg sec²/rad
4
0.2 MKg sec/rad
0.5 MKg sec/rad
ω(t)
1
3
6
0
1
2
3
20 MKg/rad
10 MKg/rad
35 MKg/rad
2
5
7
0.0005 MKg sec²/rad
ω(t)
40.
20.
0.
0
10
20
30
40
t μsec

Fig. P.3

```
ROTATIONAL
C
C      EXAMPLE # 3
C
C      DYNAMIC ANALYSIS OF A
C      ROTATIONAL SYSTEM WITH TIME VARIANT
C      ANGULAR VELOCITY APPLIED AT ONE END
C
       DYNAMIC ANALYSIS;
P1     N(1,0),K=20;
P2     N(1,0),J=5E-4;
P3     N(1,2),K=10.;
P4     N(1,2),J=0.2;
P5     N(2,0),J=2E-4;
P6     N(2,3),K=35;
P7     N(3,0),B=0.5;
W7     P(5) ,0.,20.,40.,20.,0.   ;
       TIME STEP=1E-6    ;
       OUTPUT INTERVAL=10;
       FINAL TIME=1E-4;
       PRINT ,ANGULAR VELOCITIES,TORQUES;
       EX;
       EN;
```

```
T =    0.0

NODES                    ANGULAR VELOCITY AT NODES

  1-   3      0.0                  0.0                  0.0

BRANCHES                 TORQUE DEVELOPED IN ELEMENTS

  1-   4      0.0                  0.0                  0.0                  0.0
  5-   7      0.0                  0.0                  0.0


T =    0.9999994E-05

NODES                    ANGULAR VELOCITY AT NODES

  1-   3    -0.16874941E-06    -0.13430273E-03    -0.39985901E 02

BRANCHES                 TORQUE DEVELOPED IN ELEMENTS

  1-   4    -0.74900199D-11    -0.26830336D-04     0.35475983D-08     0.26826796D-04
  5-   7    -0.69715012D-02     0.69983315D-02     0.69983315D-02


T =    0.1999996E-04

NODES                    ANGULAR VELOCITY AT NODES

  1-   3    -0.19112531E-05    -0.72990451E-03     0.28066780E-01

BRANCHES                 TORQUE DEVELOPED IN ELEMENTS

  1-   4    -0.17452842D-09    -0.14564261D-03     0.44130439D-07     0.14559866D-03
  5-   7    -0.13844355D-01     0.13989999D-01     0.13989998D-01
```

```
     T =    0.2999989E-04

NODES                  ANGULAR VELOCITY AT NODES

  1-  3      -0.65014792E-05   -0.15523736E-02   -0.39957870E 02

BRANCHES               TORQUE DEVELOPED IN ELEMENTS

  1-  4      -0.96128597D-09   -0.30932822D-03    0.15474781D-06    0.30917443D-03
  5-  7      -0.20668806D-01    0.20978136D-01    0.20978136D-01


     T =    0.3999982E-04

NODES                  ANGULAR VELOCITY AT NODES

  1-  3      -0.15391200E-04   -0.28288364E-02    0.56081839E-01

BRANCHES               TORQUE DEVELOPED IN ELEMENTS

  1-  4      -0.30641003D-08   -0.56305824D-03    0.37027281D-06    0.56269103D-03
  5-  7      -0.27396316D-01    0.27959377D-01    0.27959377D-01


     T =    0.4999975E-04

NODES                  ANGULAR VELOCITY AT NODES

  1-  3      -0.29785922E-04   -0.43249428E-02   -0.39929871E 02

BRANCHES               TORQUE DEVELOPED IN ELEMENTS

  1-  4      -0.74824352D-08   -0.85974714D-03    0.72270868D-06    0.85903192D-03
  5-  7      -0.34077106D-01    0.34936861D-01    0.34936861D-01
```

```
T =    0.5999969E-04

NODES                 ANGULAR VELOCITY AT NODES

 1-  3     -0.51119365E-04   -0.62679090E-02    0.84053814E-01

BRANCHES              TORQUE DEVELOPED IN ELEMENTS

 1-  4     -0.15444268D-07   -0.12445888D-02    0.12459449D-05    0.12433583D-02
 5-  7     -0.40662617D-01    0.41907221D-01    0.41907221D-01


T =    0.6999962E-04

NODES                 ANGULAR VELOCITY AT NODES

 1-  3     -0.80579426E-04   -0.84234662E-02   -0.39901932E 02

BRANCHES              TORQUE DEVELOPED IN ELEMENTS

 1-  4     -0.28473140D-07   -0.16705203D-02    0.19710131D-05    0.16685778D-02
 5-  7     -0.47203050D-01    0.48873599D-01    0.48873599D-01


T =    0.7999955E-04

NODES                 ANGULAR VELOCITY AT NODES

 1-  3     -0.11953272E-03   -0.11013906E-01    0.11198092E 00

BRANCHES              TORQUE DEVELOPED IN ELEMENTS

 1-  4     -0.48313635D-07   -0.21827577D-02    0.29308431D-05    0.21798752D-02
 5-  7     -0.53649826D-01    0.55832632D-01    0.55832632D-01
```

T = 0.8999948E-04

NODES ANGULAR VELOCITY AT NODES

1- 3	-0.16914043E-03	-0.13820048E-01	-0.39874023E 02

BRANCHES TORQUE DEVELOPED IN ELEMENTS

1- 4	-0.76998517D-07	-0.27342601D-02	0.41555186D-05	0.27301816D-02
5- 7	-0.60053126D-01	0.62787464D-01	0.62787464D-01	

T = 0.9999941E-04

NODES ANGULAR VELOCITY AT NODES

1- 3	-0.23073291E-03	-0.17054256E-01	0.13986140E 00

BRANCHES TORQUE DEVELOPED IN ELEMENTS

1- 4	-0.11677430D-06	-0.33702653D-02	0.56770331D-05	0.33647051D-02
5- 7	-0.66364352D-01	0.69734734D-01	0.69734734D-01	

T = 0.1009994E-03

NODES ANGULAR VELOCITY AT NODES

1- 3	-0.23760376E-03	-0.17386075E-01	-0.38600082E 01

BRANCHES TORQUE DEVELOPED IN ELEMENTS

1- 4	-0.12145766D-06	-0.34354200D-02	0.58468930D-05	0.34296945D-02
5- 7	-0.66363692D-01	0.69799234D-01	0.69799234D-01	

Example 4*

This example has been considered to show the validity of the results of UHSAP analysis with those by analytical methods.

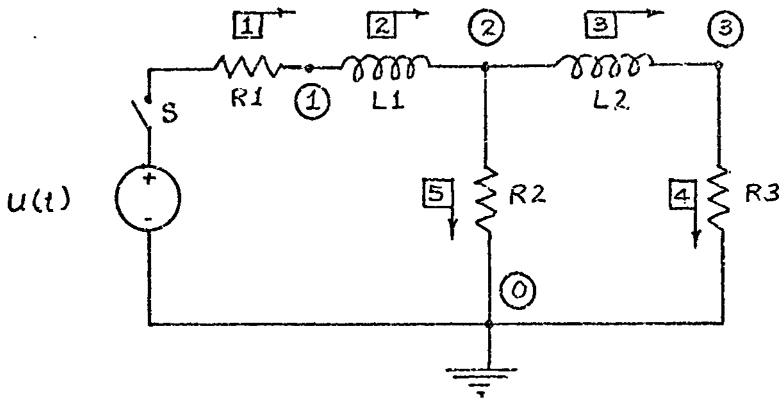
Consider the electrical circuit in Figure P.4. The switch S is closed at time t = 0 and the transient current in the resistor R3 at time t = t is given by

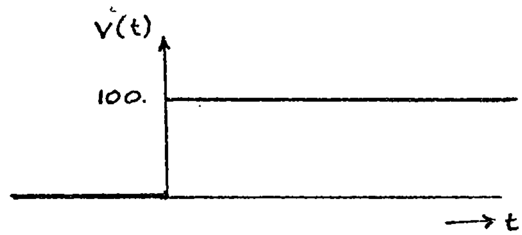
$$i(t) = 3.33 - 5e^{-10t} + 1.67e^{-30t}$$

The numerical results given by this expression at different value of t are tabulated in Figure P4.1 and UHSAP results which are in the following pages for actual comparison.

Figure P4.2 shows the computed values with both the methods.

*Adapted from Ref. 1, pages 159-160.





R1 = 10 Ω L1 = 1 H

R2 = 10 Ω L2 = 1H

R3 = 10 Ω

Fig P.4

```
ELECTRICAL SYSTEM
C
C      EXAMPLE # 4
C
       TR
C
B1     N(0,1),R=10,E=100
B2     N(1,2),L=1
B3     N(2,3),L=1
B4     N(3,0),R=10
B5     N(2,0),R=10
       TI=0.001
       DU=20
       FI=0.2
       PRINT, CURRENTS
       EX
       EN
```

```
   T =    0.0

BRANCHES                ELEMENT CURRENTS

  1-  4       0.99999766D-05    0.99999766D-05    0.99999567D-11    0.99999567D-11
  5-  5       0.99999666D-05


   T =    0.1999997E-01

BRANCHES                ELEMENT CURRENTS

  1-  4       0.165838120 01    0.165838120 01    0.154332600 00    0.154332600 00
  5-  5       0.150404860 01
```

```
T =    0.3999994E-01

BRANCHES                ELEMENT CURRENTS

  1-  4      0.28131364D 01    0.28131364D 01    0.48369116D 00    0.48369116D 00
  5-  5      0.23294452D 01


T =    0.5999990E-01

BRANCHES                ELEMENT CURRENTS

  1-  4      0.36471636D 01    0.36471636D 01    0.86475150D 00    0.86475150D 00
  5-  5      0.27824121D 01


T =    0.7999963E-01

BRANCHES                ELEMENT CURRENTS

  1-  4      0.42688687D 01    0.42688687D 01    0.12378743D 01    0.12378743D 01
  5-  5      0.30309945D 01


T =    0.9999937E-01

BRANCHES                ELEMENT CURRENTS

  1-  4      0.47443255D 01    0.47443255D 01    0.15769123D 01    0.15769123D 01
  5-  5      0.31674132D 01
```

T = 0.1199991E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.51151833D 01	0.51151833D 01	0.18729054D 01	0.18729054D 01
5- 5	0.32422780D 01			

T = 0.1399989E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.54087110D 01	0.54087110D 01	0.21253481D 01	0.21253481D 01
5- 5	0.32833628D 01			

T = 0.1599986E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.56434854D 01	0.56434854D 01	0.23375757D 01	0.23375757D 01
5- 5	0.33059096D 01			

T = 0.1799983E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.58326589D 01	0.58326589D 01	0.25143759D 01	0.25143759D 01
5- 5	0.33182830D 01			

T = 0.1999981E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.59858708D 01	0.59858708D 01	0.26607974D 01	0.26607974D 01
5- 5	0.33250734D 01			

T = 0.2009981E 00

BRANCHES	ELEMENT CURRENTS			
1- 4	0.59927258D 01	0.59927258D 01	0.26674083D 01	0.26674083D 01
5- 5	0.33253175D 01			

TIME	CURRENT IN BRANCH NO. 4
0.0	-0.95367432D-06
0.20000000D-01	0.15286110
0.39999999D-01	0.48139375
0.59999999D-01	0.86199072
0.79999998D-01	1.2348540
0.99999998D-01	1.5737470
0.12000000	1.8696594
0.14000000	2.1220577
0.16000000	2.3342610
0.18000000	2.5110481
0.20000000	2.6574630

Figure P4.1

CURRENT IN BRANCH NO.4

TIME	UHSAP VALUE	ANALYTICAL VALUE
0.00	0.00	0.00
0.02	0.1543	0.1528
0.04	0.4836	0.4814
0.06	0.8647	0.8620
0.08	1.2379	1.2348
0.10	1.5770	1.5737
0.12	1.8729	1.8696
0.14	2.1253	2.1221
0.16	2.3375	2.3342
0.18	2.5148	2.5110
0.20	2.6607	2.6574

Figure P4.2