

AD 685191

AFFDL-TR-68-56
VOLUME III

MAGIC: AN AUTOMATED GENERAL PURPOSE SYSTEM FOR STRUCTURAL ANALYSIS

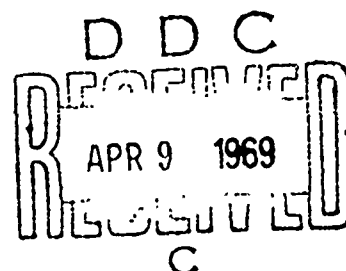
VOLUME III: PROGRAMMER'S MANUAL

DANIEL DeSANTIS

Bell Aerosystems, a Textron Company

TECHNICAL REPORT AFFDL-TR-68-56

JANUARY, 1969



This document has been approved for public
release and sale; its distribution is unlimited.

AIR FORCE FLIGHT DYNAMICS LABORATORY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO

Reproduction by the
CLEARINGHOUSE
for Federal Scientific & Technical
Information Springfield, Va. 22151

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This document has been approved for public release and sale; its distribution is unlimited.

ACQUISITION FOR	
CFSTI	WHITE SECTION ☒
DDC	BUFF SECTION ☐
UNANNOUNCED	☐
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
DIST.	AVAIL. AND/OR SPECIAL
/	

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

**AFFDL-TR-68-56
VOLUME III**

**MAGIC: AN AUTOMATED GENERAL PURPOSE
SYSTEM FOR STRUCTURAL ANALYSIS**

VOLUME III: PROGRAMMER'S MANUAL

DANIEL DeSANTIS

**This document has been approved for public
release and sale; its distribution is unlimited.**

This Document Contains
Missing Page/s That Are
Unavailable In The
Original Document

OR are
Blank pgs.
that have
Been Removed

**BEST
AVAILABLE COPY**

FOREWORD

This report was prepared by Textron's Bell Aerosystems Company (BAC), Buffalo, New York, under USAF Contract No. AF33 (615)-67-C-1505. The contract was initiated under Project No. 1467, "Structural Analysis Methods," Task No. 146702, "Thermal Elastic Analysis Methods." The program was administered by the Air Force Dynamics Laboratory (AFFDL), Air Force Systems Command, Wright-Patterson Air Force Base, Ohio 45433 under the cognizance of Mr. G. E. Maddux, AFFDL Program Manager. The program was carried out by the Structural Systems Department, Bell Aerosystems Company, during the period 15 March 1967 to 15 March 1968 under the direction of Dr. Robert H. Mallett, Program Manager.

This report, "MAGIC: An Automated General Purpose System for Structural Analysis" is published in three volumes, "Volume I: Engineer's Manual", "Volume II: User's Manual", and "Volume III: Programmer's Manual." The manuscript for Volume III was released by the author in March 1968 for publication.

The author wishes to thank Miss Beverly J. Dale and Mr. John A. Bruno for their contribution to the development of the MAGIC System, and to acknowledge the assistance of the following personnel: D. Dupree, W. Lubracki, P. Braunschweig and R. Frauendorfer.

This technical report has been reviewed and is approved.



FRANCIS J. JANIK, SR.
Chief, Theoretical Mechanics Branch
Structures Division

ABSTRACT

An automated general purpose system for analysis is presented. This system, identified by the acronym "MAGIC" for "Matrix Analysis via Generative and Interpretive Computations," provides a flexible framework for implementation of the finite element analysis technology. Powerful capabilities for displacement, stress and stability analyses are included in the subject MAGIC System for structural analysis.

The matrix displacement method of analysis based upon finite element idealization is employed throughout. Six versatile finite elements are incorporated in the finite element library. These are: frame, shear panel, triangular cross-section ring, toroidal thin shell ring, quadrilateral thin shell and triangular thin shell elements. These finite element representations include matrices for stiffness, incremental stiffness, prestrain load, thermal load, distributed mechanical load and stress.

The MAGIC System for structural analysis is presented as an integral part of the overall design cycle. Considerations in this regard include, among other things, preprinted input forms, automated data generation, data confirmation features, restart options, automated output data reduction and readable output displays.

Documentation of the MAGIC System is presented in three parts; namely, Volume I: Engineer's Manual, Volume II: User's Manual and Volume III: Programmer's Manual. The subject Volume, Volume III, is designed to facilitate implementation, operation, modification, and extension of the MAGIC System.

TABLE OF CONTENTS

Section	Page
I INTRODUCTION	1
II COORDINATION OF STRUCTURAL GENERATIVE SYSTEM WITH FORMAT II	3
A. Detailed Analysis of .USER04. Instruction	3
1. Input and Output Matrix Position Functions	3
2. Suppression Option	4
B. Use of Format II Data Sets	5
1. Master Input and Master Output Use for Material Library	5
2. Instruction Input Data Sets	6
3. Instruction Output Data Sets	6
4. Scratch Data Sets	6
III ORGANIZATION OF STRUCTURAL GENERATIVE SYSTEM	8
A. Basic Logic Flow	8
B. Input Phase Logic Flow	8
1. Report Form Input	8
2. Interpreted Input	10
C. Element Matrices Generation Phase Logic Flow	10
D. Output Phase Logic Flow	11
1. Organization of Output Matrices	11
2. Sequence of Output of Matrices	15
IV USAGE OF STRUCTURAL GENERATIVE SYSTEM	17
A. Implementation	17
1. Direct Machine Control	17
2. SUBSYS Control	17
B. Examples	19
APPENDIX I Structural System Overlay Chart	21
II List of Subroutine Functions	31
III Table of Error Messages	45
IV Example Stress and Stability Instruction Sequences	57
V Subroutine Documentation	60
VI SUBSYS Documentation	326
VII Details on LNKSTK	334
VIII LNKSTK as an Executable Subsystem	340
IX Description of the Modified .LØVRY with CPYLKO	342
X Description of the Search Routine	344
XI Deck Numbering Sequence	346
XII Revisions to FORMAT System Decks	356
XIII Logical Flow Charts	363

SECTION I

INTRODUCTION

A Structural Generative System has been developed and inserted into FORMAT II for the purpose of generating structural matrices for use by FORMAT II. The insertion of a Structural Generator into FORMAT II resulted in a computer program retaining ease of implementation and use, yet offering diversified capabilities.

Machine compatibility has been retained by the complete use of FORTRAN IV in the development of the Structural Generative System. The absence of machine or assembler language from every portion of the program eliminates the problems of machine dependency and implementation difficulty.

Input to the Structural Generative System is accomplished by filling in preprinted structural engineering oriented input sheets. The combination of these sheets and the normal matrix abstraction instructions of FORMAT II allows minimal training for use of the program, thus decreasing the possibility of input errors.

The program is capable of restart at any point in the abstraction instruction sequence stipulated at the discretion of the User. Input data, intermediate results, final results or any matrix whatsoever may be automatically saved, by use of the proper instruction, and used as a starting point or new input to subsequent applications on continuing or independent projects.

The Structural Generative System consists of a maximum of 175 subroutines logically designed into 60 overlay links on an IBM 7090 with 32K words of storage. The overlay design reflects the optimum use of available storage, yet maintains respectable execution efficiency. The Structural Generative System requires a minimum of 13000₁₀ words of work storage area assigned to an unlabeled common block. A minimum of eight external storage units available to the FORMAT II System are required for use of the Structural Generative System, including at least one assigned to the Master Input FORMAT function, one assigned to the Master Output FORMAT function and four assigned to the Utility FORMAT function.

The number of subroutines contained in the FORMAT II program has necessitated the use of SUBSYS, a software package developed by Westinghouse, which improves the loading capabilities of IBSYS on the IBM 7090/94. In addition to allowing the program to be loaded, SUBSYS allows the program overlay tape to be saved, thereby improving execution time. Programs may be stacked on this overlay tape. Taking advantage of this fact, FORMAT II with the Structural Generative System insertion, is actually three programs executed automatically with no intervention by IBSYS. The first program consists of the FORMAT II Preprocessor, the second consists of the FORMAT II Execution Monitor and the third contains the Structural Generative System. Although the Structural Generative System is actually a separate program when operating under SUBSYS control on the IBM 7090/94, it is activated and controlled as a normal User Module under the FORMAT II System. Explicitly, the Structural Generative System is the fourth User Module (USER04) available under FORMAT II.

SECTION II

COORDINATION OF STRUCTURAL GENERATIVE SYSTEM WITH FORMAT II

A. DETAILED ANALYSIS OF USER04 INSTRUCTION

1. Input and Output Matrix Position Functions

The Structural Generative System may have as many as twelve actual output matrices and require as many as four actual input matrices. The basic form of the USER04 instruction may be represented as follows:

OMP1, OMP2, OMP3, OMP4, OMP5, OMP6, OMP7, OMP8, OMP9, OMP10, OMP11, OMP12 = IMP1, IMP2, IMP3, IMP4, .USER04.; where OMP is read as output matrix position and IMP as input matrix position. All matrix positions, whether input or output, must be present. They may contain matrix names or be blank, but there must be sixteen matrix positions represented by the appropriate number of commas. Blank matrix positions are discussed in the next section. The output matrix positions, if nonblank, will contain the following matrices upon exit from the Structural Generative System:

OMP1	-	copy of input structure data deck
OMP2	-	revised material library
OMP3	-	interpreted input (structure input data as stored after being read and interpreted)
OMP4	-	external system grid point loads matrix
OMP5	-	transformation matrix for application of boundary conditions
OMP6	-	transformation matrix for assembly of element matrices
OMP7	-	element stiffness matrices stored as one matrix
OMP8	-	element generated load matrices stored as one matrix
OMP9	-	element stress matrices stored as one matrix
OMP10	-	element thermal stress matrices stored as one matrix
OMP11	-	element incremental stiffness matrices stored as one matrix
OMP12	-	element mass matrices stored as one matrix

The input matrix positions, if non-blank must contain the following matrices:

- IMP1 - structure data deck (this would be a previously generated matrix saved in OMP1)
- IMP2 - interpreted input (this would be a previously generated matrix saved in OMP3)
- IMP3 - existing material library (this would be a previously generated matrix saved in OMP2)
- IMP4 - input displacement matrix to be used for stability analyses

It should be noted that the following matrix positions are called matrices only in the sense that all input and output entities are considered matrices by FORMAT II - OMP1, OMP2, OMP3, IMP1, IMP2 and IMP3.

2. Suppression Option

Incorporated into the Structural Generative System is an option to suppress the generation and output of any of the output matrices and also to indicate the absence of any of the input matrices. This option is indicated to the Structural Generative System by the absence of a matrix name in the desired position in the USER04 instruction. A matrix name is considered to be absent if the matrix position contains all blanks or the character length of the name is zero. For example, an instruction of the form:,,INTINP, LOADS, TR, TA, KEL, FEL, SEL, SZALFL,, = ,,MATLB1, .USER04.; would cause suppression of the copy of the data deck, the revised material library, the element incremental stiffness matrices and the element mass matrices. The instruction also indicates that there is no input data deck on tape, (directing the Structural Generative System to read data from cards), no interpreted data on tape and no input displacements. It should be noted that certain sections of the data deck are necessary for the generation of each of the output matrices and that error checking is done to determine if the required sections are present. A table of the required data sections for generation of each matrix appears in the User's Manual. Accordingly, error checking is invoked for the input matrix positions to determine if ambiguous or conflicting input indications have been made.

Internally, the logic flow of the suppression option is controlled by inserting key characters for suppressed matrices. Upon detection of a suppressed matrix by Subroutine INST (deck FO30), a matrix name of the form ///XX is inserted into that matrix position. The four slashes are inserted for recognition by the Structural Generative System of a suppressed matrix and the last two positions may each contain the digits 0-9 assigned sequentially starting from 00 for each suppressed matrix encountered. The last two positions in the inserted name for suppressed matrices ensure that each suppressed matrix name will be unique, thereby eliminating inconsistencies in the FORMAT II Preprocessor.

Suppressed input matrices, i.e. those occurring to the right of the equal sign in the input .USERO4. abstraction instruction, are recorded on NDATA, the data set reserved for card input matrices, as null matrices to satisfy FORMAT II Preprocessor input matrix existence requirements. This operation is accomplished by subroutine MATSUP (deck FO3A).

B. USE OF FORMAT II DATA SETS

1. Master Input and Master Output Use for Material Library

References to the Material Library are indicated by output matrix position two and input matrix position three in the .USERO4. abstraction instruction. Retention of a newly generated or revised Material Library is governed solely by use of the SAVE abstraction instruction at the discretion of the User. If retention is desired, the matrix name in output matrix position two must appear in a SAVE abstraction instruction, in which case it will be placed on a Master Output tape. If a non-blank matrix name appears in input matrix position three, the Master Input tape will be searched for that name.

Usage and generation of the Material Library is controlled by the three legal combinations of suppression of output matrix position two and input matrix position three. If the matrix name in output matrix position two is non-blank, but input matrix position three is suppressed, a new Material Library will be generated and used. If both involved matrix positions are non-blank, the old Material Library will be located on the Master Input tape, will be revised, stored as the matrix named in the specified output position, and then this revised Material Library will be used. If output matrix position two is suppressed and input matrix position three is non-blank, then the named input Material Library will be used. Suppression of both involved matrix positions results in an error condition.

Since the material library is stored under a matrix name on Master Output tapes, and also, therefore Master Input tapes, any other matrices may also be saved on the same tape, including other Material Libraries.

2. Instruction Input Data Sets

An instruction input data set is an external storage unit that contains at least one of the non-blank matrices named in input matrix positions one, two, three or four in the USER04 abstraction instruction. The Structural Generative System conforms to all the rules of FORMAT II with regard to use of instruction input data sets. All searching, reading, and re-winding is accomplished by use of the FORMAT II data set handling subroutines EUTL1-EUTL9 (decks F101-F109). No attempt is ever made to write on an instruction input data set.

3. Instruction Output Data Sets

An instruction output data set is an external storage unit which has been designated by the FORMAT II Preprocessor to contain at least one of the non-blank matrices in output matrix positions one to twelve in the USER04 abstraction instruction. The Structural Generative System conforms to all rules of FORMAT II with regard to instruction output data sets by using the FORMAT II data set handling subroutines EUTL1-EUTL9 (decks F101-F109) to write all matrix headers, matrix trailers, data set trailers and end of files on instruction output data sets. All matrices are stored by column in the required record format. No attempt is ever made by the Structural Generative System to rewind an instruction output data set.

4. Scratch Data Sets

Scratch data sets are external storage units that have been assigned by the FORMAT II System to the Structural Generative System to be used as temporary storage areas. There are no reading, writing or rewinding rules imposed on scratch data sets by the FORMAT II System. The required four scratch data sets are assigned to the following functions by the Structural Generative System:

- SCRATCH DATA SET 1 - 1st use - external storage areas for report form input preprocessor
- 2nd use - contain structure control information including system orders, boundary conditions and system print operations

- SCRATCH DATA SET 2 - 1st use - contain temporary
copy of direct input structure
data deck
- 2nd use - contain generated
element matrices in compact
form
- SCRATCH DATA SET 3 - 1st use - contain temporary
copy of actual input deck
- 2nd use - contain element input
data after reading and interpre-
tation
- SCRATCH DATA SET 4 - 1st use - external storage area
for report form input
preprocessor
- 2nd use - contain input loads
matrix
- 3rd use - contain input dis-
placements, if any

SECTION III

ORGANIZATION OF STRUCTURAL GENERATIVE SYSTEM

A. BASIC LOGIC FLOW

The Structural Generative System has three basic phases of operational flow; the input phase, the element matrices generation phase, and the output phase. The input phase consists of reading, interpreting and storing the information contained in the structure data deck. From the stored input, the element matrices selected are generated in the second phase. Phase three outputs all non-suppressed matrices in output matrix position six through twelve indicated by the USER04. abstraction instruction. Output matrix positions one through five are generated directly from the input structure data deck and for this reason are actually output during the first or input phase. Subroutine US04 (deck S000) controls the three logical phases by directly controlling subroutine US04A (deck S100) which controls the input phase and US04B (deck S200) which controls the generation and output phases. Normally, the basic logical flow of the Structural Generative System would be sequentially through the three phases, however, by use of the suppression option, it is possible to completely skip a given phase. The actual logic flow of the system is created by subroutine LOGFLO (deck S010) as determined by the .USER04. abstraction instruction. For example, if the .USER04. instruction was written such that only the boundary conditions had changed and the remainder of the necessary matrices were saved from a previous application as indicated by the suppression option, subroutine LOGFLO would eliminate the second and third phases.

B. INPUT PHASE LOGIC FLOW

The logic flow of the input phase is determined by the type of input encountered. The two types of input are report form input and interpreted input.

1. Report Form Input

The location of the input data deck is determined by examining IMPl of the input .USER04. abstraction instruction. If this input position was blank, then the data deck is assumed to be on NPIT, the system input unit. If IMPl contained a non-blank matrix name, then the input data deck exists as a matrix and the original card form deck is reconstructed by subroutine INDECK (deck S102).

Report Form Input is a highly flexible, engineering oriented type of input for the Structural Generative System. From a programming viewpoint, report form input allows ease of use by the Analyst and by translation allows logical readability by the program.

Encountering a report form input deck causes the input phase to pass control to the Report Form Input Preprocessor. Basically, the report form input preprocessor translates the flexible report form input deck into a sophisticated direct input deck. Translation is accomplished by two steps controlled by subroutine REFORM (deck S126).

The first step is to read and store the report form input deck. This step is accomplished by subroutine PHASE1 (deck S128) with support by subroutines LATCH and FORMIN (decks S130 and S132). PHASE1 controls all storage, both internal core storage and external storage on scratch data sets one and four. LATCH performs label matching tests to determine the various sections of input and FORMIN reads all table form input, sections; non-table form input sections are read directly in PHASE1.

The second step in processing a report form input deck is to merge the data stored by the first step into a direct data deck. These two operations are performed by subroutine PHASE2 (deck S134) supported by subroutine OPEN (deck S136). The information stored by the first step is merged into a compact direct data deck by PHASE2 and output on scratch data set two. The OPEN subroutine aids PHASE2 by locating, in any order designated by PHASE2 the input sections stored on scratch data sets one and/or four. At this point, a complete direct data input deck is resident on scratch data set two and control returns to US04A. Once a direct data deck is resident on scratch data set two, reading, interpreting and storage is controlled by subroutine INPUT (deck S108) with each input section handled as indicated by the following table:

INPUT SECTION	SUBROUTINE	INTERPRETED STORAGE
Title	INPUT (deck S108)	None
System Control	INPUT	Scratch data set 1
Grid Points	INPUT	Scratch data set 3
Boundary Conditions	BOUND (deck S112)	Scratch data set 3
Element Definitions	ELEM (deck S114)	Scratch data set 3
Grid Point Loads	EGRIDS (deck S120)	Scratch data set 4
Grid Point Axes	FRED (deck S110)	Scratch data set 3
Material Library Requests	FMAT (deck S122)	Master Output data set
Grid Point Temperatures	INPUT	Scratch data set 3
Grid Point Pressures	INPUT	Scratch data set 3
Initial Displacements	BOUND	Scratch data set 3

If output matrix position one was non-blank, then a copy of the actual input data deck is also written on the instruction output data set specified by the FORMAT II System by subroutine COPYDK (deck S106).

2. Interpreted Input

After the data deck has been read and interpreted under control of subroutine INPUT, all pertinent data exists on scratch data sets one and three. If output matrix position three in the USER04 abstraction instruction is non-blank, then the contents of scratch data sets one and three are output under that matrix name onto the instruction output data set specified by the FORMAT II System by subroutine OUTINT (deck S140). If this "matrix" is saved and input at input matrix position two in the USER04 instruction, the Structural Generative System is capable of restart at the second or element generation phase, thereby eliminating a repeat of the input phase. This feature is recommended for usage on large applications where the procedure would be to run the data deck, stop after interpreting and storing the data, check for input errors, and if no errors are present restart at the element generation phase.

Before exiting from the input phase, subroutine CHEK (deck S138) is called to perform input error cross-checking. While determining the logical flow at the Structural Generative System, subroutine LOGFLO (deck S010) also recorded the input sections required to generate the requested output matrices. If any of the required input sections have not been processed, then execution will be terminated after the input phase.

C. ELEMENT MATRICES GENERATION PHASE LOGIC FLOW

The second phase of operation of the Structural Generative System consists of generation of the element matrices.

If input matrix position two of the input .USER04. abstraction instruction is non-blank, then subroutine ININT (deck S202) is called to reconstruct the data on scratch data sets one and three from the input matrix.

If input matrix position four of the input .USER04. abstraction instruction is non-blank, then subroutine DEFLEX (deck S204) is called to store the input displacements on scratch data set four.

At this point all the necessary data is located on scratch data sets one and three, placed there by either phase one or restart using input matrix position two of the USER04 abstraction instruction. Basic control of the second phase is accomplished by subroutine FELEM (deck S206) under subroutine USO4B. FELEM reads scratch data set one to obtain system control information and sets suppression controls to eliminate generation of undesired element matrices by calling subroutine SQUISH (deck S208). Scratch data set three contains the necessary input for each element, one set of element input per record. For each element, subroutine ELPLUG (deck S210) reads an element input record, selects the proper element to calculate the matrices and then writes the generated matrices on scratch data set two in compact form.

Prior to being written upon scratch data set two, the element matrices are temporarily stored in the blank common work area. Also, all work areas that are needed by the specific element are allocated from the blank common work area. For these reasons, the Structural Generative System requires a blank common work area of at least 13,000 words of internal core storage.

Imbedded into the Element Matrices Generation Phase, at strategic locations, are utility packages accessible by the specific elements which require their capabilities. Integration packages and small scale matrix operation packages are examples of utility sections commonly accessible to the necessary elements. The exact locations of these packages are indicated by the Structural System Overlay Chart (Appendix I). Overlay to each element has been avoided wherever possible to reduce execution process time. However, an area of approximately 1000 locations between the longest link and the origin of the common area has been kept clear to allow for future substantial alterations to be made without redesigning the complete overlay structure.

D. OUTPUT PHASE LOGIC FLOW

1. Organization of Output Matrices

All output entities from the Structural Generative System are written following the rules of the FORMAT II System. Each output entity is written as a matrix, consisting of a matrix header, matrix column records and a matrix trailer. The following list exhibits the contents, interpretation of matrix header information (number of rows, number of columns) and interpretation of matrix column records for each output position in the USER04 abstraction instruction.

a. Output Matrix Position One (OMP1)

Contents	- Copy of card input data deck
Number of rows	- Set to eighty (80)
Number of columns	- Number of cards in data deck
Column records	- One data card per column record, one card column per row

b. Output Matrix Position Two (OMP2)

Contents	- Material library
Number of rows	- 306 (maximum number of words possible for one material entry)
Number of columns	- Number of material tables in library plus one
Column records	- One material table per column record

c. Output Matrix Position Three (OMP3)

Contents	- Interpreted input
Number of rows	- Set to number of words in maximum record created
Number of columns	- Number of elements plus
Column records	- One element input block per record

d. Output Matrix Position Four (OMP4)

Contents	- External system grid point loads
Number of rows	- Number of degrees of freedom in total system
Number of columns	- Number of load conditions
Column records	- One load condition per column record

e. Output Matrix Position Five (OMP5)

Contents	- Transformation matrix for application of boundary conditions
Number of rows	- Number of degrees of freedom in reduced system
Number of columns	- Number of degrees of freedom in total system

- Column records - (1) for desired degrees of freedom - contain a one in the assigned reduced degree of freedom row
- (2) for undesired degrees of freedom - column record is omitted (null column)

f. Output Matrix Position Six (OMP6)

- Contents - Transformation matrix for assembly of element matrices
- Number of rows - Number of degrees of freedom in total system
- Number of columns - Summation of element degrees of freedom
- Column records - Contain a one in the assigned degree of freedom row for that summed element degree of freedom

g. Output Matrix Position Seven (OMP7)

- Contents - Element stiffness matrices
- Number of rows - Summation of element degrees of freedom
- Number of columns - Summation of element degrees of freedom
- Column records - Each record contains a column of an element stiffness matrix

h. Output Matrix Position Eight (OMP8)

- Contents - Element applied load matrices
- Number of rows - Summation of element degrees of freedom
- Number of columns - One
- Column record - Contains all element applied load matrices

i. Output Matrix Position Nine (OMP9)

- Contents - Element stress matrices
- Number of rows - Summation of element stress point and component orders
- Number of columns - Summation of element degrees of freedom
- Column records - Each record contains a column of an element stress matrix

j. Output Matrix Position Ten (OMP10)

Contents	- Element thermal stress matrices
Number of rows	- Summation of element stress point and component orders
Number of columns	- One
Column record	- Contains all element thermal stress matrices

k. Output Matrix Position Eleven (OMP11)

Contents	- Element incremental stiffness matrix
Number of rows	- Summation of element degrees of freedom
Number of columns	- Summation of element degrees of freedom
Column records	- Each record contains a column of an element incremental stiffness matrix

l. Output Matrix Position Twelve (OMP12)

Contents	- Element mass matrices
Number of rows	- Summation of element degrees of freedom
Number of columns	- Summation of element degrees of freedom
Column records	- Each record contains a column of an element mass matrix

It should be noted that OMP1, OMP2 and OMP3 are not actually matrices and, therefore, should never be referenced as input to an algebraic matrix operation. OMP7, OMP9, OMP11 and OMP12 are formed by placing the element matrices into the output matrix such that the main diagonal of the element matrix coincides with the next available main diagonal positions in the output matrix. For example, if the first two element stiffness matrices represented 48 element degrees of freedom each (such as 8 element defining points with 6 degrees of freedom each) then the first would be located in rows one to 48 and column one to 48 in the output matrix and the second would be placed into rows 49 to 96 and columns 49 to 96. Output matrices in these positions are almost always written in FORMAT II compressed column format due to the inherent sparseness of non-zero matrix elements.

OMP8 and OMP10 are formed by placing each element matrix, which is a column matrix, into the succeeding available row positions in the output matrix.

2. Sequence of Output of Matrices

Output matrix positions one to five are output sequentially in numerical order by the Structural Generative System. Since these first five matrices are generated directly from data contained in the input deck, they are output, if non-blank, as part of phase one or input phase operations. Specifically, the first five output matrices are placed into the FORMAT II system by the following subroutines in phase one:

OMP1	- Subroutine COPYDK	(deck S106),
OMP2	- Subroutine FMAT	(deck S122),
OMP3	- Subroutine OUTINT	(deck S140),
OMP4	- Subroutine FLOADS	(deck S142),
OMP5	- Subroutine FTR	(deck S144),

Output matrix positions six through twelve are released into the FORMAT II System during phase three of the Structural Generative System. Output of matrices is controlled by subroutine OUTMAT (deck S268) using utility subroutines US461, US462 and US463 (decks S270, S272 and S274). In contrast to output of the first five matrices, which is achieved consecutively, output of matrices six through twelve will usually occur concurrently.

Operational flow in the output phase consists of extracting the compacted element matrices from scratch data set two and releasing them to the FORMAT II System in the required form. Due to the fact that more than one output matrix may have been assigned to the same instruction output data set by the FORMAT II System, direct output at matrix generation time (phase two) is impossible, thus necessitating the use of scratch data set two. However, at output time, the optimum procedure is determined by subroutine OUTMAT to achieve multiple matrix output per pass of scratch data set two. The procedure involves determining which matrices may be output during the same pass of scratch data set two by one, comparing the assigned instruction output data set number and two, type of matrix being output. Output matrix positions eight and ten, if non-blank, are always output on the first pass. Output matrix positions six, seven, nine, eleven, and twelve may require from one to five passes of scratch data set two, recognizing the best and worst possible cases. In general, OUTMAT may only output one matrix per pass on a given instruction output data set with the exception of output matrix positions eight and ten which are always

output on the first pass regardless of their instruction output data set numbers.

For example, given the following instruction output data set assignments by the FORMAT II System (all output matrix positions referenced are non-blank):

Output Matrix Position	Format Assigned Instruction Output Data Set
6	4
7	8
8	3
9	3
10	8
11	4
12	3

OUTMAT would release all the requested matrices (6-12) to the FORMAT II System in two passes of scratch data set two as indicated below.

PASS 1 - 6, 7, 8, 9, 10
PASS 2 - 11, 12

Output Matrix Positions 6, 7, and 9 may be output concurrently on pass one since they are to be located on different data sets. Positions eight and ten will always be output on pass one. Since positions 11 and 12 are to be located on different data sets, they may be output on the same pass.

If a matrix is less than 50% dense, the compressed column record format is invoked.

SECTION IV

USAGE OF STRUCTURAL GENERATIVE SYSTEM

A. IMPLEMENTATION

1. Direct Machine Control

Under direct machine control the only changes required for implementation on any system are contained in one deck, subroutine MRES (deck F020). The implementation operations involved are explained in detail in Volume II (Description of Digital Computer Program), Section IV, Subsection 3 of "FORMAT II - Second Version of Fortrar Matrix Abstraction Technique" (AFFDL-TR-66-207). In general, the information which must be supplied consists of defining system parameters; such as system input unit, system output unit, size of blank common work area, and limiting size of matrix capability; and assigning FORMAT II System functions to the available external storage units.

Under direct machine control the Structural Generative System has been inserted as a normal user module with the same origin and accessibility as any other user module.

Operation of the Structural Generative System requires the common area to be at least 13000₁₀ storages and the number of external storage units to be at least eight. Both of these facts must be inserted into MRES at implementation time.

2. SUBSYS Control

Implementation upon an IBM 7090/94 requires an improvement of the loading capabilities of IBSYS. The software package selected is SUBSYS, developed by Westinghouse Corporation. A software package was selected in deference to multiple passes at IBJOB due to the inflexibilities of IBLDR under IBJOB. For example, IBLDR requires the use of at least three tape drives to load each portion, thereby removing units from use by FORMAT II. Also, data would be inserted in the middle of program deck and printed output would be interspersed with IBJOB Processor Output. The most decisive advantage, however, was the saving of load time under SUBSYS. Normal load time under IBLDR for the complete program is approximately eight minutes on a 7090, whereas under SUBSYS control the program is placed into core and execution started with a load time of fifteen to twenty seconds.

The SUBSYS package consists of four subroutines written in MAP. The first subroutine, .LOVRY, is placed in the program deck, thus replacing the normal .LOVRY that IBSYS would have provided. The function of this altered .LOVRY is to receive control after the program has been loaded and to then copy the main link (LINK 0), which is now resident in core storage, onto a specified tape unit. Entry is then made into LNKSTK, the second SUBSYS subroutine, which will perform the function of copying LINK 0 from the tape written by .LOVRY onto another tape. Also, LNKSTK will place the overlay load file generated in the IBLDR phase and place it on the same tape as LINK 0. Upon completion of a LNKSTK execution, the entire program will be on tape in absolute load mode in two files; the first containing LINK 0 and the second containing the overlay structure. At this point the program may now be edited onto the System Library with the aid of the third SUBSYS subroutine, COPYDK, in which case it may be invoked by a \$EXECUTE XXXXXX card, or the tape may be saved in its two file per program form accessible by the fourth SUBSYS subroutine, SEARCH. SEARCH has the capability of locating any program on a SUBSYS generated program tape, loading that program's LINK 0 into core and then transferring control to it.

Usage of a SUBSYS generated program tape is accomplished by writing a FORTRAN load program that need contain only one executable statement, CALL SEARCH (6HPROGNM). This will cause SEARCH to locate the program, read the main link into core and execute the main deck. The overlay is contained in the next file and the modified .LOVRY, now resident in core with the main link, will control the loading of the overlay links. The modified .LOVRY will also substitute backspace file commands in place of rewind selections on the \$ORIGIN cards in order to keep inside of the overlay file on the SUBSYS generated program tape. Complete SUBSYS documentation is located in Appendices VI through X.

A SUBSYS generated program tape may contain more than one program, each being identified and located by the name that was assigned to it by the User during the LNKSTK phase. Execution of each program is initiated by a call to SEARCH supplying the program name.

FORMAT II, with the Structural Generative System insertion, is contained on one SUBSYS generated program tape as three separate programs, named AFMTII, BFMTII and USER04, which are, respectively, the FORMAT II Preprocessor, the FORMAT II Execution Monitor and the Structural Generative System. Sequence of usage of the three programs is indicated on the following two lists, the first reflecting an application in which the USER04 module (Structural Generative System) is accessed and the second reflecting an application in which the USER04 module is not accessed.

B. EXAMPLES

1. USER04 Module Accessed:

The FORTRAN load program will cause the loading of -

- (a) AFMTII, which upon completion of processing the input will issue a call to SEARCH to load -
- (b) BFMTII, which upon encountering the USER04 instruction will issue a call to SEARCH, to load -
- (c) USER04, which upon completion of matrix generation will issue a call to SEARCH to load -
- (d) BFMTII, which upon completion of execution of the input abstraction instructions will call SEARCH to load -
- (e) AFMTII, which will begin processing the next input data deck, if any.

2. USER04 Module Not Accessed:

The FORTRAN load program will cause the loading of -

- (a) AFMTII, which upon completion of processing the input will issue a call to SEARCH to load -
- (b) BFMTII, which upon completion of execution of the abstraction instructions will call SEARCH to load -
- (c) AFMTII, which will begin processing the next input data deck, if any

USER04 and non-USER04 data decks may be batched together on a single loading of the program.

Due to the fact that FORMAT II with the Structural Generative System is actually three separate programs, the necessary changes required for implementation on a given system must be made in each program. The same information must be supplied to subroutine MRES (deck F020) in AFMTII that was needed for direct machine control. Main programs BFMTII and USER04 each have a subroutine RESET (decks SS20 and SS30) which must re-establish the size of blank common.

The sequence of operations to generate a SUBSYS program tape would be as follows:

1. IBSYS - start job
2. IBJOB - load AFMTII
3. LNKSTK - place AFMTII on SUBSYS program tape
4. IBJOB - load BFMTII
5. LNKSTK - place BFMTII after AFMTII on SUBSYS program tape
6. IBJOB - load USER04
7. LNKSTK - place USER04 after AFMTII and BFMTII on SUBSYS program tape

It is extremely helpful, but not necessary, to the above procedure that LNKSTK be placed into IBSYS as a subsystem prior to executing the above procedure. Further examples are given in Appendix VI, SUBSYS Documentation.

APPENDIX I

STRUCTURAL SYSTEM OVERLAY CHART

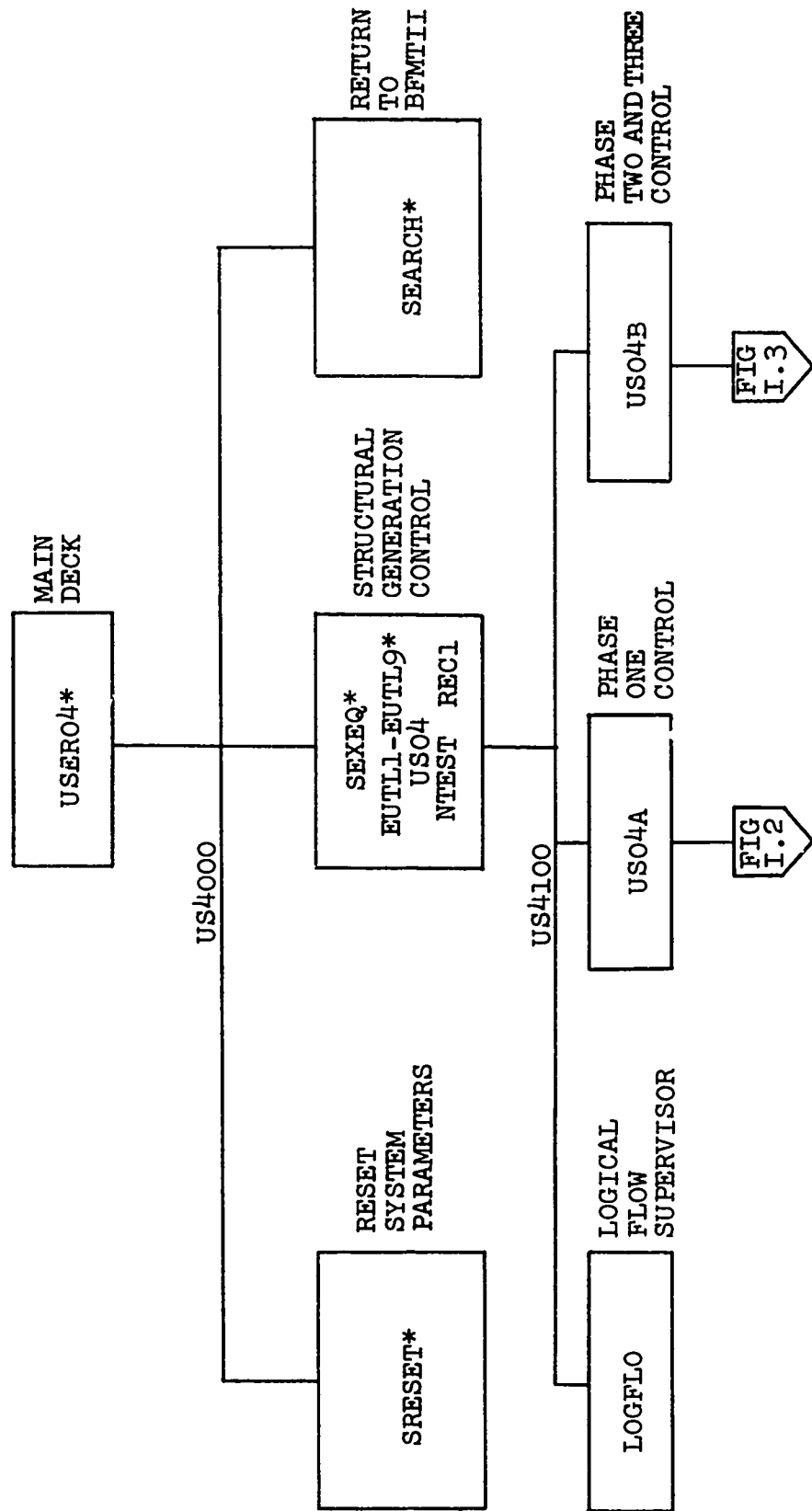


FIGURE I.1 CONTROL SECTION

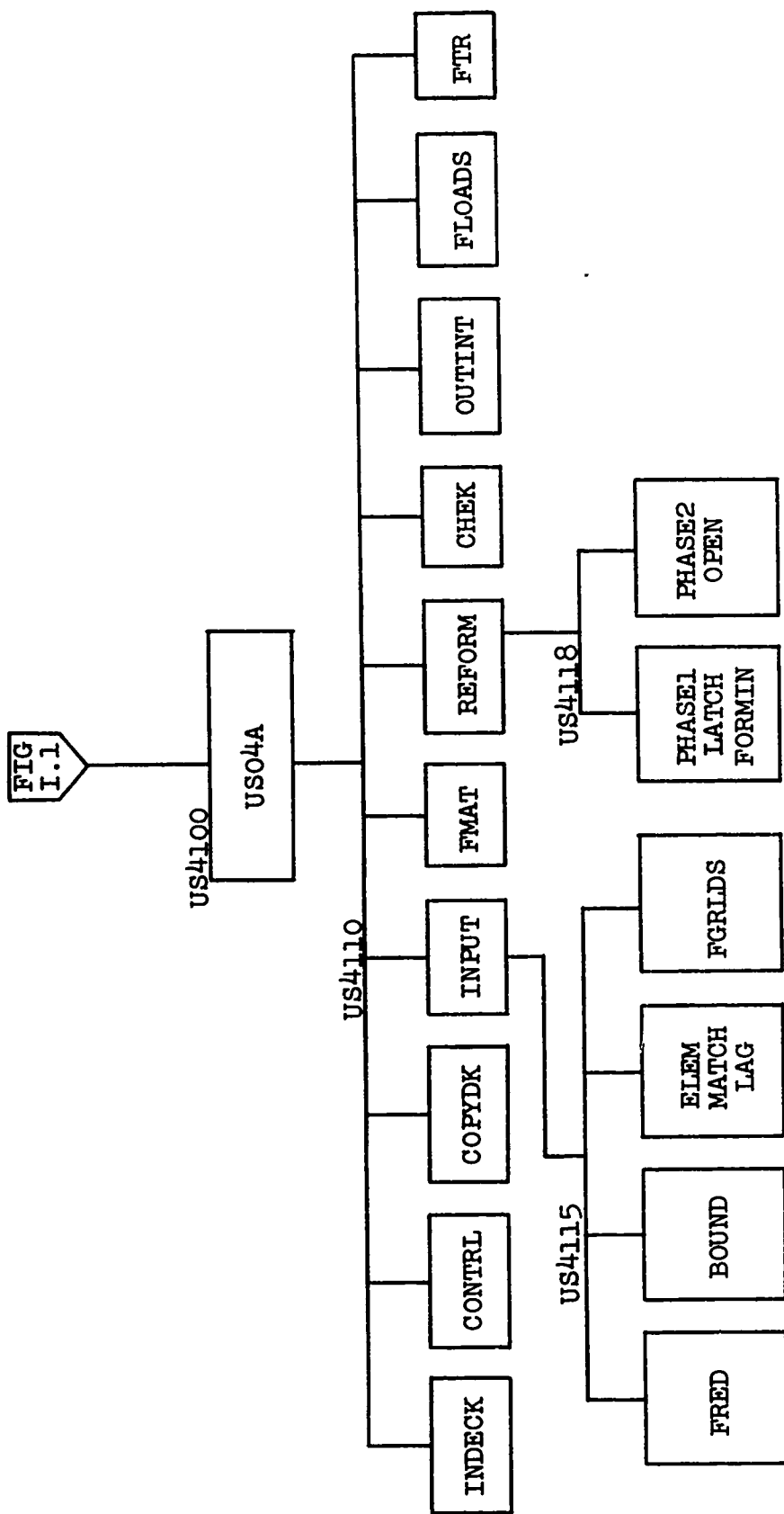


FIGURE I.2 PHASE ONE SECTION

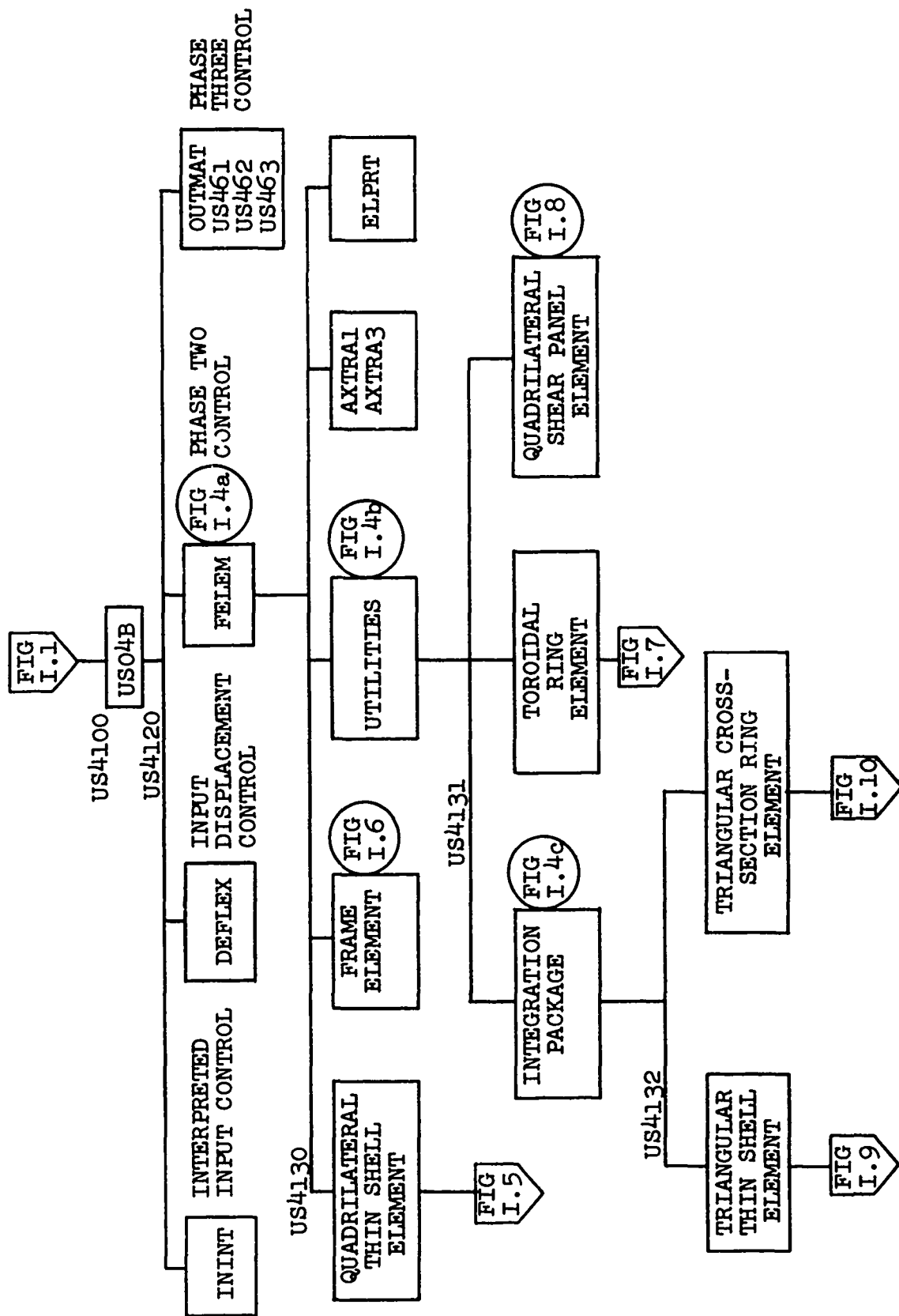


FIGURE I.3 PHASE TWO AND PHASE THREE SECTION

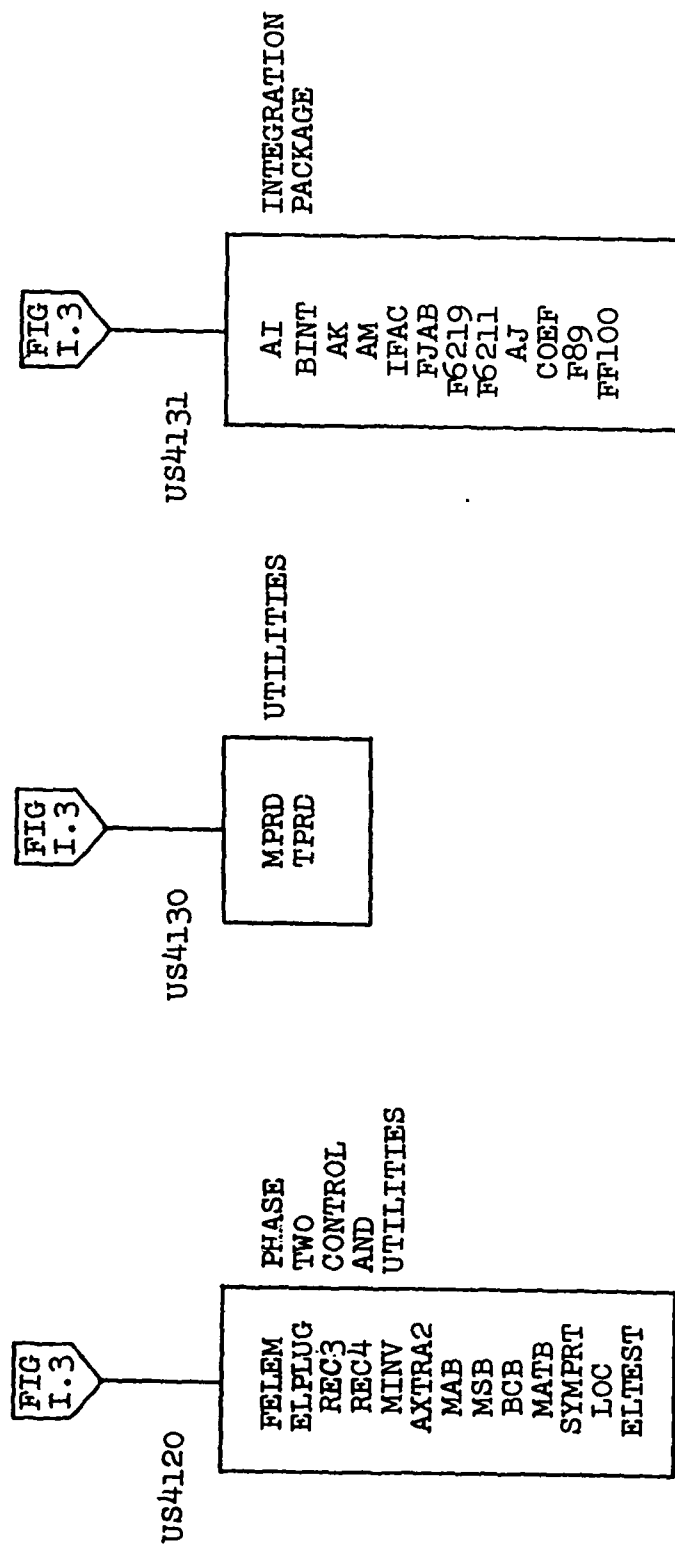


FIG. I.4a

FIG. I.4b

FIG. I.4c

FIGURE I.4 PHASE TWO CONTROL AND UTILITY PACKAGES

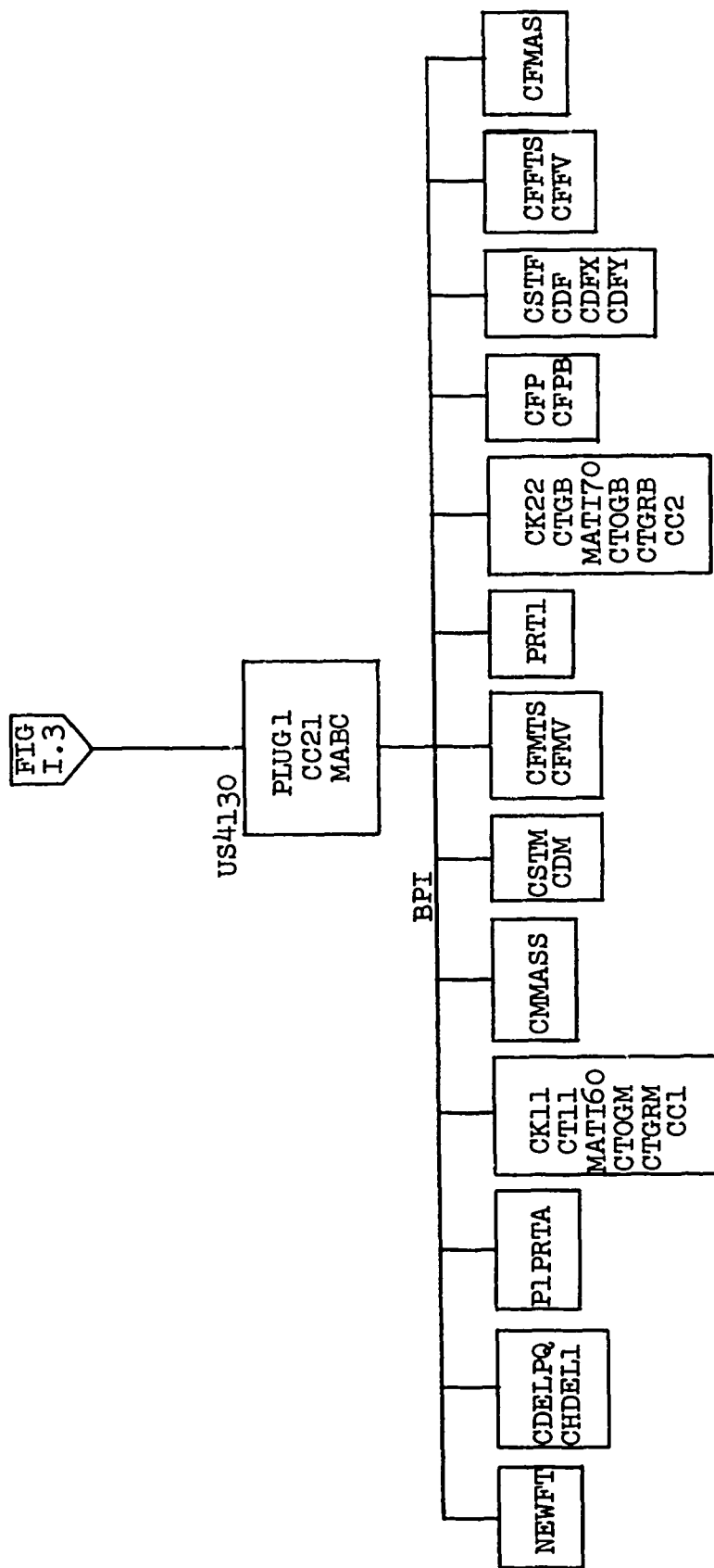


FIGURE I.5 QUADRILATERAL THIN SHELL ELEMENT

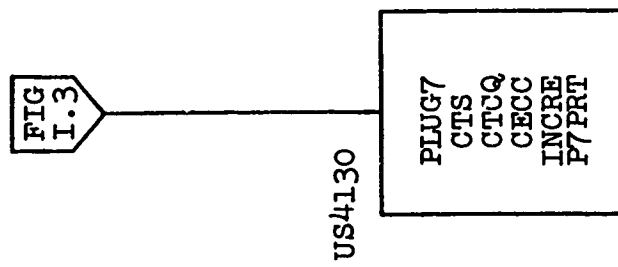


FIGURE I.6 FRAME ELEMENT

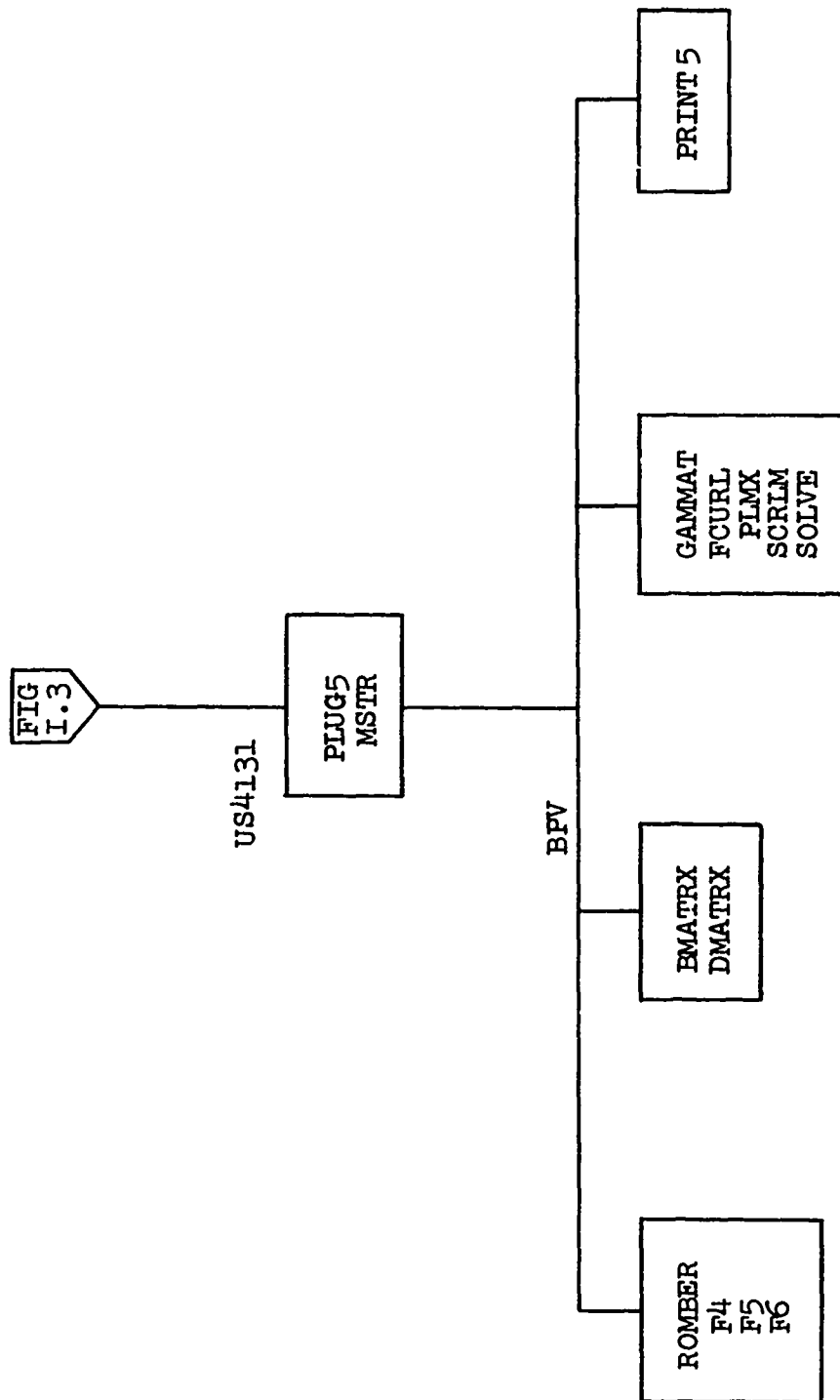


FIGURE I.7 TOROIDAL RING ELEMENT

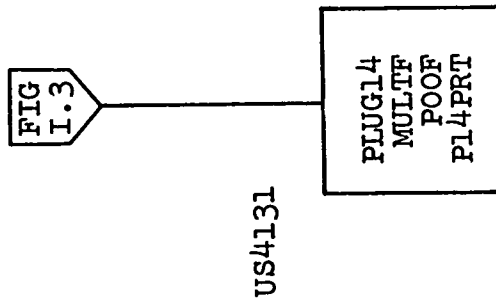


FIGURE I.8 QUADRILATER SHEAR PANEL ELEMENT

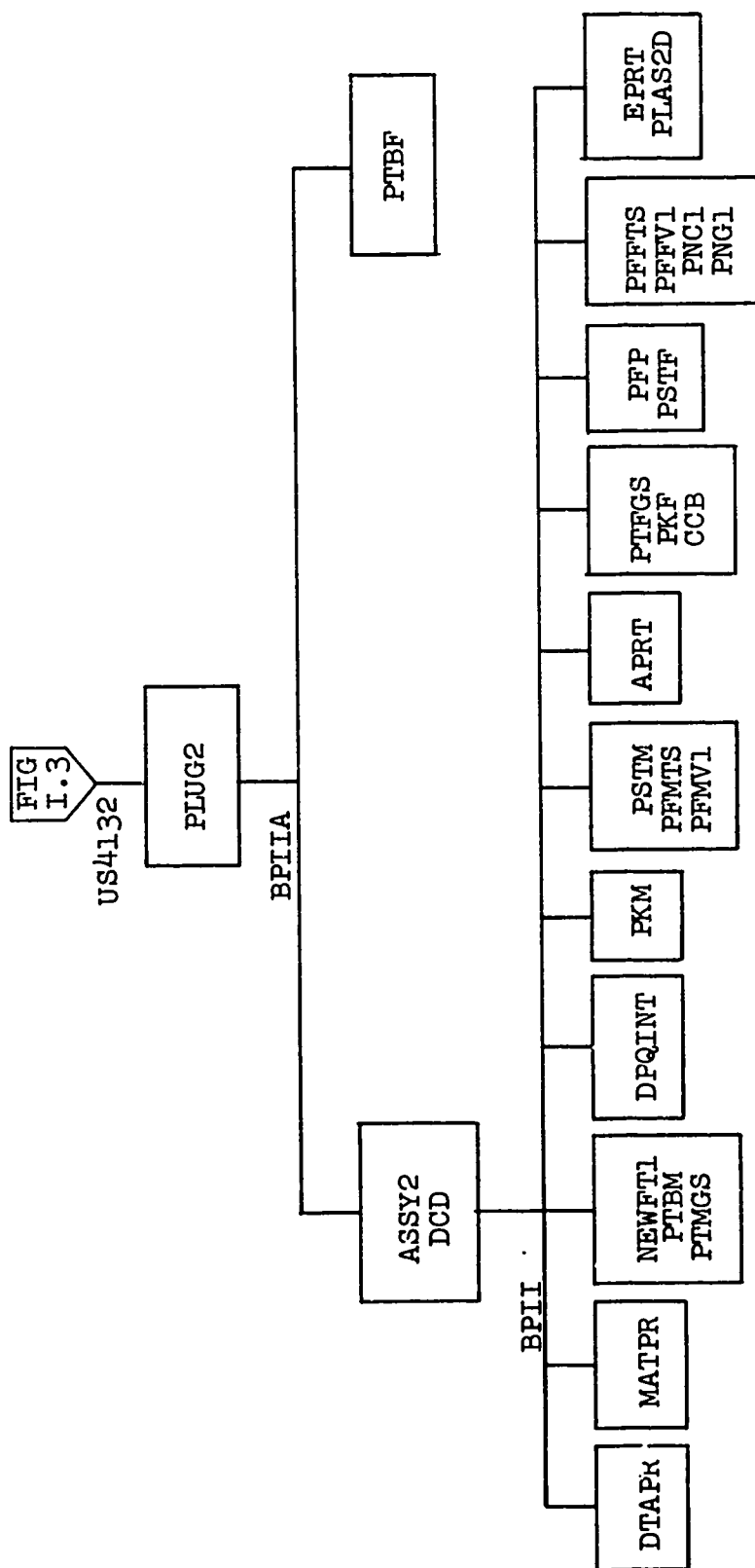


FIGURE I.9 TRIANGULAR THIN SHELL ELEMENT

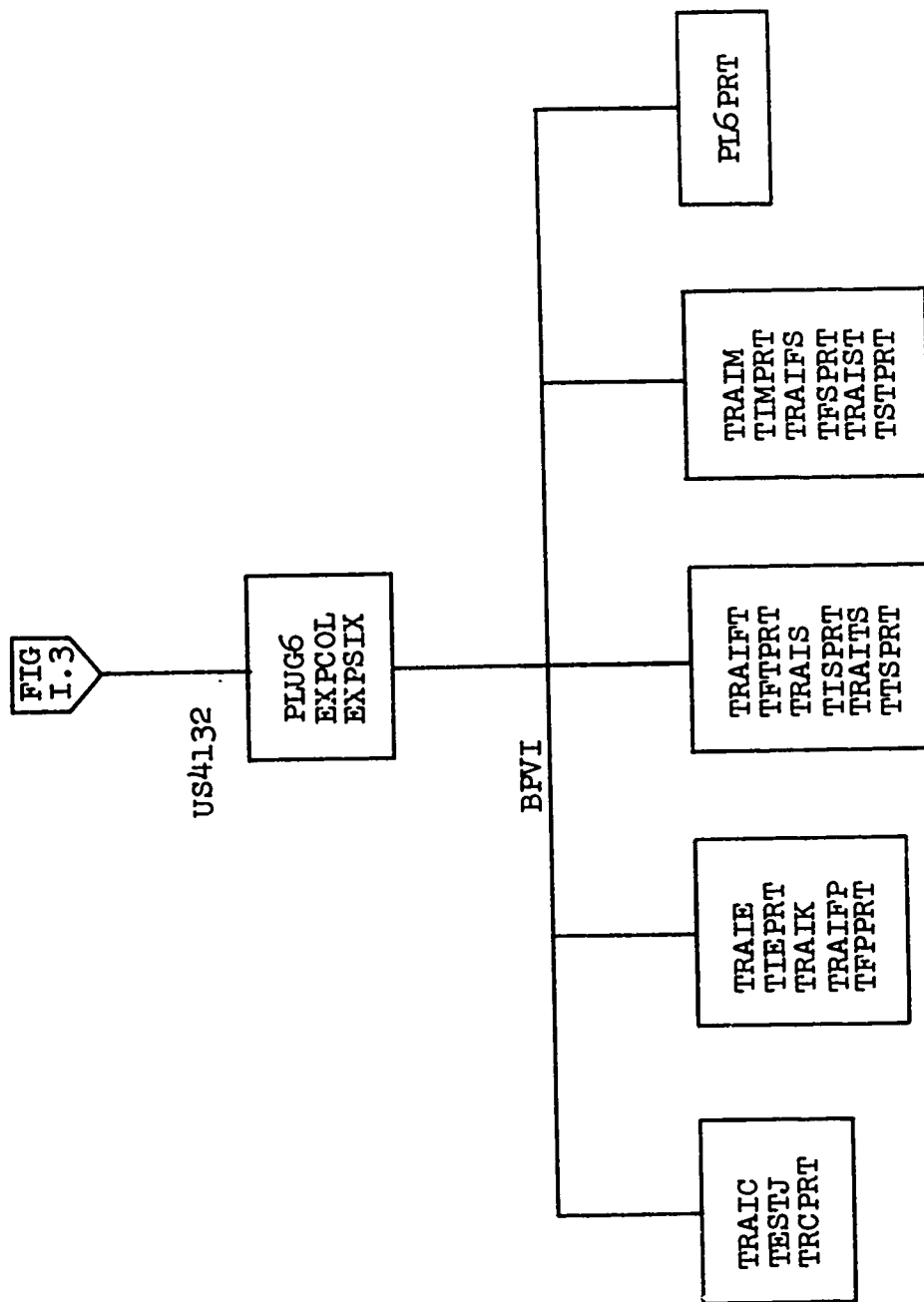


FIGURE I.10 TRIANGULAR CROSS SECTION RING ELEMENT

APPENDIX II
LIST OF SUBROUTINE FUNCTIONS

A. Control and Utility Subroutines

USER04*	(Main deck) Control reset of system parameters, call SEXEQ and return control to BFMTII
SRESET*	Reset system input unit, system output unit, maximum matrix limit, size of work area, print control and re-establish blank common area
SEXEQ*	Read and interpret USER04 instruction and pass control to USC4
US04	Control three phases of operation of USER04 module
NTEST	Examine matrix name for suppression code
REC1	Perform writing and reading of tape records for interpreted element input
US04A	Control first phase (input phase) of operation of USER04 module
CONTRL	Select scratch tape unit for copying structural data deck, extracting structural system information in the process
INPUT	Master control subroutine for reading and storing of structural input data
FRED	Generate grid point axes transformation matrices
BOUND	Read and store boundary constraints
ELEM	Read and store element input data
MATCH	Compare a material name to an entry name in the material library
LAG	Interpolate material properties with respect to temperature

* Required for SUBSYS implementation only

FGRLDS	Read and store grid point load conditions
FMAT	Generate, revise and/or display material library information
REFORM	Control report form input preprocessing
PHASE1	Read and store report form input data deck
LATCH	Compare an input label to list of legal input labels
FORMIN	Read and store report form table input
PHASE2	Merge data stored by PHASE1 into logical sequence for INPUT
OPEN	Control scratch tape manipulations for report form input
FLOADS	Output grid point load conditions as Format matrix
FTR	Output boundary constraints as a Format matrix
USO4B	Control second and third phases (element matrix generation and output) of operation of USERO4 module
FELEM	Control generation of element matrices
ELPLUG	Allocate work storage for elements, read interpreted element input, select proper element and store element matrices on scratch tape in compact form
REC3	Perform writing of tape records for element control data
REC4	Perform compact writing of tape records for generated element matrices
MINV	Perform in-core matrix inversion
AXTRA2	Apply grid point axes transformation
MAB	Perform in-core matrix multiplication
MSB	Perform in-core matrix multiplication where first matrix is symmetric

BCB	Perform in-core matrix triple product of the form $T^T K T$ where K is symmetric
MATB	Perform in-core matrix multiplication of the form $A^T B$
SYMPRT	Print symmetrically stored matrix
LOC	Compute single subscript index given double subscript indices
ELTEST	Compare input element control information to required element control information
MPRD	Perform generalized in-core matrix multiplication
TPRD	Perform generalized in-core matrix transpose multiplication
AI	Controls calculation procedures of triangular integration package
BINT	Perform integration by expansion of binomial theorem
AK	Calculate slope of line between two points of a triangle
AM	Calculate intercept of line between two points of a triangle
IFAC	Calculate n factorial for a given n
FJAB	Perform defined integration
F6219	Perform defined integration
F6211	Perform defined integration
AJ	Perform defined integration
COEF	Calculate binomial coefficients
F89	Perform defined integration
FF100	Perform defined integration

AXTRA1	Apply grid point axes transformations
AXTRA3	Apply grid point axes transformations
ELPRT	Print generated element matrices
OUTMAT	Output generated element matrices as Format matrices
US461	Write a matrix column record in compressed format
US462	Generate each elements contribution to the assembly transformation matrix
US463	Generate full column from symmetrically stored matrix
LOGFLO	Determine logical path for USER04 module
INDECK	Create data deck from input deck matrix
COPYDK	Create input deck matrix from data deck
SHIFT	Manipulate material library internal storage area
CHEK	Perform input cross checking
OUTINT	Output interpreted input as a matrix
ININT	Create interpreted input from a matrix
DEFLEX	Sort and store input displacements
SQUISH	Set non-generation indicators for suppressed matrices
MATSUP	Insert suppressed input matrix names into the Format system

B. Quadrilateral Thin Shell Element Subroutines

PLUG1	Master control
CC21	Form intermediate stiffness matrix by summation
MABC	Perform in-core matrix triple product multiplication
NEWFT	Calculate revised thermal load formulation
CDELPQ	Calculate coordinate integrals
CHDELL	Arrange coordinate integrals in storage
PLPRTA	Print results of coordinate and material properties calculations
CK11	Control generation of membrane stiffness matrix
CT11	Generate membrane stiffness transformation sub-matrix
MATI60	Invert 8 x 8 matrix in-core
CTOGM	Generate membrane transformation matrix for transformation from oblique to geometric coordinates
CTGRM	Generate membrane transformation matrix for transformation from geometric to reference system coordinates
CC1	Generate membrane stiffness sub-matrices
CMMASS	Generate membrane contribution to element mass matrix
CSTM	Generate membrane contribution to element stress matrix
CDM	Generate membrane displacement derivative matrix for element stress matrix control
CFMTS	Control generation of membrane contribution to element thermal stress and element thermal load matrices

CFMV	Generate membrane thermal load matrix
PRT1	Print membrane and flexure transformation matrices and contribution to element stiffness, stress, thermal stress, thermal load and pressure
CK22	Control generation of flexure stiffness matrix
CTGB	Generate flexure transformation sub-matrix
MATI70	Invert 16 x 16 matrix in-core
CTOGB	Generate flexure transformation matrix for transformation from oblique to geometric coordinates
CTGRB	Generate flexure transformation matrix for transformation from geometric to reference system coordinates
CC2	Generate flexure stiffness sub-matrices
CFP	Control generation of element pressure load matrix
CFPB	Generate intermediate element pressure load matrix
CSTF	Generate flexure contribution to element stress matrix
CDF	Generate flexure displacement derivative matrix for element stress matrix
CDFX	Generate flexure displacement partial with respect to X derivative matrix for element stress matrix
CDFY	Generate flexure displacement partial with respect to Y derivative matrix for element stress matrix
CFFS	Control generation of flexure contribution to element thermal stress and thermal load matrices
CFV	Generate flexure contribution to element thermal load matrix

CFMASS

Generate flexure contribution to element
mass matrix

C. Frame Element Subroutines

PLUG7	Master control, generation of element matrices (except incremental stiffness)
CTS	Generate transformation matrix for transformation from geometric to reference
CTCQ	Generate transformation matrix for transformation from material to geometric axes
CECC	Evaluate effect of eccentricities
INCRE	Generate element incremental stiffness matrix
P7PRT	Print transformation matrices and intermediate calculations

D. Triangular Thin Shell Element Subroutine

PLUG2	Master control
ASSY2	Assemble membrane and flexure stiffness sub-matrices
DCD	Perform in-core matrix multiplication of the form TST where T is a diagonal matrix and S is a symmetric matrix
DTAPR	Process coordinate data
MATPR	Generate material properties matrices
NEWFT1	Calculate revised thermal matrices
PTBM	Generate membrane transformation matrix for transformation from oblique to geometric coordinate systems
PTMGS	Generate membrane transformation matrix for transformation from geometric to reference system coordinates
DPQINT	Calculate coordinate integrals
PKM	Generate membrane contribution to element stiffness matrix
PSTM	Generate membrane contribution to element stress matrix
PFMTS	Generate membrane contribution to element thermal load and thermal stress matrices
PFMV1	Generate intermediate membrane thermal load matrix
APRT	Print membrane and flexure transformation matrices and contributions to element stiffness, stress, thermal stress, thermal load and pressure load matrices
PTFGS	Generate flexure transformation matrix for transformation from geometric to reference system coordinates
PKF	Generate flexure contribution to element stiffness matrices

CCB	Perform in-core matrix triple product of the form $T^T K T$ where K is symmetric and accuracy criteria is imposed
PFP	Generate element pressure load matrix
PFFTS	Generate flexure contribution to element thermal stress and thermal load matrices
PFFV1	Generate intermediate flexure thermal load matrix
PSTF	Generate flexure contribution to element stress matrix
PTBF	Generate flexure transformation matrix for transformation from oblique to geometric coordinate systems
EPRT	Print final element matrices
PLAS2	Non-functional
PNCI	Non-functional
PNG1	Non-functional

E. Triangular Cross Section Ring Element Subroutines

PLUG6	Master control.
EXPCOL	Expand column matrix to six degrees of freedom per point
EXPSIX	Expand symmetric matrix to six degrees of freedom per point
TRAIC	Generate coordinate transformation matrices and integrals
TESTJ	Impose accuracy criteria upon integrals
TRCPRT	Print coordinate transformation matrices and integrals
TRAIE	Generate material properties matrices
TIEPRT	Print material properties matrices
TRAIK	Generate element stiffness matrix
TIKPRT	Print element stiffness matrix
TRAIFP	Generate element pressure load matrix
TFPPRT	Print element pressure load matrix
TRAIFT	Generate element thermal load matrix
TFTPRT	Print element thermal load matrix
TRAIS	Generate element stress matrix
TISPRT	Print element stress matrix
TRAITS	Generate element thermal stress matrix
TTSPRT	Print element thermal stress matrix
TRAIM	Generate element mass matrix
TIMPRT	Print element mass matrix
TRAIFS	Generate element pre-strain load matrix
TFSVRT	Print element pre-strain load matrix

TRAIST	Generate element pre-stress load matrix
TSTPRT	Print element pre-stress load matrix
PL6PRT	Print all element matrices generated

F. Toroidal Ring Element Subroutines

PLUG5	Master control, generate element stiffness, thermal load, pressure load, stress and thermal stress matrices
MSTR	Change storage arrangement of a matrix
ROMBER	Perform integration by Romberg Method
F4	Evaluate a defined function for ROMBER
F5	Evaluate a defined function for ROMBER
F6	Evaluate a defined function for ROMBER
BMATRX	Generate coordinate transformation matrix
DMATRX	Generate material properties matrix
GAMMAT	Generate material transformation matrix
FCURL	Generate intermediate thermal load matrix
PLMX	Generate intermediate pressure load matrix
SCRLM	Generate intermediate stress matrix
SOLVE	Solve for element stress coefficients
QUADI	Performs integration using numerical quadrature methods
PRINT5	Print generated element matrices

G. Quadrilateral Shear Panel Element Subroutine

PLUG14	Master control, generate element stiffness, stress and mass matrices
MULTF	Performs in-core specialized matrix multiplication
POOF	Expands element matrices to displacement degrees of freedom
PL4PRT	Prints intermediate calculations and generated element matrices

APPENDIX III

TABLE OF ERROR MESSAGES

AN ERROR HAS OCCURRED IN THE USER04 MODULE. FORMAT WILL ATTEMPT TO CONTINUE.

Subroutine: USER04
Explanation: Self explanatory

ASSEMBLY TRANSFORMATION MATRIX SIZE XXXXXX EXCEEDS LIMIT XXXXXX OF FORMAT SYSTEM.

Subroutine: US04A
Explanation: Self explanatory

AVAILABLE SCRATCH DATA SETS XXXX IS LESS THAN THE REQUIRED 4.

Subroutine: US04A
Explanation: The USER04 module requires at least four scratch data sets. The addition of more data sets is required by the program.

DUE TO ABOVE ERROR CONDITION CHECK CARD WILL BE INSERTED. EXECUTION WILL BE SUPPRESSED.

Subroutine: PHASE2
Explanation: Self explanatory

DUE TO ABOVE ERROR MESSAGE THIS SECTION WILL BE OMITTED AND CHECK CARD INSERTED.

Subroutine: PHASE1
Explanation: Self explanatory

DUE TO PREVIOUSLY ENCOUNTERED ERROR CONDITION THIS SECTION IS BEING SKIPPED. PROGRAM WILL FLUSH DATA DECK UNTIL NEXT RECOGNIZABLE INPUT SECTION IS ENCOUNTERED.

Subroutine: PHASE1
Explanation: Self explanatory

DUE TO THE OMISSION OF THIS SECTION THE FOLLOWING SECTIONS MAY BE IGNORED - XXXXXX XXXXXX XXXXXX ...

Subroutine: PHASE2
Explanation: The final processing of certain sections requires data from other sections which by omission or other input error are not present.

ELEMENT CONTROL ERROR IN SUBROUTINE ELEM

ELEMENT NUMBER XXXXX CALLS PLUG NUMBER XXX. PLUG NUMBER SHOULD BE GREATER THAN ZERO. EXECUTION TERMINATED.

Subroutine: ELEM

Explanation: All element type code numbers are greater than zero. Proper element type cannot be selected.

ELEMENT CONTROL ERROR IN SUBROUTINE ELEM

ELEMENT NUMBER XXXXX HAS MATERIAL NUMBER XXXXXX. MATERIAL IDENTIFICATION MUST BE DIFFERENT FROM ZERO. EXECUTION TERMINATED.

Subroutine: ELEM

Explanation: Self explanatory

ELEMENT CONTROL ERROR IN SUBROUTINE ELEM

ELEMENT NUMBER XXXXX HAS NUMBER OF INPUT POINTS = XX. NUMBER OF INPUT POINTS MUST BE POSITIVE. EXECUTION TERMINATED.

Subroutine: ELEM

Explanation: Self explanatory

ELEMENT CONTROL ERROR IN SUBROUTINE ELEM

ELEMENT NUMBER XXXXX HAS NUMBER OF GRID POINTS = XXX. NUMBER OF GRID POINTS MUST BE GREATER THAN ZERO AND NO GREATER THAN EIGHT. EXECUTION TERMINATED.

Subroutine: ELEM

Explanation: Self explanatory

ELEMENT INPUT ERROR NO. X PLUG NO. XX ELEMENT NO XXXX

Subroutine: ELPLUG

Explanation: Error number 1 - incorrect plug number (element type code)
Error number 2 - incorrect number of element defining points
Error number 3 - incorrect value for extra element input indicator
Error number 4 - incorrect matrix orders for element (number of degrees of freedom per point incorrect)

ELEMENT GENERATION CORE STORAGE REQUIRED XXXXXX EXCEEDS THAT AVAILABLE XXXXXX TO DISPLACEMENT METHOD MATRIX GENERATOR.

Subroutine: US04A

Explanation: Blank common work area is not large enough for generation of element matrices.

ELEMENT SORT ROUTINE CORE STORAGE REQUIRED XXXXXX EXCEEDS
THAT AVAILABLE XXXXXX TO DISPLACEMENT METHOD MATRIX
GENERATOR.

Subroutine: US04B

Explanation: Blank common work area is not large
enough for output of generated
matrices.

ERROR MESSAGE FROM SUBROUTINE MAT

ATTEMPT TO DELETE MATERIAL NUMBER XXXXXX USING LOCK CODE XX.
INCORRECT LOCK CODE, REQUEST IGNORED.

Subroutine: FMAT

Explanation: Self explanatory

ERROR MESSAGE FROM SUBROUTINE MAT

ATTEMPT TO DELETE MATERIAL THAT WAS NOT ON MATERIAL TAPE.
MATERIAL NUMBER XXXXXX. MATERIAL IDENTIFICATION IS
XXXXXXXXXXXXXXXXXXXXXXXXXXXX. INPUT CODE IS XXX.
REQUEST IGNORED.

Subroutine: FMAT

Explanation: Self explanatory

ERROR MESSAGE FROM SUBROUTINE MAT

ATTEMPT TO INPUT PLASTIC DATA ONLY FOR MATERIAL WHICH WAS
NOT ON TAPE. MATERIAL NUMBER XXXXXX. MATERIAL IDENTIFICA-
TION IS XXXXXXXXXXXXXXXXXXXXXXXX. INPUT CODE IS XXX.
REQUEST IGNORED.

Subroutine: FMAT

Explanation: Usage of an input code of "P" requires
that the material to be revised already
exists in the material library.

ERROR MESSAGE FROM SUBROUTINE MAT

ATTEMPT TO REVISE MATERIAL NUMBER XXXXXX USING LOCK CODE XX.
INPUT LOCK CODE DOES NOT MATCH TAPE LOCK CODE FOR THIS
MATERIAL. REVISIONS OR DELETIONS NOT ALLOWED WITHOUT
PROPER LOCK CODE. EXECUTION TERMINATED.

Subroutine: FMAT

Explanation: Self explanatory

ERROR MESSAGE FROM SUBROUTINE MAT

NUMBER OF REQUESTS RECEIVED IS ZERO.

Subroutine: FMAT

Explanation: Number of requests must not be zero.
Value of zero indicates improper
operation of program.

ERROR MESSAGE FROM SUBROUTINE MAT

REQUEST FOR PRINT OF MATERIAL THAT WAS NOT ON TAPE.
MATERIAL NUMBER XXXXXX. MATERIAL IDENTIFICATION IS
XXXXXXXXXXXXXXXXXXXXXXXXXXXX. INPUT CODE IS XXX.
REQUEST IGNORED.

Subroutine: FMAT

Explanation: Self explanatory

ERROR MESSAGE FROM SUBROUTINE MAT

UNRECOGNIZABLE DATA INPUT CODE. LEGAL CODES ARE PI, PO,
I, O, P, OUT, ALL, SEE, SUM. MATERIAL NUMBER XXXXXX.
MATERIAL IDENTIFICATION IS XXXXXXXXXXXXXXXXXXXXXXXX.
INPUT CODE IS XXX. EXECUTION TERMINATED.

Subroutine: FMAT

Explanation: Self explanatory

ERROR MESSAGE FROM SUBROUTINE MAT

ADDITIONS REQUESTED EXCEED CAPACITY OF MATERIAL TAPE.
MAXIMUM NUMBER OF MATERIALS CANNOT EXCEED XXX.

Subroutine: FMAT

Explanation: Self explanatory

FOR I = XX AND N = XX INTEGRAL DOES NOT CONVERGE

Subroutine: PLUG5

Explanation: No convergence has been obtained for
the given integral calculated by the
Romberg technique in the Toroidal
Ring Element.

GRID POINT LOADS MATRIX STORAGE REQUIRED XXXXXX EXCEEDS
THAT AVAILABLE XXXXXX TO DISPLACEMENT METHOD MATRIX
GENERATOR.

Subroutine: US04A

Explanation: Blank common work area is not large
enough for generation of grid point
loads matrix.

GRID POINT LOAD MATRIX SIZE XXXXXX EXCEEDS LIMIT XXXXXX
OF FORMAT SYSTEM.

Subroutine: US04A

Explanation: Self explanatory

ILLEGAL MODAL CARD ENCOUNTERED. CARD WILL BE IGNORED.

Subroutine: PHASE1

Explanation: A modal card has been found while
reading an input section for which
no modal card has been defined.

INPUT ERROR IN SUBROUTINE ELEM, AFTER INTERPOLATION
POISSON VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN MATERIAL
NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD
BE GREATER THAN -1.0 AND LESS THAN 1.0. EXECUTION
TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM, AFTER INTERPOLATION
RIGIDITY VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN MATERIAL
NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD
BE GREATER THAN 1.0. EXECUTION TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM, AFTER INTERPOLATION
THERMAL COEFFICIENT VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN
MATERIAL NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX.
VALUE SHOULD BE GREATER THAN -1.0 AND LESS THAN 1.0.
EXECUTION TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM, AFTER INTERPOLATION
VALUE OF YOUNG'S MODULUS EQUALS $\pm$.XXXXXXXX $\pm$ XX IN
MATERIAL NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX.
VALUE SHOULD BE GREATER THAN 1.0. EXECUTION TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM
ELEMENT NODE POINT IS NEGATIVE OR ZERO IN ELEMENT
NUMBER XXXXX.

Subroutine: ELEM
Explanation: No element defining point number may
be negative and only mid-points may
be zero.

INPUT ERROR IN SUBROUTINE ELEM
ELEMENT NUMBER XXXXXX IS DEFINED BY NODE POINTS FOR
WHICH NO COORDINATES HAVE BEEN INPUT. CALCULATION OF
MATERIAL TEMPERATURE IMPOSSIBLE. EXECUTION TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM
MASS DENSITY VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN MATERIAL
NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD
BE GREATER THAN ZERO. EXECUTION TERMINATED.

Subroutine: ELEM
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE ELEM
VALUE OF IP = XXX, VALUE OF IPRE = XXX FOR ELEMENT
NUMBER ONE. REQUEST TO REPEAT DATA FROM ELEMENT
PREVIOUS TO FIRST ELEMENT IS ILLOGICAL. EXECUTION
TERMINATED.

Subroutine: ELEM
Explanation: IP and IPRE cannot be negative
for first element.

INPUT ERROR IN SUBROUTINE MAT
MASS DENSITY VALUE EQUALS $\pm$.XXXXXXXX $\pm$ XX IN MATERIAL
NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD
BE NON-NEGATIVE. EXECUTION TERMINATED.

Subroutine: FMAT
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE MAT
MATERIAL NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX.
NUMBER OF MATERIAL TEMPERATURE POINTS IS XXX. NUMBER
OF PLASTIC TEMPERATURE POINTS IS XXX. NUMBER OF
TEMPERATURE POINTS IN EITHER CASE CANNOT EXCEED 9.
EXECUTION TERMINATED.

Subroutine: FMAT
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE MAT
POISSON VALUE EQUALS $\pm$.XXXXXXXX $\pm$ XX IN MATERIAL NUMBER
XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD BE
GREATER THAN -1.0 AND LESS THAN 1.0. EXECUTION
TERMINATED.

Subroutine: FMAT
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE MAT
RIGIDITY VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN MATERIAL NUMBER
XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX. VALUE SHOULD BE
GREATER THAN 1.0. EXECUTION TERMINATED.

Subroutine: FMAT
Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE MAT

THERMAL COEFFICIENT VALUE EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN
MATERIAL NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX.
VALUE SHOULD BE GREATER THAN -1.0 AND LESS THAN 1.0.
EXECUTION TERMINATED.

Subroutine: FMAT

Explanation: Self explanatory

INPUT ERROR IN SUBROUTINE MAT

VALUE OF YOUNG'S MODULUS EQUALS $\pm$.XXXXXXXXXE $\pm$ XX IN
MATERIAL NUMBER XXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXXX.
VALUE SHOULD BE GREATER THAN 1.0.

Subroutine: FMAT

Explanation: Self explanatory

INPUT ERROR NUMBER OF REFERENCE POINTS INPUT EXCEEDS XXXX.

Subroutine: INPUT

Explanation: Program cannot accommodate more than
the given number of input points.

INPUT ERROR NUMBER OF DIRECTIONS OF GRID POINTS NOT
EQUAL TO NUMBER OF DIRECTIONS OF TRANSFORMATION MATRIX.
EXECUTION TERMINATED.

Subroutine: INPUT

Explanation: Order of grid point axes transformation
matrices must be equal to three.

INPUT ROUTINE CORE STORAGE REQUIRED XXXX.X EXCEEDS THAT
AVAILABLE XXXXXX TO DISPLACEMENT METHOD MATRIX GENERATOR.

Subroutine: US04A

Explanation: Blank common work area is not large
enough for processing input.

INTERNAL TAPE ERROR HAS OCCURRED. PROCESSING ABANDONED.

Subroutine: PHASE2

Explanation: Report form input preprocessor cannot
retrieve information stored on a
scratch data set.

LABEL CARD ERROR XXXXXX

Subroutine: INPUT

Explanation: Input card read should have been label
card. Execution will be terminated.

LOAD CONDITION XXX SUB-LABEL IS INCORRECT. PROGRAM
CANNOT DISTINGUISH BETWEEN LOAD CONDITIONS.

Subroutine: PHASE1

Explanation: Load condition sub-label in report
form input is in error.

MAXIMUM NUMBER OF ITERATIONS REACHED IN ROMBERG
INTEGRATION ROUTINE.

Subroutine: PLUG5

Explanation: Convergence was not obtained in 15
iterations for an integral in the
toroidal thin shell element.
Processing will continue, using
15 iteration result.

MAXIMUM NUMBER OF LOAD CONDITIONS ALLOWED IS 100. THIS
PROBLEM CONTAINS XXXX.

Subroutine: PHASE1

Explanation: Self explanatory

MORE THAN ONE OPTION HAS BEEN SELECTED FOR REQUEST NUMBER XXX
OF MATERIAL LIBRARY. ONLY THE FIRST SELECTION WILL
BE RETAINED.

Subroutine: PHASE1

Explanation: Self explanatory

NEW MATERIAL TAPE NOT GENERATED. ALL REVISIONS AND/OR
DELETIONS REQUESTED BY THIS CASE HAVE BEEN IGNORED.

Subroutine: FMAT

Explanation: Due to a previous error, generation
of a new material library has been
abandoned. Execution will be terminated.

NO END OR CHECK CARD HAS BEEN FOUND. CHECK CARD WILL BE
INSERTED, SUPPRESSING EXECUTION.

Subroutine: PHASE2

Explanation: Self explanatory

NO OPTION HAS BEEN SELECTED FOR REQUEST NUMBER XXX OF
MATERIAL LIBRARY.

Subroutine: PHASE1

Explanation: Self explanatory

NUMBER OF ELEMENTS READ XXXXX IS GREATER THAN 9999.

NUMBER OF ELEMENTS WILL BE SET AT 9999.

Subroutine: PHASE 2

Explanation: Self explanatory, execution will be suppressed.

NUMBER OF ENTRIES READ FOR THIS SECTION, XXXXX, DOES NOT AGREE WITH NUMBER THAT WAS TO BE READ, XXXXX. ACTUAL NUMBER READ WILL BE USED.

Subroutine: PHASE 2

Explanation: Self explanatory

PLUG7 ERROR - THIRD POINT TO DEFINE PLANE WAS NOT GIVEN - INPUT ERROR.

Subroutine: P7PRT

Explanation: Three element defining points are required for the frame element, the third supplying definition of the plane.

REDUCTION OF TRANSFORMATION MATRICES STORAGE XXXXXX

EXCEEDS THAT AVAILABLE TO DISPLACEMENT METHOD

MATRIX GENERATOR.

Subroutine: US04A

Explanation: Blank common work area is not large enough for generation of reduction transformation matrix.

REDUCTION TRANSFORMATION MATRIX SIZE XXXXXX EXCEEDS LIMIT

XXXXXX OF FORMAT SYSTEM.

Subroutine: US04A

Explanation: Self explanatory

REPEAT FOR FIRST POINT IGNORED.

Subroutine: FORMIN

Explanation: Repeat option on table forms of report form input cannot be used for first value entered.

REPORT ROUTINE CORE STORAGE REQUIRED XXXXXX EXCEEDS THAT
AVAILABLE XXXXXX TO DISPLACEMENT METHOD MATRIX GENERATOR.
Subroutine: US04A
Explanation: Blank common work area is not large enough
for processing report form input data.

STIFFNESS MATRIX SIZE XXXXXX EXCEEDS LIMIT OF FORMAT SYSTEM.
Subroutine: US04A
Explanation: Self explanatory

STRESS MATRIX SIZE XXXXXX EXCEEDS LIMIT XXXXXX OF FORMAT
SYSTEM.
Subroutine: US04A
Explanation: Self explanatory

SUBROUTINE MINV HAS DETERMINED ARRAY GAMABQ TO BE SINGULAR,
EXECUTION TERMINATED BY SUBROUTINE TRAIC.
Subroutine: TRAIC
Explanation: Transformation matrix to system
coordinates in triangular cross-section
ring element cannot be inverted,
usually because three element defining
points do not define a triangle.

SYSTEM INFORMATION CARD MISSING. CANNOT ALLOCATE STORAGE.
Subroutine: CONTRL
Explanation: All input data decks must have SYSTEM
section to allocate storage for
processing of input.

THE INTEGRAL OF $(\ln(A+B*X)/X) DX$ IS NOT ALLOWED FOR $A+B*X=0$
 $A = \pm .XXXXXXXXXE\pm XX$ $B = \pm .XXXXXXXXXE\pm XX$ $X = \pm .XXXXXXXXXE\pm XX$
Subroutine: F6211
Explanation: Natural log of zero is undefined.

THERE IS A MISTAKE IN THE COORDINATES FOR THIS TRANSFORMATION,
WE WILL CALCULATE THE REMAINING IN SPITE OF THIS.
Subroutine: FRED
Explanation: An error has occurred in generating a
grid point axes transformation matrix.
Execution will continue.

THIS SECTION HAS EITHER BEEN OMITTED OR FLUSHED BY PHASE ONE ERROR. IN EITHER CASE THIS SECTION IS CONSIDERED CRITICAL AND EXECUTION WILL NOT BE ALLOWED.

Subroutine: PHASE2

Explanation: Self explanatory

THIS SECTION IS TO BE MERGED WITH XXXXXX AND XXXXXX FOR WHICH MODAL CARDS HAVE BEEN ENCOUNTERED FOR BOTH. TWO VALUES CANNOT BE ASSIGNED TO THE SAME POINT. BOTH MODAL CARDS WILL BE IGNORED.

Subroutine: PHASE2

Explanation: Self explanatory

THIS SECTION IS TO BE MERGED WITH XXXXXX AND XXXXXX FOR WHICH VALUES HAVE BEEN ASSIGNED BY BOTH FOR POINT NUMBER XXXXX. TWO VALUES CANNOT BE ASSIGNED TO THE SAME POINT. NEITHER VALUE WILL BE USED.

Subroutine: PHASE2

Explanation: Self explanatory

TOROIDAL RING ELEMENT WITH CO-ORDINATES $R1 = \pm .XXXXXXXXXE\pm XX$
 $R2 = \pm .XXXXXXXXXE\pm XX$ $Z1 = \pm .XXXXXXXXXE\pm XX$
 $Z2 = \pm .XXXXXXXXXE\pm XX$ IS NOT DIAGONALLY DOMINANT AND SHOULD BE SUBDIVIDED.

Subroutine: PRINT5

Explanation: Element stiffness matrices must be diagonally dominant.

UNEXPECTED BLANK LABEL CARD ENCOUNTERED.

Subroutine: PHASE2

Explanation: Card read should have contained an input section label. Input processor will attempt to continue.

UNEXPECTED LABEL CARD READ - POINT XXXXX

Subroutine: FORMIN

Explanation: Input section label card encountered while reading table form input. Point reflects entry now being processed.

UNRECOGNIZABLE INPUT SECTION.

Subroutine: PHASE1

Explanation: Input section label has been read
which is undefined in input
processor.

VALUE OF SIN (ALPHA) IS ZERO - RUN TERMINATED.

Subroutine: PLUG1

Explanation: Element defining points are in
error for Quadrilateral Thin Shell
Element.

APPENDIX IV

EXAMPLE STRESS AND STABILITY INSTRUCTION SEQUENCES

A. DISPLACEMENT AND STRESS ANALYSIS INSTRUCTION SEQUENCE

,MATLBB,,LOADS,TR,TA,KEL,FTEL,SEL,SZALEL,, = ,,MATLBA,
.USERO4.

```
C
C      FORM TAR MATRIX (ASSEMBLY AND APPLICATION OF
C      BOUNDARY COND.)
C
TRT = TR .TRANSP.
TAR = TA .TMULT. TRT

C
C      ASSEMBLE AND REDUCE ELEMENT STIFFNESS MATRICES
C
KTEMP = KEL .TMULT. TAR
STIFF = TAR .TMULT. KTEMP
PRINT (FORCE ,DISP. , ,) STIFF

C
C      ASSEMBLE AND REDUCE ELEMENT APPLIED LOADS
C
FTELAR = TAR .TMULT. FEL
PRINT (REDDOF,COND. , ,) FTELAR

C
C      APPLY BOUNDARY CONDITIONS TO SYSTEM LOADS
C
LOADR = TR .MULT. LOADS
PRINT (REDDOF,COND. , ,) LOADR

C
C      COMBINE ELEMENT AND SYSTEM LOADS
C
TLOAD = FTELAR .ADD. LOADR
PRINT (REDDOF,COND. , ,) TLOAD

C
C      SOLVE FOR DISPLACEMENTS
C
DISPR = STIFF .SEQEL. TLOAD
PRINT (REDDOF,COND. , ,) DISPR

C
C      SOLVE FOR ELEMENT STRESSES
C
```

```

STREL = SEL .MULT. TAR
STRESF = STREL .MULT. DISPR
STRESS = STRESF .SUBT. SZALEL
PRINT (NRSEL ,COND. , ,) STRESS
C
C
C      SOLVE FOR ELEMENT FORCES
C
FORCEL = KTEMP .MULT. DISPR
FORCES = FORCEL .SUBT. FEL
PRINT (D.O.F.,COND. , ,) FORCES
C
C
C      SOLVE FOR SYSTEM REACTIONS
C
REACTN = TA .MULT. FORCES
REACT = REACTN .SUBT. LOADS
PRINT (D.O.F.,COND. , ,) REACT

B.  STABILITY ANALYSIS INSTRUCTION SEQUENCE

, ,INTINP,LOADS,TR,TA,KEL,FTEL,, , , ,MATLBA, .USER04.

C
C
C      FORM TAR MATRIX (ASSEMBLY AND APPLICATION OF
C      BOUNDARY CONDITIONS)
C
TRT = TR .TRANSP.
TAR = TA .TMULT. TRT
C
C
C      ASSEMBLE AND REDUCE ELEMENT STIFFNESS MATRICES
C
KTEMP = KEL .TMULT. TAR
STIFF = TAR .TMULT. KTEMP
PRINT (FORCE,DISP,,) STIFF
C
C
C      ASSEMBLE AND REDUCE ELEMENT APPLIED LOADS
C
FTELAR = TAR .TMULT. FTEL
PRINT (DISP,COND,,) FTELAR
C
C
C      APPLY BOUNDARY CONDITIONS TO SYSTEM LOADS
C
LOADR = TR. MULT. LOADS
PRINT (DISP,COND,,) LOADR

```

```

C
C      COMBINE ELEMENT AND SYSTEM LOADS
C
TLOAD = FTELAR .ADD. LOADR
PRINT (DISP,COND,,)TLOAD
C
C      CREATE FLEXIBILITY MATRIX
C
FLEX = STIFF .INVERS.
PRINT (DISP,FORCE ,,) FLEX
C
C      SOLVE FOR DISPLACEMENTS
C
DISPR = FLEX .MULT. TLOAD
PRINT (DISP,COND,,) DISPR
C
C      RETURN DISPLACEMENTS TO FULL SYSTEM ORDER
C
DISP = TR .TMULT. DISPR
C
C      GENERATE ELEMENT INCREMENTAL STIFFNESS MATRICES
C
,,,,,,,,,NEL, = ,INTINP,MATLBA,DISP .USER04.
C
C      ASSEMBLE AND REDUCE ELEMENT INCREMENTAL STIFFNESS
C      MATRICES
C
NTEMP = NEL .TMULT. TAR
INCRE = TAR .TMULT. NTEMP
PRINT (,,,) INCRE
C
C      CREATE INPUT EIGENVALUE MATRIX
C
KINVN = FLEX .MULT. INCRE
PRINT (,,,) KINVN
C
C      OBTAIN EIGENVALUES AND EIGENVECTORS
C
EIGVAL, EIGVEC = KINVN .EIGEN. (5)
PRINT (,,,) EIGVAL, EIGVEC
C
C      COMPUTE FREQUENCIES (EXP(1,1) = -.5)
C
FREQ = EIGVAL .POWER. EXP(1,1)
PRINT (FREQ,,,) FREQ

```

APPENDIX V
SUBROUTINE DOCUMENTATION

1. Subroutine Name: USER04 (Main Deck)
2. Purpose: Provide main deck control under SUBSYS implementation
3. Equation and Procedures: Logical variable ERROR is set to false. Subroutine SRESET is called to reset system parameters. Subroutine SEXEQ is then called to execute the .USER04. abstraction instruction. SUBSYS subroutine SEARCH is then called to return to the BFMTII program.
4. Input Argument: None
5. Output Argument: None
6. Error Returns: If logical variable ERROR is found to be true after performing subroutine SEXEQ, then an error message to this effect is printed and continuation of execution is attempted.
7. Calling Sequence: None
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 120_8 (80_{10}).
12. Subroutine User: None
13. Subroutines Required: SRESET
SEXEQ
SEARCH
14. Remarks: None

1. Subroutine Name: SEXEQ
2. Purpose: Extract and separate the required information from the USER04 instruction on the instruction tape
3. Equation and Procedure: The USER04 instruction is read from the instruction tape into the common work storage area. From information contained in the first six words of the instruction record the succeeding data in the record is separated into its component sections and placed into the calling sequence to US04.
4. Input Arguments:
NINST : Instruction tape number
IPRINT : Output print control
5. Output Arguments:
ERROR : Error condition indicator
6. Error Returns: None
7. Calling Sequence: (NINST, IPRINT, ERROR)
8. Input Tape: NINST - Abstraction instruction tape
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total Storage is 217_8 (136_{10}).
12. Subroutine User: USER04
13. Subroutine Required: US04
14. Remarks: None

1. Subroutine Name: SRESET
2. Purpose: Reset system parameters under SUBSYS implementation
3. Equations and Procedures: There are seven system parameters which must be reset due to operating under SUBSYS. They are:

- (1) NPIT : System input unit number
- (2) NPOT : System output unit number
- (3) KONST : Maximum matrix order capability
- (4) NWORK : Number of storages in work area
- (5) IPRINT : Output print control
- (6) WORK : Dimensioned work storage area (to be in blank common)
- (7) NINST : Unit number containing instructions

NINST is defined to have a value of one. NPIT, NPOT, KONST, NWORK and IPRINT are reset by reading them from the return instruction on NINST. NINST is searched until the return instruction is located, then NINST is backspaced and the return instruction is read again, this time the required system parameters are read, thus resetting their values. The work storage area, WORK, is allocated into blank common by a COMMON statement in SRESET.

4. Input Arguments: None
5. Output Arguments:
NINST : Fortran logical unit number containing instructions
IPRINT : Output print control
NPOL : System output unit number
6. Error Returns: None
7. Calling Sequence: (NINST, IPRINT, NPOL)
8. Input tape: NINST - Abstraction instruction input tape.
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 163₈ (115₁₀).
12. Subroutine User: USER04
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: US04
2. Purpose: Control operation of the structural generative system (USER04 module)
3. Equations and Procedures: The error indicator, ERROR, is initially set to .FALSE.. Subroutine US04A is then called to control the input operations. Subroutine US04B is called to control the element matrix generation and output phases. If an error has occurred in the input phase then the call to US04B is skipped. All information received from the Format Monitor is relayed to US04A and US04B.
4. Input Arguments:
 - NUMOT: Number of output matrices
 - NAMOUT: Array containing output matrix names
 - IOSPEC: Unit specifications for output matrices
 - NUMIN: Number of input matrices
 - NAMIN: Array containing input matrix names
 - INSPEC: Unit specifications for input matrices
 - NUMSR: Number of available scratch units
 - ISSPEC: Scratch unit specifications
 - NUMSC: Number of scalars
 - SCALAR: Array containing scalars
 - NWORKR: Number of available storages in blank common work area
 - WORK: Work storage area
 - IPRINT: System print control
5. Output Argument:
 - ERROR: Error condition indicator
6. Error Returns: If error has occurred in US04A or US04B then ERROR will be .TRUE. upon return to the calling program.
7. Calling Sequence:
 - CALL US04 (NUMOT, NAMOUT, IOSPEC, NUMIN, NAMIN, INSPEC, NUMSR, ISSPEC, NUMSC, SCALAR, ERROR, NWORKR, WORK, IPRINT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 204_8 (132_{10}).
12. Subroutine User: SEXEQ

13. Subroutines Required:

USO4A
USO4B

14. Remarks: None

1. Subroutine Name: NTEST
2. Purpose: To determine if output matrix is to be generated by US04
3. Equations and Procedures: The first position in the output name is compared to a slash (/). If this first character is a slash then the matrix is not to be calculated. If the first character is not a slash then the matrix will be calculated and output.
4. Input Arguments: NAME - array containing output matrix name
5. Output Arguments: KØDE - control code
if KØDE equals zero then matrix is calculated
if KØDE equals one then matrix is not calculated
6. Error Returns: None
7. Calling Sequence: Call NTEST (NAME, KØDE)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total Storage = $44_8 = 36_{10}$
12. Subroutine User: US04A, US04B
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: RECl
2. Purpose: Write or read element input tape record
3. Equations and Procedures: The decision to write or read the tape record is determined by examining the input variable IOPT in the following manner:
 - (1) If $\text{IOPT} \geq 2$ then the tape record will be written
 - (2) If $\text{IOPT} \leq 1$ then the tape record will be read
4. Input Arguments: (when $\text{IOPT} \geq 2$)
 - IOPT : Read/write indicator
 - K : Involved unit number
 - NIL : Number of words in tape record, excluding NIL
 - IPL : Element type number (plug number)
 - X : "X" coordinates of element definition points
 - Y : "Y" coordinates of element definition points
 - Z : "Z" coordinates of element definition points
 - T : Temperatures at element definition points
 - P : Pressures at element definition points
 - NLIST : Total degrees of freedom in element
 - LISTEL : Boundary condition information list
 - NNO : Number of element defining points
 - NODES : Grid point numbers of element defining points
 - IP : Extra element input and matrix repeat indicator
 - DISPEL : Input displacements for element degrees of freedom
 - PCOLEL : External loads for element degrees of freedom
 - LISTDL : Not used
 - IG : Maximum number of element defining points
 - NEL : Element number
 - GPAXEL : Grid point axes transformation matrices for element defining points
 - NUMMAT : Length of MAT array
 - MAT : Array containing interpolated material properties
 - NUMEPS : Length of EPSIO array
 - EPSIO : Pre-strain load vector
 - NUMSO : Length of SO array
 - SO : Pre-stress load vector
 - EXTRA : Extra element input
5. Output Arguments: (when $\text{IOPT} \leq 1$)

With the exception of IOPT and K, which are always input arguments, all of the above input arguments are output arguments when IOPT = 1.
6. Error Returns: None

7. Calling Sequence: (IOPT, K, NIL, IPL, X, Y, Z, T, P, NLIST, LISTEL, NNO, NODES, IP, DISPEL, PCOLEL, LISTDL, IG, NEL, GPAXEL, NUMMAT, MAT, NUMEPS, EPSIO, NUMSO, SO, EXTRA)
8. Input Tapes: When $\text{IOPT} \leq 1$ the input tape number is the variable K.
9. Output tapes: When $\text{IOPT} \geq 2$ the output tape number is the variable K.
10. Scratch Tapes: None
11. Storage Required: Total storage is 1024_8 (580_{10}).
12. Subroutine User: ELEM, ELPLUG
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: US04A
2. Purpose: Control input phase operations of structural system (USER04 module)
3. Equations and Procedures: Input, output and scratch units supplied by the Format Monitor are assigned to their respective functions. Subroutine CONTRL is called to copy the entire structural data input onto a scratch tape, extracting structural system information in the process. From this point, subroutine INPUT controls the selection of all other subroutines which process input (see INPUT). The function of US04A is to partition the blank common work storage area and select the proper subroutine for the following operations: If material library requests are present then subroutine FMAT is called, if report form input processing is required then subroutine REFORM is called, if generation of the loads matrix is not suppressed then subroutine FLOADS is called and finally if the boundary condition transformation matrix is not suppressed then FTR is called.

4. Input Arguments:

NUMOT: Number of output matrices (12)
 NAMOUT: Array containing output matrix names
 IOSPEC: Unit specifications for output matrices
 NUMIN: Number of input matrices (4)
 NAMIN: Array containing input matrix names
 INSPEC: Unit specifications for input matrices
 NUMSR: Number of available scratch units
 ISSPEC: Scratch unit specifications
 NUMSC: Number of scalars (0)
 SCALAR: Array containing scalars
 NWORKR: Number of available work storages in blank common area (WORK)
 WORK: Work storage area
 IPRINT: System print control

5. Output Arguments:

ERROR: Error condition indicator
 KNMD: Array containing structural system control information
 KNMD (1) - NSYS - Total number of degrees of freedom in application
 KNMD (2) - NL - Number of load conditions
 KNMD (3) - NALB - Number of degrees of freedom after application of boundary conditions
 KNMD (4) - NNORD - Summation of element degrees of freedom
 KNMD (5) - NELEM - Number of elements
 KNMD (6) - NNRSEL - Summation of element stress orders
 KNMD (7) - NTD - Number of degrees of freedom per point
 KNMD (8) - NRSELM - Maximum element stress order
 KNMD (9) - NORDM - Maximum element degrees of freedom
 KNMD (10) - NOINRM - Maximum number of storages required for an element stiffness matrix

6. Error Returns: If at any time the number of required work storages exceeds NWORKR or a generated matrix will have a dimension greater than KONST (matrix size limitation), the appropriate message will be written, ERROR set to .TRUE. and control returned to the calling program.

7. Calling Sequence:

CALL US04A (NUMOT, NAMOUT, IOSPEC, NUMIN, NAMIN, INSPEC, NUMSR, ISSPEC, NUMSC, SCALAR, ERROR, NWORKR, WORK, IPRINT, KNMD)

8. Input Tapes:

ITAPE1 - INSPEC (1,1) - Unit containing input structure data deck
ITAPE2 - INSPEC (1,2) - Unit containing interpreted input
ITAPE3 - INSPEC (1,3) - Unit containing existing material library
ITAPE4 - INSPEC (1,4) - Unit containing input displacements

9. Output Tapes:

JTAPE1 - IOSPEC (1,1) - Unit which will contain copy of input structure data deck
JTAPE2 - IOSPEC (1,2) - Unit which will contain revised or new material library
JTAPE3 - IOSPEC (1,3) - Unit which will contain interpreted input
JTAPE4 - IOSPEC (1,4) - Unit which will contain grid point loads matrix
JTAPE5 - IOSPEC (1,5) - Unit which will contain boundary condition application transformation matrix
JTAPE6 - IOSPEC (1,6) - Unit which will contain assembly transformation matrix
JTAPE7 - IOSPEC (1,7) - Unit which will contain element stiffness matrices
JTAPE8 - IOSPEC (1,8) - Unit which will contain element load matrices
JTAPE9 - IOSPEC (1,9) - Unit which will contain element stress matrices
JTAP10 - IOSPEC (1,10) - Unit which will contain element thermal stress matrices
JTAP11 - IOSPEC (1,11) - Unit which will contain element incremental stiffness matrices
JTAP12 - IOSPEC (1,12) - Unit which will contain element mass matrices

10. Scratch Tapes:

- NTAPE1 - ISSPEC (1,1) - External storage area for report form input preprocessor and later will contain structural control information
- NTAPE2 - ISSPEC (1,2) - Contain temporary copy of translated input data deck and later contain generated element matrices in compact form
- NTAPE3 - ISSPEC (1,3) - Contain temporary copy of actual input deck and later contain interpreted element input data
- NTAPE4 - ISSPEC (1,4) - External storage area for report form input preprocessor and later contain input load conditions

11. Storage Required: Total storage is 22458 (1189₁₀).

12. Subroutine User: US04

13. Subroutines Required:

CONTRL
INPUT
FMAT
REFORM
NTEST
FLOADS
FTR

14. Remarks: None

1. Subroutine Name: CONTRL
2. Purpose: Generate BCD tape from system input tape data and read constants needed by US04 for dynamic storage and matrix sizes.
3. Procedure: The input data is read in BCD format of 12 words/card. A scanning of the data is made for certain card types.
 - a. REPORT card - defines NBCD to be NTAPE3
 - b. SYSTEM card - defines NBCD to be NTAPE2
 - c. CHECK card - end of file of NBCD
 - d. END card - end of file placed on NBCD
 - e. SYSTEM card - NREF, NREFP, NTD, NL, NELEM are read to allocate storage
4. Input Arguments: NTAPE2 - tape storage number for defining NBCD
NTAPE3 - tape storage number for defining NBCD
NPIT - system input tape number
5. Output Arguments:
NBCD : tape unit number on which data is stored
NREF : number of reference points on system
NREFP : number of reference points in grid point table
NTD : number of degrees of freedom per point
NL : number of grid point load conditions
NELEM : number of elements
6. Error Returns: None
7. Calling Sequence: CALL CONTRL (NREF, NREFP, NTD, NL, NELEM, NTAPE2, NPIT, NBCD, NTAPE3)
8. Input Tapes: NPIT - Input data tape
9. Output Tapes: NBCD - Output BCD tape
10. Scratch Tapes: None
11. Storage Required: CARD (12)
12. Subroutine User: US04A
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: INPUT
2. Purpose: Process directly or control processing of all structural input data.
3. Equations and Procedures: The input variable IN designates the Fortran logical unit number containing a direct label card input deck. If the input deck was actually direct it was copied onto IN by subroutine CONTRL. If report form input was used then the report form input preprocessor placed the generated direct label card input deck on IN.

The logic in INPUT is to read a label card and branch to the appropriate section to process the indicated data. The available label sections and the action taken upon encountering each is indicated in the following list.

Input Section

Label	Action Taken
TITLE	Title cards are read and printed on system output unit in INPUT
PRINT	Not used, the data card is flushed through
NREF	Processed directly in INPUT, data eventually stored on scratch tape (NTAPE1)
GRID	Processed directly in INPUT, data eventually stored on scratch tape (NTAPE3)
BOUND	Processed by direct call to subroutine BOUND, data stored on scratch tapes (NTAPE1 and NTAPE3)
ELEM	Processed by direct call to subroutine ELEM, data stored on scratch tape (NTAPE3)
LOADS	Processed by direct call to subroutine FGRLDS data stored on scratch tape (NTAPE4)
END	Processed directly in INPUT, terminates input processing
TRANS	Processed directly in INPUT, data eventually stored on scratch tape (NTAPE3)
GRAXES	Processed by direct call to subroutine FRED, data eventually stored on scratch tape (NTAPE3)
MATER	Processed by setting input/output variable ITRACE equal to number of requests and returning to US04A where ITRACE will be tested causing subroutine FMAT to be called; after the MATER section is processed US04A will again call INPUT

TZERO: Processed directly in INPUT, eventually stored on scratch tape (NTAPE3)

CHECK: Processed directly in INPUT, terminates input processing for a case but does not execute data

REPORT: Processed by setting input/output variable IN to the value of NTAPE2 and returning to US04A where IN will be tested causing subroutine REFORM to be called; after report form input processing is completed US04A will again call INPUT

SYSTEM: Processed directly in INPUT

4. Input Arguments:

NTAPE1: Scratch unit number
 NTAPE2: Scratch unit number
 NTAPE3: Scratch unit number
 NTAPE4: Scratch unit number
 ITAPE1: Existing material library unit number
 JTAPE1: Revised or new material library unit number
 NREFP1: Not used
 NSYS: Total degrees of freedom in application (adjustable dimension)
 IN: Data deck unit number
 IPRINT: System print control
 NPIT1: Scratch input control for report form input
 ITRACE: Material library residence indicator
 NAMIN: Existing material library matrix name
 INSPEC: Existing material library unit number
 NAMOUT: Revised or new material library name
 IOSPEC: Revised or new material library unit number
 NRF: Number of total reference points in application (must be equal to highest point number)
 X,Y,Z: Storage allocated for coordinate data
 T: Storage allocated for grid point temperatures
 P: Storage allocated for grid point pressures
 TGRA: Storage allocated for grid point axes transformation matrices
 IZR: Not used
 LIST: Storage allocated for boundary conditions
 DISPL: Storage allocated for input displacements
 LNOD: Not used
 NZEL: Not used
 PCOL: Storage allocated for grid point loads

5. Output Arguments:

ICALC: Execution indicator

if END card read, ICALC is set to 1 and US04A will relinquish control to US04B for matrix generation

if CHECK card read, ICALC is set to zero and subroutine US04A will set controls to return to the Format Monitor (execution of data is suppressed)

ITRACE: Material request indicator

if ITRACE is not equal to zero upon exit from INPUT then US04A will call FMAT

IN: Report form input preprocessor indicator

if IN is equal to NTAPE2 upon exit from INPUT then US04A will call REFORM

6. Error Returns: If any errors are detected then INPUT will set ERROR to .TRUE. and return.

7. Calling Sequence:

CALL INPUT (X, Y, Z, T, P, TGRA, IZR, LIST, DISPL, LNOD, NZEL, PCOL, ITRACE, ICALC, NTAPE1, NTAPE2, NTAPE3, NTAPE4, ITAPE1, JTAPE1, NREFP1, NSYS, IN, IPRINT, NMD, NPIT1, ERROR, NAMIN, INSPEC, NAMOUT, IOSPEC, NRF)

8. Input Tape:

ITAPE1 - Contains existing material library
JTAPE1 - Contains revised or new material library

9. Output Tapes: None

10. Scratch Tapes:

NTAPE1 - Temporary storage for structure control information including system orders, boundary conditions and system print operations
NTAPE2 - Scratch unit used when rewriting NTAPE3 for grid point axes data storage
NTAPE3 - Storage for interpreted element input
NTAPE4 - Storage for input grid point load conditions

11. Storage Required: Total storage is 50718 (2617₁₀).

12. Subroutine User: US04A

13. Subroutines Required:

BOUND
ELEM
FGRLDS
FRED
RECL

14. Remarks: None

1. Subroutine Name: FRED

2. Purpose: To compute transformation matrices when input for GRAXES is encountered.

3. Equations and Procedures:

$$\begin{Bmatrix} u^1 \\ v^1 \\ w^1 \end{Bmatrix} = [T] \begin{Bmatrix} u \\ v \\ w \end{Bmatrix}$$

where (1) u, v, w , are the displacements in the global x, y, z system

(2) u^1, v^1, w^1 are the displacements in the new x^1, y^1, z^1 system

(3) $[T]$ contains the direction cosines

4. Input Arguments:

X :X coordinates of plane defined by 3 pts.
Y :Y coordinates of plane defined by 3 pts.
Z :Z coordinates of plane defined by 3 pts.
KID :See Remarks
L :Point 1 of Plane
M :Point 2 of Plane
N :Point 3 of Plane

5. Output Arguments: TRANSC - transformation matrix $[T]$

6. Error Returns:

- (1) If points 1 and 2 have same coordinates, no plane defined.
- (2) If point 3 lies on the line connecting points 1 and 2, there is no plane defined.

7. Calling Sequence:

CALL FRED (X, Y, Z, TRANSC, KID, L, M, N)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: X(1), Y(1), Z(1), TRANSC (3,3)

12. Subroutine User: INPUT

13. Subroutines Required: None

14. Remarks:

1. Since 3 points define a plane, KID may be

- (a) 0 when the 1st 2 points define the x' axis
- (b) 1 when the 1st 2 points define the y' axis
- (c) 2 when the 1st 2 points define the z' axis

The direction cosines are first computed for points 1 and 2 defining the x' axis. If KID is $\neq 0$, then the direction cosines are rearranged to give the respective notation described above.

2. In spite of error returns indicated, analysis does not terminate.

1. Subroutine Name: BOUND
2. Purpose: Read and process boundary condition data and input displacement data
3. Equations and Procedures: The boundary conditions are read for each point input and the data is stored in the array LIST to be later written on scratch tape NTAP1 by subroutine INPUT. Omitted points are constrained for all degrees of freedom. Only unconstrained degrees of freedom are stored in LIST, giving LIST a length equal to the actual degrees of freedom for which solution will be obtained (NMDB). For each degree of freedom for which a solution is desired, its appropriate total system degree of freedom location, which is $NTD \cdot (IN - 1) + L$, where NTD is the number of degrees of freedom per point, IN is the point number and L is the subject degree of freedom for that point number, is placed in the next available position in LIST. The same procedure is followed for input displacements, which are stored in DISPL.
4. Input Arguments:

IVEC	-	Not used
NDIR, NDEG	-	Product equals NTD, number of degrees of freedom per point
NREF	-	Total number of points referenced in application
NREF4	-	Number of points for which boundary conditions have been input
IN	-	Input unit containing boundary condition data
NSYS	-	Total number of degrees of freedom in application
5. Output Arguments:

NMDB	-	Number of degrees of freedom for which solutions are desired
NMDB2	-	Number of degrees of freedom for which displacements have been input
LIST	-	Array containing degree of freedom numbers for which solutions are to be obtained and displacements have been input
DISPL	-	Array containing input displacements
6. Error Returns: None
7. Calling Sequence:

CALL BOUND (IVEC, NDIR, NDEG, NREF, NMDB, NMDB2, LIST, DISPL, NREF4, IN, NSYS)
8. Input Tape:

IN - Unit containing boundary condition and input displacement data

9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 1014_8 (524_{10}).
12. Subroutine User: INPUT
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: ELEM
2. Purpose: Process element input data (input section ELEM)
3. Equations and Procedures: Processing of element input data begins by reading the element definition input for an element and checking the values for errors and inconsistencies. Error messages for subroutine ELEM are exhibited in Appendix III. The information read is then printed on the system output unit. If no errors have been detected then the element definition input is merged with the required system input. Specifically, the following operations are performed for each element to assimilate the required information for generation of element matrices:

the coordinates, temperatures and pressures are extracted and stored for each of the element definition node points;

the grid point axes transformation matrices are initialized as identity matrices and stored for each of the element definition node points;

the interpolation temperature for material properties is read or calculated dependent upon input, the material library is searched to locate the requested material, the interpolation is performed and the results stored;

the element generation print control is stored;

the boundary conditions for the degrees of freedom referenced by the element defining node points are extracted from the system boundary condition list and stored;

the input displacements, if any, for the degrees of freedom referenced by the element defining node points are extracted from the system input displacement list and stored;

the pre-strains and pre-stresses, if input, are read and stored;

the extra element input data, if any, is read and stored and finally, subroutine RECL is called to place all of the above interpreted element data on scratch tape NTAPE3 (see RECL).

4. Input Arguments:

NELEM:	Number of elements
X,Y,Z:	Arrays containing coordinates of system grid points
T,P:	Arrays containing temperatures and pressures respectively for system grid points
IVEC:	Not used
LIST:	Array containing boundary condition information for system grid points
NMDB2:	Number of entries in array LIST
NDIR,NDEG:	Product equals number of degrees of freedom per grid point
IG:	Maximum number of element defining points possible for an element
NMDB:	Number of system degrees of freedom for which solutions are desired
DISPL:	Array containing input displacements
LNOD:	Not used
GPAXEL:	Work storage reserved for grid point axes transformation matrices
NUTAPE:	Logical variable indicating that new or revised material library has been generated
TZERO:	Base temperature for application
NUMSEQ:	Material library sequence number
XEL,YEL, ZEL:	Work storage reserved for extracting coordinates for element definition node points
TEL,PEL:	Work storage reserved for extracting temperatures and pressures for element definition node points
LISTEL:	Work storage for extracting boundary condition information for element definition node points
NODES:	Array containing element definition node point numbers
DISPEL:	Work storage reserved for extracting input displacements for element definition node points
PCOLEL:	Not used
MAT:	Work storage reserved for interpolated material properties, element print control, mass density and TZERO
EPSIO:	Work storage reserved for pre-strain load vector
SO:	Work storage reserved for pre-stress load vector
EXTRA:	Work storage area reserved for extra element input
IN:	Element data input unit number
NREFP:	Number of input system grid points
ITAPEL:	Existing material library unit number
JTAPE:	Not used

JTAPE1: New or revised material library unit number
NTAPE3: Scratch unit number
NAMIN: Name of existing material library
INSPEC: Same as ITAPE1
NAMOUT: Name of new or revised material library
IOSPEC: Same as JTAPE1

5. Output Arguments:

IFLAG: Error indicator
NNORD: Summation of element degrees of freedom
NNRSEL: Summation of element stress orders
NORDM: Maximum element degrees of freedom for this application
NOINKM: Maximum number of storages for element stiffness matrix for this application
NRSELM: Maximum element stress order for this application

6. Error Returns: If an error is encountered then IFLAG is set to minus one and control is returned to the calling program.

7. Calling Sequence:

CALL ELEM (NELEM, X, Y, Z, T, P, IVEC, LIST, NMDB2, NDIR, NDEG, IG, NMDB, DISPL, LNOD, GPAXEL, NUTAPE, TZERO, IFLAG, NUMSEQ, KEL, YEL, ZEL, TEL, PEL, LISTEL, NODES, DISPEL, PCOLEL, MAT, EPSIO, SO, EXTRA, IN, NREFF, NNORD, NNRSEL, NORDM, NOINKM, NRSELM, ITAPE1, JTAPE, JTAPE1, NTAPE3, NAMIN, INSPEC, NAMOUT, IOSPEC)

8. Input Tape:

IN - Contains element input data

9. Output Tapes:

NTAPE3 - Contains interpreted element input

10. Scratch Tapes: None

11. Storage Required: Total storage is 74078 (3847₁₀).

12. Subroutine User: INPUT

13. Subroutines Required:

MATCH
EUTL3
LAG
RECL

14. Remarks: In calculating the interpolated material properties, if the requested material and the interpolation temperature of the present element being processed are the same as the previous element then the results calculated for the previous element are used and no searching or interpolation is done; if the requested material is in core but the interpolation temperature is different then just the searching is eliminated.

1. Subroutine Name: MATCH
2. Purpose: Compare a material number and its interpolation temperature to the material number and interpolation temperature last referenced in order to determine if a search of the material library tape and/or interpolation is necessary.
3. Equations and Procedures: The material number, TAG1, is compared to the material number now residing in core, NSAVE1. If they do not match, then they are tested again to see if they differ only by an asterisk in the first position. If they still do not match then control is returned to the calling program at the statement following the CALL MATCH statement. If a match was obtained while testing for an asterisk then STAR is set to TRUE. Once a match has been obtained for the material number, the following procedure is followed:

If IELEM equals one then control is returned to the statement number replacing the first asterisk since interpolation must be done for the first element. If IELEM is not one then a check is made to see if a search of the material library was in progress to find this material number. If this is the case then control is returned to the calling program at the statement number replacing the first asterisk since this material table has just been placed in core and interpolation will be necessary. If a search was not in progress then TEMP is compared to SAVTEM. If they are equal then interpolation of the material table has already been calculated and control is returned to the calling program at the statement number replacing the second asterisk. If TEMP does not equal SAVTEM then control returns through the first asterisk in order to perform the interpolation.

4. Input Arguments:

TAG1 : Material number desired
NSAVE1 : Material number now residing in core
TEMP : Interpolation temperature desired
SAVTEM : Last interpolation temperature processed
NDIFF : Constant used to determine if asterisk is present in material number
IELEM : Element number
SEARCH : Logical variable indicating if a search of the material library is in progress

, : Non-standard returns to calling program
(See 7. Calling Sequence)

5. Output Arguments:

STAR : Logical variable indicating presence of asterisk
in material number.

6. Error Returns: None

7. Calling Sequence: CALL MATCH (TAG1, NSAVE1, TEMP, SAVTEM,
NDIFF, STAR, IELEM, SEARCH, *,*)

Where the asterisks are statement numbers, preceded by a dollar sign (\$), that MATCH will return control to in the calling program. Control will pass to the statement number replacing the first asterisk if TAG1 matches NSAVE1 but TEMP does not match SAVTEM (i.e. the material is the same but the interpolation temperatures differ). Control will pass to the statement number replacing the second asterisk if TAG1 matches NSAVE1 and TEMP matches SAVTEM (i.e. the material is the same as the last material referenced and the interpolation temperatures are also the same). If TAG1 does not match NSAVE1 then control is returned to the calling program at the statement following the CALL MATCH statement.

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total Storage = $126_8 = 86_{10}$

12. Subroutine User: ELEM

13. Subroutines Required: None

14. Remarks: None

1. Subroutine Name: LAG
2. Purpose: Linear interpolation routine for material properties
3. Equations and Procedures:
$$ZAPX = \frac{X(I) Y(I-1) - X(I-1) Y(I) + P(Y(I) - Y(I-1))}{X(I) - X(I-1)}$$
4. Input: P - temperature at which material properties will be interpolated
K - number of pairs of coordinates
X - X coordinate
Y - Y coordinate
5. Output; ZAPX - value of the material property being interpolated
6. Error Returns: None
7. Calling Sequence: CALL LAG (P, ZAPX, K, X, Y)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: None
12. Subroutine User: ELEM
13. Subroutines Required: None
14. Remarks: If there is only one X-Y pair, ZAPX will be set equal to Y.

1. Subroutine Name: FGRLDS
2. Purpose: Read and print grid point loads data
3. Equations and Procedures: System input is read from NTAPE4 and includes LIST which is an array containing row numbers of degrees of freedom which are to be retained in the reduced load column. Grid point loads are read for each input point and printed. If grid point axis transformations are present, this transformation is applied. The assembled PCOL is stored on tape NTAPE4, followed by the reduced PCOL. This process is repeated for each load condition.
4. Input Arguments:
 - NL :Number of grid point load conditions
 - TGRA :Grid point axes transformation matrices
 - NØGPA :Number of grid point axes transformations
 - LIST :Reduction array
 - IT :System input tape number
 - NTAPE1:Input tape number
 - NTAPE4:Output tape number
5. Output Arguments: PCOL - Loads Column
6. Error Returns: None
7. Calling Sequence:


```
CALL FGRLDS (NL, TGRA, NØGPA, LIST, IT, PCOL, NTAPE1,
              NTAPE4, NSYS)
```
8. Input Tapes: NTAPE1: Record 1 - not used
Record 2 - NMDB1, NMDB, LIST
9. Output Tapes: NTAPE4: Record 1 - N1, NMDB1, NMDB
Record 2 - PCOL (assembled)
Record 3 - PCOL (reduced)
Repeat Record 2 and 3 for each load condition
10. Scratch Tapes: None
11. Storage Required:
 - LIST (NSYS)
 - ELØAD (12)
 - PCOL (NSYS)
 - COL (3)
 - ISAVE (3)
 - TGRA (3, 3, NREFF)

- 12. Subroutine User: INPUT
- 13. Subroutines Required: None
- 14. Remarks: None

1. Subroutine Name: FMAT

2. Purpose:

- (a) Generate material library tape
- (b) Update material library tape
- (c) Print material library information

3. Equations and Procedures:

Subroutine FMAT operates in three distinct phases.

First, a test is made on NM. If NM is positive, then it is assumed that this is an update run and the original material library is read into PRØPER from ITAPE1. Each table in the library is placed in PRØPER in a block of length NTØT, where NTØT is computed as the necessary storage needed. If NM is zero, it is considered an error condition and a message is printed and control is returned to the calling program. If NM is negative, then it is assumed that this is a generation of a new material library tape and the section which reads the original material library tape is skipped.

The second phase consists of processing the requests. The requests are controlled by an input code read into location D. The legal input codes are:

- (1) I : add or revise isotropic material
- (2) Ø : add or revise orthotropic material table
- (3) PI : add or revise plastic isotropic material table
- (4) PØ : add or revise plastic orthotropic material table
- (5) P : add plastic section to existing material table
- (6) ØUT : delete material table with correct lock code
- (7) ALL : print entire material library
- (8) ØEE : print material table
- (9) ØUM : print summary of material library
- (10) /*/ : print lock code for material table
- (11) ZAP : delete material table regardless of lock code

If NM was negative, then the only allowable codes are I, $\emptyset$, PI and P $\emptyset$ and the requests are processed and placed into the array PR $\emptyset$ PER starting from the beginning and ending at NT $\emptyset$ L. If NM was positive, then the material number is checked against the materials in PROPER to see if it already existed in the original library. If no match is obtained, then the material is added at the next open block in PR $\emptyset$ PER and NT $\emptyset$ L is updated accordingly. If a match occurred, then the revised table will be placed in same position as the original table. If the locations for the material is greater or lesser than before, the remaining contents of PROPER, i.e. those tables after the one in question, are shifted down or up respectively. If the request is of the type that will alter or delete the original table, then the lock code (TAG2) must match the lock code of the original table, otherwise an error condition is encountered and control returns to the calling program. Once it has been decided where the table is to be placed, then the table is read into PR $\emptyset$ PER by material temperature points and plastic temperature points. The material properties are as follows:

- E - Young's Modulus
- ν - Poisson's Ratio
- α - Coefficients of Thermal Expansion
- G - Rigidity Modulus

For an input code of I or PI only E, ν , and α are read and G is computed from $E/2(1+\nu)$ for each material temperature point. For an input code of $\emptyset$ or P $\emptyset$, then E_x , E_y , E_z , ν_{xy} , ν_{yz} , ν_{zx} , α_x , α_y , α_z , G_{xy} , G_{yz} and G_{zx} are read for each material temperature point. If the input code contains a P, then for each plastic temperature point the following data is read:

- N - exponent of stress-strain function assumption
- K - scalar of stress-strain function assumption
- X - nondimensionalizing factor for
- Y - " "
- Z - " "
- R - " "
- S - " "
- T - " "

The procedure for input codes 6-11 is as follows:

- ØUT - If the material is not located in PRØPER, then a message is printed to this effect and the request is ignored. If the material is located and the lock codes do not match, then a message is printed and the request ignored. If the material is located and the lock codes match, then the deletion occurs when the remaining contents of PRØPER are merely shifted up over the deleted material.
- ALL - A flag (WRTALL) is set for phase three and control passes to the next request.
- SEE - If the material is not located, a message is printed and the request ignored. If the material is located, the table is printed and control passes to the next request.
- SUM - All the tables in PRØPER are scanned and the following information is printed for each table:
 - Material Number (TAG1)
 - Material Identification (MIDENT)
 - Analysis Capability (derived from I, Ø, PI, PØ)
 - Number of Material Temperature Points (NP1)
 - Number of Plastic Temperature Points (NP2)
 - Temperature Range of Material Table
 - Temperature Range of Plastic Table
- /*/ - If the material is located, the lock code is printed. If the material is not located, the request is ignored.
- ZAP - If the material is not located, the request is ignored. If the material is located, it is deleted regardless of lock code.

Phase two ends when all of the requests have been processed.

Phase three consists of writing the new or updated material library on JTAPE1 and printing the entire tape if it has been requested. Writing of the tape and a print of the entire material library, if requested, are done in a parallel processing manner; i.e., a table is written on tape and then printed, if requested. Either process may be done separately or together depending upon the requests received. Finally, if a tape has been written, a summary is printed.

4. Input Arguments:

NM : Number of Requests
 MATTAP : Code Controlling Selection of Input and Output Tapes
 IN : Input Tape Unit
 TABMAT : Material Properties Work Storage Area
 TABPLA : Plastic Properties Work Storage Area
 PRØPER : Material Library Work Storage Area
 NWØRK : Number of Available Work Storages
 ITAPE1 : Input Material Library Tape Unit
 JTAPE1 : Output Material Library Tape Unit
 NAMØUT : Array Containing Output Material Library Name
 NAMIN : Array Containing Input Material Library Name

5. Output Arguments:

MATTAP : Code signifying error condition has been encountered, if $MATTAP \geq 0$, then no error has been encountered, if $MATTAP < 0$, then error condition exists.

6. Error Returns:

Message	Action Taken
(1) Value of Young's Modulus (E) ≤ 1.0	RETURN
(2) Value of Poisson's Ratio < -1.0 or > 1.0	RETURN
(3) Value of thermal expansion coefficient (α) < -1.0 or > 1.0	RETURN
(4) Value of Rigidity Modulus (G) ≤ 1.0	RETURN
(5) Value of mass density is negative	RETURN
(6) Lock codes do not match for revision	RETURN
(7) Lock codes do not match for deletion	IGNORE REQUEST
(8) Capacity of material library exceeded	RETURN
(9) Number of material or plastic temperature points > 9	RETURN
(10) Attempt to delete nonexistent material	IGNORE REQUEST
(11) Attempt to input plastic data only for nonexistent material	IGNORE REQUEST
(12) Unrecognizable input code	RETURN
(13) Request to print nonexistent material	IGNORE REQUEST
(14) Number of requests is zero	RETURN

7. Calling Sequence:

Call FMAT (NM, MATTAP, IN, TABMAT, TABPLA, PRØPER, NWØRK,
ITAPE1, JTAPE1, NAMØUT, NAMIN)

8. Input Tapes

9. Output Tapes

Input and output tapes are identical with respect to information contained and record format. Records are as follows from the matrix header to the matrix trailer:

Format Matrix Header Record

Record number 1 - ICØL, KØDE, IWØRDS, NUMTAB, NUMSEQ

Record numbers 2 to NUMTAB+1 - ICØL, KØDE, IWØRDS, NTØT,
D, TAG1, TAG2, NP1, NP2,
DENSTY, MIDENT, ((TABMAT
(I,J), J=1, NMAT),
I=1, NP1), ((TABPLA(I,J),
J=1, NPLA), I=1, NP2)

Format Matrix Trailer Record

where ICOL : Dummy Variable
KØDE : Dummy Variable
IWØRDS : Number of Words Remaining in Record
NUMTAB : Number of Material Tables in Library
NUMSEQ : Sequence Number of Library
NTØT : Total Number of Words in the Specific
Table
D : Input Code
TAG1 : Material Number
TAG2 : Lock Code
NP1 : Number of Material Temperature Points
NP2 : Number of Plastic Temperature Points
DENSTY : Mass Density
MIDENT : Material Identification (Short
Description or Name)
TABMAT : Material Properties Table
NMAT : Number of Material Properties per
Temperature Point + 1
TABPLA : Plastic Properties Table
NPLA : Number of Plastic Properties per
Temperature Point + 1

10. Scratch Tapes: None

11. Storage Required:

CØM(10), MIDENT(4), G(16), HEADER(20), TAG1(6), NFIX1A(6),
Total Storage = $7262_8 = 3762_{10}$ FL1A(6)

12. Subroutine User: USØ4A

13. Subroutines Required: SHIFT

14. Remarks:

Whenever new or updated material tape is written, all changes and/or additions and a summary of the output tape are printed.

1. Subroutine Name: REFORM
2. Purpose: Control generation of BCD input tape from Report Form Input Sheets
3. Equations & Procedures: Storage is allocated for all variables needed by PHASE1 and PHASE2 combined. All valid input section names are stored by a data statement in array NAMES. Temporary tape storage for input sections which must be merged are assigned to scratch tapes NTAPE1 and NTAPE2. Subroutine PHASE1 is entered to read and store all data. Subroutine PHASE2 is entered to merge and output on INTAPE the data that was read in PHASE1. If a dump has been requested then the contents of INTAPE are printed on the system output unit. Control is then returned to the calling program.
4. Input Arguments:

INTAPE:	Tape unit number on which BCD input data is to be generated
NTAPE1,NTAPE2:	Scratch tape unit numbers
IN:	Input tape unit number
NRFP,NSS,NRF:	Adjustable dimension variables
COORD:	Storage area reserved for grid point coordinates
T:	Storage area reserve for grid point temperatures
P:	Storage area reserved for grid point pressures
IBOUND:	Storage area reserved for grid point boundary conditions
5. Output Arguments:

ERROR:	logical variable indicating error condition.
6. Error Returns: If an error has occurred in PHASE1 or PHASE2 then ERROR is set to .TRUE..
7. Calling Sequence:


```
CALL REFORM (INTAPE, NTAPE1, NTAPE2, IN, NRFP, NSS, NRF,
COORD, T, P, IBOUND, ERROR)
```
8. Input Tapes:

IN	- Scratch tape containing card images of data deck

9. Output Tapes:

INTAPE - BCD tape containing sorted data generated for sub-routine INPUT

10. Scratch Tapes:

NTAPE1 - Temporary storage for grid point axes input, initial displacement input and element definition input

NTAPE2 - Temporary storage area for grid point loads input, prescribed displacement input and special element input

11. Storage Required: Total storage = $3005_8 = 1541_{10}$

12. Subroutine User: US04A

13. Subroutines Required:

PHASE1
PHASE2

14. Remarks: None

1. Subroutine Name: PHASE1
2. Purpose: Read, sort and store temporarily, all report form input data.
3. Equations & Procedures: First, all core storage areas are initialized with either blanks or zeroes. The following core storage areas are initialized with blanks: IBOUND, COORD, BM, LM, INM, PRM, EM and ERRMOD. The following core storage areas are initialized with zeroes: P, T, MEMORY, TM and PM.

Reading of input is controlled entirely by label cards for each input section. Correlation between label codes and input sections is as follows:

Code	Input Section
TITLE	Title cards
COORD	Grid point coordinates
TEMP	Grid point temperatures
PRESS	Grid point pressures
BOUND	Grid point boundary conditions
MATER	Material library requests
LOADS	Grid point external loads
GRAXES	Grid point axes (matrices generated)
TRANS	Grid point axes (matrices input)
INITA	Grid point initial displacements
PRDISP	Grid point prescribed displacements
ELEM	Element definition data
EXTERN	Special element data
INPUT	Master input control
PRINT	Print controls
CALC	Calculation controls
END	End card
CHECK	Check card
SYSTEM	System control information

After initialization, the data may be read from IN. The only restriction placed upon order of input sections is that SYSTEM may only be preceded by TITLE, MATER and/or INPUT.

The procedure for a typical input section is as follows:

- (1) Subroutine LATCH is called to determine the identity of the input section.
- (2) Control is transferred to the corresponding section of PHASE1 that will read and store the data. This step is accomplished either directly in PHASE1 itself or by a call to FORMIN.

- (3) Data storing for a section terminates upon reading of a section label card which differs from the section being read.

Upon reading a CHECK or END card, PHASE1 returns control to the calling program.

4. Input Arguments:

NAMES: Array containing valid input section labels
INTAPE: Tape unit number on which BCD input data is to be generated
LOCATE: Array containing tape unit numbers locating temporary tape storage for input sections. For each entry in NAMES there is a corresponding entry in LOCATE pointing to a temporary storage area. If the entry in LOCATE is a zero then storage is in core. If the entry is non-zero then storage is on the tape number indicated.
NUMCAL: Number of possible solution techniques
NUMNAM: Number of valid input section labels
ICASE: Case number
NDIR: Number of directions per grid point
NEND: Last word of every input section placed on tape
IN: Input tape unit number
NRFP: Adjustable dimensions for COORD, T and P
NRF: Adjustable dimension for IBOUND
DINFO: Not used

5. Output Arguments:

COORD: Array containing grid point coordinates
T: Array containing grid point temperatures
P: Array containing grid point pressure
MEMORY: Array containing indicators which record input sections that have been encountered during processing of data
IBOUND: Array containing grid point boundary conditions
TM: Array containing grid point temperature modal values
PM: Array containing grid point pressure modal values

BM:	Array containing grid point boundary condition modal values
SM:	Array containing grid point load modal values for each load condition
INM:	Grid point initial displacement modal values
PRM:	Grid point prescribed displacement modal values
EM:	Special element input modal values
NLOAD:	Array containing number of points in each load condition
NINITA:	Array containing numbers of points in each initial displacement condition
NPRDIS:	Array containing number of points in each prescribed displacement condition
ICALC:	Array containing solution procedures desired
NREF:	Number of system referenced grid points
NREFP:	Number of input grid points
NTD:	Number of degrees of freedom per grid point
NL:	Number of load conditions
NID:	Number of initial displacement conditions
NPD:	Number of prescribed displacement conditions
NAXES:	Number of grid point axes systems
NELEM:	Number of elements
NM:	Number of requests of the material library tape
NREF4:	Number of input boundary condition grid points
TZERO:	System reference temperature
NREF4C:	Number of boundary condition points read by PHASE1
NREFPC:	Number of grid points read by PHASE1
NELEMC:	Number of elements read by PHASE1
NGRAXC:	Number of grid point axes systems read by PHASE1
NTRANC:	Number of grid point axes transformation matrices read by PHASE1
ERROR:	Error indicator
DUMPT:	Debug dump indicator

6. Error Returns:

Message:	Action Taken:
Unexpected blank label card encountered.	Flush to next recognizable label card and insert check card.
No option has been selected for request number xxx of material library.	Flush to next recognizable label card and insert check card.
More than one option has been selected for request number xxx of material library.	Retain first selection encountered.
Maximum number of load conditions allowed is 100. This problem contains xxx.	Flush to next recognizable label card and insert check card.
Load condition xxx sub-label is incorrect. Program cannot distinguish between load conditions.	Flush to next recognizable label card and insert check card.
Illegal modal card encountered. Card will be ignored.	Self-explanatory
Due to previously encountered error condition this section is being skipped. Program will flush data deck until next recognizable section is encountered.	Self-explanatory
Unrecognizable input section.	Flush to next recognizable label card and insert check card.
Due to above error message this section will be omitted and check card inserted.	Self-explanatory

7. Calling Sequence:

CALL PHASE1 (COORD, T, P, MEMORY, IBOUND, NAMES, TM, PM, BM, SM, INM, PRM, EM, NLOAD, NINITA, NPRDIS, ICALC, NREF, NREFP, NTD, NL, NID, NPD, NAXES, NELEM, NM, NREF4, TZERO, INTAPE, LOCATE, NUMCAL, NUMNAM, ICASE, NDIR, NEND, NREF4C, NREFPC, NELEMC, NGRAXC, NTRANC, IN, NRFP, NRF, ERROR, DUMPT, DINFO)

8. Input Tapes:

IN - BCD tape containing card images of data deck

9. Output Tapes:

- NTAPE1 - Temporary storage for grid point axes input, initial displacement input, and element definition input
- NTAPE2 - Temporary storage for grid point loads input, prescribed displacement input and special element input
- INTAPE - TITLE, MATER, PRINT sections are output if they were present.

10. Scratch Tapes: None

11. Storage Required: Total storage = $6270_8 = 3256_{10}$

12. Subroutine User: REFORM

13. Subroutines Required:

LATCH
FORMIN

14. Remarks: None

1. Subroutine Name: LATCH
2. Purpose: Compare a six character name to the recognizable list of input section names for Report Form Input.
3. Equations and Procedures: The six character name LABEL is compared to each of the legal input section names (array NAMES). If a match is found then LEADER is set to the position number in NAMES which contained the matching name. If no match is found then LEADER is set equal to one plus the number of legal section names.
4. Input Arguments:
LABEL - name to be matched
NUMNAM - number of valid input section names
NAMES - array containing valid input section names
5. Output Arguments:
LEADER - position number in NAMES of input section name which matches LABEL

If no match was found then LEADER is set equal to
NUMNAM + 1
6. Error Returns: None
7. Calling Sequence: CALL LATCH (LABEL, LEADER, NUMNAM, NAMES)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required) TOTAL STORAGE = 638 = 51₁₀
12. Subroutine User: PHASE1
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: FORMIN
2. Purpose: Read and store on tape or in core all table form input to Phase One of Report Form Input Preprocessor.
3. Equations and Procedures: The decision to store data on tape or in core is determined by examining the input variable NTAPE. If NTAPE is less than or equal to zero then the data is stored in core, otherwise the data is stored on the unit specified by NTAPE. Any modal values read are always stored in core.
4. Input Arguments:

LEADER	: Index number referring to input section being processed
MEMORY	: Not used
NAMES	: Array containing legal input section labels
NTAPE	: Storage indicator, if#0 then NTAPE contains unit number for external storage
AMODAL	: Storage reserved for modal values read, if any
MODAL	: Modal card label
NUMBER	: Number of input values to be read per card
REPEAT	: Logical variable indicating legality of repeat option
FMT1-5	: Input formats
MSG1-3	: Error message formats
WARN	: Error message warning flag
FATAL	: Error message fatal flag
NCARD	: Number of input cards per table entry
CORE	: Core storage area if data is to remain in core
NR, NC	: Adjustable dimensions of CORE
LABSUB	: Sub-label for multiple condition input sections
IN	: Unit number containing input data

5. Output Arguments:

LABEL	: Input section label encountered which was different from input section label now being processed
KOUNT	: Number of input table entries read
NERROR	: Error indicator
NCOND	: Condition number for encountered sub-label
SCALAR	: Constant for encountered sub-label

6. Error Returns: Error conditions are indicated in NERROR as follows:

If NERROR equals zero, then no error has occurred
If NERROR is less than zero, then a sub-label has been encountered
If NERROR is greater than zero, then a fatal error has occurred and an appropriate message will be printed
7. Calling Sequence: Call FORMIN

(LEADER, MEMORY, NAMES, LABEL, KOUNT, NTAPE, AMODAL, MODAL, NUMBER, REPEAT, FMT1, FMT2, FMT3, FMT4, FMT5, MSG1, MSG2, MSG3, WARN, FATAL, NERROR, NCARD, CORE, NR, NC, LABSUB, NCOND, SCALAR, IN)
8. Input Tape: IN contains input data
9. Output Tape: If NTAPE is greater than zero then it will contain the stored input, otherwise there is no output tape.
10. Scratch Tapes: None
11. Storage Required: Total storage is $751_8(489_{10})$.
12. Subroutine User: PHASE1
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PHASE2
2. Purpose: Merge, order and output form input data stored by PHASE1.
3. Equations and Procedures: The input sections stored by PHASE1 are detected by examining the array MEMORY. The exact procedure is to check the MEMORY array in the order required for output and if the MEMORY value for that section is greater than zero then output that section's stored data; otherwise continue to the next section. The order in which the stored input sections are output, if present, and the sections that they are to be merged with is as follows:

Input Section Generated from Report Form Input Sections

NREF	SYSTEM
TZERO	SYSTEM
GRID	COORD, TEMP, PRESS
BOUNDS	BOUND, CALC, INITA, PRDISP
ELEM	ELEM, EXTERN
TRANS	TRANS
GRAXES	GRAXES
LOADS	LOADS
END	END
CHECK	CHECK

4. Input Arguments:

COORD	: Array containing system grid point coordinates
T	: Array containing grid point temperatures
P	: Array containing grid point pressures
MEMORY	: Array indicating report form input sections read
IBOUND	: Array containing grid point boundary conditions
NAMES	: Array containing legal report form input section names
TM	: Array containing modal values for grid point temperatures
PM	: Array containing modal values for grid point pressures
BM	: Array containing modal values for grid point boundary conditions
SM	: Array containing modal values for grid point load conditions
INM	: Array containing modal values for initially displaced grid points
PRM	: Array containing modal values for pre-scribed displaced grid points

EM	: Array containing modal values for special element input
NLOAD	: Number of loaded grid points per load condition
NINITA	: Number of initial displacement conditions
NPRDIS	: Number of prescribed displacement conditions
ICALC	: Array containing solution codes
NREF	: Number of grid points in system
NREFP	: Number of input grid points
NTD	: Number of degrees of freedom per grid point
NL	: Number of load conditions
NID	: Number of initially displaced grid points
NPD	: Number of prescribed displaced grid points
NAXES	: Number of grid point axes transformation systems
NELEM	: Number of elements
NM	: Number of requests of material library
NREF4	: Number of input boundary condition points
TZERO	: System base temperature
INTAPE	: Unit on which processed output is to be written
LOCATE	: Array indication storage location of input sections
NUMCAL	: Number of solution codes
NUMNAM	: Number of legal report form input section labels
ICASE	: Not used
NDIR	: Number of directions per grid point
NEND	: Not used
NREF4C	: Number of input boundary condition points actually read
NREFPC	: Number of input grid points actually read
NELEMC	: Number of input elements actually read
NGRAXC	: Number of input grid point axes systems actually read (transformation matrices generated)
NTRANC	: Number of input grid point axes systems actually read (transformation matrices input)
IN	: Not used
NRFP	: Not used
NRF	: Adjustable dimension for COORD, T, P, and IBOUND

5. Output Argument:

ERROR	: Logical variable indicating

6. Error Returns: Error messages are indicated in Appendix. If an error occurs logical variable ERROR is set to TRUE and control is returned to the calling program.
7. Calling Sequence: Call PHASE2
(COORD, T, P, MEMORY, IBOUND, NAMES, TM, PM, BM, SM, INM, PRM, EM, NLOAD, NINITA, NPRDIS, ICALC, NREF, NREFP, NTD, NL, NID, NPD, NAXES, NELEM, NM, NREF4, TZERO, INTAPE, LOCATE, NUMCAL, NUMNAM, ICASE, NDIR, NEND, NREF4C, NREFPC, NELEMC, NGRAXC, NTRANC, IN, NRFP, NRF, ERROR)
8. Input Tapes: The array LOCATE contains the unit number, if any, on which data was stored by subroutine PHASE1.
9. Output Tape: INTAPE contains processed output.
10. Scratch Tapes: None
11. Storage Required: Total storage is $6357_8(3311_{10})$.
12. Subroutine User: REFORM
13. Subroutine Required: OPEN
14. Remarks: None

1. Subroutine Name: OPEN
2. Purpose: Select a unit and then locate the requested input section on that unit
3. Equations and Procedures: The correct unit number is extracted from the array LOCATE. The unit is then searched for the requested input section. Searching starts from the present position of the unit and allows the end of the unit's extent to be reached twice before the search is abandoned.
4. Input Arguments:
 - LEADER : Identification number of input section being processed
 - NAMES : Array containing valid labels
 - LOCATE : Array containing corresponding logical units for valid labels
 - * : Non-standard return for error condition
5. Output Arguments:
 - NTAPE : Unit containing requested input section
6. Error Returns: If the requested input section is not located on the selected unit the non-standard return is used.
7. Calling Sequence: Call OPEN
(LEADER, NAMES, LOCATE, NTAPE, \$XXXXX) where XXXXX is the statement number to which control is returned in the calling program if an error occurs.
8. Input Tapes: The array LOCATE contains the logical unit numbers which may be input tapes.
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 136_8 (94_{10}).
12. Subroutine User: PHASE2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: FLOADS
2. Purpose: To generate a matrix of external grid point loads which is acceptable to Format.
3. Equations and Procedures: A grid point load matrix, PCOL, is read from NTAPE4 for each load condition. It is then converted into compressed format and stored on tape IOSPEC.

The matrix dimensions are NSYS x NL, where NSYS is the size of the total assembled load column and NL is the number of grid point load conditions.
4. Input Arguments:

NSYS - Size of total assembled load column
NAMOUT- Array containing output matrix name for load matrix
IOSPEC- Output tape unit number for loads matrix
NTAPE4- Input tape unit number containing loads matrix
PCOL - Core storage area for loads matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: CALL FLOADS (NSYS, NAMOUT, IOSPEC, NTAPE4, PCOL)
8. Input Tapes: NTAPE4
9. Output Tapes: IOSPEC
10. Scratch Tapes: None
11. Storage Required: PCOL (NSYS)
12. Subroutine User - US04A
13. Subroutines Required - None
14. Remarks: None

1. Subroutine Name: FTR
2. Purpose: To generate a matrix which will transform another matrix from full system coordinates to "reduced" system, i.e. boundary condition constrained.
3. Equations and Procedures: The matrix TR is of order NMDB X NSYS such that if $J = \text{LIST}(I)$, then the element $\text{TR}(I, J) = 1.0$. LIST contains the row numbers of the full system which are to be retained in the reduced matrix. Only fixed bounds are reduced out as indicated by $\text{KODE} = 0$ in input data bounds.

Each column is generated and stored on tape as defined by FORMAT. Each column record consists of: J, 1, 2, 1.0, where $J = \text{LIST}(i)$.
4. Input Arguments: NMDB - order of reduced matrix
NSYS - order of full system
NAMØUT - matrix name of TR
IØSPEC - matrix output tape for TR
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:

CALL FTR (NSYS, LIST, NTAPE1, NAMØUT, IØSPEC)
8. Input Tape: NTAPE1
Record #1 COM1 (not required)
Record #2 NMDB1, NMDB, (LIST (I), I=1, NMDB)
9. Output Tapes: IØSPEC - Format Output Tape Number
10. Scratch Tapes: None
11. Storage Required: LIST (NSYS)
12. Subroutine User: USØ4A
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: US04B
2. Purpose: Control Phase Two and Phase Three operations (element matrix generation and element matrix output, respectively).
3. Equations and Procedures: System control information is extracted from the array KNMD. Scratch units are assigned from the array ISSPEC. If input displacements are present then subroutine DEFLEX is called to record the input displacements on scratch unit NTAPE4. If the interpreted input matrix position is non-blank then subroutine ININT is called to generate input tapes NTAPE1 and NTAPE3. Subroutine FELEM is called to control the generation of the element matrices. And, finally, subroutine OUTMAT is called to place the generated matrices into the Format System.
4. Input Arguments:

NUMOT	: Number of output matrices
NAMOUT	: Array containing names of output matrices
IOSPEC	: Array containing unit specifications for output matrices
NUMIN	: Number of input matrices
NAMIN	: Array containing names of input matrices
INSPEC	: Array containing unit specification for input matrices
NUMSR	: Number of available scratch units
ISSPEC	: Array containing scratch unit specifications
NUMSC	: Number of scalars
SCALAR	: Array containing scalars
NWORKR	: Number of available storages in work area
WORK	: Work area
IPRINT	: System print control
KNMD	: System control information
MASTER	: Array containing input/output cross-checking codes
NUMAST	: Length of MASTER
NUMK	: Length of KNMD
5. Output Arguments:

ERROR	: Logical variable indicating error condition
6. Error Return: If an error is detected in element matrix generation or in element matrix output then ERROR is set to TRUE and control is returned to the calling program.

7. Calling Sequence: Call US04B

(NUMOT, NAMOUT, IOSPEC, NUMIN, NAMIN, INSPEC, NUMSR,
ISSPEC, NUMSC, SCALAR, ERROR, NWORKR, WORK, IPRINT,
KNMD, MASTER, NUMAST, NUMK)

8. Input Tapes:

NTAPE1 : Contains system control information
NTAPE3 : Contains interpreted element input

9. Output Tapes:

IOSPEC(1,6) : Reserved for assembly transformation
matrix
IOSPEC(1,7) : Reserved for element stiffness matrices
IOSPEC(1,8) : Reserved for element load matrices
IOSPEC(1,9) : Reserved for element stress matrices
IOSPEC(1,10) : Reserved for element thermal stress
matrices
IOSPEC(1,11) : Reserved for element incremental
stiffness matrices
IOSPEC(1,12) : Reserved for element mass matrices

10. Scratch Tape:

NTAPE2 : Contains element generated matrices
in compact form
NTAPE4 : Contains input displacements, if
present

11. Storage Required: Total storage is 421_8 (273_{10})

12. Subroutine User: US04

13. Subroutines Required:

NTEST
ININT
DEFLEX
FELEM
OUTMAT

14. Remarks: None

1. Subroutine Name: FELEM
2. Purpose: Set element matrix generation controls and initiate matrix generation.
3. Equations and Procedures: Logical unit definitions are assigned to their structural system functions. An array, IWORK, is reserved for storage of generation controls and system information. The generation controls are determined by examining the output matrix names and the system information is retrieved from unit NTAPE1. Subroutine SQUISH is called to compute matrix suppression controls. The number of elements is read from unit NTAPE3 and subroutine ELPLUG, which selects the correct element type, is called for each element.
4. Input Arguments:
 - KP : Not used
 - NTAPE1 : Logical unit containing system control information
 - NTAPE2 : Logical unit reserved for generated element matrices
 - NTAPE3 : Logical unit containing interpreted element input
 - NORDM : Maximum element degrees of freedom
 - NRSELM : Maximum element stress order
 - NOINKM : Maximum storage required for element stiffness matrix
 - NIAM : Maximum storage required for element matrix record on NTAPE2
 - NTAPE4 : Logical unit containing input displacements, if present
5. Output Arguments:
 - ERROR : Logical variable indicating error condition
6. Error Returns: If an error occurs in generation of element matrices then ERROR is set to .TRUE. and control is returned to the calling program.
7. Calling Sequence: Call FELEM
(KP, NTAPE1, NTAPE2, NTAPE3, NORDM, NRSELM, NOINKM, NIAM, ERROR, NTAPE4)
8. Input Tapes:
 - NTAPE1 : Contains system control information
 - NTAPE3 : Contains interpreted element input

9. Output Tapes:

NTAPE2 : Reserved for compact storage of element
generated matrices

10. Scratch Tapes: None

11. Storage Required: Total storage is 521_8 (328_{10})

12. Subroutine User: US04B

13. Subroutines Required: ELPLUG, SQUISH

14. Remarks: None

1. Subroutine Name: ELPLUG
2. Purpose: Select proper element type to generate requested element matrices.
3. Equations and Procedures: Subroutine RECl is called to obtain the interpreted element input. If input displacements were present then the values are retrieved from unit NTAPE4. Included in the interpreted element input is the element type code number (plug number). From this data the proper plug subroutine is called and the requested element matrices are generated. If the plug number is five, six or fourteen the grid point axes transformations are then applied. If the plug number was one, two or seven then grid point axes transformations were applied inside the plug. Subroutines REC3 and REC4 are called to write as external units element control data and the generated element matrices, respectively. Finally, if an element matrix print has been requested then subroutine ELPRT is called to perform the printing.

There is only one exception to the above procedure. If the option to repeat element matrices has been selected (IP = -2), then the plug subroutine is bypassed and element matrices from the previous element are written again by REC3 and REC4.

4. Input Arguments: The input arguments contained in JWORK are:

JWORK(1)-IEL	: Element generation sequence number (IEL = 1,2,3, . . . , NELEM)
JWORK(2)-ITAPE	: Indicator controlling writing of matrices on external unit
JWORK(3)-KK	: Element stiffness matrix suppression control
JWORK(4)-KF	: Element load matrix suppression control
JWORK(5)-KS	: Element stress matrix suppression control
JWORK(6)-KM	: Element mass matrix suppression control
JWORK(7)-KDS	: Not used
JWORK(8)-KDV	: Not used
JWORK(9)-KN	: Element incremental stiffness matrix suppression control
JWORK(11)-NMDB	: Not used
JWORK(12)-NDIR	: Number of directions per grid point
JWORK(13)-NDEG	: Number of solution degrees of freedom per grid point
JWORK(14)-ICONT	: Grid point axes transformation indicator

JWORK(15)-NTAPE2 : Unit number reserved for generated element matrices
 JWORK(16)-NTAPE3 : Unit number containing interpreted element input
 JWORK(18)-ILP : Internal element type code
 JWORK(19)-IPL : Input element type code
 JWORK(20)-NTAPE4 : Unit number containing input displacement, if present
 JWORK(21)-INDISP : Variable indicating presence of input displacements

Other input arguments are:

NUMOT : Number of output matrices
 NAMOUT : Array containing output matrices names

5. Output Arguments: Input and output arguments are contained in the array JWORK. The output arguments contained in JWORK are:

JWORK(10)-NORD : Element degrees of freedom
 JWORK(17)-NIAM : Maximum number of storages required to write a record on unit NTAPE2
 JWORK(20)-NERR : Returning error code,
 if NERR is zero then no error has occurred,
 if NERR is one then element type code number is incorrect,
 if NERR is two then the number of element defining points is incorrect,
 if NERR is three then the special element input is incorrect, and
 if NERR is four then the number of element degrees of freedom is incorrect.

6. Error Returns: If NERROR is not zero upon return from ELPLUG, then an error has occurred.

7. Calling Sequence: Call ELPLUG (JWORK, NUMOT, NAMOUT)

8. Input Tape:

NTAPE3 : Unit containing interpreted element input

9. Output Tape:

NTAPE2 : Unit reserved for generated element matrices

10. Scratch Tapes: None

11. Storage Required: Total storage is 1216_8 (590_{10})

12. Subroutine User: FELEM

13. Subroutines Required:

REC1
PLUG1
PLUG2
PLUG5
PLUG6
PLUG7
PLUG14
AXTRA3
AXTRA2
AXTRA1
REC3
REC4
ELPRT

14. Remarks: Storage for the generated element matrices and work areas required by ELPLUG is allocated by equivalencing into the blank common work area starting at location 1001 and extending to location 6000. Work storage for the various element types is allocated by equivalencing into the blank common work area at location 6001.

1. Subroutine Name: REC3
2. Purpose: Write or read element control information tape records.
3. Equations and Procedures: The decision to read or write the record is determined by examining the input variable IOPT in the following manner:

if $\text{IOPT} \leq 1$ the record is read
if $\text{IOPT} \geq 2$ the record is written
4. Input Arguments: (if $\text{IOPT} \geq 2$)

IOPT: Read/write indicator
K: Fortran logical unit number
NI3: Number of words in record (excluding NI3)
JEL: Element number
IPL: Element type code number (plug number)
NLIST: Element order (number of degrees of freedom per point * number of points)
LISTEL: Vector containing boundary condition information for element
NIA: Not used (set equal to one)
IAKEL: Not used
5. Output Arguments: (if IOPT 1)

Given the proper value of IOPT, all of the above input arguments will be output arguments with the exception of IOPT and K, which are always input arguments.
6. Error Returns: None
7. Calling Sequence:

CALL REC3 (IOPT, K, NI3, JEL, IPL, NLIST, LISTEL, NIA, AKEL)
8. Input Tape: If $\text{IOPT} \leq 1$, then K is an input tape.
9. Output Tape: If $\text{IOPT} \geq 2$, then K is an output tape.
10. Scratch Tape: None
11. Storage Required: Total storage is 2208 (130₁₀).
12. Subroutine User: ELPLUG
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: REC4
2. Purpose: Read or write generated element matrices records.
3. Equations and Procedures: The decision to read or write the record is determined by examining the input variable IOPT in the following manner:

if IOPT $\leq$ 1 then a record is read
if IOPT $\geq$ 2 then a record is written
4. Input Arguments: (when IOPT $\geq$ 2)

IOPT: Read/write indicator
K: Fortran logical unit number
NOINK: Number of storages required for stiffness and incremental stiffness matrices
AKELT: Element stiffness matrix
NORD: Number of storages required for element loads matrix
FTEL: Element loads matrix
NNO: Number of element defining points (node points)
NODES: Grid point numbers defining element
NSEL: Number of storages required for element stress matrix
NRSEL: Number of rows in element stress and thermal stress matrices, also number of storages required for element thermal stress matrix
SEL: Element stress matrix
SZALEL: Element thermal stress matrix
ANEL: Element incremental stiffness matrix
FNEL: Not used
NMASS: Number of storages required for element mass matrix
AMASS: Element mass matrix
NDMPV: Number of storages required for element viscous damping matrix
DAMPV: Element viscous damping matrix
NDMPS: Number of storages required for element structural damping matrix
DAMPS: Element structural damping matrix

5. Output Arguments: (when $IOPT \leq 1$)
NI4 - number of words contained in record (excluding NI4)
All of the above input arguments are output arguments given the correct value of IOPT except for IOPT and K which are always input arguments.
6. Error Returns: None
7. Calling Sequence:
CALL REC4 (IOPT, K, NI4, NOINK, AKELT, NORD, FTEL, NNO, NODES, NSEL, NRSEL, SEL, SZALEL, ANEL, FNEL, NMASS, AMASS, NDMPV, DAMPV, NDMPS, DAMPS)
8. Input Tape: If $IOPT \leq 1$ then K is an input unit.
9. Output Tape: If $IOPT \geq 2$ then K is an output unit.
10. Scratch Tapes: None
11. Storage Required: Total storage is 604_8 (388_{10}).
12. Subroutine User: ELPLUG
13. Subroutines Required: None
14. Remarks: None

1. Subroutine name: MINV
2. Purpose: Invert a matrix.
3. Equations and Procedures: The standard Gauss-Jordan Method is used in which the inverted matrix is stored back on itself.
4. Input Arguments:
 - A: Matrix to be inverted
 - N: Order of matrix
 - D: Determinant of matrix
 - L: Work vector of length N
 - M: Work vector of length N
5. Output Arguments: A - Contains the inverted matrix
6. Error Returns: If $D = 0$, matrix is singular.
7. Calling Sequence: CALL MINV (A, N, D, L, M)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A (1), L (1), M (1)
12. Subroutine User: TRAIC, NEWFT, PLUG1, PTBM, PTBF, MATPR, NEWFT1
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: AXTRA2

2. Purpose: Apply grid point axes transformation by post-multiplication using either the actual transformation matrix or its transpose.

3. Equations and Procedures:

$$[M_{OUT}] = [M_{IN}] [GPA] \quad \text{or} \quad [M_{OUT}] = [M_{IN}] [GPA]^T$$

where: $[M_{IN}]$ is the input element matrix,

$[GPA]$ is the element grid point axes transformation matrix,

$[M_{OUT}]$ is the output transformed element matrix,

$[M_{OUT}]$ is stored in the same location as $[M_{IN}]$, therefore, the input element matrix is lost once the multiplication has been effected. Advantage is taken, during multiplication, of the fact that $[GPA]$ is structured as a set of (3 x 3) or (2 x 2) matrices with main diagonal positions lying on the main diagonal of $[GPA]$.

4. Input Arguments:

GPAXEL : Element grid point axes transformation matrix, $[GPA]$

SEL : Input element matrix $[M_{IN}]$

NROW : Number of rows in SEL

NNO : Number of element node points

NDEG : Number of degrees of freedom

NDIR : Number of directions

IPL : Element plug number

ITRAN : Control code, if ITRAN = 0, then $[M_{OUT}] = [M_{IN}] [GPA]$
if ITRAN = 1, then $[M_{OUT}] = [M_{IN}] [GPA]^T$

5. Output Arguments:

SEL : Output transformed element matrix, $[M_{OUT}]$

6. Error Returns: None

7. Calling Sequence:

CALL AXTRA2 (GPAXEL, SEL, NROW, NNO, NDEG, NDIR, IPL, ITRAN)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total Storage = $346_8 = 230_{10}$
ROW (3)
ISAVE (3)
12. Subroutine User: ELPLUG, PLUG7, PLUG2, CK22, CK11
13. Subroutine Required: None
14. Remarks: The output matrix is stored in the input matrix storage. Grid point axes transformation is not applied to the rotation terms at the mid-points of the quadrilateral thin shell and the triangular thin shell elements.

1. Subroutine Name: MAB
2. Purpose: To evaluate the matrix product $A * B = AN$
3. Equations & Procedures:

$$AN_{nm} = \sum_j A_{nj} * B_{jm}$$
4. Input Arguments:
 - A: Elements of [A] matrix
 - B: Elements of [B] matrix
 - N: Number of rows in [A] matrix
 - L: Number of columns/rows in [A] [B] matrix
 - M: Number of columns in [B] matrix
 - N1,M1: Dimension of [A] matrix
 - N2,M2: Dimension of [B] matrix
5. Output Arguments:
 - AN: The matrix product
6. Error Returns: None
7. Calling Sequence: CALL MAB (A,B,AN,N,L,M,N1,M1,N2,M2)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 - A(1)
 - B(1)
 - AN(1)
12. Subroutine User: Used by many subroutines within the MAGIC program
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: MSB

2. Purpose: To evaluate the matrix product of a symmetric-bottom half matrix and a rectangular matrix

3. Equations & Procedures:

$$AN_{nm} = \sum_e S_{ne} * B_{em}$$

4. Input Arguments:

S: Elements of [S] matrix (symmetric)
B: Elements of [B] matrix
N: Number of rows in the [S], [B], and [AN] matrices (order)
M: Number of columns in the [B] and [AN] matrices (order)
N1 and M1: Dimensions of the [B] and [AN] matrices

5. Output Arguments: AN: Matrix product

6. Error Returns: None

7. Calling Sequence: CALL MSB (S,B,AN,N,M,N1,M1)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required:

B(1)
S(1)
AN(1)

12. Subroutine User: Used by various subroutines within the MAGIC Program.

13. Subroutines Required: None

14. Remarks: [S] is of the form

$$\begin{bmatrix} S_{11} & & & \\ S_{21} & S_{22} & & \\ \vdots & \vdots & \ddots & \\ S_{N1} & S_{N12} & \dots & S_{N1N1} \end{bmatrix}$$

1. Subroutine Name: BCB
2. Purpose: To evaluate the triple product of the transpose of a matrix A, a symmetric matrix S and the A matrix.
3. Equations and Procedures:

$$AN_{mm} = \sum_n \sum_n A_{mn}^T * S_{nn} * A_{nm} \quad (\text{See remark 1})$$
4. Input Arguments:
 A: The elements of the [A] matrix
 SYM: The elements of the [S] matrix (symmetric-bottom half)
 ND,MD: Dimensions of a matrix
 N,M: Order of A matrix
 N1: Number of rows to be deleted in multiplication
 SCAL: Scalar quantity
 IASSY: (see remark 2)
5. Output Arguments:
 AN: Elements of the matrix AN which is the final product
6. Error Returns: None
7. Calling Sequence:
 CALL BCB (A, SYM, AN, ND, MD, N, M, N1, SCAL, IASSY)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A(1), AN(1), SYM(1), ROW(48)
12. Subroutine User: Various routines within MAGIC
13. Subroutines Required: None
14. Remarks:
 1. In the summations, the n's must be replaced by dummy subscripts, running from 1 to n. The dummy must be used (ie. $\sum_{j=1}^n \sum_{r=1}^n$) to ensure proper summing.

2. IASSY controls the summation procedure.

If IASSY = 1, AN will be the sum of the calculated AN and all previous calculations of AN.

If IASSY = 0, AN will be the triple product for this calculation.

1. Subroutine Name: MATB
2. Purpose: Subroutine to evaluate the matrix product of A transpose and B.
3. Equations and Procedures:

$$AN_{nm} = \sum_e A_{en}^T * B_{em}$$
 where
 A_{en}^T is the transpose of A_{ne} .
4. Input Arguments:
 A: elements of [A] matrix
 B: elements of [B] matrix
 N: number of rows in [A] matrix (order)
 L: number of columns in [A] matrix (order)
 M: number of rows in [B] matrix (order)
 N1,M1: dimension of [A] matrix
 N2,M2: dimension of [B] matrix
5. Output Arguments:
 AN: elements of matrix product
6. Error Returns: None
7. Calling Sequence: CALL MATB (A, B, AN, N, L, M, N1, M1, N2, M2)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A(1), B(1), AN(1)
12. Subroutine User: Various subroutines in MAGIC
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: SYMPRT
2. Purpose: To print a symmetric matrix as output
3. Equations and Procedures: Not Applicable
4. Input Arguments:
SYM: Elements of the symmetric matrix
N1: Matrix identification number
N2: Dimension of matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: CALL SYMPRT (SYM, N1, N2)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: SYM(1)
12. Subroutine User: Various subroutines in MAGIC System
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: LØC
2. Purpose: Compute a vector subscript for an element in a matrix of specified storage mode
3. Equations and Procedures: The routine determines the type of matrix and computes the subscript accordingly.
4. Input Arguments:
 - I: Row number of element
 - J: Column number of element
 - N: Number of rows in matrix
 - M: Number of columns in matrix
 - MS: Storage mode of matrix
 - 0 General
 - 1 Symmetric (Upper Half)
 - 2 Diagonal
5. Output Arguments: IR - Resultant vector subscript
6. Error Returns: None
7. Calling Sequence: CALL LØC (I, J, IR, N, M, MS)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: MPRD, TPRD, AXTRA3
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: ELTEST
2. Purpose: Check on input variables (plug number, number of nodes, order of matrix), for a specific element.
3. Equations & Procedures: Logical "IF" statement is used to check equivalence of variables with predefined program constants.
4. Input: IPL & IPL1 - plug number & check constant
NNO & NNO1 - number of nodes & check constant
NORD & NORD1 - order of matrix & check constant
5. Output: NERR (error return)
6. Error Returns: NERR = 0 No error
NERR = 1 Plug number incorrect
NERR = 2 Number of nodes incorrect
NERR = 4 Order of matrix incorrect
7. Calling Sequence: CALL ELTEST (IPL, IPL1, NNO, NNO1, IP, IPL, NORD, NORD1, NERR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: Total storage is 75₈ (61₁₀)
12. Subroutine User: All plugs
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: MPRD
2. Purpose: Multiply two matrices to form a resultant matrix
3. Equations and Procedures:

$$[R] = [A] [B]$$

4. Input Arguments

A: First input matrix
 B: Second input matrix
 N: Number of rows in A matrix
 L: Number of columns in B
 MSA: Control on storage mode of A
 MSB: Control on storage mode of B

} See Remarks

5. Output Arguments: R - Resultant matrix
6. Error Returns: None
7. Calling Sequence: CALL MPRD (A, B, R, N, M, MSA, MSB, L)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A (1), B (1), R (1)
12. Subroutine User: Utility subroutine
13. Subroutines Required: LØC
14. Remarks:

Storage Control of A and B matrix (MSA and MSB)

- 0 - General
- 1 - Symmetric (upper half)
- 2 - Diagonal

1. Subroutine Name: TPRD
2. Purpose: Transpose a matrix and postmultiply by another to form a resultant matrix.
3. Equations and Procedures

$$[R] = [A]^T [B]$$

A is not actually transposed. Instead, elements in matrix A are taken column-wise rather than row-wise for multiplication by B.
4. Input Arguments

A:	First input matrix	
B:	Second input matrix	
N:	Number of rows in A and B	
M:	Number of columns in A and rows in R	
L:	Number of columns in B and rows in R	
MSA:	Control of storage mode of A	} See Remarks
MSB:	Control of storage mode of B	
5. Output Arguments: R - Resultant matrix
6. Error Returns: None
7. Calling Sequence: CALL TPRD (A, B, R, N, M, MSA, MSB, L)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A (1), B (1), R (1)
12. Subroutine User: Utility subroutine
13. Subroutines Required: LOC
14. Remarks

Storage Control of A and B Matrix (MSA and MSB)

0	-	General
1	-	Symmetric (upper half by columns)
2	-	Diagonal

1. Subroutine Name: AI (Function)
2. Purpose: Control operation of the triangular integration package.
3. Equations and Procedures: The integration package will calculate the value of a double definite integral of the form

$$\iint r^p z^q dz dr \quad \left| \begin{array}{c} r_j \\ r_i \end{array} \right| \quad \left| \begin{array}{c} z_{mn} \\ z_{kl} \end{array} \right|$$

The procedure is to call a series of function subprograms dependent upon the values of p and q. The variables in the above integral are represented by the following program variables, which are defined in the input arguments section below:

Integral Variable	Corresponding Program Variable
r	R
z	Z
p	IP
q	IQ
i	I
j	J
k	K
l	L
m	M
n	N

4. Input Arguments:

I : r coordinate subscript of i th element
defining point

J : z coordinate subscript of j th element
defining point

K, L : Slope of element line passing through the
element defining point z_{kl}

M, N : Slope of element line passing through element
defining point z_{mn}

IP : Exponent of r coordinate

IQ : Exponent of z coordinate

- R : Array containing r coordinates of element
defining points
Z : Array containing z coordinates of element
defining points
5. Output Argument:
AI(Function) : Result of performing the indicated
integration
6. Error Return: None
7. Calling Sequence:
AI(I, J, K, L, M, N, IP, IQ, R, Z)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 1014_8 (524_{10}).
12. Subroutine User: TRAIC, DPQINT
13. Subroutines Required:
AM
AK
BINT
F89
FF100
FJAB
F6219
F6211
14. Remarks: None

1. Subroutine Name: BINT
2. Purpose: Perform integration

$$r_i \int_{r_i}^{r_j} r^v (a+br)^w dr$$
3. Equations and Procedures:
Expand $r^v (a+br)^w$ by binomial theorem
and integrate term by term.
4. Input Arguments: I, J, A, B, IV, IW, R, Z
5. Output Arguments: BINT
6. Error Returns: None
7. Calling Sequence: BINT(I, J, A, B, IV, IW, R, Z)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: AI
13. Subroutines Required: COEF, AJ
14. Remarks: None

1. Subroutine name: AK
2. Purpose: Generate slope of line between two points of a triangle
3. Equations and Procedures:
$$AK = [Z(J) - Z(I)] / [R(J) - R(I)]$$
4. Input Arguments: I, J, R, Z
5. Output Arguments: AK
6. Error Returns: None
7. Calling Sequence: AK(I, J, R, Z)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 65₈ (53₁₀).
12. Subroutine User: AI
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: AM
2. Purpose: Generate intercept of line between two points of triangle.
3. Equations and Procedures:
$$AM = [R(J)Z(I) - R(I)Z(J)] / [R(J) - R(I)]$$
4. Input Arguments: I, J, R, Z
5. Output Arguments: AM
6. Error Returns: None
7. Calling sequence: AM (I, J, R, Z)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 73₈ (59₁₀).
12. Subroutine User: AI
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: IFAC
2. Purpose: Compute N factorial
3. Equations and Procedures: $N! = \text{IFAC} = n(n-1)(n-2) \dots (1)$
4. Input Arguments: N
5. Output Arguments: IFAC
6. Error Returns: None
7. Calling Sequence: IFAC(N)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 62_8 (50_{10}).
12. Subroutine User: FF100
F89
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: FJAB (function)
2. Purpose: To generate

$$\int [x^{m-1}/(a + bx)] dx$$
3. Equations and Procedures:

$$F = [(x^m \log (a+bx))/m] - [(b/m) \int (x^n/(a-bx)^n) dx]$$

evaluated at $x = x(I)$
4. Input Arguments: I, A, B, M, N, X
5. Output Argument: FJAB
6. Error Returns: None
7. Calling Sequence: FJAB (I, A, B, M, N, X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 143₈ (99₁₀).
12. Subroutine User: AI
13. Subroutines Required: F89
14. Remarks: None

1. Subroutine Name: F6219 (function)
2. Purpose: To generate integral of

$$\int (\log (a + bx) / (x^m + 1)) \, dx$$
3. Equation and Procedures:

$$F = (- \log (a + bx) / (mx^m)) + \left(\int (b / (m(a + bx) x^m)) \, dx \right)$$
 evaluated at $x = X(I)$
4. Input Arguments: I, A, B, M, N, X
5. Output Arguments: F6219
6. Error Returns: None
7. Calling Sequence: Function F6219 (I, A, B, M, N, X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: AI
13. Subroutines Required: FF100
14. Remarks: None

1. Subroutine Name: F6211
2. Purpose: To generate

$$\int [(\log (A+BX))/X] dx$$
3. Equations and Procedures:

$$F = \log (A) \log (X) + \frac{BX}{A} - \frac{B^2 X^2}{4A^2} + ...$$

evaluated at $X = X(I)$
4. Input Arguments: I, A, B, X
5. Output Arguments: F6211
6. Error Returns: None
7. Calling Sequence: Function F6211 (I, A, B, X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 376₈(254₁₀).
12. Subroutine User: AI
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: AJ (function)
2. Purpose: Generates
for $M + 1 > 0$ $[R(J)^M - R(I)^M] / (M+1)$
for $M + 1 = 0$ $\log [R(J)/R(I)]$
3. Equations and Procedures: None
4. Input Arguments: I, J, R, M
5. Output Arguments: AJ
6. Error Returns: None
7. Calling Sequence: Function AF(I, J, R, M)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is $136_8(94_{10})$.
12. Subroutine User: BINT
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: COEF
2. Purpose: Generate binomial coefficient
3. Equations and Procedures:
$$\text{COEF} = \binom{n}{r} = n^C_r = \frac{n!}{r! (n-r)!}$$

(the combination of n items taken r times)
4. Input Arguments: N,R
5. Output Arguments: COEF
6. Error Returns: None
7. Calling Sequence: COEF (N,R)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 117₈(79₁₀).
12. Subroutine User: BINT
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: F89 (Function)
2. Purpose: To generate integral

$$\int (x^m / (a+bx)^n) dx$$
3. Equations and Procedures:

$$F89 = \frac{1}{b^{m+1}} \left[\sum_{s=0}^m \frac{m! (-a)^s x^{m-n-s+1}}{(m-s)! s! (m-n-s+1)} \right]$$

where $X = a+bx$
evaluated at x
4. Input Arguments: I, A, B, M, N, X
5. Output Arguments: F89
6. Error Returns: None
7. Calling Sequence: F89 (I, A, B, M, N, X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 345_8 (233_{10})
12. Subroutine User: AI
13. Subroutines Required: IFAC
14. Remarks: None

1. Subroutine Name: FF100 (function)

2. Purpose: generate

$$\int (1/(x^m x^n)) dx$$

where $X = a + bx$

3. Equations and Procedures:

$$FF100 = \frac{-1}{a^{m+n-1}} \left[\sum_{s=0}^{m+n-2} \frac{(m+n-2)! x^{m-s-1} (-b)^s}{(m+n-s-2)! s! (m-s-1) x^{m-s-1}} \right]$$

evaluated at x_1

4. Input Arguments: I, A, B, M, N, X

5. Output Arguments: FF100

6. Error Returns: None

7. Calling Sequence: FF100 (I, A, B, M, N, X)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 403_8 (259_{10}).

12. Subroutine User: F6219

13. Subroutines Required: IFAC

14. Remarks: None

1. Subroutine Name: AXTRAl

2. Purpose: Apply grid point axes transformation by pre-multiplication using either the actual transformation matrix or its transpose.

3. Equations and Procedures:

$$[M_{OUT}] = [GPA] [M_{IN}] \text{ or } [M_{OUT}] = [GPA]^T [M_{IN}]$$

where $[M_{IN}]$ is the input element matrix,

$[GPA]$ is the element grid point axes transformation matrix,

$[M_{OUT}]$ is the output transformed element matrix.

$[M_{OUT}]$ is stored in the same location as $[M_{IN}]$, therefore the input element matrix is lost once the multiplication has been effected. Advantage is taken, during multiplication, of the fact that $[GPA]$ is structured as a set of (3 x 3) or (2 x 2) matrices with main diagonal positions lying on the main diagonal of $[GPA]$.

4. Input Arguments:

GPAXEL : Element grid point axes transformation matrix, $[GPA]$
QSEL : Input element matrix, $[M_{IN}]$
NCOL : Number of columns in QSEL
NNO : Number of element node points
NDEG : Number of degrees of freedom
NDIR : Number of directions
ITRAN : Control code, if ITRAN = 0, then $[M_{OUT}] = [GPA] [M_{IN}]$
if ITRAN = 1, then $[M_{OUT}] = [GPA]^T [M_{IN}]$

5. Output Arguments:

QSEL : Output transformed element matrix, $[M_{OUT}]$

6. Error Returns: None

7. Calling Sequence:

CALL AXTRAl (GPAXEL, QSEL, NCOL, NNO, NDEG, NDIR, ITRAN)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total Storage = 277_8 = 191_{10}

COL (3)
ISAVE (3)

12. Subroutine User: ELPLUG

13. Subroutines Required: None

14. Remarks: The output matrix is stored in the input matrix storage.

1. Subroutine Name: AXTRA3
2. Purpose: Apply grid point axes transformation by triple product multiplication.
3. Equations and Procedures:

$$[AN] = [GPA]^T * [SYM] * [GPA]$$

where

[GPA] is the element grid point axes transformation matrix

[SYM] is symmetric input element matrix

[AN] is symmetric output transformed element matrix

The triple product is obtained by computing a row of the intermediate product of $[GPA]^T * [SYM]$ and then multiplying this intermediate row with $[GPA]$ to obtain a row in $[AN]$. Advantage is taken, during multiplication, of the facts that $[SYM]$ and $[AN]$ are symmetric and also that $[GPA]$ is structured as a set of (3x3) or (2x2) matrices with main diagonal elements lying on the main diagonal of $[GPA]$.

4. Input Arguments:

GPAXEL : Element grid point axes transformation matrix, $[GPA]$
 SYM : Input element matrix, symmetric, singly subscripted, stored lower half by rows, $[SYM]$
 NCOL : Number of columns in SYM (also number of rows in SYM)
 NNO : Number of element node points
 NDEG : Number of degrees of freedom
 NDIR : Number of directions

5. Output Arguments:

AN : Output transformed element matrix, symmetric, singly subscripted, stored lower half by rows, $[AN]$

6. Error Returns: None

7. Calling Sequence: Call AXTRA3
(GPAXEL, SYM, AN, NCOL, NNO, NDEG, NDIR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:
ROW(48)
Total Storage = $517_8 = 335_{10}$.
12. Subroutine User: ELPLUG
13. Subroutines Required: LOC
14. Remarks:

SYM must be stored lower half by rows,
AN will be stored lower half by rows.

Internal intermediate storage in variable ROW is dimensioned 48. If the order of [SYM] is greater than 48, an appropriate increase must be made in this intermediate storage.

1. Subroutine Name: ELPRT
2. Purpose: Print generated element matrices.
3. Equations and Procedures: Non-suppressed matrices are printed, complete with titles.
4. Input Arguments:

NOINK:	: Number of storages in element stiffness, incremental stiffness and mass matrices
AKEL	: Array containing element stiffness matrix
NORD	: Number of element degrees of freedom
FTEL	: Vector containing element load matrix
NNO	: Number of element defining points
NODES	: Array containing element defining grid point numbers
NSEL	: Number of storages in element stress matrix
NRSEL	: Element stress order
SEL	: Array containing element stress matrix
SZALEL	: Vector containing element thermal stress matrix
ANEL	: Array containing element incremental stiffness matrix
INEL	: Element number
NMASS	: Number of storages in element mass matrix
AMASS	: Array containing element mass matrix
NDMPV	: Not used
DAMPV	: Not used
NDMPS	: Not used
DAMPS	: Not used
ILP	: Element type code number
NUMOT	: Number of output matrices
NAMOUT	: Array containing output matrix names
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: Call ELPRT
(NOINK, AKEL, NORD, FTEL, NNO, NODES, NSEL, NRSEL, SEL, SZALEL, ANEL, INEL, NMASS, AMASS, NDMPV, DAMPV, NDMPS, DAMPS, ILP, NUMOT, NAMOUT)
8. Input Tapes: None
9. Output Tapes: None

10. Scratch Tapes: None
11. Storage Required: Total storage is 605_8 (389_{10}).
12. Subroutine User: ELPLUG
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: OUTMAT
2. Purpose: Sort element matrices on scratch tape and output to Format Execution Monitor in an optimal manner
3. Equations and Procedures: First the array controlling the selection and order of output of the matrices (IKNOW) is formed. The IKNOW array will contain the pass number on which each computed output matrix will be written on an output tape. Correspondence between the IKNOW array and the output matrices is as follows:

IKNOW(6)	: Transformation assembly matrix (TA)
IKNOW(7)	: Master element stiffness matrix (KEL)
IKNOW(8)	: Master element applied load matrix (FTEL)
IKNOW(9)	: Master element stress matrix (SEL)
IKNOW(10)	: Master element thermal stress matrix (SZALEL)
IKNOW(11)	: Master element incremental matrix (N)
IKNOW(12)	: Master element mass matrix (M)

For each output matrix except the element applied load and thermal stress matrices the following procedure takes place:

- a. If the matrix has not been calculated, as determined by a slash in its position in the NAMOUT array, its position in the IKNOW array is set equal to zero.
- b. If the matrix has been calculated, then its corresponding output tape number is obtained from the IOSPEC array and a search is done from the beginning of the IOSPEC array to this matrix's position, counting the number of times this tape number has been encountered. This final count is the pass number on which this matrix will be written and is placed into the matrix's corresponding position in the IKNOW array.

After the IKNOW array has been formed it is searched for the greatest number. This number will be the number of passes required to output all of the computed matrices.

On each pass the following procedure is used. The scratch tape containing the element matrices is rewound. This tape consists of two records per element. The first record contains element definition data, the second contains the matrices for that element. The second record is read into a dynamic storage area and interpreted by locating key numbers that appeared in the record. A loop is entered from one to NELEM. The contents of the IKNOW array are

compared to the pass number. When a match is found the corresponding matrix is written on its output tape. Before writing the first element's contribution on its output tape, the appropriate matrix header is written. In most cases the matrices will be output in compressed format. However, in small applications when the maximum element order (NORDM) or the maximum element stress order (NRSELM) is greater than one-half the sum of the element orders (NORSUM) or the element stress orders (NRSSUM), respectively, then the matrices will be output in uncompressed format. A count is maintained in IR and IC for each output matrix in order to place each element's contribution in the correct position in the output matrix. At the end of the pass the appropriate matrix trailer and data set trailer labels are written. The TA matrix is a special case in that it is generated from the element definition data and then placed on its output tape. For output of the element applied load and element thermal stress matrices the following procedure is invoked. During the first pass of the tape, if they were not suppressed, the element applied load and thermal stress matrices were stored in the blank common work area. Following the first pass these two matrices are output in either compressed or uncompressed format, dependent upon the same criteria as all other matrices.

4. Input Arguments:

NUMOT	: Number of output matrices
NAMOUT	: Names of output matrices
IOSPEC	: Unit information regarding output matrices
NTAP3	: Scratch tape containing system information
NTAP4	: Scratch tape containing element matrices
NSYS	: System order
NTD	: Number of degrees of freedom per grid point
NORSUM	: Summation of element orders
NRSSUM	: Summation of element stress rows
NELEM	: Number of elements
NWORKR	: Number of working storages available
WORK	: Common work area
NORDM	: Maximum element order
NRSELM	: Maximum element stress order

5. Output Arguments: None

6. Error Returns: None

7. Calling Sequence:

```
CALL US460(NUMOT, NAMOUT, IOSPEC, NTAP3, NTAP4, NSYS,
            NTD, NORSUM, NRSSUM, NELEM, NWORKR, WORK, NORDM, NRSELM)
```

8. Input Tapes:

NTAP3 : Contains system information
NTAP4 : Contains element matrices in compact form

9. Output Tapes: Output tape units are supplied by the Format Execution Monitor; matrices are output by columns in compressed format. Appropriate matrix header and trailer labels are written. An output matrix consists of all the element matrices of that type placed such that their main diagonal positions lie on the main diagonal of the output matrix in succeeding positions.

10. Scratch Tapes: None

11. Storage Required:

IR(15), IC(15), IKNOW(15) : Total Storage = $1540_8 = 864_{10}$

12. Subroutine User: US04B

13. Subroutines Required:

US461
US462
US463

14. Remarks: None

1. Subroutine Name: US461
2. Purpose: Write a column of an output matrix in uncompressed or compressed format.
3. Equations and Procedures: If KODE is zero, the IWORK array has NSUM zeros placed into it. Then, starting at ISTART, NROW's of ISTORE are placed into the corresponding positions in IWORK. The variable NUM = NSUM is the number of words from IWORK that will be written on tape. If KODE is one, each element of ISTORE is compared to zero. If it is zero, it is ignored. If the element is not zero, then it is placed in the IWORK array in the first unused position and the next position in IWORK is filled by the row number in the output matrix of the non-zero element. The row number is corrected by ISTART in order to place the contribution in the correct row of the output matrix. NUM is a counter used to record the number of non-zero numbers found and the number of words that will be written from IWORK (NUM = 2* number of non-zero elements in ISTORE).
4. Input Arguments:

ISTORE	: Matrix column to be written
ICOL	: Column number of ISTORE in matrix
ISTART	: Starting row number in output matrix
NROW	: Number of rows in ISTORE
NTAPE	: Output tape number
IWORK	: Work area for compression of ISTORE
KODE	: Determines whether matrix is to be put into compressed form
NSUM	: Sum of element orders
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: Call US461
(ISTORE, ICOL, ISTART, NROW, NTAPE, IWORK, KODE, NSUM)
8. Input Tapes: None
9. Output Tape: NTAPE

Record format is ICOL, KODE, NUM, (IWORK(I), I=1, NUM) where ICOL is column number, KODE equals one or zero, NUM is number of words remaining in record and IWORK is the compressed or uncompressed version of ISTORE. Each record then contains NUM + 3 words.

10. Scratch Tapes: None
11. Storage Required: Total storage = $147_8 = 103_{10}$
12. Subroutine User: US460
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: US462
2. Purpose: Create a list which defines the location of the contributions of an element to the assembly transformation matrix.
3. Equations and Procedures: The degrees of freedom for each node point, with respect to the system of grid points, are calculated and placed in LIST. LIST is therefore of length $NNO \times NTD$. The formula for determining this location is:

$$LIST(K) = (NODES(I) - 1) \times NTD + L$$

where $K = 1, 2, \dots, NNO \times NTD$
 $I = 1, 2, \dots, NNO$
 $L = 1, 2, \dots, NTD$
4. Input Arguments:
 NNO - number of element node points
 NODES - array containing element node point numbers
 NTD - number of degrees of freedom per grid point
5. Output Arguments:
 LIST - array containing row number in TA matrix for each degree of freedom for each element node point.
6. Error Returns: none
7. Calling Sequence: CALL US462 (NNO, NODES, NTD, LIST)
8. Input Tapes: none
9. Output Tapes: none
10. Scratch Tapes: none
11. Storage Required: total storage = $114_8 = 76_{10}$
12. Subroutine User: US460
13. Subroutines Required: none
14. Remarks: none

1. Subroutine Name: US463
2. Purpose: Obtain full column from symmetrically stored matrix
3. Equations and Procedures: For a symmetric matrix column is equivalent to row. The corresponding row to ICOL is located and the elements of that row up to and including the diagonal element are placed in the first and succeeding position of COL. If ICOL was the last column of the matrix the process is complete and control is returned to the calling program. If ICOL was not the last column then each element in the ICOL position of the remaining rows is placed into COL and control is returned to the calling program.
4. Input Arguments:
SYM symmetric matrix stored lower half by rows, singly sub-scripted
N order of SYM
ICOL Column number of SYM desired
5. Output Arguments:
COL - full column number ICOL
6. Error Returns: none
7. Calling Sequence: CAL US463 (SYM, N, ICOL, COL)
8. Input Tapes: none
9. Output Tapes: none
10. Scratch Tapes: none
11. Storage Required: Total storage = $1618 = 113_{10}$
12. Subroutine User: US460
13. Subroutines Required: none
14. Remarks: none

1. Subroutine Name: LOGFLO
2. Purpose: Set logical execution controls for USER04 module
3. Equations and Procedures: APHASE, BPHASE and ERROR are initially set to .FALSE. All positions in MASTER are set to zero. If any of the first five output matrix positions are non-blank then APHASE is set to .TRUE. If any of the last seven output matrix positions is non-blank then BPHASE is set to .TRUE. MASTER is then filled by packing in the output matrix position number the requires that input section. At present there are six possible required input sections indicated in MASTER:

MASTER (1)	- System control input indicator
MASTER (2)	- Grid point coordinates input indicator
MASTER (3)	- Boundary condition input indicator
MASTER (4)	- Element definition input indicator
MASTER (5)	- Grid point loads input indicator
MASTER (6)	- Material library input indicator
4. Input Arguments:

NUMOT	: Number of output matrices
NAMOUT	: Array containing output matrix names
NUMIN	: Number of input matrices
NAMIN	: Array containing input matrix names
APHASE	: Logical variable indicating necessity to execute subroutine US04A
BPHASE	: Logical variable indicating necessity to execute subroutine US04B
NUMAST	: Length of MASTER
MASTER	: Array indicating required input sections
5. Output Arguments:

ERROR	: Logical variable indicating error condition
6. Error Returns: If output matrix position eleven is non-blank and input matrix position four is blank, then ERROR is set to .TRUE.
7. Calling Sequence:

(NUMOT, NAMOUT, NUMIN, NAMIN, APHASE, BPHASE, NUMAST, MASTER, ERROR)
8. Input Tapes: None

9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 512_8 (330_{10}).
12. Subroutine User: US04
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: INDECK
2. Purpose: Translate input matrix containing a data deck into a BCD input deck
3. Equations and Procedures: The matrix is located by utilizing subroutine EUTL3. Each column of the matrix contains one input card divided into eighty rows. Each column is read in binary from the unit specified in INSPEC(1) and written on NOUT by an 80A1 format. The number of columns, as contained in the matrix header, is actually the number of cards in the data deck.
4. Input Arguments:
 - NAMIN : Array containing input matrix name
 - INSPEC : Array containing unit specification for input matrix
 - NOUT : Logical unit reserved for output data deck
 - CARD : Work storage
5. Output Arguments:
 - IER : Logical variable indicating error condition
6. Error Returns: For each column of the input matrix, the compression code must be zero and the number of words must be eighty. If either condition is not satisfied then the matrix does not qualify as an input deck matrix and IER will be set to .TRUE..
7. Calling Sequence:
 - (NAMIN, INSPEC, NOUT, CARD, IER)
8. Input Tapes:
 - INSPEC(1) : unit containing input data deck matrix
9. Output Tapes:
 - NOUT : unit which will contain BCD data deck
10. Scratch Tapes: None
11. Storage Required: Total storage is 404_8 (260_{10}).
12. Subroutine User: US04A
13. Subroutines Required: EUTL3
14. Remarks: None

1. Subroutine Name: COPYDK
2. Purpose: Output a data deck in matrix form
3. Equations and Procedures: A matrix header is written in which the number of rows is set to eighty and the number of columns is set equal to the number of cards in the data deck. Each card of the data deck is read from NINPUT in 80A1 format and then written on the unit specified in IOSPEC(1) in a binary matrix column record containing eighty words. The process continues until an END, CHECK or \$END card is encountered. Finally the matrix trailer is written and control is returned to the calling program.
4. Input Arguments:
 - NAMOUT : Array containing output matrix name
 - IOSPEC : Array containing unit specifications for the output matrix
 - CARD : Work storage
 - NINPUT : Unit containing data deck
 - JMAX : Number of cards in data deck
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:
 - (NAMOUT, IOSPEC, CARD, NINPUT, JMAX)
8. Input Tapes:
 - NINPUT : unit containing input data deck
9. Output Tapes:
 - IOSPEC(1): unit which will contain output data deck matrix
10. Scratch Tapes: None
11. Storage Required: Total storage is 300_8 (192_{10}).
12. Subroutine User: US04A
13. Subroutines Required:
 - EUTL5
 - EUTL6
14. Remarks: None

1. Subroutine Name: SHIFT
2. Purpose: Given a one-dimensional array, this routine can relocate a block of data, within the array.
3. Equations and Procedures: The routine computes the size of the block to be shifted. It checks the direction of shift, and initializes the shift constants, finally performing the shift.
4. Input Arguments:
 - PROPER : Array in which shifting is to occur
 - IFROM : Initial subscript of block to be shifted
 - ITO : Final subscript of block to be shifted
 - ISIZE : Size of shift
 - NDIR : Direction of shift
5. Output Arguments:
 - IERROR : Error return
6. Error Returns: If the size of the block to be shifted is computed to be negative (IFROM ITO) IERROR is set equal to 1 (one).
7. Calling Sequence:
 - SHIFT (PROPER, IFROM, ITO, ISIZE, NDIR, IERROR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 142_8 (98_{10}).
12. Subroutine User: FMAT
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CHEK
2. Purpose: Perform input/output cross-checking for USER04 module
3. Equations and Procedures: The required input sections for the selected output matrices are indicated in the array MASTER (see subroutine LOGFLO). The actual input sections processed are indicated in the array ICONT. The logical array, GO, is set according to the information in MASTER as compared with ICONT. If an output matrix requires an input section that is not present then a message is printed giving the matrix name and corresponding position in the GO array is set to .FALSE.
4. Input Arguments:
 - NAMOUT : Array containing output matrix names
 - NUMOT : Number of output matrices
 - NAMIN : Array containing input matrix names
 - NUMIN : Number of input matrices
 - MASTER : Array indicating required input sections
 - NUMAST : Length of MASTER
 - ICONT : Array indicating processed input sections
 - NCONT : Length of ICONT
5. Output Arguments:
 - GO : Array indicating input requirements have been satisfied, one position for each possible output matrix
6. Error Returns: None
7. Calling Sequence:
 - (NAMOUT, NUMOT, NAMIN, NUMIN, MASTER, NUMAST, ICONT, NCONT, GO)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 404_8 (260_{10}).
12. Subroutine User: US04A
13. Subroutines Required: NTEST
14. Remarks: None

1. Subroutine Name: OUTINT
2. Purpose: Output interpreted input data as a matrix
3. Equations and Procedures: After processing the input data deck, all necessary information is stored in three areas. System control information is stored in the array KNMD and in the first two records on scratch unit NTAPE1. Element generation data is stored on scratch unit NTAPE3. All of this data is output as a matrix, the first column containing KNMD, the second and third columns containing the first two records from NTAPE, the fourth column containing two words (number of elements, NELEM, and grid point axes indicator) and the last 2*NELEM columns containing the input element generation data.
4. Input Arguments:
 - NAMOUT : Array containing output matrix name
 - IOSPEC : Array containing unit specifications for output matrix
 - NTAPE1 : Unit containing system control information
 - NTAPE3 : Unit containing element generation data
 - KNMD : Array containing system control information
 - NUMK : Length of KNMD
 - IWORK : Work storage area
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:

(NAMOUT, IOSPEC, NTAPE1, NTAPE3, KNMD, NUMK, IWORK)
8. Input Tapes:
 - NTAPE1 : Unit containing system control information
 - NTAPE3 : Unit containing element generation data
9. Output Tapes:

IOSPEC(1): Unit which will contain interpreted input data matrix
10. Scratch Tapes: None
11. Storage Required: Total storage is 640_8 (416_{10}).
12. Subroutine User: US04A

13. Subroutine Required:

EUTL5
EUTL6

14. Remarks: None

1. Subroutine Name: ININT
2. Purpose: Restore data from interpreted input matrix
3. Equations and Procedures: Subroutine EUTL3 is called to locate the input matrix. The first column of the matrix contains system control information and is read into KNMD. Columns two and three contain further system information and are recorded as the first two records on NTAPE1. Column four and all succeeding columns contain element generation input data and are recorded on NTAPE3.
4. Input Arguments:
 - NAMIN : Array containing input matrix name
 - INSPEC : Array containing unit specifications for input matrix
 - NTAPE1 : Unit reserved for system control information
 - NTAPE3 : Unit reserved for element generation input data
 - KNMD : Array reserved for system control information
 - IWORK : Work storage area
 - NUMK : Length of KNMD
5. Output Arguments:
 - IER : Logical variable indicating error condition
6. Error Returns: If the input matrix cannot be located, or a word count error occurs for columns one or four, or the matrix trailer record is encountered unexpectedly, then IER is set to .TRUE..
7. Calling Sequence:

(NAMIN, INSPEC, NTAPE1, NTAPE3, KNMD, IWORK, NUMK, IER)
8. Input Tapes:

INSPEC(1) : Unit containing interpreted input matrix
9. Output Tapes:
 - NTAPE1 : Unit reserved for system control information
 - NTAPE3 : Unit reserved for element generation input data
10. Scratch Tapes: None
11. Storage Required: Total storage is 1002₈ (514₁₀).

- 12. Subroutine User: US04B
- 13. Subroutines Required: EUTL3
- 14. Remarks: None

1. Subroutine Name: DEFLEX
2. Purpose: Sort input displacement matrix into separate element input sections
3. Equations and Procedures: The input displacements for the system are read into the IWORK array and restored at the end of the IWORK array. For each element, the following procedure is invoked: the element generation input data is read from scratch unit NTAPE3; the array containing the element definition points is extracted; the input displacements corresponding to these points are selected from the system input displacements and written on scratch unit NTAPE4.
4. Input Arguments:
 - NSYS : Total degrees of freedom in system
 - NAMIN : Array containing input matrix name
 - INSPEC : Array containing unit specifications for input matrix
 - NTAPE3 : Unit containing element generation input
 - NTAPE4 : Unit reserved for element input displacements
 - IWORK : Work storage area
 - NWORK : Length of IWORK
 - MAXNIL : Maximum length of record on NTAPE3
5. Output Arguments:
 - IER : Logical variable indicating error condition
6. Error Returns: If the input matrix cannot be found, or its dimensions are not NSYS by one or IWORK does not contain sufficient storage locations then IER is set to .TRUE..
7. Calling Sequence:
 - (NSYS, NAMIN, INSPEC, NTAPE3, NTAPE4, IWORK, NWORK, MAXNIL, IER)
8. Input Tapes:
 - NTAPE3 : Unit containing element generation input data
 - INSPEC(1): Unit containing system input displacement matrix
9. Output Tapes:
 - NTAPE4 : Unit reserved for element input displacements

10. Scratch Tapes: None
11. Storage Required: Total storage is 777_8 (511_{10}).
12. Subroutine User: US04B
13. Subroutines Required:
 EUTL3
 EUTL9
14. Remarks: None

1. Subroutine Name: SQUISH
2. Purpose: Set matrix suppression codes for element generation phase
3. Equations and Procedures: The indicators are initially set to zero, signifying suppression is desired. Subroutine NTEST is called to examine the output matrix names for suppression selections. For each non-suppressed matrix position encountered the corresponding indicator is reset to one.
4. Input Arguments:

NAMOUT : Array containing matrix names
NUMOT : Number of output matrices
5. Output Arguments:

KK : Suppression indicator for element stiffness matrices
KF : Suppression indicator for element load matrices
KS : Suppression indicator for element stress matrices
KN : Suppression indicator for element incremental stiffness matrices
KM : Suppression indicator for element mass matrices
KDS : Suppression indicator for element structural damping matrices
KDV : Suppression indicator for element viscous damping matrices
KTS : Suppression indicator for element thermal stress matrices
6. Error Returns: None
7. Calling Sequence:

(NAMOUT, KK, KF, KS, KN, KM, KDS, KDV, KTS, NUMOT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None

11. Storage Required: Total storage is 212_8 (138_{10}).
12. Subroutine User: FELEM
13. Subroutines Required: NTEST
14. Remarks: None

1. Subroutine Name: MATSUP
2. Purpose: Insert suppressed input matrix names into the Format System
3. Equations and Procedures: Scratch unit NPREP is backspaced to the beginning of the instruction section. If scratch unit NDATA already contains matrices then it is positioned at the data set trailer; otherwise it is rewound and a data set header written upon it. Each instruction record is then read to determine if the op-code is capable of containing input suppressed matrices as indicated in the array LEGAL. If the operation is capable of containing suppressed input matrices then the input matrix names are checked to see if they contain a slash in the first position. If this is the case the suppression name is entered as a null matrix on NDATA. NDATA is then returned to the first suppressed matrix name and re-read so that each added matrix on NDATA is recorded on NPREP after the instructions. Control is then returned to the calling program.
4. Input Arguments:
 - NUMD : Number of matrices on NDATA
 - NUMSUP : Number of suppressed input matrices to be added to NDATA
 - NDATA : Logical unit containing card input matrices
 - NPREP : Logical unit containing preprocessor data
 - NUMI : Number of instructions on NPREP
 - IWORK : Work storage area
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:

(NUMD, NUMSUP, NDATA, NPREP, NUMI, IWORK)
8. Input Tapes:
 - NDATA : contains card input matrices, if present
 - NPREP : contains input abstraction instructions in coded form
9. Output Tapes:
 - NDATA : will contain suppressed input matrices
 - NPREP : will contain suppressed input matrix names

- 10. Scratch Tapes: None
- 11. Storage Required: Total storage is 717_8 (463_{10}).
- 12. Subroutine User: PREP
- 13. Subroutines Required: None
- 14. Remarks: None

1. Subroutine Name: PLUG1
2. Purpose: To formulate the element matrices for a quadrilateral plate
3. Equations and Procedures: The following sequence of operations are necessary in order to obtain the element matrices. Equations are found in Volume I.
 - A. The material and geometric properties are obtained from MAT and EXTRA respectively.
 - B. From the Appendix of reference 1, the corner points defining the element are redefined to local oblique system by TRAQB. Provision is made to also account for different material axis orientation (due to orthotropy) or for a specific input stress direction.
 - C. The following operations are performed as formulated in the appropriate equations:
 - (1) Call NEWFT to form matrices necessary for thermal loadings,
 - (2) Call CDELPQ to determine integrals of each zone of the quadrilateral,
 - (3) The material property matrix dependent upon the stress-strain input of EXTRA (4) is coded as EM,
 - (4) The strain, stress and displacement transformations are coded as TES, TESS and TW respectively,
 - (5) Compute $[EG] = [TE\$]^T [EM] [TE\$]$,
 - (6) Store transpose of $[TE\$]$ into $[T\$AVE]$, $[T\$AVE]$ is then stored back into $[TE\$]$ and inverted,
 - (7) If print option equals -1, call PLPRTA for print of intermediate computations,
 - (8) Initialize the thermal load, pressure, thermal stress, stress and mass matrices to zero.
 - D. Membrane computations are performed in the following manner:
 1. Call CK11 to formulate the $[K21S]$ element stiffness matrix in global system,
 2. Formulate the transformation from local to global system by forming the product $[TAOM] [TOGM] [TGM] = [TMS]$,

3. Equations and Procedures: Continued

- (3) If mass matrix is requested then
 - a. Call CMMASS to form the membrane mass matrix in local systems (CMM),
 - b. The mass matrix is then transformed to global systems as $[AMASS] = [TGSM]^T [CMM] [TGSM]$.
- (4) If stress and/or force matrices are requested then
 - a. Call C\$TM to formulate the membrane stress matrix $[S]$,
 - b. Call CFMTS to formulate the membrane thermal force and stress matrices.
- (5) If print controls equal -1, call PRT1 to print out intermediate matrices.

E. Flexural computations are then performed in the following manner:

- (1) Call CK22 to add the flexural contributions to the stiffness matrix $[K21S]$,
- (2) Apply transformation to global system by performing $[TFM] = [TGAMB][TOGB][TGRB]$,
- (3) If stress and/or force matrices are requested then
 - a. If input pressure not equal to 0, call CFP to formulate the pressure matrix,
 - b. The flexural contributions to the stress matrix are formulated by calling C\$TF,
 - c. If flexural input temperature not equal to zero, calls CFFTS to formulate the thermal force and stress matrices.
- (4) If mass is requested then
 - a. Call CFMASS to form the membrane mass matrix in local system $[CMF]$,
 - b. The mass matrix is transformed to global system as $[AMASS] = [TGFS]^T [CMF] [TGFS]$
- (5) Again if the print option is -1, intermediate element computation printout is obtained from PRT1.

4. Input Arguments:

IPL : Plug number
NNO : Number of nodes (8)
XC,YC,ZC : Coordinates of element node points
TEL : Temperature array of element node points
PEL : Pressures at element node points
NN : Number of nodes
NL : Node point numbers
KK,KN : Control for computation of matrices (see remarks)
GPAXEL : Grid point axes transformations
MAT : Array containing material properties
EXTRA : Array containing geometric properties

5. Output Arguments:

K21S : Stiffness matrix
FTEL : Element force matrix
S : Stress matrix
SZALEL : Thermal stress matrix
AMASS : Mass matrix for dynamic analysis

6. Error Returns:

- a. Standard error returns by ELPLUG (NERR)
- b. $\sin \alpha = 0$ indicates coordinate input data error

7. Calling Sequence:

CALL PLUG1 (IPL, NNO, XC, YC, ZC, TEL, PEL, QS, IP, NORD,
NERR, NOINK, K21S, AN1, FTEL, S, SZALEL, AMASS, DAMPV,
DAMPS, NRSEL, NN, NL, NMASS, NDMPV, NDMPS, NSEL, KK, KF, K8,
KTS, KM, KDS, KDV, KN, IUSEL, EPSLON, SIGZER, MAT, EXTRA,
GPAXEL, NDIR, NDEG, ICONT)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is (13141)₁₀

12. Subroutine User: ELPLUG

13. Subroutines Required:

ELTEST	CSTM
NEWFT	CFMTS
CDELPQ	PRT1
MINV	CK22
BCB	CFP
CK11	CSTF
MABC	CFFTS
CMMASS	CFMASS

14. Remarks:

The following is a list of control indicators for PLUG1. For all indicators shown a value of one will cause the operation to be performed and a value of zero will cause the operation to be skipped.

LT1	-	compute membrane contributions
LT2	-	compute flexural contributions
KK	-	compute element stiffness matrix
KF	-	compute element force matrix (thermal and/or pressure)
K8	-	compute element stress matrix
KTS	-	compute element thermal stress matrix
KM	-	compute element mass matrix
KDS	-	not used
KDV	-	not used
KN	-	compute element incremental stiffness matrix

1. Subroutine Name: CC21
2. Purpose: To assemble a submatrix into an assembled matrix
3. Equations and Procedures: None
4. Input Arguments:
 - K : Control on positioning of elements for assembly
 - NI : Constants from PLUG1
 - C : elements of input matrix
5. Output Arguments:
 - C21 - elements of the expanded matrix
6. Error Returns: None
7. Calling Sequence: CALL (K, NI, C, C21)
8. Input Tapes: None
9. Output: None
10. Scratch Tapes: None
11. Storage Required: NI(8,10), C(1), C21(105) and total storage is $(145)_{10}$
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: MABC
2. Purpose: To evaluate the triple product of
 $[AN] = [A] [B] [C]$
3. Equations and Procedures:
 - a. Each row of the $[A]$ matrix is multiplied by the corresponding column of the $[B]$ matrix and stored in the $[AM]$ matrix by column.
 - b. Then each row of the $[AM]$ matrix is multiplied by the corresponding column of the $[C]$ matrix and the final product stored in the $[AN]$ matrix by column.
4. Input Arguments:

A:	elements of $[A]$ matrix
B:	elements of $[B]$ matrix
C:	elements of $[C]$ matrix
AM:	working storage
N:	number of rows in $[A]$ matrix (order)
L:	number of rows in $[B]$ matrix (order)
K:	number of rows in $[C]$ matrix (order)
M:	number of columns in $[C]$ matrix (order)
N1, M1:	dimension of $[A]$ matrix
N2, M2:	dimension of $[B]$ matrix
N3, M3:	dimension of $[C]$ matrix
5. Output Arguments:

AN:	Elements of triple product matrix
6. Error Returns: None
7. Calling Sequence:

(A, B, C, AN, AM, N, L, K, M, N1, M1, N2, M2, N3, M3)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

A (1), B (1), C (1), AN (1), AM (1)

12. Subroutine User: general subroutine used by many other subroutines
13. Subroutines Required: None
14. Remarks: Standard matrix multiplication routine; but caution must be exercised when the dimensions and orders of input and output matrices are different

1. Subroutine Name: NEWFT
2. Purpose: Generate membrane and flexural thermal loads for quadrilateral thin shell in local coordinates
3. Equations and Procedures:

$$\begin{aligned} [BCT] &= [F]^{-1} [CT] \\ [BMT] &= [BCT] \{TEMM\} \\ [BFT] &= [BCT] \{TEMF\} \end{aligned}$$

where $[F]$ and $[CT]$ are geometric matrices of local coordinates

$$\begin{aligned} \{TEMM\} &= \{TEL(I,1)\} \text{ membrane temperatures} \\ \{TEMF\} &= \{TEL(I,2)\} \text{ flexural temperatures} \end{aligned}$$
4. Input Arguments:

DELTM : Average membrane temperature
 DELTF : Average flexure temperature
 TEL : Temperature array of element
 R1B : Local X coordinate of node 1
 R2B : " Y " of node 2
 R3B : " X " of node 3
 R4B : " Y " of node 4
 IPRINT : Print option
 TZ : Initial membrane temperature
5. Output Arguments:

BMT : Membrane thermal load in local coordinates
 BFT : Flexural thermal load in local coordinates
6. Error Returns: None
7. Calling Sequence:

(DELTM, DELTF, TEL, R1B, R2B, R3B, R4B, BMT, BFT, IPRINT, TZ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None

11. Storage Required:

F(3,3), BCT(3,4), CT(3,4), BMT(4,1), BFT(4,1), TEMM(4),
TEMF (4), TEL(12,2), R1B(1), R2B(1), R3B(1), R4B(1)

Total Storage is (227₁₀).

12. Subroutine User: PLUG1

13. Subroutines required: MINV, MAB

14. Remarks:
- a. If print option equals -1, intermediate computations are printed out.
 - b. The membrane or flexural contribution is by passed if the respective thickness is 0.

1. Subroutine Name: CDELPQ
2. Purpose: To compute the integrals from equations in documentation for PLUG1 in Volume I.
3. Equations and Procedures:

$$\text{DELPQ}^j = Cx_j Y_j \quad \text{where } \begin{array}{l} p = 0,1,2,3,4 \\ q = 0,1,2,3,4 \\ j = 1,2,3,4 \\ C = \text{constant} \end{array}$$
4. Input Arguments:
 AJ - x distance from centroid to respective node point
 BJ - y distance from centroid to respective node point
5. Output Arguments:
 DELPQ - table of integrals for the 4 zones of the quadrilateral
6. Error Returns: None
7. Calling Sequence:
 Call CDELPQ (AJ, BJ, DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 DELPQ (4,5,5)
 AJ $\left\{ \begin{smallmatrix} 4 \\ 4 \end{smallmatrix} \right\}$
 BJ $\left\{ \begin{smallmatrix} 4 \\ 4 \end{smallmatrix} \right\}$
 Total Storage is $(241)_{10}$.
12. Subroutine User: PLUG1
13. Subroutines Required: CHDEL1
14. Remarks: None

1. Subroutine Name: CHDELL
2. Purpose: To rearrange the integrals generated by CDELPQ
3. Equations and Procedures: None
4. Input Arguments: DELPQ - integrals generated by CDELPQ
5. Output Arguments: DELPQ - rearranged integrals
6. Error Returns: None
7. Calling Sequence: CALL CHDELD1 (DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: DELPQ (4,5,5)
Total Storage is (70)₁₀
12. Subroutine User: CDELPQ
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PLPRTA
2. Purpose: Print variables generated by PLUG1, if IPRINT equals -1.
3. Equations and Procedures: Not applicable
4. Input Arguments:

EX, EY	: Youngs modulus in X and Y directions respectively
MUXY	: Poisson's Ratio
GXY	: Shear modulus
GAMMA	: Material angle
ALPHAX, ALPHAY	: Thermal coefficients of expansion in X and Y directions
TF, TM	: Flexural and membrane thickness
RZB	: Vector normal to plane of quadrilateral element
R24	: deviation of local coordinates between points 2 and 4 of the quadrilateral
LAMDA	: Coefficient of normal vector so that element lies in a plane
R24BP	: Sum of the inplane vector and normal vector
THETA	: Angle for calculating centroid of element
E	: Column vector colinear with local geometric X, Y and Z system
TPRIME	: Transformation matrix
NL	: Node point numbers
SINAL, COSAL	: Sine and cosine of oblique coordinate system
SINA, COSA	: Sine and cosine for stress angles
SING, COSG	: Sine and cosine of material angle
EM	: Coefficient matrix utilizing Hook's Law
ALPHM	: Matrix containing coefficients of thermal expansion
CORRDL	: local coordinates
DELPQ	: table of integrals for the 4 zones of the quadrilateral
ALPHG	: Dummy
EG	: E matrix transformed
TES	: Strain transformation matrix
TW	: Displacement function transformation matrix
5. Output Arguments: None
6. Error Returns: None

7. Calling Sequence:

CALL PLPRTA (EX, EY, MUXY, GXY, GAMMA, ALPHAX, ALPHAY,
TF, TM, RZB, R24B, LAMDA, R24BP, RØB,
THETA, E, TPRIME, NL, SINAL, CØSAL, SINA,
CØSA, SING, CØSG, EM, ALPHM, CØØRDL, DELPQ,
ALPHG, EG, TES, TW)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total Storage is (629_{10}) .

12. Subroutine User: PLUG1

13. Subroutines Required: None

14. Remarks: None

1. Subroutine Name: CK11

2. Purpose: To generate the membrane stiffness for PLUG1, quadrilateral thin shell element

3. Equations and Procedures: The following sequence of operations takes place to formulate the membrane stiffness matrix:

- (1) Call CT11 to formulate the membrane displacement coordinate transformation as TAO.
- (2) Call MATI60 to invert the above matrix.
- (3) Call CTOGM to form the transformation from oblique to geometric coordinates as TOGM.
- (4) Generate the transformation matrix from geometric to reference system coordinates (TGRM) by calling CTGRM.
- (5) If grid point axes transformations to another system other than global are to be formulated, call AXTRA2 to generate the new TGRM matrix.
- (6) Generate the displacement function transformation as TU.
- (7) Call BCB to form the product
 $[TU]^T [EG] [TU] = [EO]$

This matrix is then multiplied by the constant $T \times SINA$ and renamed the JPQ matrix.

- (8) Generate the membrane stiffness (C matrix) by calling CC1. The C matrix is then expanded by CC21 and C21.
- (9) The transformation matrix TAO is expanded as TAOM.
- (10) Call BCB to form the following products:

$$(a) [K11O] = [TAOM]^T [C11] [TAOM]$$

$$(b) [K11G] = [TOGM]^T [K11O] [TOGM]$$

$$(c) [K21S] = [TGRM]^T [K11G] [TGRM]$$

The final product, $[K21S]$, is the desired membrane stiffness matrix.

4. Input Arguments:

NDIR	: Number of directions of movement for each grid point, control needed for AXTRA2
NDEG	: Number of degrees of freedom for each grid point, control needed for AXTRA2
ICONT	: Control set equal to 1 if grid point axes transformations are required from input data
GPAXEL	: The grid point axis transformation matrix
NNJ	: Number of grid points (8) describing the element
NL	: Array containing the grid point numbers
EEZ	: Input on element data card for eccentricity
AJ, BJ	: Local X and Y coordinates of the element

SINA, COSA : Sine and cosine of the angle defined by the
 diagonals of the element between grid points
 1 and 2
 TPRIME : Transformation matrix
 IPRINT : Print option
 T : Membrane thickness
 LT1 : Control set equal to 1 when membrane thickness
 is not zero
 EG : Material properties matrix
 DELPQ : Table of integrals
 NI : Array for assembly purposes

5. Output Arguments:

K21S : Membrane stiffness matrix
 EO : Material properties matrix
 TU }
 TAO } : Transformation matrices defined in item 3 above
 TAOM }
 TOGM }
 TGRM }
 K11O }
 K11G } : Intermediate matrices formed and defined in
 C11 } item 3 above.
 JPQ }
 C21 }

6. Error Returns: None

7. Calling Sequence:

CALL CK11, (K21S, NDIR, NDEG, ICONT, GPAXEL, NNO, NL, EEZ,
 AJ, BJ, SINA, COSA, TPRIME, IPRINT, T, NI, LT1, EG,
 DELPQ, TAO, TAOM, TOGM, TGRM, K11O, K11G, C11, JPQ,
 C21, TU, EO, TFS, TMS, C)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required:

NL (8), GPAXEL (3, 3, 12), AJ (1), BJ (1), TAO (8, 8),
 TPRIME (3,3), F (10), EO (10), C (55), C21 (105), NI (8,
 10), TU (3, 4), K11O (136), K11G (210), C11 (105), TAOM
 (16, 16), TOGM (16, 20), TGRM (20, 48), JPQ (10), TGRA
 (16, 48), DELPQ (4, 5, 5), TFS (16, 48), TMS (16, 48)
 Total Storage is (464)10

12. Subroutine User: PLUG1

13. Subroutines Required:

CT11	CTOGM	AXTRA2	CC1
MATI60	CTGRM	BCB	CC21

14 Remarks: None

1. Subroutine Name: CT11
2. Purpose: To formulate the membrane displacement coordinate transformation as $TA\emptyset$
3. Equations and Procedures: The formulation is given in the documentation for PLUG1 in Volume I.
4. Input Arguments:
 - AJ : Local X coordinates
 - BJ : Local Y coordinates
 - IPRINT : Print indicator
5. Output Arguments:
 - $TA\emptyset$: Transformation matrix
6. Error Returns: None
7. Calling Sequence: CALL CT11 (AJ, BJ, $TA\emptyset$, IPRINT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: AJ (1), BJ (1), $TA\emptyset$ (8,8)
Total Storage is (227)₁₀
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: If IPRINT equals -1, the $TA\emptyset$ matrix is printed.

1. Subroutine Name: MATI60
2. Purpose: Invert the TAØ matrix
3. Equations and Procedures: None
4. Input Arguments: N - order of matrix to be inverted
A - to be inverted
5. Output Arguments: ISING - error messages
DETR - value of determinant
A - contains elements of the inverted matrix
6. Error Returns: ISING = 0 No error
ISING = 1 Singular matrix
7. Calling Sequence:
Call MATI60(N, A, ISING, DETR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: LIST (8), A(8,8) total storage (321)₁₀
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CTØGM
2. Purpose: To formulate the transformation matrix from oblique to geometric coordinates
3. Equations and Procedures: See writeup for PLUG1
4. Input Arguments:
CØSA : Cosine and sine of the angle defined by the
SINA diagonals of the element between grid points
1 and 2
5. Output Arguments:
TØGM : Transformation matrix
6. Error Returns: None
7. Calling Sequence:
CALL CTØGM (CØSA, SINA, TØGM)
8. Input tapes: None
9. Output tapes: None
10. Scratch Tapes: None
11. Storage Required:
TØGM (16,20)
Total Storage (67)₁₀
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CTGRM
2. Purpose: Formulate the transformation from geometric to reference system coordinates
3. Equations and Procedures: See writeup of PLUG1.
4. Input Arguments:
 - NL : Node point numbers
 - EEZ : Eccentricity factor
 - TRIME : Transformation matrix to be expanded
5. Output Arguments:
 - TGRM : Transformation matrix
6. Error Returns: None
7. Calling Sequence:
 - CALL CTGRM (NL, EEZ, TPRIME, TGRM)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required:
 - NL (1), TPRIME (3,3), TGRM (20,48),
 - Total Storage is (275) 10.
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CCl
2. Purpose: Generate the bottom half of the membrane contribution to the element stiffness matrix for the quadrilateral element
3. Equations and Procedures: Contained in documentation for quadrilateral element in Volume I.
4. Input Arguments:
KI : Control for appropriate computation
JPQ : Matrix containing material properties
DELPQ : Table of integrals
5. Output Arguments:
C : Membrane contribution to stiffness matrix
6. Error Returns: None
7. Calling Sequence: CALL CCl (KI, JPQ, DELPQ, C)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
DELPQ (4,5,5), C (55), JPQ (10)
Total Storage is (666)₁₀.
12. Subroutine User: CK11
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CMMASS
2. Purpose: Generate the membrane contribution to the mass matrix in local coordinates
3. Equations and Procedures: Contained in documentation for the quadrilateral element in Volume I
4. Input Arguments:
 - T : Membrane thickness
 - DOO : Area of each zone of quadrilateral
 - SINA : Sine of angle defined by points 1 and 2 and the diagonal of the quadrilateral
 - DENS : Density of the plate material
5. Output Arguments:
 - AMS : Membrane mass contribution
6. Error Returns: None
7. Calling Sequence: CALL CMMASS (T, DOO, SINA, DENS, AMS)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 - DOO (1), AMS (1), Total Storage (112)₁₀.
12. Subroutine User: PLUG1
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CSTM
2. Purpose: Evaluate the membrane stress matrix in local coordinates for the quadrilateral element
3. Equations and Procedures: The following sequence of operations is performed:
 - (1) Call CDM to formulate the membrane displacement derivative matrix as [DFM].
 - (2) Call MAB to form $[AM4] = [DFM] [TMS]$
 - (3) Call MAB to form $[AM5] = [TU] [AM4]$
 - (4) Call MSB to form $[AM6] = [EG] [AM5]$
 - (5) Call MAB to form $[AM5] = [TES] [AM6]$
 - (6) Multiply [AM5] by the thickness and store in appropriate location of the stress matrix.
4. Input Arguments:

$\left. \begin{array}{l} R1B \\ R2B \\ R3B \\ R4B \end{array} \right\} : \text{Local coordinates}$
 TU : Displacement function transformation
 EG : Material properties matrix
 TES : Strain displacement matrix
 T : Membrane thickness
 TMS : Transformation matrix to system coordinates
5. Output Arguments:

S : Stress matrix in system coordinates.
6. Error Returns: None
7. Calling Sequence:

CALL C\$TM (R1B, R2B, R3B, R4B, TU, EG, TES, T, S, TFS, TMS, DFM, AM4, AM5, AM6)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

AM4 (4, 48), DFM (4, 16), TU (3, 4), EG (10), TES (3, 3),
 S (40, 48), AM5 (3, 48), AM6 (3, 48), TFS (16, 48), TMS (16, 48),
 Total Storage is (176)₁₀

- 12. Subroutine User: PLUG1
- 13. Subroutines Required: MAB, MSB
- 14. Remarks: None

1. Subroutine Name: CDM
2. Purpose: To evaluate membrane displacement derivative matrix for the 4 zones of the quadrilateral
3. Equations and Procedures:
See Writeup on PLUG 1 for equations
4. Input Arguments:
IZ - constant for zone to be evaluated
R1B, R2B, R3B, R4B - local coordinates of element
5. Output Arguments:
DFM - membrane displacement displacement matrix
6. Error returns: None
7. Calling Sequence:
Call CDM (IZ, R1B, R2B, R3B, R4B, DFM)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch tapes: None
11. Storage required:
R1B (1), R2B (1), R3B (1), R4B (1), DFM (4, 16)
Total Storage is (257)₁₀
12. Subroutine User: CSTM
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CFMTS
2. Purpose: To evaluate the membrane thermal load and thermal stress matrices
3. Equations and Procedures:

(1) The thermal load is computed as follows:

$$\begin{aligned}
 \{AM1\} &= [EM] \{ALPHM\} \\
 \{AM2\} &= [TES]^T \{AM1\} \\
 \{IT\} &= [TU]^T \{AM2\} \\
 \{IT\} &= T [SINA] \{IT\}, \text{ then}
 \end{aligned}$$

Call CFM to formulate the thermal load FPB then

$$\{FT\} = [TMS]^T \{FPB\}$$

(2) The thermal stress matrix is computed as follows:

$$\begin{aligned}
 \{AM2\} &= DELTM(T) \{AM2\} \\
 \{SZLM\} &= [TESS] \{AM2\}
 \end{aligned}$$

The SZLM array is assembled into SZALEL.

4. Input Arguments:

EM : Material properties matrix
 ALPHM : Coefficients of thermal expansion
 TES : Strain transformation matrix
 TU : Displacement function transformation
 T : Membrane plate thickness
 SINA : Sine of angle determined by the intersection of diagonals and grid points 1 and 2
 DELPQ : Table of integrals for the 4 zones of the quadrilateral
 BMT : Transformation matrix
 DELTM : Membrane temperature
 TESS : Stress transformation
 TMS : Transformation to global system
 WK1 : Array containing DELPQ

5. Output Arguments:

SZALEL : Thermal stress matrix
 FT : Thermal load matrix
 AM4 } : Working arrays
 AM7 }
 FPB }

6. Error Results: None
7. Calling Sequence: (EM, ALPHM, TES, TU, T, SINA, DELPQ, BMT, DELTM, TESS, SZALEL, FT, TFS, TMS, FPB, AM4, AM7, WK1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: EM (1), ALPHM (1), TES (3,3), TU (3,4), FPB (16), IT (4), DELPQ (4,5,5), FT (1), AM1 (3), AM2 (3), AM4 (4,48), AM7 (2,48), SZLM (3), SZALEL (1), TESS (3,3), TFS (16,48), TMS (16,48), WK1 (100)
Total Storages is (195)₁₀
12. Subroutine User: PLUG1
13. Subroutines Required: MAB, MATB, CFMF
14. Remarks: None

1. Subroutine Name: CFMV
2. Purpose: To generate the membrane thermal load matrix in local coordinates
3. Equations and Procedures: Formulations are given in the documentation on the quadrilateral element in Volume I.
4. Input Arguments:
DELIC, : Table of integrals for 4 zones of quadrilateral
DELPQ
IT : Thermal vector
BMT : Transformation matrix
5. Output Arguments:
FPB1 : Thermal vector
6. Error Returns: None
7. Calling Sequence:
CALL CFMV (DELIC, FPB1, IT, BMT, DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: DELIC (4,5,5), FPB1 (16), IT (4), BMT (4), FPB (16),
DELPQ (4,5,5)
Total Storage is (310)₁₀
12. Subroutine User: CFMTS
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PRT1
2. Purpose: If IPRINT equals -1, intermediate matrices generated are printed out
3. Equations and Procedures: not applicable
4. Input Argument:
 - LT - Control on either membrane or flexural output
 - TU, TAØ, TGAMB, TØGBM, TGRBM - transformation matrices
 - FP, FT, CM, EO, IJPQ, C21, K21Ø, K21G - intermediate element matrices
 - KK - Control for dynamics print
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:
 - CALL PRT (LT, TU, EO, IJPQ, C21, K21Ø, K21G, TAØ, TGAMB, TØGBM, TGRBM, KM, CM, FP, FT)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage: EO(10), IJPQ(10), C21(105), K21Ø(136), K21G(210(, TAØ(8,8), TU(3,4), TGAMB(16,16), TØGBM(16,20), TGRBM(20,48), CM(1), FP(1), ET(1)
Total Storage is (538)₁₀
12. Subroutine User: PLUG 1
13. Subroutine Required: SYMPRT
14. Remarks: Matrices above defined in other writeups.

1. Subroutine Name: CK22

2. Purpose: Formulate the flexural stiffness matrix in local coordinates.

3. Equation and Procedures:

- (1) The following operations take place to formulate the transfer motion matrices
 - (a) Calls CTGB to evaluate the transformation to geometric coordinates as TGAMB
 - (b) Inverts TGAMB
 - (c) Calls CTØGB to formulate the transformation from oblique to geometric coordinates as TØGB
 - (d) Calls CTGRB to formulate the transformation from geometric to reference system coordinates as TGRBM
 - (e) If grid point axes transformations are used, call AXTRA2 to revise the flexural transformation TGRB.

(2) The flexural stiffness is then obtained by:

- (a) Formulating the rigidity as IPQ
- (b) Evaluating the [C] matrix for each zone by calling CC2
- (c) Assembling the [C] matrix for each zone into C21 by calling CC21.
- (d) Forming the following products:

$$[K22Ø] = [TGAMB]^T [C22] [TGAMB]$$

$$[K22G] = [TØGB]^T [K22Ø] [TØGB]$$

$$[K21\$] = [TGRB]^T [K22G] [TGRB]$$

Where [K21\$] is the desired flexural stiffness-matrix

4. Input Arguments:

K21\$	: Input Stiffness matrix from membrane contribution
IA\$\$Y	: control to add membrane plus flexural stiffness
NL	: Node points of element
NDIR	: Number of directions
NDEG	: Number of degrees of movement
ICØNT	: Control on grid point axis transformations
GPAXEL	: Grid point axis transformations
NNO	: Number of node points being transformed
AJ;BJ	: Local coordinates

TMS	}	: Transformation matrices
TF\$		
AMATT		
TRAØBQ		
TGN		
TPRIME	}	: Sine and cosine of angle defined by intersection of diagonals and points 1 and 2
\$INA		
COSA		
LT2		
EG		
T	}	: Modified materials property matrix
NI		
DELPO		
DELPO		
DELPO		
DELPO	}	: Flexural plate thickness
DELPO		
DELPO		
DELPO		
DELPO		
DELPO	}	: Array for assembly purposes
DELPO		
DELPO		
DELPO		
DELPO		
DELPO	}	: Table of integrals for 4 zones of quadrilateral
DELPO		
DELPO		
DELPO		
DELPO		

5. Output Arguments

K21\$	}	: Flexural contribution to stiffness matrix
TGAMB		
TØGB		
TGRB		
TGRBM		
C	}	: Transformation matrices
EO		
K220		
K226		
C22		
IPQ	}	: Intermediate matrices
C21		
C21		
C21		
C21		

6. Error Returns: None

7. Calling Sequence:

```
CALL CK22 (K21$, IASSY, NL, NDIR, NDEG, ICØNT, GPAXEL,
          NNO, AJ, BJ, AMAT, TRAØBQ, $INA, CØSA, TGN, TPRIME,
          LT2, TW, EG, T, NI, DELPO, TGAMB, TØGB, TGRB,
          K22Ø, K22G, C22, IPQ, C21, TGRMB, EO, TF$, TMS, C)
```

8. Input tapes: None

9. Output tapes: None

10. Scratch tapes: None

11. Storage:

```
AJ (1), BJ(1), AMAT (3,4), TRAØBQ (3,3), TGN (4,2,2),
TPRIME (3,3), TW (3,3), EG (10), EO (10), NI (8, 10),
DELPQ (4,5,5), TGAMB (16,16), C (28), C21 (105),
TGRBM (20, 48) K220 (136), K22G (210), C22 (105),
TØGB (16,20), TGRB (20, 48), IPQ (10), TF$ (16, 48)
TM$ (16, 48)
```

Total Storage is (269) 10

12. Subroutine user: PLUG1

13. Subroutines required are:

CTGB, MATI70, CTØGB, CTGRB, AXTRA2, BCB, CC2, CC21.

14. Remarks:

All formulations are given in the report for the quadrilateral thin shell element.

1. Subroutine Name: CTGB
2. Purpose: To formulate the flexural transformation matrix from local to geometric coordinates.
3. Equations and Procedures:
 - (1) The TGB matrix is formulated from local coordinates
 - (2) Using elements from AMAT, the lengths of the sides of each zone are computed and assembled in the TGN matrix
 - (3) The TØN matrix is evaluated for the 4 zones by first storing [TRAØBQ] into [TØG] and then solving $[TØN] = [TØG] [TGN]$
 - (4) The TGB matrix is then evaluated for the 4 zones as $[TGB] = [TØN]\{WX\} + [TØN]\{WY\}$
Where {WX} and {WY} are arrays of local coordinate values for the respective zones.
4. Input Variables:

AJ, BJ : Local coordinates
 AMAT : Transformation to local coordinates
 TRAØBQ : Transformation from local to oblique coordinates
5. Output Variables

TGAMB }
 TGB } : Transformation matrices
 TGN }
6. Error Returns: None
7. Calling Sequence:

CALL CTGB (AJ, BJ, AMAT, TRAØBQ, TGAMB, TGB, TGN)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage:

TØN (4,2,2), TGB (16, 16), TØG (2,2), XD (4), YD (4),
 WX (16), WY (16), L (4), TGAMB (16, 16), AJ (1), BJ (1),
 AMAT (3,4), TGN (4,2,2), TRAØBQ (3,3)
 Total storage is (681)₁₀
12. Subroutine User: CK22

13. Subroutines Called: None

14. Remarks:

All formulations are given in the report on the quadrilateral thin shell element.

1. Subroutine Name: MATI70
2. Purpose: To invert the [TGAMB] matrix
3. Equations: standard inverse technique where inverted matrix is stored back on top of itself.
4. Input Arguments:
 - N - order of matrix = 16
 - A - matrix to be inverted
5. Output Arguments
 - A - inverted matrix
 - I\$ING - error return
 - DETR - value of determinant
6. Error Return:
 - IF I\$ING = 1, singular matrix
7. Calling Sequence: CALL MATI70 (N, A, I\$ING, DETR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:
 - A(6,16), LI\$T(16), Total Storage is (329)₁₀
12. Subroutine User: CK22
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CTØGB
2. Purpose: formulate the flexural transformation matrix from oblique to geometric coordinates
3. Equations and procedures: The formulation is given in the report on the quadrilateral plate .
4. Input Arguments:
SINA, CØSA - sine and cosine of the angle defined by the diagonals and points 1 and 2
TGN - Transformation matrix
5. Output Arguments:
TØGB - the required transformation matrix
6. Error Returns: None
7. Calling Sequence :
CALL CTØGB (SINA, CØSA, TGN, TØGB)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: TGN(4,2,2), TØGB(16,20)
Total Storage is (78)₁₀
12. Subroutine User: CK22
13. Subroutines Required: None
14. Remarks : None

1. Subroutine Name: CTGRB
2. Purpose: formulate the flexural transformation matrix from geometric to reference system coordinates.
3. Equations and Procedures:
 - (1) Elements of the TPRIME matrix are first assembled into their respective positions
 - (2) If any midpoints are suppressed, the contribution of the midpoints is redistributed to the respective corner points
4. Input Arguments:

NL - node point numbers
TGN - transformation matrix for midpoints
TPRIME - transformation matrix to local coordinates
5. Output Arguments:

TGRB, TGRBM - transformation from geometric to reference system coordinates
6. Error Returns: None
7. Calling Sequence:

CALL CTGRB(NL, TGN, TPRIME, TGRBM, TGRB)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: NL(1), TGN(4,2,2), AI(4), BI(4), TPRIME(3,3)
TGRB(20,48), TGRBM(20,48)
Total storage is (345)₁₀
12. Subroutine User: CK22
13. Subroutines Required: None
14. Remarks: Formulation is given in report on Quadrilateral Plate

1. Subroutine Name: CC2
2. Purpose: Form for the 4 zones of the quadrilateral the flexural contributions to an intermediate matrix C.
3. Equations and Procedures: The formulation is given in report on quadrilateral thin shell element.
4. Input Arguments:
 - K : Control on zone contribution
 - IPQ : Rigidity matrix
 - DELPQ : Table of integrals for the 4 zones of the quadrilateral
5. Output Arguments:
 - C : Elements of the intermediate matrix
6. Error Returns: None
7. Calling Sequence:
 - CALL CC2 (K, IPQ, DELPQ, C)
8. Input tapes: None
9. Output tapes: None
10. Scratch Tapes: None
11. Storage:
 - IPQ (1), DELPQ (4,5,5), C (1), Total Storage is (448) 10
12. Subroutine User: CK22
13. Subroutine Required: None
14. Remarks: None

1. Subroutine Name: CFP
2. Purpose: Formulate the pressure load for the quadrilateral plate in reference system coordinates
3. Equations and Procedures:
 Call CFPB to generate the pressure load vector in reference system coordinates as FPB as defined by

$$\{FP\} = [TF\$]^T \{FPB\}.$$
4. Input Arguments:
 DELPQ : Table of integrals for the 4 zones of the quadrilateral
 P : Pressures at node points
 SINA : Sine of angle defined by intersection of diagonals and points 1 and 2 of the element
 TFB : Flexural transformation matrix
5. Output Arguments:
 FP : Pressure load vector in reference system coordinates
 FPB : Pressure load in local system.
6. Error Returns: None
7. Calling Sequence:
 Call CFP (DELPQ, P, SINA, FP, TFS, TMS, FPB)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:
 DELPQ (4,5,5), FP (4,8), FPB (16), TFB (16,48), TMS (16,48)
 Total Storage is (57)₁₀.
12. Subroutine User: CK22
13. Subroutine Required: CFPB, MATB
14. Remarks: The formulation is given in the documentation on the quadrilateral element in Volume I.

1. Subroutine Name: CFPB
2. Purpose: Formulate the pressure load in local coordinates for the quadrilateral thin shell element.
3. Equations and Procedures: Formulation is given in the report on the quadrilateral thin shell element.
4. Input Arguments:
 - DELPQ : Table of integrals for the 4 zones of the element
 - P : Pressure value.
 - SINA : Sine of angle defined by intersection of diagonals and points 1 and 2 of the element
5. Output Arguments:
 - FPB : Pressure load in local coordinates
6. Error Returns: None
7. Calling Sequence:
 - CALL CFPB (DELPQ, P, SINA, FPB)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage:
 - DELPQ (4,5,5), FPB (16)
 - Total Storage is (227)₁₀.
12. Subroutine User: CFP
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: CSTF
2. Purpose: To evaluate the flexural contribution to the stress matrix in reference system coordinates for the quadrilateral element.

3. Equations and Procedures:

- (1) Call CDF to evaluate the membrane displacement derivative matrix DFM.

- (2) Perform the following operations:

$$\begin{aligned} \text{(a)} \quad [AM5] &= [DFM] [TFS] \\ \text{(b)} \quad [AM6] &= [TW] [AM5] \\ \text{(c)} \quad [AM5] &= [EG] [AM6] \\ \text{(d)} \quad [AM6] &= [TE\$] [AM5] \end{aligned}$$

The AM6 matrix is then assembled into the stress matrix S.

- (3) Call CDX to evaluate the flexural derivatives matrix DFM.

- (4) After generating the G matrix, perform the following:

$$\begin{aligned} \text{(a)} \quad [AM5] &= [DFM] [TF\$] \\ \text{(b)} \quad [AM6] &= [TW] [AM5] \\ \text{(c)} \quad [AM5] &= [EG] [AM6] \\ \text{(d)} \quad [AM8] &= [G] [AM5] \end{aligned}$$

- (5) Evaluate another G matrix and call CDFX and CDFY to formulate the flexural derivate matrix DFM.

- (6) Perform the following operations:

$$\begin{aligned} \text{(a)} \quad [AM5] &= [DFM] [TR\$] \\ \text{(b)} \quad [AM6] &= [TW] [AM5] \\ \text{(c)} \quad [AM5] &= [EG] [AM6] \\ \text{(d)} \quad [AM7] &= [G] [AM5] \end{aligned}$$

The AM7 and AM8 matrices are then assembled into the stress matrix S.

4. Input Arguments:

T	:	Flexural thickness
TW	}	: Transformation matrices
TE		
TF\$		
TM\$		
EG	:	Material properties matrix

R1B }
 R2B } :Local coordinates
 R3B }
 R4B }
 CØSA, :Sine and cosine of angle defined by the intersection
 SINA of the diagonals and points 1 and 2 of the element

5. Output Arguments:

S :Stress matrix
 DFM }
 AM5 } :Intermediate matrices
 AM6 }
 AM7 }
 AM8 }

6. Error Returns: None

7. Calling Sequence:

CALL C\$TF (T, TW, EG, TES, R1B, R2B, R3B, R4B, CØSA, SINA,
S, TFS, TMS, DFM, AM5, AM6, AM7, AM8)

8. Input tapes: None

9. Output tapes: None

10. Scratch Tapes: None

11. Storage Required:

R1B (3), R2B (3), R3B (3), R4B (3), DFM (4, 16), TW (3,3),
 EG (10), TES (3,3), S (40, 48), AM5 (3, 48), AM6 (3, 48),
 AM7 (2, 48), AM8 (2, 48), G (2, 3), TF\$ (16, 48), TMS
 (16, 48).
 Total Storage is (446)₁₀.

12. Subroutine User: PLUG1

13. Subroutines Required:

MAB, MSB, CDF, CDFX, CDFY

14. Remarks:

The formulations are given in the documentation on the quadrilateral element.

1. Subroutine Name: CDF
2. Purpose: To evaluate the flexure derivative matrices for the 4 zones of the quadrilateral element
3. Equations and Procedures: Formulation is given in the documentation on the quadrilateral element.
4. Input Arguments:
 - IZ : Control on zone computation
 - $\left. \begin{array}{l} R1B \\ R2B \\ R3B \\ R4B \end{array} \right\}$: Local coordinates
5. Output Arguments:
 - DFM : Flexural derivative matrix
6. Error Returns: None
7. Calling Sequence:
 - CALL CDF (IZ, R1B, R2B, R3B, R4B, DFM)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required:
 - R1B (1), R2B (1), R3B (1), R4B (1), DFM (4, 16)
 - Total storage is (271)₁₀.
12. Subroutine User: CSTF
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: CDFX
2. Purpose: To evaluate the partial derivatives with respect to x of the flexural displacement matrix for the 4 zones of the quadrilateral element
3. Equations and Procedures: Formulation is given in the documentation on the quadrilateral element.
4. Input Arguments:
 - ITE : Control on constant (T_1)
 - IZ : Control on zone computation
 - CØSA, : Sine and cosine of angle defined by the intersection of the diagonals and points 1 and 2 of the element.
5. Output Arguments:
 - DFM : Flexural derivative matrix
6. Error returns: None
7. Calling Sequence:
 - CALL CDFX (ITE, CØSA, SINA, IZ, DFM)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required: DFM (4, 16)
Total Storage is $(176)_{10}$.
12. Subroutine User: CSTF
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: CDFY
2. Purpose: To evaluate the partial derivatives with respect to y of the flexural displacement derivative matrix for the 4 zones of the quadrilateral element
3. Equations and Procedures: Formulation is given in the documentation on the quadrilateral element
4. Input Arguments:
 - IZ : Control on zone computation
 - SINA : Sine of angle defined by the intersection of the diagonals and points 1 and 2 of the element
5. Output Arguments:
 - DFM : Flexural derivative matrix
6. Error Returns: None
7. Calling Sequence:
 - CALL CDFY (IZ, SINA, DFM)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required: DFM (4, 16), Total Storage is (143)₁₀
12. Subroutine User: CSTF
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: CFFTS

2. Purpose: To evaluate the flexural contribution to the thermal load and stress matrices for the quadrilateral element.

3. Equation and Procedures:

(1) The thermal stress is obtained by:

(a) $\{AM1\} = [EM] \{ALPHM\}$

(b) $\{AM2\} = [TES]^T \{AM1\}$

(c) $\{JT\} = [TW]^T \{AM2\}$

(d) $\{AM2\} = C^4 \{AM2\}$ where C^4 is a flexural constant

(e) $\{SZLF\} = [TESS] \{AM2\}$

SZLF is assembled into the thermal stress matrix SZALEL.

(2) The thermal load is obtained by:

(a) Define a flexural constant $C3$,

(b) $\{JT\} = C3 \times \{JT\}$,

(c) Call CFFV to formulate the thermal load in local system coordinates as $\{FPB\}$,

(d) Transform the thermal load to reference system coordinates as $[AM3] = [TFS]^T \{FPB\}$,
 $\{AM3\}$ is assembled into the thermal load matrix FT.

4. Input Arguments:

EM : Material properties matrix

ALPHM : Thermal coefficient matrix

TMS

TE\$

TW

TE\$\$

BMT

TF\$

} : Transformation matrices

DELTF : Flexural temperature

T : Flexural thickness

SINA : Sine of angle defined by intersection of diagonals and points 1 and 2 of the element

DELPQ : Table of integrals for the 4 zones

5. Output Arguments:

SZALEL : Thermal stress matrix

FT : Thermal load matrix

FPB

AM3

WK1

} : Intermediate matrices

6. Error Returns: None
7. Calling Sequence:
CALL CFFTS (EM, ALPHM, TES, TW, DELTF, T, TESS, SINA,
DELPQ, BMT, SZAEL, FT, TFS, TMS, FPB, AM3, WK1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
EM (10), ALPHM (3), TES (3,3), TW (3,3), DELPQ (4,5,5),
FT (48), FPB (16), JT (3), BMT (3,1), SZLF (3), SZAEL (1),
TESS (3,3), AM3 (48), AM1 (3), AM2 (3), TFS (16, 48),
TMS (16, 48), WK1 (100)
Total Storage is (222)₁₀.
12. Subroutine User: PLUG1
13. Subroutines Required: MAB, CFFV, MATB
14. Remarks: Formulation is given in documentation on the
quadrilateral element in Volume I.

1. Subroutine Name: CFFV
2. Purpose: To evaluate the flexural thermal load matrix in local system coordinates for the quadrilateral element
3. Equations and Procedures: Formulation is given in the report on the quadrilateral element in Volume I
4. Input Arguments:
DELC, :Table of integrals for the 4 zones of the quad-
DELPQ rilateral
JT :Flexural rigidity
BMT :Transformation matrix
5. Output Arguments:
FPB1 :Flexural load matrix in local coordinates
6. Error Results: None
7. Calling Sequence:
CALL CFFV (DELC, FPB1, JT, BMT, DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
DELC (4,5,5), DELPQ (4,5,5) FPB1 (16), JT (4), BMT (4),
FPB (16)
Total Storage is (365)₁₀.
12. Subroutine User: CFFTS
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CFMASS
2. Purpose: Evaluate the flexural mass matrix in local system coordinates for the quadrilateral thin shell element
3. Equations and Procedures: Formulation is given in report on the quadrilateral thin shell element.
4. Input Arguments:
 - T : Flexural thickness
 - DØØ : Area array of the 4 zones of the quadrilateral
 - SINA : Sine of angle defined by intersection of the diagonals and points 1 and 2 of the element
 - DENS : Density of element material
5. Output Arguments:
 - AMS : Elements of the mass matrix in local coordinate system
6. Error Returned: None
7. Calling Sequence:
 - CALL CFMA\$\$ (T, DØØ, SINA, DENS, AMS)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required:
 - DØØ (1), AMS (1), Total storage is (82)₁₀
12. Subroutine User: PLUG1
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PLUG7
2. Purpose: To formulate the element matrices for a frame element
3. Equations and Procedures: The following sequence of operations take place:
 - (1) Plug constants are set and checked against plug input
 - (2) Data is processed for:
 - (a) grid points
 - (b) element data such as area, inertia, etc.
 - (3) The length for the element and the direction cosines are determined and stored in TPRIME.
 - (4) Call CTS and CTCQ to formulate transformation matrices TS and TCQ. The eccentricity of the element is taken into account by calling CECC and modifying the TS matrix.
 - (5) The transformation to systems coordinates is performed as $[TCQS] = [TCQ][TS]$ and if grid point axes transformations are necessary, $[TCQS]$ is modified.
 - (6) The matrix $[KS]$ is evaluated and then pre and post multiplied by $[TCQS]$ to form the stiffness matrix as $[KSEL]$.
 - (7) Dependent on the type of analysis, the incremental and mass matrices may be computed.
 - (8) The thermal load is set equal to zero.
 - (9) The stiffness matrix is rearranged into the stress matrix and the thermal stress matrix set equal to zero.
 - (10) If the print option is not equal to 0 calls P7PRT to print out intermediate computations.

4. Input Arguments:

IPL	:	Plug number
NN	}	: Number of nodes
NNO		
XC	}	: Element coordinates
YC		
ZC		
TEL		
TEL	:	Temperature array
PEL	:	Pressure array
QS	:	Initial displacements
NØRD	:	Order of stiffness matrix
NERR	:	Error return
KK	}	: Controls on element matrices to be computed
KF		
K8		
KM		
ET		
KVM		
KN		

EPSIO } : Prestress and prestrain values
 SØ
 MAT : Material properties array
 EXTRA : Element geometric properties
 GPAXEL : Grid point axis transformations
 NDIR } : Number of directions and degree control for
 NDEG } grid point axis transformation
 ICØNT : Control on grid point axis

5. Output Arguments:

KSEL : Stiffness matrix
 GT : Gradient
 FTEL : Thermal load matrix
 SEL : Stress matrix
 SZALEL : Thermal stress matrix
 AMASS : Mass Matrix
 DAMPV } : Viscous and Structural Damping Matrices
 DAMPS }
 NRSEL : Number of rows in stress matrix
 NL : Node point numbers
 NØINK } : Number of elements in the stiffness, mass, viscous
 NMASS } damping, structural damping and stress matrices
 NDMPV }
 NDMP\$ }
 N\$EL }

6. Error Returns: If third node point is not present, then exit. Standard tests on plug constants.

7. Calling Sequence:

CALL PLUG7 (IPL, NNO, XC, YC, ZC, TEL, PEL, QS, IP, NØRD,
 NERR, NØINK, K\$EL, AN1, FTEL, SEL, SZALEL, AMASS,
 DAMPV, DAMPS, NRSEL, NN, NL, NMASS, NDMPV, NDMPS,
 NSEL, KK, KF, K8, KM, ET, KVM, KN, IUSEL, EPSIO, SC,
 MAT, EXTRA, GPAXEL, NDIR, NDEG, ICØNT)

8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 8476₁₀.
12. Subroutine User: ELPLUG
13. Subroutines Required: ELTEST, CTCQ, MAB, AXTRA2, CECC, BCB, MSB, MATB, P7PRT, CTS, INCRE
14. Remarks: Formulations are given in report on Frame Element.

1. Subroutine Name: CTS
2. Purpose: To evaluate the transformation matrix from local to referenced system coordinates for the frame element
3. Equations and Procedures: Formulation is given in documentation on frame element.
4. Input Arguments:
TPRIME : Local coordinates transformation matrix
5. Output Arguments:
TS : Required transformation matrix
6. Error Returns: None
7. Calling Sequence: CALL CTS (EE1, EE2, TS, TPRIME)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required: TS (12,12), TPRIME (3,3)
Total Storage is (105)₁₀
12. Subroutine User: PLUG7
13. Subroutines Required: None
14. Remarks: EE1, EE2 - Dummy arguments

1. Subroutine Name: CTCQ
2. Purpose: To formulate the transformation matrix to local system coordinates for the frame element
3. Equations and Procedures: Formulations are given in documentation on Frame Element.
4. Input Arguments:
 TGQ : Elements of input transformation
 L }
 L2 }
 L3 }: Length, Length squared, etc.
 L4 }
 L5 }
5. Output Arguments:
 TCQ : Required transformation matrix
6. Error Returns: None
7. Calling Sequence: CALL CTCQ (TCQ, L, L2, L3, L4, L5)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage required: TCQ (12,12) Total storage is (183)₁₀.
12. Subroutine User: PLUG7
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: CECC
2. Purpose: To compute modifications to the transformation matrix to account for eccentricity for the frame element
3. Computations and Procedures: Formulation is given in documentation on Frame Element.
4. Input Arguments:
 - EE1 } : Eccentricity matrices
 - EE2 }
 - TS : Transformation matrix to be modified
5. Output Arguments:
 - TS : Modified transformation matrix
6. Error Returns: None
7. Calling Sequence: CALL CECC (EE1, EE2, TS)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage required: TS (12,12), EE1 (3), EE2 (3)
Total Storage is (146)₁₀
12. Subroutine User: PLUG7
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: INCRE
2. Purpose: To evaluate the incremental matrix for the frame element.
3. Equations and Procedures: Formulation is given in report on Frame Element.
4. Input Arguments:
 - CØN1 } : Constants set equal to 1.0
 - CØN2 }
 - L } : Physical properties of element
 - J1 }
 - C : Input displacement matrix
 - TCQS : Transformation matrix
5. Output Arguments:
 - AN1 : Element incremental stiffness matrix transformed to reference system coordinates.
 - AI } : Intermediate matrices
 - CI }
 - N }
 - AN2 }
 - AN3 }
6. Error Returns: None
7. Calling Sequence:


```
CALL INCRE (CØN1, CØN2, L, J1, AN1, AN2, C, TCQS, N, AN3,
            AI, CI)
```
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:


```
AN1 (171), AN2 (78), N (78), AN3 (78), AI (3,5), C (1),
TCQS (12,12), CI (18)
```
12. Subroutine User: PLUG7
13. Subroutines Required: BCB
14. Remarks: None

1. Subroutine Name: P7PRT
2. Purpose: To print out intermediate computations and matrices from the frame element
3. Equations and Procedures: Not applicable.
4. Input Arguments:

NERR	:	Error test
GRI	}	: Gradient terms
GR4		
GRT		
PHI1	}	: Energy terms
PHI4		
AMMAS	:	Mass Matrix
EX	}	: Material and geometric properties
G		
A		
AJ1		
J1		
L		
AIY		
AIZ		
EEI	}	: Control on element matrix computation
ET		
RN		
R1		
R2		
R3	}	: Intermediate computations
RM		
TPRIME		
TCQ		
T\$	}	: Transformation matrices
TCQS		
AN1	}	: Incremental matrices
AN2		
KS	:	Stiffness matrix
C	:	Intermediate displacement matrix
IPRINT	:	Print option
5. Output Arguments: Not applicable
6. Error Returns: If node point 3 equal to zero, then exit.
7. Calling Sequence:

```
CALL P7PRT (NERR, GR1, GR4, PHI1, PHI4, AMMAS, G, A, AJ1,
            L, AIY, AIZ, RN, R1, R2, R3, AJ, TPRIME, KS, TCQ,
            TS, TCQS, C, QS, AN2, AN1, RM, EX, EE1, PRINT, AN1
            ET)
```

- 8. Input tapes: None
- 9. Output tapes: None
- 10. Scratch tapes: None
- 11. Storage Required:
GR (1), GR⁴ (1), AMMAS (1), RN (1), RM (1), R1 (1), R2 (1),
R3 (1), AJ (1), TPRIME (3,3), KS (1), TCQ (12,12), TS
(12,12), C (1), QS (1), AN2 (1), AN1 (1), TCQS (12,12),
GRT (1). Total storage is (687)₁₀.
- 12. Subroutine User: PLUG7
- 13. Subroutines Required: SYMPRT
- 14. Remarks: None

1. Subroutine Name: PLUG2
2. Purpose: Control generation of element matrices for the triangular thin shell element
3. Equations and Procedures:
 - a) Call subroutine ELTEST to verify input control values
 - b) Call subroutine DTAPR to calculate sub-element coordinates and boundaries
 - c) Call subroutine MATPR to generate material properties matrices
 - d) Call subroutine NEWFT1 to apply revised thermal load formulation, if necessary
 - e) Call subroutine PTBM to generate sub-element to local geometric coordinate system transformation matrix
 - f) Call subroutine PTMGS to generate local geometric coordinates to reference system coordinates transformation matrix
 - g) Call subroutine MAB to combine transformation matrices generated in e) and f) above into one matrix that will apply transformation from sub-element to reference system coordinates
 - h) If grid point axes are to be applied then call subroutine AXTRA2 to appropriately modify final transformation matrix generated in g) above
 - i) Call subroutine DPQINT to evaluate the integrals over the three sub-elements
 - j) Call subroutine PKM to generate the membrane contribution to the element stiffness matrix
 - k) Call subroutine PSTM to generate the membrane contribution to the element stress matrix
 - l) Call subroutine PFMTS to generate membrane contribution to element thermal load and thermal stress matrices
 - m) If requested, call subroutine APRT to print intermediate results
 - n) The flexural contributions to the element matrices are then generated with the following flexure sub-routines performing the same function as their membrane counterparts

PTBF	is the flexural counterpart to	PTBM
PTFGS	" " " " "	PTMGS
PKF	" " " " "	PKM
PSTF	" " " " "	PSTM
PFFTS	" " " " "	PFMTS

- o) Call subroutine PFP to generate element pressure load matrix

- p) Call subroutines PNC1 and PNG1 to generate element incremental stiffness matrix (non-functional)
- q) Call subroutine PLAS2 to generate plasticity pre-multipliers (non-functional)

4. Input Arguments:

IPL	: Internal element identification number (2)
NNO	: Number of element defining points (6)
XC	: Coordinates of element defining points
YC	
ZC	
TTL	: Temperatures at element defining points
PEL	: Pressures at element defining points
QS	: Input displacements at element defining points (not used)
IP	: Not used
NORD	: Total element degrees of freedom (36)
NOINK	: Number of storages required for element stiffness matrix $((NORD * (NORD + 1))/2)$
NN	: Not used
NL	: Array containing grid point numbers of element defining points
KK	: Suppression control for element stiffness matrix
KF	: Suppression control for element thermal and pressure load matrices
K8	: Suppression control for element stress matrix
KTS	: Suppression control for element thermal stress matrix
KM	: Suppression control for element mass matrix
FN	: Not used
KVM	: Not used
KN	: Suppression control for element incremental stiffness matrix
IUSEL	: Not used
EPSLON	: Input pre-strains (not used)
SIGZER	: Input pre-stresses (not used)
MAT	: Input temperature interpolated material properties
EXTRA	: Special element input
GPAXEL	: Grid point axes transformation matrices
NDIR	: Number of directions of element defining points (3)
NDEG	: Number of solution degrees of freedom (2 - translation and rotation)
ICONT	: Grid point axes indicator

5. Output Arguments:

NERR : Error indicator
AK : Element stiffness matrix
ANEL : Element incremental stiffness matrix
FTEL : Element thermal and pressure load matrix
S : Element stress matrix
SZALEL : Element thermal stress matrix
AMASS : Element mass matrix
DAMPV : Element viscous damping matrix
DAMPS : Element structural damping matrix
NRSEL : Number of rows in element stress and thermal stress matrices
NMASS : Number of storages required for element mass matrix
NDMPV : Number of storages required for element viscous damping matrix
NDMPS : Number of storages required for element structural damping matrix
NSEL : Number of storages required for element stress matrix

6. Error Returns:

If no error, then NERR is set to zero
If IPL $\neq$ 2, then NERR is set to one
If NNO $\neq$ 6, then NERR is set to two
If NORD $\neq$ 36, then NERR is set to four

7. Calling Sequence:

(IPL, NNO, XC, ZC, YC, TTL, PEL, QS, IP, NORD, NERR, NOINK, AK, ANEL, FTEL, S, SZALEL, AMASS, DAMPV, DAMPS, NRSEL, NN, NL, NMASS, NDMPV, NDMPS, NSEL, KK, KF, K8, KTS, KM, FN, KVM, KN, IUSEL, EPSLON, SIGZER, MAT, EXTRA, GPAXEL, NDIR, NDEG, ICONT)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 3226_8 (1686_{10}).

12. Subroutine User: ELPLUG

13. Subroutines Required:

ELTEST
DTAPR
MATPR
NEWFT1
PTBM
PTMGS
MAB
AXTRA2
DPQINT
MINV
PKM
PSTM
PFMTS
APRT
PTBF
PTFGS
PKF
PFP
PSTF
PFFTS
PNC1
PNG1
EPRT
PLAS2

14. Remarks: None

1. Subroutine Name: ASSY2
2. Purpose: Assemble membrane and flexure contributions into element stiffness matrix for triangular thin shell element
3. Equations and Procedures: The elements of the C1 matrix are summed into the C2 matrix as directed by the input array IASY.
4. Input Arguments:
 - C1 : Array containing input elements to be assembled
 - IASY : Array containing assembly instructions
 - N1 : Order of C1
5. Output Arguments:
 - C2 : Assembled matrix
6. Error Returns: None
7. Calling Sequence:
 - (C2, C1, IASY, N1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 172_8 (122_{10}).
12. Subroutine User:
 - PKM
 - PKF
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: DCD
2. Purpose: To evaluate the triple matrix product of a diagonal matrix D, a symmetric matrix S, and the diagonal matrix D.
3. Equations and Procedures:

$$AN_{nn} = \sum_n \sum_n D_{nn} * S_{nn} * D_{nn} \quad (\text{See remarks})$$
4. Input Arguments:
 SYM: Elements of symmetric matrix [S]
 D: Elements of a diagonal matrix [D]
 N: Order of [S] and [D] matrices
5. Output Arguments:
 AN: Elements of matrix product
6. Error Returns: None
7. Calling Sequence:
 CALL DCD (SYM, D, AN, N)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 SYM(1), D(1), AN(1), ROW(20)
12. Subroutine User: PKF, PKM
13. Subroutines Required: None

14. Remarks: The summations occur over

$$\begin{bmatrix} d_{11} & 0 & . & . & . & 0 \\ 0 & d_{22} & & & & . \\ . & . & . & & & . \\ . & & . & . & & . \\ . & & & . & . & . \\ 0 & . & . & . & . & d_{nn} \end{bmatrix} * \begin{bmatrix} S_{11} & & & & & \\ S_{21} & S_{22} & & & & \\ . & . & & & & \\ . & . & & & & \\ . & . & & & & \\ S_{n1} & S_{n2} & \dots & S_{nn} \end{bmatrix} * \begin{bmatrix} d_{11} & 0 & . & . & . & 0 \\ 0 & d_{22} & & & & . \\ . & . & . & & & . \\ . & & . & . & & . \\ . & & & . & . & . \\ 0 & . & . & . & . & d_{nn} \end{bmatrix}$$

All redundant multiplications (i.e. those where zero elements exist in the D matrix and those where the upper elements of the S matrix would be considered) are dispensed within the program and only significant multiplications take place.

1. Subroutine Name: DTAPR
2. Purpose: Create three sub-elements and transformation matrix from system to local coordinates
3. Equations and Procedures: The sub-element coordinates are calculated from the system coordinates by generating a transformation matrix and applying it to the system coordinates array.
4. Input Arguments:
 - R1,R2,R3 : Reference system coordinates
 - E1,E2,E3,E : Arrays containing coordinate differences
 - R12,R13 : Work storage
 - COORDS : Reference system coordinates
5. Output Arguments:
 - RO : Origin of sub-element's coordinate system
 - RL1,RL2,RL3 : Local sub-elements coordinates
 - TGS : Transformation from reference system to local sub-element coordinates matrix
 - COORDL : Local sub-elements coordinates
6. Error Returns: None
7. Calling Sequence:
 - (R1, R2, R3, RL1, RL2, RL3, E1, E2, E3, E, TGS, RO, R12, R13, COORDS, COORDL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 560_8 (368_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: MAB
14. Remarks: None

1. Subroutine Name: MATPR
2. Purpose: Generate material properties matrices for triangular thin shell element
3. Equations and Procedures: The material properties matrix, EM, is generated dependent upon the formulation option selected; plane stress, plane strain or normal. The matrix angle and stress angle is determined by examining the extra element defining points. The material properties matrix is then oriented to the desired material angle and the stress angle transformation matrix is generated.
4. Input Arguments:

NL	: Array containing grid point numbers of element defining points
XC,YC,ZC	: Arrays containing reference system coordinates for element defining points
EX,EY,EZ	: Young's Moduli
GXY	: Rigidity Modulus
VXY,VZX,VYZ	: Poisson's Ratios
ALPHAX,ALPHAY	: Coefficients of thermal expansion
GAMXY	: Material angle
T	: Thickness
EXGRID	: Array containing coordinate differences for stress angle definition points
EXGRDL	: Array containing coordinate differences for material angle definition points
ALPHAM	: Not used
ALPHAG	: Not used
TGS	: Transformation matrix from reference system to sub-element coordinates
IST	: Plane strain, stress control
RL,R2,R3	: Not used
ROB	: Origin of sub-element coordinate system
RL1,RL2,RL3	: Local sub-element coordinates
EES	: Work storage
NEXGR	: Work storage
AMAT	: Local sub-element coordinates
L,M	: Work storage
5. Output Arguments:

EM	: Material properties matrix
EG	: Transformed material properties matrix (oriented to material angle)
TES	: Material angle transformation matrix
TESS	: Stress angle transformation matrix

6. Error Returns: None

7. Calling Sequence:

(NL, XC, YC, ZC, EX, EY, GXY, VXY, EZ, VZX, VYZ,
ALPHAX, ALPHAY, GAMXY, T, EM, EG, EXGRID, EXGRDL,
ALPHM, ALPHG, TGS, IST, R1, R2, R3, ROB, RL1, RL2,
RL3, EES, TES, TESS, NEXGR, AMAT, L,M)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 1277_8 (703_{10}).

12. Subroutine User: PLUG2

13. Subroutines Required:

MINV
MAB
BCB

14. Remarks: None

1. Subroutine Name: NEWFT1
2. Purpose: Generate membrane and flexural thermal loads for triangular thin shell element in local coordinates
3. Equations and Procedures:

$$\begin{aligned} \text{BCT} &= \text{F} * \text{CT} \\ \text{BMT} &= \text{BCT} * \text{TEMM} \\ \text{BFT} &= \text{BCT} * \text{TEMF} \end{aligned}$$

where F and CT are geometric matrices of local coordinates, and TEMM and TEMF are membrane and flexure temperatures, respectively, at the element defining points.

4. Input Arguments:

DELTM : Average membrane temperature
 DELTF : Average flexure temperature
 RL1, RL2, RL3 : Local coordinates
 TZ : T₀ for structure
 F, BCT, CT : Work storage
 TEL : Temperatures at element defining points
 TEMM, TEMF, L, M : Work storage

5. Output Arguments:

BMT : Membrane thermal load in local coordinates
 BFT : Flexure thermal load in local coordinates

6. Error Returns: None

7. Calling Sequence:

(DELTM, DELTF, RL1, RL2, RL3, TZ, BMT, BFT, F, BCT, CT, TEL, TEMM, TEMF, L, M)

8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 407₈ (263₁₀).
12. Subroutine User: PLUG?

13. Subroutine Required:

MINV
MAB

14. Remarks: None

1. Subroutine Name: PTBM
2. Purpose: Generate membrane transformation matrix from sub-element to geometric coordinate system
3. Equations and Procedures: The transformation matrix is generated directly from sub-element coordinate values and inversion.
4. Input Arguments:
 - TGSM : Not used
 - RL1,RL2,RL3 : Sub-element coordinates
 - L,M : Work storage
5. Output Argument:
 - TBM : Sub-element to geometric coordinate system membrane transformation matrix
6. Error Returns: None
7. Calling Sequence:
 - (TBM, TGSM, RL1, RL2, RL3, L, M)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 440_8 (288_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: MINV
14. Remarks: None

1. Subroutine Name: PFMGS
2. Purpose: Generate geometric to reference coordinate system membrane transformation matrix
3. Equations and Procedures: The transformation matrix is generated by utilizing the TGS matrix. The effect of eccentricities and mid-point suppression is also reflected in the generation of the TGSM matrix.
4. Input Arguments:
 - NL : Array containing element defining grid point numbers
 - EEZ : Eccentricity
 - TBM : Not used
 - TGS : Reference system to sub-element transformation matrix
5. Output Arguments:
 - TGSM : Geometric to reference coordinate system membrane transformation matrix
6. Error Returns: None
7. Calling Sequence:
 - (NL, EEZ, TBM, TGSM, TGS)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 374_8 (252_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: DPQINT
2. Purpose: Generate integrals over the three sub-elements of a triangular thin shell element
3. Equations and Procedures: The integrals are calculated by using the triangular integration package controlled by the function subprogram AI. The output values of the integrals are placed in the array DELPQ.
4. Input Arguments:
RL1,RL2,RL3 : Sub-element coordinates
R,Z,TEMP : Work storage
5. Output Arguments:
DELPQ : Array containing integral values
6. Error Returns: None
7. Calling Sequence:
(DELPQ, RL1, RL2, RL3, R, Z, TEMP)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 572_8 (378_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: AI (Function)
14. Remarks: None

1. Subroutine Name: PKM
2. Purpose: Generate membrane contribution to triangular thin shell element stiffness matrix
3. Equations and Procedures: The membrane contribution to the element stiffness matrix is formed by generating sub-element stiffness matrices, assembling them into a work area and then transforming from the work area to the reference coordinate system.
4. Input Arguments:

AK1	: Work storage
DELPQ	: Sub-element integrals
EM	: Material properties matrix
EG	: Material properties matrix oriented to material angle
TMS	: Sub-element to reference coordinate system transformation matrix
TFS	: Not used
IASSEM	: Array containing assembly parameters
AD	: Work storage
CM	: Work storage
AIJ	: Work storage
IPRT	: Element print control
EX	: Not used
EY	: Not used
GXY	: Not used
VXY	: Not used
ALPHAX	: Not used
ALPHAY	: Not used
GAMXY	: Not used
T	: Membrane thickness
5. Output Argument:

AK	: Membrane contribution to element stiffness matrix
6. Error Returns: None
7. Calling Sequence:

(AK, AK1, DELPQ, EM, EG, TMS, TFS, IASSEM, AD, CM, AIJ, IPRT, EX, EY, GXY, VXY, ALPHAX, ALPHAY, GAMXY, T)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None

11. Storage Required: Total storage is 513_8 (331_{10}).

12. Subroutine User: PLUG2

13. Subroutines Required:

SYMPRT
DCD
ASSY2
BCB

14. Remarks: None

1. Subroutine Name: PSTM
2. Purpose: Generate membrane contribution to element stress matrix for the triangular thin shell element
3. Equations and Procedures: The membrane contributions to the element stress matrix are generated by first forming the stress values in local coordinates, then transforming to reference system coordinates and finally applying the stress angle transformation.
4. Input Arguments:

RL1	: Sub-element coordinates
RL2	
RL3	
TMS	: Sub-element to reference coordinate system transformation matrix
TFS	: Not used
EM	: Not used
EG	: Material properties matrix oriented to material angle
SN	: Work storage
AM1	: Work storage
AM2	: Work storage
TES	: Stress angle transformation matrix
EX	: Not used
EY	: Not used
GXY	: Not used
VXY	: Not used
ALPHAX	: Not used
ALPHAY	: Not used
GAMXY	: Not used
T	: Membrane thickness
R	: Work storage
U	: Work storage
X	: Work storage
Y	: Work storage
5. Output Arguments:

S	: Membrane contribution to element stress matrix
6. Error Returns: None

7. Calling Sequence:
(S, RL1, RL2, RL3, TMS, TFS, EM, EG, SN, AM1, AM2,
TES, EX, EY, GXY, VXY, ALPHAX, ALPHAY, GAMXY, T, R,
U, X, Y)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 406_8 (262_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required:
MAB
MSB
MATB
14. Remarks: None

1. Subroutine Name: PFMTS
2. Purpose: Generate membrane thermal load and membrane thermal stress matrices for the triangular thin shell element
3. Equations and Procedures: Subroutine PFMV1 is called to generate the thermal load matrix in geometric coordinates from BMT. This matrix is then transformed to reference system coordinates by TMS. The thermal stress matrix is generated and the stress angle applied by TESS.
4. Input Arguments:

DELTM	: Average membrane temperature
TES	: Material angle transformation matrix
TESS	: Stress angle transformation matrix
BMT	: Membrane thermal load contribution in sub-element coordinate system
EM	: Not used
EG	: Material properties matrix oriented to material angle
TMS	: Sub-element to reference coordinate system transformation matrix
TFS	: Not used
EX	: Not used
EY	: Not used
GXY	: Not used
VXY	: Not used
FMV	: Work storage
ALPHAX, ALPHAY	: Coefficients of thermal expansion
GAMXY	: Not used
T	: Membrane thickness
TO	: Not used
TI	: Not used
FME	: Work storage
EMI	: Work storage
EM1	: Work storage
SZLM	: Work storage
SZLM1	: Work storage
WRK	: Work storage
DELFQ	: Array containing sub-element integral values

5. Output Arguments:

FT	: Membrane contribution to element thermal load matrix
SZALEL	: Membrane contribution to element thermal stress matrix

6. Error Returns: None
7. Calling Sequence:
(FT, DELTM, SZALEL, TES, TESS, BMT, EM, EG, TMS, TFS,
EX, EY, GXY, VXY, FMV, ALPHAX, ALPHAY, GAMXY, T, TO,
TI, FME, EMI, EM1, SZLM, SZLM1, WRK, DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 314_8 (204_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required:
PFMV1
MAB
MATB
MSB
14. Remarks: None

1. Subroutine Name: PFMV1
2. Purpose: Generate membrane contribution to element thermal load matrix in local coordinates
3. Equations and Procedures: The integral values across the sub-elements are re-arranged. The membrane contribution for each sub-element is generated in FMV by direct formulation as a function of the integral values and the material properties matrix. The sub-element matrices are placed in FMV1 and pre-multiplied by BMT.
4. Input Arguments:
 - DELC : Array containing sub-element integral values
 - EG : Material properties matrix oriented to material angle
 - BMT : Array containing revised formulation for membrane thermal load matrix in local coordinates
 - FMV : Work storage
 - T : Membrane thickness
5. Output Arguments:
 - FMV1 : Membrane thermal load matrix in local coordinates
 - DELPQ : Re-arranged sub-element integral values
6. Error Returns: None
7. Calling Sequence:

(FMV1, DELC, EG, BMT, FMV, DELPQ, T)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 672_8 (442_{10}).
12. Subroutine User: PFMTS
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: APRT
2. Purpose: Provide print of intermediate triangular thin shell element computations
3. Equations and Procedures: None
4. Input Arguments:

LT	: Membrane/flexure indicator
LT1	: Not used
LT2	: Not used
DELPQ	: Array containing sub-element integral values
RL1, RL2, RL3	: Sub-element coordinates
R1, R2, R3	: Reference system element coordinates
RO	: Origin of sub-element coordinate system
E1, E2, E3, E	: Sub-element coordinate differences
TGS	: Sub-element to geometric coordinates transformation matrix
TBF	: Flexure sub-element to geometric system coordinates transformation matrix
TGSF	: Flexure geometric to reference system coordinates transformation matrix
TMS	: Membrane sub-element to reference system coordinates transformation matrix
TFS	: Flexure sub-element to reference system coordinates transformation matrix
EM	: Material properties matrix
EG	: Material properties matrix oriented to material angle
TES	: Material angle transformation matrix
TBM	: Membrane sub-element to geometric coordinates transformation matrix
TGSM	: Membrane geometric to reference system coordinates transformation matrix
5. Output Arguments: None
6. Error Returns: None

7. Calling Sequence:

(LT, LT1, LT2, DELPQ, RL1, RL2, RL3, R1, R2, R3, R0,
E1, E2, E3, E, TGS, TBF, TGSF, TMS, TFS, EM, EG,
TES, TBM, TGSM)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 1105_8 (581_{10}).

12. Subroutine User: PLUG2

13. Subroutines Required: None

14. Remarks: None

1. Subroutine Name: PTFGS
2. Purpose: Generate flexure geometric to reference system coordinates transformation matrix
3. Equations and Procedures: The flexure geometric to reference system coordinates transformation matrix is generated from the TGS matrix and the sub-element coordinates. The effect of mid-point suppress contained in this transformation matrix suppression.
4. Input Arguments:
 - NL : Array containing element definition grid point numbers
 - TGS : Sub-element to geometric transformation matrix
 - TBF : Not used
 - XD, : Work storage
 - YD,
 - L,
 - AI,
 - BI
 - AMAT : Array containing sub-element coordinates
5. Output Arguments:
 - TGSF : Flexure geometric to reference system coordinates transformation matrix
6. Error Returns: None
7. Calling Sequence:

(NL, TGS, TBF, TGSF, XD, YD, L, AI, BI, AMAT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 532_8 (346_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PKF
2. Purpose: Generate the flexure contribution to the triangular thin shell element stiffness matrix
3. Equations and Procedures: The sub-element flexure contributions are generated and assembled into a work area. A transformation is then applied to the reference coordinate system.
4. Input Arguments:
 - IASSY : Control indicating flexure contribution will supplement membrane contribution or flexure contribution alone is requested
 - DELPQ : Array containing sub-element integrals
 - EM : Not used
 - EG : Material properties matrix oriented to material angle
 - TMS : Not used
 - TFS : Flexure sub-element to system reference coordinates transformation matrix
 - IASSEM : Work storage for assembly control array
 - AD : Work storage
 - CM : Work storage
 - AIJ : Work storage
 - EX : Not used
 - EY : Not used
 - GXY : Not used
 - VXY : Not used
 - ALPHAX : Not used
 - ALPHAY : Not used
 - GAMXY : Not used
 - T : Flexure thickness
 - IPRT : Intermediate results print control
 - AK1 : Work storage
 - ROW : Work storage
 - ROWN : Work storage
5. Output Argument:
 - AK : Flexure contribution to element stiffness matrix
6. Error Returns: None

7. Calling Sequence:

(AK, IASSY, DELPQ, EM, EG, TMS, TFS, IASEM, AD, CM,
AIJ, EX, EY, GXY, VXY, ALPHAX, ALPHAY, GAMXY, T,
IPRT, AK1, ROW, ROWN)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 467_8 (311_{10}).

12. Subroutine User: PLUG2

13. Subroutines Required:

DCD
ASSY2
CCB

14. Remarks: None

1. Subroutine Name: CCB
2. Purpose: Perform triple product multiplication, $A^T S A$, where S is a symmetric matrix stored lower half by rows
3. Equations and Procedures: A row of the intermediate matrix product $A^T S$ is generated at a time. From the product of this row and A, a row of the final triple product is generated.

Options are present for scalar multiplication of the triple product, summing the triple product into an existing matrix, and deleting upper rows of the matrices from the operation.

4. Input Arguments:

A : First input matrix, doubly dimensioned in calling program
 SYM : Second input matrix, symmetric, singly subscripted, stored lower half by rows
 ND, MD : Dimensioned size of A
 N, M : Actual size of A
 N1 : Number of upper rows to be deleted in the operation
 SCAL : Scalar multiplier
 IASSY : Sum option indicator
 ROW, ROWN: Work storage

5. Output Argument:

AN : Triple product of $A^T S A$, symmetric, singly subscripted, stored lower half by rows

6. Error Returns: None

7. Calling Sequence:

(A, SYM, AN, ND, MD, N, M, N1, SCAL, IASSY, ROW, ROWN)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is $430_8(280_{10})$.

12. Subroutine User: PLUG2

13. Subroutines Required: None

14. Remarks: None

1. Subroutine Name: PFP
2. Purpose: Generate element pressure load matrix for the triangular thin shell element
3. Equations and Procedures: The element pressure load matrix is generated in local coordinates and then transformed to reference system coordinates.
4. Input Arguments:
 - TMS : Not used
 - TFS : Flexure sub-element to reference system transformation matrix
 - DELPQ : Array containing sub-element integral values
 - P : Pressures at element definition points
 - FPB : Work storage
5. Output Arguments:
 - FP : Element pressure load matrix
6. Error Returns: None
7. Calling Sequence:
 - (FP, TMS, TFS, DELPQ, P, FPB)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 - Total storage is 200_8 (128_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: MATB
14. Remarks: None

1. Subroutine Name: PFFTS
2. Purpose: Generate flexure contribution to element thermal load and thermal stress matrices for the triangular thin shell element
3. Equations and Procedures: The flexure contribution to the element thermal load matrix in local coordinates is generated by calling subroutine PFFV1. The material angle transformation is applied and the transformation from local to reference system coordinates is performed.

The flexure contribution to the element thermal stress matrix is generated and transformed to the selected stress angle.

4. Input Arguments:

DELTF	: Average flexural temperature
TES	: Material angle transformation matrix
TESS	: Stress angle transformation matrix
BFT	: Flexural thermal load formulation revision
EM	: Not used
EG	: Material properties matrix, oriented to material angle
TMS	: Not used
TFS	: Flexure sub-element to reference system coordinates transformation matrix
EX	: Not used
EY	: Not used
GXY	: Not used
VXY	: Not used
FFV	: Not used
ALPHAX	: Not used
ALPHAY	: Not used
GAMXY	: Not used
T	: Flexure thickness
TO	: Not used
TI	: Not used
EFI	: Work storage
FFE	: Work storage
FF	: Work storage
SZLF	: Work storage
SZLF1	: Work storage
EFl	: Work storage
WRK	: Work storage
DELPQ	: Array containing sub-element integrals

5. Output Arguments:

FT : Flexure contribution to element thermal
load matrix
SZALEL : Flexure contribution to element thermal
stress matrix

6. Error Returns: None

7. Calling Sequence:

(FT, DELTF, SZALEL, TES, TESS, BFT, EM, EG, TMS,
TFS, EX, EY, GXY, VXY, FFV, ALPHAY, ALPHAY, GAMXY,
T, TO, TI, EFI, FFE, FF, SZLF, SZLF1, EF1, WRK,
DPLPQ)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required: Total storage is 335₈(221₁₀).

12. Subroutine User: PLUG2

13. Subroutine Required: PFFV1

PFFV1
MAB
MATB
MSB

14. Remarks: None

1. Subroutine Name: PFFV1
2. Purpose: Generate flexure contribution to element thermal load matrix in local coordinates
3. Equations and Procedures: The array containing the subelement integral values is re-arranged. The sub-element thermal load matrices are generated from the integral values and the material properties matrix. The sub-element thermal load matrices are assembled into one matrix and then multiplied by BFT to apply the revised thermal load formulation.
4. Input Arguments:
 - DELC : Array containing sub-element integral values
 - EG : Material properties matrix, oriented to material angle
 - BFT : Array containing revised thermal load formulation
 - FFV : Work storage
5. Output Arguments:
 - FFV1 : Flexure contribution to element thermal load matrix in local coordinates
 - DELPQ : Array containing re-arranged sub-element integral values
6. Error Returns: None
7. Calling Sequence:

(FFV1, DELC, EG, BFT, FFV, DELPQ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 662₈ (434₁₀).
12. Subroutine User: PFFTS
13. Subroutine Required: None
14. Remarks: None

1. Subroutine Name: PSTF

2. Purpose: Generate flexure contribution to element stress matrix for the triangular thin shell element

3. Equations and Procedures: The sub-element stress matrices are generated and assembled into one matrix. This matrix is then transformed to reference system coordinates and the stress angle is applied.

4. Input Arguments:

RL1
RL2 : Sub-element coordinates
RL3
TMS : Not used
TFS : Flexure sub-element to reference system
coordinates transformation matrix
EM : Not used
EG : Material properties matrix, oriented the
material angle
SNM : Work storage
TES : Stress angle transformation matrix
EX : Not used
EY : Not used
GXY : Not used
VXY : Not used
ALPHAX : Not used
ALPHAY : Not used
GAMXY : Not used
T : Flexure thickness
R : Not used
U : Not used
X : Work storage
Y : Work storage
AM1 : Work storage
AM2 : Work storage
AM3 : Work storage
AM4 : Work storage
G : Work storage

5. Output Argument:

S : Flexure contribution to element stress matrix

6. Error Returns: None

7. Calling Sequence:

(S, RL1, RL2, RL3, TMS, TFS, EM, EG, SNM, TES, EX, EY,
GXY, VXY, ALPHAX, ALPHAY, GAMXY, T, R, U, X, Y, AM1,
AM2, AM3, AM4, G)

8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 1062_8 (562_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required:
 - MAB
 - MSB
 - MTB
14. Remarks: None

1. Subroutine Name: PTBF
2. Purpose: Generate flexure sub-element to geometric axes transformation matrix
3. Equations and Procedures: The inverse of the desired matrix is generated by direct assignment into a work area. Inversion is performed to obtain the final transformation matrix.
4. Input Arguments:
 - TGSF : Not used
 - RL1
 - RL2 : Sub-element coordinates
 - RL3
 - IPRT : Intermediate element print indicator
 - L : Work storage
 - M : Work storage
 - U : Work storage
 - TI : Work storage
 - B : Work storage
 - BFF : Work storage
 - BFO : Work storage
5. Output Arguments: None
 - TBF : Flexure sub-element to geometric transformation matrix
6. Error Returns: None
7. Calling Sequence:
 - (TBF, TGSF, RL1, RL2, RL3, IPRT, L, M, U, TI, B, BFF, BFO)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 1767₈ (1015₁₀).
12. Subroutine User: PLUG2
13. Subroutines Required:
 - MAB
 - MINV
14. Remarks: None

1. Subroutine Name: EPRT
2. Purpose: Print generated triangular thin shell element matrices
3. Equations and Procedures: None
4. Input Arguments:
 - AK : Final element stiffness matrix
 - S : Final element stress matrix
 - ANEL : Non-functional
 - FN : Non-functional
 - FT : Final element thermal load matrix
 - FP : Final element pressure load matrix
 - SZALEL : Final element thermal stress matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:
 - (AK, S, ANEL, FN, FT, FP, SZALEL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 377₈ (255₁₀).
12. Subroutine User: PLUG2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PLAS2
2. Purpose: Non-functional
3. Equations and Procedures: None
4. Input Arguments: None
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: None
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 22_8 (18_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PNG1
2. Purpose: Non-functional
3. Equations and Procedures: None
4. Input Arguments: None
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: None
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 22_8 (18_{10}).
12. Subroutine User: PLUG2
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PNC1
2. Purpose: Non-functional
3. Equations and Procedures: None
4. Input Arguments: None
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: None
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: Total storage is 22_8 (18_{10}).
12. Subroutine User: PLUG2
13. Subroutine Required: None
14. Remarks: None

1. Subroutine Name: PLUG 6
2. Purpose: To form the element matrices for a triangular cross section ring discrete element with applications towards the analysis of thick walled and solid axisymmetric structures of finite length. It may be used to form the assembly of any axisymmetric structure taking into account:
 - (1) Arbitrary axial variations in geometry
 - (2) Axial variation in orientation of material axes of orthotropy
 - (3) Radial and axial variations in material properties
 - (4) Any axisymmetric loading systems including pressure, prestrain, prestress, and temperature

The complete discrete element representation, consists of the algebraic expressions for the following matrices:

- (1) Stiffness
- (2) Pressure load
- (3) Thermal load
- (4) Pre-strain load
- (5) Pre-stress load
- (6) Stress
- (7) Mass
- (8) Structural damping
- (9) Viscous damping

3. Equations and Procedures: The development of the complete element representation arises from the Lagrangian (variational) equation

$$\frac{\partial \phi_1}{\partial q_r} + i \frac{\partial \phi_2}{\partial q_r} + \frac{\partial \phi_3}{\partial q_r} + \frac{d}{dt} \left(\frac{\partial \phi_4}{\partial q_r} \right) = 0$$

where

- q_r = r generalized displacement coordinates
- ϕ_1 = total potential energy
- ϕ_2 = structural damping dissipation energy
- ϕ_3 = viscous
- ϕ_4 = kinetic energy

The subsequent development of the element matrices is then provided in algebraic form to the coded program, which follows the format:

- (1) The input data, used in forming the matrices, is processed and organized for computation.
- (2) By subroutine TRAIC, the coordinate transformation matrices, and the table of integrals is formed. In routine TRAIE, the material properties matrices are formed.
- (3) Using the above mentioned matrices and integrals, the program then generates, the stiffness, pressure load, thermal load, stress, pre-strain, pre-stress, mass, structural damping, and viscous damping matrices, and the stress, thermal stress, pre-strain, pre-stress, pre-strain load, and pre-stress load vectors.
- (4) After each significant matrix, vector, etc., is formed, the program prints out the desired results.

4. Input Arguments:

IPL : Plug number
 NNØ : Number of node points
 XC : X - coordinates of nodes points
 YC : Y - coordinates of node points
 ZC : Z - coordinates of node points
 TEL : Temperatures at the node points
 PEL : Pressures at the node points
 Q\$EL : Input displacements of the node points
 IP : Number of extra cards
 NØRD : Order of element stiffness matrix
 NR\$EL : Number of rows in the stress matrix
 INØ : Number of nodes
 NØDØRD : Node point numbers
 KK : Code for computation of element stiffness matrix
 KF : Code for computation of element thermal load
 K\$: Code for computation of element stress matrix
 KM : Code for computation of element mass matrix
 KD\$: Code for computation of structural damping matrix
 KDV : Code for computation of viscous damping matrix
 KN : Code for computation of incremental damping matrix
 IU\$EL : Dummy
 EP\$LØN : Pre-strain load vector
 \$IGZER : Pre-stress load vector
 MAT : Material properties matrix
 EXTRA : Extra information (angles, etc.)
 NDIR : Number of directions of movement per grid point
 NDEG : Number of types of movement allowed per grid point
 ICØNT : Code for use of grid point axes

5. Output Arguments:

NERR : Error return
 NØINK : Number of elements in lower half matrices
 AKELXP : Stiffness matrix
 ANEL : Incremental stiffness matrix
 FTXP : Thermal load + pressure load matrix
 \$TR\$XP : Stress matrix
 T\$: Thermal stress matrix
 XMA\$XP : Mass matrix
 DAMPV : Viscous damping matrix
 DAMP\$: Structural damping matrix
 N\$EL : Number of elements in stress matrix
 NMA\$\$: Number of elements in mass matrix
 NP\$L : Number of elements in viscous damping matrix
 NP\$\$: Number of elements in structural damping matrix
 GPAXEL : Grid point axes transformation

6. Error Returns

NERR = 0 No Error
 = 1 Plug Number Incorrect
 = 2 Number of Nodes Incorrect
 = 3 Number of Input Points Incorrect
 = 4 Order of Matrix (nord) Incorrect

7. Culling Sequence:

(IPL, NNØ, XC, YC, ZC, TEL, PEL, Q\$EL, IP, NØRD, NERR, NØINK, AKELXP, ANEL, FTXP, \$TR\$XP, T\$, XMA\$XP, DAMPV, DAMP\$, NR\$EL, INØ, NØDØRD, NMA\$\$, NP\$L, NP\$\$, N\$EL, KK, KF, K\$, KM, KD\$, KDV, KN, IU\$EL, EP\$LØN, \$IGZER, MAT, EXTRA, GPAXEL, NDIR, NDEG, ICØNT)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage:

GAMABQ(6,6) DELINT(12) DCURL(4,6) EM(10) E(10) TEØ(4,4)
 AKEL(21) FP(6) F\$(6) F\$T(6) EP\$LØN(6) \$IGZER(6) EXTRA(1)
 ALFBAR(4) FT(6) \$TRE\$\$ (4,6) T\$(4) XMA\$\$ (21) D\$M(4)
 D\$(21) DV(21) AKELXP (45) XMA\$XP (45) D\$XP(45) DVXP(45)
 XC(3) YC(3) ZC(3) NODORD(3) X(3) Y(3) Z(3) P\$LMAT (6,4)
 PEL (6) Q\$EL(6) ANEL(6) TEMP2(6,6) IU\$EL (6) LI\$TP(6)
 TEL (12,3) E1 (4,4) DAMP\$(6) DAMPV(6) A(9,6) B(4,6)
 ALI\$TP (6) FTXP(9) \$TR\$XP (4,9) P\$LXP (9,4) P\$\$ (4)
 MAT(1) AKEL1 (6,6) AKEL2 (6,6) AMCURL (21) TEMP (6)
 TEMPl (4) XMA\$\$1 (6,6) TMG (2,2) AMCURL (21) AMBAR (2,2)
 DZERØ (6,4)

12. Subroutine User: ELPLUG

13. Subroutines Required:

ELTE\$T	TRAIE	TRAIK	TRAIFP	TFTPRT
TRAIT\$	TRAIM	TF\$PRT	EXPCØL	TRAIC
TIEPRT	TIKPRT	TFPPRT	TRAI\$	TT\$PRT
TIMPRT	TRAI\$T	FL6PRT	TRCPRT	TPRD
EXP\$IX	TRAIFT	TI\$PRT	MPRD	TRAIF\$
T\$TPRT				

14. Remarks: None

1. Subroutine Name: EXPCØL
2. Purpose: To generate a matrix $[B]$, given a specific input matrix $[A]$, for Plug 6.

The purpose of this operation is to impose the conditions that flexure terms "v" are zero.
3. Equations and Procedures: The matrix terms are formed by direct assignment.
4. Input Arguments: $[A]$: Input Matrix
5. Output Arguments: $[B]$: Output Matrix
6. Error Returns: None
7. Calling Sequence: (A, B)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A (6), B (9)
12. Subroutine User: PLUG 6
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: EXP\$IX
2. Purpose: To generate a symmetric matrix $[B]$, given a specific input symmetric matrix $[A]$, for Plug 6. The purpose of this operation is to impose the condition that flexure terms "v" are zero.
3. Equations and Procedures: The matrix terms are formed by direct assignment.
4. Input Arguments: $[A]$: Input Matrix
5. Output Arguments: $[B]$: Output Matrix
6. Error Returns: None
7. Calling Sequence: (A, B)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A(21) B(45)
12. Subroutine User: Plug 6
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: TRAIC
2. Purpose: To generate coordinate transformation matrices for triangular ring which vary with coordinates and generate integrals for future use.
3. Equations and Procedures: The coordinate matrix [GAMABQ] is formed by algebraic assignment. The table of integrals, DELINT, is formed by algebraic methods using the function subprogram AI.
4. Input Arguments: R,Z: Coordinates of node points
WIPR: Print control
5. Output Arguments:
GAMABQ: Coordinate matrix
DELINT: Table of integrals
DCURL: Matrix transformation
ISING: Error return code
6. Error Returns: If GAMABQ cannot be generated due to singular matrix then ISING is set to one.
7. Calling Sequence: (R, Z, GAMABQ, DELINT, DCURL, ISING, WIPR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: R(3), Z(3), GAMABQ (6,6), DELINT (12),
DCURL (4,6), LL(6), MM(6)
12. Subroutine User: PLUG6
13. Subroutines Required: MINV, AI, TESTJ, TRCPRT
14. Remarks: None

1. Subroutine Name: TESTJ
2. Purpose: To check DELINT (PLUG6) for any negative or incorrect integrals; If any errors are noted, the integrals are recomputed by an approximation method.
3. Equations and Procedures: The checks are performed by logical if statements. The integral approximation is

$$\iint X^P Z^Q dx dz \approx \bar{X}^P \cdot \bar{Z}^Q \cdot A$$

where

$$\bar{X} = \frac{1}{3} [X_1 + X_2 + X_3]; \quad \bar{Z} = \frac{1}{3} [Z_1 + Z_2 + Z_3]$$

$$A = \frac{1}{2} [X_1(Z_2 - Z_3) + X_2(Z_3 - Z_1) + X_3(Z_1 - Z_2)]$$

4. Input Arguments: DELINT (I) value of the i th integral
X: X coordinates
Z: Z coordinates
WIPR: print control
5. Output Arguments: DELINT (I): recomputed integral
6. Error Returns: None
7. Calling Sequence: CALL TESTJ (DELINT, X, Z, WIPR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

DELINT (12), DLINT (12), X (1), Z (1), XO (3), ZO (3),
DELTAX (1), DELTAZ (1), XHAT (1), ZHAT (1)
12. Subroutine User: TRAIC
13. Subroutine Required: None
14. Remarks: If the test necessitates recomputation, the new integral values will be stored in the old locations, thus destroying the originals.

1. Subroutine Name: TRCPRT
2. Purpose: To print elements formed in TRAIC
3. Equations and Procedures: None
4. Input Arguments:
 - GAMABQ: coordinate matrix
 - DELINT: table of integrals
 - DCURL : matrix of integrals
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (GAMABQ, DELINT, DCURL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: GAMABQ (6,6), DELINT (12), DCURL (4,6)
12. Subroutine User: PLUG6
13. Subroutine Required: None
14. Remarks: None

1. Subroutine Name: TRAIE
2. Purpose: To generate the transformed matrix of elastic constants
3. Equations and Procedures: The routine
 - a) Generates the transformation matrix
 - b) Generates the elastic constants matrix
 - c) Generates the transformed elastic constant matrix
4. Input Arguments:

ER, ETHETA, EZ : Moduli of elasticity (Young's)
 VRØ, VØZ, VZR : Poissons ratio
 GRZ : Modulus of rigidity
 GAM : Angle between material axes and
 element axes
 El : Work storage
5. Output Arguments:

TEØ : Transformation matrix
 EM : Elastic constants matrix
 E : Transformed elastic constant matrix
6. Error Returns: None
7. Calling Sequence:

(ER, ETHETA, EZ, VRØ, VØZ, VZR, GRZ, GAM, TEØ, EM, E, El, WIPR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: EM(10), TEØ(4,4), E(10), El(4,4), E2(4,4)
12. Subroutine User: PLUG6
13. Subroutines Required: MPRD, TPRD
14. Remarks: None

1. Subroutine Name: TIEPRT
2. Purpose: To print matrices formed in TRAIE
3. Equations and Procedures: None
4. Input Arguments: TEØ : Transformation matrix
EM : Elastic constant matrix
E : Transformed elastic constant matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (TEØ, EM, E)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: TEØ(4,4), EM(10), E(10)
12. Subroutine User: PLUG6
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: TRAIK
2. Purpose: Generate stiffness matrix for triangular ring
3. Equations and Procedures: The program uses the table of integrals to form the first intermediate matrix. This matrix is then transformed to form the final stiffness matrix.
4. Input Arguments:
GAMABQ: Transformation matrix
E : Transformed elastic constant matrix
DELINT: Table of integrals
WIPR : Print control
AKEL1, AKEL2, ACURL: Work storage
5. Output Arguments: AKEL : Stiffness matrix
6. Error Returns: None
7. Calling Sequence: (GAMABQ, E, DELINT, AKEL, WIPR, AKEL1, AKEL2, ACURL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
GAMABQ (6,6), E(10), DELINT (12), AKEL (21), AKEL1 (6,6), AKEL2 (6,6), ACURL(21)
12. Subroutine User: PLUG6
13. Subroutines Required: TPRD, MPRD
14. Remarks: None

1. Subroutine Name: TIKPRT
2. Purpose: To display matrices generated in TRAIK
3. Equations and Procedures: None
4. Input Arguments: AKEL : Stiffness matrix
ACURL : Intermediate stiffness matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (AKEL, ACURL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: AKEL (21), ACURL (21)
12. Subroutine User: PLUG6
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: TRAIFF
2. Purpose: To generate the pressure load vector for triangular ring.
3. Equations and Procedures: The program
 - 1.) Generates necessary constants
 - 2.) Generates pressure load vector (non-transformed)
 - 3.) Transforms pressure load vector
4. Input Arguments:

R,Z: Coordinates of node points
P: Node point pressures
GAMABQ: Coordinate transformation matrix
WIPR: Print control
5. Output Arguments:

FCURLP: Non-transformed pressure load vector
FP: Transformed pressure load vector
6. Error Returns: None
7. Calling Sequence:

(R, Z, P, GAMABQ, FP, WIPR, FCURLP)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: R(3), Z(3), P(3), GAMABQ(6,6), FP(6), F(3), FCURLP(6), DELTA(6)
12. Subroutine User: PLUG6
13. Subroutines Required: TPRD
14. Remarks: None

1. Subroutine Name: TFPVRT
2. Purpose: To display the non-transformed and transformed pressure load vectors.
3. Equations: None
4. Input arguments:
 FP: transformed pressure load vector
 FCURLP: non-transformed pressure load vector
5. Output arguments: None
6. Error returns: None
7. Calling sequence: (FP, FCURLP)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: FP(6), FCURLP(6)
12. Subroutine User: PLUG6
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: TRAIPT
2. Purpose: To generate a thermal load vector for a triangular ring element
3. Equations & Procedures: The input matrices are manipulated by matrix algebra to form the thermal load vector.
4. Input arguments:
 - ALFBAR: vector of coefficients of linear thermal expansion
 - TMTZRO: base temperature
 - GAMABQ: transformation matrix
 - DCURL: matrix containing integral values
 - E: transformed elastic constant matrix
 - WIPR: print control
5. Output arguments:
 - FT: thermal load vector
6. Error returns: None
7. Calling sequence:
 - (ALFBAR, TMTZRO, GAMABQ, DCURL, E, FT, WIPR)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage: ALFBAR(4), GAMABQ(6,6), DCURL(4,6), E(10), FT(6),
TEMP1(4), TEMP2(6), SAVE(4).
12. Subroutine User: PLUG6
13. Subroutines Used: MPRD, TPRD
14. Remarks: None

1. Subroutine Name: TFTPRT
2. Purpose: To display thermal load vector for triangular ring element
3. Equations: None
4. Input arguments:
 FT: thermal load vector
 ALFBAR: coefficients of linear expansion
 TMTZRØ: base temperature
5. Output arguments: None
6. Error Returns: None
7. Calling Sequence: (FT, ALFBAR, TMTZRØ)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: FT(6), ALFBAR(4)
12. Subroutine User: PLUG6
13. Subroutines Used: None
14. Remarks: None

1. Subroutine Name: TRAI\$
2. Purpose: To generate the stress matrix for triangular ring element
3. Equations and Procedures: Given input constants, an intermediate matrix is formed, which is then multiplied by the system matrices to form the final matrix
4. Input Arguments:
 - R, Z: coordinates of node points
 - GAMABQ: coordinate transformation matrix
 - E: elastic constant matrix
 - WIPR: print control
 - DZERØ: work space
 - TEMP: node point temperatures
5. Output Arguments: \$TRE\$\$: stress matrix
6. Error returns: None
7. Calling sequence: (R, Z, GAMABQ, E, STRESS, WIPR, DZERO, TEMP)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: R(3) Z(3), GAMABQ(6,6), E(10), STRESS(4,6), DZERØ(4,6), TEMP(4,6)
12. Subroutine User: PLUG6
13. Subroutines Used: MPRD
14. Remarks: None

1. Subroutine Name: TI\$PRT
2. Purpose: To display the stress matrix for a triangular ring element
3. Equations: None
4. Input Arguments:
\$TRE\$\$: stress matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (\$TRE\$\$)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: STRESS (4,6)
12. Subroutine User: PLUG6
13. Subroutines Used: None
14. Remarks: None

1. Subroutine Name: TRAIT\$
2. Purpose: To generate thermal stress vector for a triangular ring element
3. Equations and Procedures: The input matrices are combined, using matrix algebra, to form the thermal stress vector.
4. Input Arguments:
 - E: elastic constant matrix
 - ALFBAR: linear thermal expansion coefficients
 - TMTZRØ: base temperature
 - WIPR: print control
5. Output Arguments:
 - T\$: thermal stress matrix
6. Error Returns: None
7. Calling Sequence: (E, ALFBAR, TMTZRØ, T\$, WIPR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:
 - E(10), ALFBAR(4), T\$(4), SAVE(4)
12. Subroutine User: PLUG6
13. Subroutines Used: MPRD
14. Remarks: None

1. Subroutine Name: TT\$PRT
2. Purpose: to display the thermal stress vector of a triangular ring element
3. Equations: None
4. Input Arguments:
T\$: thermal stress vector
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (T\$)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: T\$(4)
12. Subroutine User: PLUG6
13. Subroutines Used: None
14. Remarks: None

1. Subroutine Name: TRAIM
2. Purpose: To generate a mass matrix for a triangular ring element
3. Equations and Procedures: The program
 - (1) Forms a transformation matrix [TMG]
 - (2) Generates a matrix [M] which is a function of the mass coefficients.
 - (3) Generates a matrix [$\tilde{M}$] which is a function of [M] and the table of integrals.
 - (4) Generates the mass matrix [M] which is a function of [M] and the transformation matrix [GAMABQ].

$$[M] = [GAMABQ]^T [\tilde{M}] [GAMABQ]$$
4. Input Arguments:

AMASS1, AMASS2: mass coefficients
 GAM: angle between material axes and element axes
 GAMABQ: coordinate transformation matrix
 DELINT: table of integrals
 WIPR: print control
 XMASS1, TEMP, AMCRUL, TEMPl, AMBAR: storage
 TMG: transformation matrix
5. Output Arguments:

XMASS: mass matrix
6. Error Return: None
7. Calling Sequence:

(AMASS1, AMASS2, GAM, GAMABQ, DELINT, XMA\$\$, WIPR, XMASS1, TEMP, TMG, AMCURL, TEMPl, AMBAR)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:

AMASS(2), GAMABQ(6,6), DELINT(12),
 XMASS(21), XMASS1(6,6), TEMP(6,6), TMG(2,2), AMBAR(2,2),
 TEMPl(2,2), AMCURL(21)
12. Subroutine User: PLUG6
13. Subroutines Used: TPRD, MPRD
14. Remarks: None

1. Subroutine Name: TIMPRT
2. Purpose: To display the mass matrix of a triangular ring element
3. Equations: None
4. Input Arguments:
 XMASS: mass matrix
 AMCURL: intermediate mass matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (XMA\$\$, AMCURL)
8. Output Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage:
 XMASS(21), AMCURL(21)
12. Subroutine User: PLUG6
13. Subroutines Used: None
14. Remarks: None

1. Subroutine Name: TRAI F\$
2. Purpose : To generate pre-strain load vector for a triangular ring element.
3. Equations and Procedures: The routine uses the inputed matrices and combines these to form the pre-strain load vector.
4. Input Arguments:
 - EP\$ LØN: Input pre-strain values
 - GAMABQ: Transformation matrix
 - DCURL: Integral matrix
 - E: Elastic constant matrix
 - WIPR: Print control
 - TEMP: Dummy storage
 - TEMP1, TEMP2: Dummy storage
 - P\$LMAT: Dummy storage
5. Output Arguments:
 - F\$: Pre-strain load vector
6. Error Returns: None
7. Calling Sequence:
 - (EP\$ LØN, GAMABQ, DCURL, E, F\$, WIPR, TEMP, TEMP1, TEMP2, P\$LMAT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 - EP\$ LØN(4), GAMABQ (6,6), DCURL (4,6), E(10), FS(6), TEMP(1), TEMP1(1), TEMP2(6,4), P\$LMAT(6,4)
12. Subroutine User: PLUG6
13. Subroutines Required: MPRD, TPRD
14. Remarks: None

1. Subroutine Name: TF\$PRT
2. Purpose: Display pre-strain load vector for triangular ring
3. Equations: None
4. Input arguments: F\$: pre-strain load vector
5. Output arguments: None
6. Error returns: None
7. Calling sequence: (F\$)
8. Input tapes: None
9. Output Tapes: None
10. Scratch tapes: None
11. Storage required: F\$(6)
12. Subroutine user: PLUG6
13. Subroutines required: None
14. Remarks: None

1. Subroutine Name: TRAI\$T
2. Purpose: To generate the pre-stress load vector for a triangular ring element
3. Equations & Procedures: The input matrices are combined by matrix manipulations to form the pre-stress load vector.
4. Input arguments:
 - \$IGZER: column of pre-stresses
 - GAMABQ: transformation matrix
 - DCURL: Integral value matrix
 - WIPR: print control
5. Output arguments:
 - F\$T: pre-stress load vector
6. Error Returns: None
7. Calling sequence:
 - (\$IGZER, GAMABQ, DCURL, F\$T, WIPR)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage Required:
 - \$IGZER(4), GAMABQ(6,6), DCURL(4,6), F\$T(6), TEMP(6)
12. Subroutine User: PLUG6
13. Subroutines required: TPRD
14. Remarks: None

1. Subroutine Name: T\$TPRT
2. Purpose: Disply pre-stress load vector for triangular ring element
3. Equations: None
4. Input arguments: F\$T
5. Output arguments: None
6. Error returns: None
7. Calling Sequence: (F\$T)
8. Input tapes: None
9. Output tapes: None
10. Scratch tapes: None
11. Storage: F\$T (6)
12. Subroutine User: PLUG6
13. Subroutines used: None
14. Remarks: None

1. Subroutine Name: PL6PRT
2. Purpose: To display structural damping, viscous damping, pre-strain and pre-stress matrices for a triangular ring element.
3. Equations: None
4. Input Arguments:
 - D\$XP: structural damping vector
 - DVXP: viscous matrix
 - E1: pre-stress multiplier matrix
 - P\$LMAT: pre-strain multiplier matrix
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (D\$XP, DVXP, E1, P\$LMAT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage: D\$XP(45), DVXP(45), E1(4,4), P\$LMAT(4,4)
12. Subroutine User: PLUG6
13. Subroutines Used: None
14. Remarks: None

1. Subroutine Name: PLUG 5
2. Purpose: To form the element matrices for a doubly curved ring (toroidal ring) discrete element. This ring configuration, defined by an arbitrary section of revolution of a complete right circular toroidal shell, enables smoothly continuous idealization of general axisymmetric thin shell problems.

The matrices which are formed are:

- (1) Stiffness matrix
 - (2) Stress matrix
 - (3) Thermal load matrix + pressure load matrix
 - (4) Thermal stress matrix
3. Equations and Procedures: There are two cases treated for this type of element. They are:
 - (1) The angles of the interior and exterior membranes are not equal (Toroidal section)
 - (2) The angles of the interior and exterior membranes are equal. (Conic section).

In the second case, the interior angle is increased by a factor of $.50^\circ$ so that they can be treated as in case one. A special case arises for the degenerate situation where the two angles equal 90° . In this case a different path is followed.

A variational (Lagrangian) approach is taken in formulating the discrete element representation. On account of this, it has been found necessary to use numerical integration techniques, namely the Romberg technique and the numerical quadrature technique.

The sequence of procedures is as follows:

- (1) The first general part of the routine processes input information, forming constants to be used in calculations. Also, several constants are extracted from the input material and extra matrices.
- (2) After testing as to the relative values of the membrane angles (i.e. equal or not), the program selects the correct path to take in forming the integrals used in later calculations. Either the Romberg or Numerical Quadrature methods are used to evaluate the integrals.
- (3) Using the integrals and the program constants, the program forms several intermediate element matrices.

- (4) By several matrix operations (multiplications, transformations, etc.), the stiffness matrix, AKEL, is formed.
- (5) In like manners, the program forms the thermal load (FTEL) matrix, the pressure load (FPEL) matrix, the combined thermal and pressure load (TPEL) matrix, the stress (\$EL) matrix, and the thermal stress (T\$EL) matrix.
- (6) After all the calculations are completed, the program calls a subroutine to print all the matrices.

4. Input Arguments:

IPL: Plug Number
 NNØ: Number of node points
 R: X - coordinates of nodes
 Y: Y - coordinates of nodes
 Z: Z - coordinates of nodes
 TEMP: Node point temperatures
 P: Node point pressures
 Q\$: Node point i puted displacements
 IP: Number of extra cards
 KK: Code for computation of element stiffness matrix
 KF: Code for computation of element thermal load
 K\$: Code for computation of element stress matrix
 KM: Code for computation of element mass matrix
 KN: Code for computation of element incremental matrix
 KD\$: Code for computation of element structural damping
 KDV: Code for computation of element viscous damping
 NØRD: Order of element stiffness matrix
 MAT: Material properties table
 EXTRA: Specific element information
 NDIR: Number of directions for each grid point
 NDEG: Number of types of movement allowed
 IU\$EL: Dummy
 EP\$LØN: Pre-strain load vector
 \$Ø: Pre-stresses
 INNØ: Number of nodes
 ICØNT: Code for use of grid point axes
 NR: Number of rows in stress matrix
 NØDE\$: Node point numbers

5. Output Arguments:

NERR: Error return
 NØINK: Number of elements in lower half matrices
 AKEL: Stiffness matrix
 ANEL: Incremental matrix
 TPEL: Thermal load + pressure load matrix
 \$EL: Stress matrix
 T\$EL: Thermal stress matrix
 AMA\$\$: Mass matrix
 DAMPV: Viscous damping matrix
 DAMP\$: Structural damping matrix
 N\$EL: Number of elements in stress matrix
 NMA\$\$: Number of elements in mass matrix
 NDAMPV: Number of elements in viscous damping matrix
 NDAMP\$: Number of elements in structural damping matrix
 GPAXEL: Grid point axes transformation matrix

6. Error Returns:

NERR = 0 No error
 = 1 Plug number incorrect
 = 2 Number of nodes incorrect
 = 3 Number of input points incorrect
 = 4 Order of matrix (nord) incorrect

7. Calling Sequence:

(IPL, NNØ, R, Y, Z, TEMP, P, Q\$, IP, NØRD, NERR, NØINK,
 AKEL, ANEL, TPEL, \$EL, T\$EL, AMA\$\$, DAMPV, DAMP\$, NR,
 INNØ, NØDE\$, NMA\$\$, NDMPV, NDMP\$, N\$EL, KI, KF, K\$, KM,
 KD\$, KDV, KN, IU\$EL, EP\$LØN, \$Ø MAT, EXTRA, GPAXEL,
 NDIR, NDEG, ICØNT)

8. Input Tapes: None

9. Output Tapes: None

10. Scratch Tapes: None

11. Storage Required:

T(21) W(10,18) W1(18,18) R(2) Y(2) Z(2) P(2) TEMP (12,3)
 NØDE\$(1) W2 (18,18) W3 (18,18) TAKEL (18,18) AKEL(171)
 GAMM (10,18) XI(6,12) YI(6,12) X(6) B(10,18) D(10,10)
 FTEL (18,1) GAM(10,18) FMEØ (10,2) FME1 (10,2) FFEQ(10,2)
 FFE1 (10,2) E(2,2) AIK(2,2) AJK (2,2) ETØ (2,1) ET1(2,1)
 ALTØ (2,1) ALT1 (2,1) V1 (18,2) V2 (18,2) V3 (18,1) V4(18,1)
 V5 (18,1) V6 (18,1) FPFL (18) FPCQ (10,1) TPEL (18) \$EL
 (15,18) XXI (3) EXTRA(1) \$CURL (15,10) T\$SEL (15) TEL (2,1)
 TE2 (2,1) EM1 (2,1) EM2 (2,1) EP\$LØN (1) \$Ø (1) MAT(1)

- 12. Subroutine User: ELPLUG
- 13. Subroutines Required:
F4, F5, F6, ELTET, MPRD, GAMMAT, \$CRLM, BMATRX, TPRD,
FCURL, \$OLVE, DMATRX, M\$TR, PLMX, PRINT5
- 14. Remarks: None

1. Subroutine Name: MSTR
2. Purpose: To change the storage mode of a matrix.
3. Equations and Procedures: MSTR will perform the operation on the right when MSA and MSR are equal to

<u>MSA</u>	<u>MSR</u>	<u>PROCEDURE</u>
0	0	[A] is moved to [R]
0	1	The upper triangle elements of a general matrix are used to form a symmetric matrix
0	2	The diagonal element of a general matrix are used to form a diagonal matrix
1	0	A symmetric matrix is expanded to form a general matrix
1	1	[A] is moved to [R]
1	2	The diagonal elements of a symmetric matrix are used to form a diagonal matrix
2	0	A diagonal matrix is expanded to form a general matrix
2	1	A diagonal matrix is expanded to form a symmetric matrix
2	2	[A] is moved to [R]

The codes for MSA and MSR stand for

0	General matrix form
1	Symmetric matrix form
2	Diagonal matrix form

4. Input Arguments:

[A]	Input matrix
N	Number of rows and columns in [A] and [R]
MSA	Code designating storage mode of [A]
MSR	Code designating storage mode of [R]

5. Output Arguments:
 [R] : Output matrix.
6. Error Returns: None
7. Calling Sequence: (A, R, N, M, $\oint$ A, M $\oint$ R)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: A (1), R (1)
12. Subroutine User: PLUG 5
13. Subroutines Required: L $\emptyset$ C
14. Remarks: Matrix [A] may not be in the same storage as [R].

1. Subroutine Name: RØMBER
2. Purpose: To integrate $f(x)$ from $x = a$ to $x = b$.
3. Equations and Procedures: The precision of large numbers in terms of number of significant digits and the accuracy of small numbers in terms of number of significant digits is measured. The subroutine terminates when either of these conditions is met.
4. Input Arguments:

A:	Lower limit
B:	Upper limit
NØSIG:	Number of correct significant digits (not more than 7)
NUM:	maximum number of halvings of (a,b) to be made (not more than 99)
KØDE:	controls the form of the print-out
FUNCT:	function of $X - F4, F5, F6$
X:	variable of integration
5. Output Arguments:

ITDØNE:	number of iterations
FINTG:	value of the integral
PRECIS:	actual number of significant digits attained
6. Error Returns: None
7. Calling Sequence: $(A, B, NØSIG, PRECIS, NUM, ITDØNE, FINTG, KØDE, FUNCT, X)$
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

$X(6), FAAAA(20), FAAAB(20)$
12. Subroutine User: PLUG5
13. Subroutines Required: FUNCT
14. Remarks: None

1. Subroutine Name: F4
2. Purpose: To set up a function to be used by RØMBER in the computation of i_5 , one of the six basic integrals used in PLUG5.
3. Equations and Procedures:

$$F4 = (X_1)^{-1} \sin^2(X_1) / DEN$$

where

$$DEN = X_3 - X_2 X_5 + X_2 X_5 \cos(X_1) + X_2 X_4 \sin(X_1)$$
4. Input Argument: X: array containing integration arguments
5. Output Arguments: F4: functional value
6. Error Returns: None
7. Calling Sequence: F4 (X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: X (6)
12. Subroutine User: RØMBER
13. Subroutines Required: SIN, COS
14. Remarks: None

1. Subroutine Name: F5
2. Purpose: To set up a function to be used by RØMBER in the computation of I_5 , one of the six basic integrals used in PLUG 5.
3. Equations and Procedures:

$$F5 = (X_1)^{x_6-1} 2 \sin(x_1) \cos(x_1) / DEN$$
 where

$$DEN = x_3 - x_2 x_5 + x_2 x_5 \cos(x_1) + x_2 x_4 \sin(x_1)$$
4. Input Arguments:
 X: array containing integration arguments
5. Output Arguments: F5 - functional value
6. Error Returns: None
7. Calling Sequence: F5(X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: X (6)
12. Subroutine User: RØMBER
13. Subroutines Required: SIN, CØS
14. Remarks: None

1. Subroutine Name: F6
2. Purpose: To set up a function to be used by RØMBER in the computation of I_6 , one of the six basic integrals used in PLUG5
3. Equations and Procedures:

$$F6 = \text{CONST} \cdot \text{Cos}(X_1) / \text{DEN}$$

where

$$\text{DEN} = x_3 - x_2 x_5 + x_2 x_5 \text{Cos}(x_1) - x_2 x_4 \sin(x_1)$$

$$\text{CONST} = \begin{cases} 1, & x_6 = 1 \\ (x_1)^{x_6-1}, & x_6 \neq 1 \end{cases}$$
4. Input Arguments:

X: array containing integration arguments
5. Output Arguments:

F6: functional value
6. Error Returns: None
7. Calling Sequence: F6(X)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: X(6)
12. Subroutine User: RØMBER
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: BMATRIX
2. Purpose: To generate a matrix [B] , given specific input, for PLUG5
3. Equations and Procedures: The routine forms the terms of the matrix by direct assignment
4. Input Arguments:
S: Variable used to form terms of matrix
5. Output Arguments
B: completed transformation matrix
6. Error Returns: None
7. Calling Sequence: (B , \$)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: B(10, 18)
12. Subroutine User: PLUG5
13. Subroutines Required: None
14. Remarks: Typical Element
$$B(6, 9) = -1.0/2.0 * S * S * S)$$

1. Subroutine Name: DMATRIX
2. Purpose: To generate a matrix $[D]$, for Plug 5, given specific input.
3. Equations and Procedures: The routine forms the terms of the matrix by direct algebraic assignment
4. Input Arguments:

V	}	: All variables used to form the terms
C		
CA		
CA2		
VA		
DM		
DB		
YI		

5. Output Arguments:
 $[D]$: Completed Matrix
6. Error Returns: None
7. Calling Sequence
(D, V, C, CA, CA2, VA, DM, DB, YI)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: B(10,10), YI (6, 12)
12. Subroutine User: Plug 5
13. Subroutines Required: None
14. Remarks: Typical Element

$$D(3,2) = DB * (2.*V*YI(4, 1) - 2.* YI(6,2) + D(4,1))$$

1. Subroutine Name: GAMMAT
2. Purpose: To generate a matrix [GAMM], given another matrix
3. Equations and Procedures: The routine rearranges the rows of the input matrix to form the output matrix.
4. Input Arguments:
 B: Input Matrix
5. Output Arguments:
 GAMM: Output Matrix
6. Error Returns: None
7. Calling Sequence: (GAMM, B)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: GAMM (10, 18), B (1, 18)
12. Subroutine User: PLUG5
13. Subroutines Required: None
14. Remarks:
 Typical Element $GAMM(4, 3) = B(10, 3)$

1. Subroutine Name: FCURL
2. Purpose: To generate 4 matrices, [FMEØ], [FME1] , [FFEØ] , and [FFE1] , given specific input, for Plug 5
3. Equations and Procedures: The routine forms the terms of the matrices by direct algebraic assignment.
4. Input Arguments:

$$\left. \begin{array}{l} \text{YI} \\ \text{S} \\ \text{LAM1} \end{array} \right\} \text{variables used to form the terms of the matrices.}$$
5. Output Arguments:

$$\left. \begin{array}{l} \text{FMEØ} \\ \text{FME1} \\ \text{FFEØ} \\ \text{FFE1} \end{array} \right\} : \text{output matrices}$$
6. Error Returns: None
7. Calling Sequence: (FMEØ, FME1, FFEØ, FFE1, YI, \$, LAM1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

$$\text{FMEØ}(10,2), \text{FFEØ}(10,2), \text{FME1}(10,2), \text{FFE1}(10,2), \text{YI}(6,12)$$
12. Subroutine User: PLUG5
13. Subroutines Required: None
14. Remarks: Typical Element

$$\text{FME1}(4,2) = \$1 * \text{YI}(4,5)$$

1. Subroutine Name: PLMX
2. Purpose : to generate a matrix $[FPCQ]$, given specific input for Plug 5.
3. Equations and Procedures: The routine forms the terms of the matrix by direct algebraic assignment.
4. Input Arguments:

$\left. \begin{array}{l} YI \\ CONST1 \\ CONST2 \\ P1 \end{array} \right\}$	:	Variables used to form the terms of the matrix
5. Output Arguments:

FPCQ	Output Matrix
6. Error Returns: None
7. Calling Sequence: (FPCQ, YI, CONST1, CONST2, P1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:

FPCQ (10,1), YI(6, 12)
12. Subroutine User

PLUG 5
13. Subroutines Required: None
14. Remarks: Typical Element

$FPCQ(6,1) = CONST1 * (P1 * YI(1,2) - CONST2 * YI(1,3))$

1. Subroutine Name: SCRLM
2. Purpose: To generate a matrix [$\$CURL$] , given specific input, for PLUG5
3. Equations and Procedures: This routine forms the terms of the matrix by direct algebraic assignment.
4. Input Arguments:
 - XXI:
 - E:
 - H: Variables used to form the terms of the matrix
 - CONT:
 - RP:
 - ALF1:
 - R1:
 - LAM1:
5. Output Arguments:
 - SCURL: output element stress matrix
6. Error Returns: None
7. Calling Sequence:
 - ($\$CURL$, XXI, E, H, CONT, RP, ALF1, R1, LAM1)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
 - $\$CURL$ (15, 10), E(2,2), XXI(3)
12. Subroutine User: PLUG5
13. Subroutines Required: None
14. Remarks:
 - Typical Element
 - $\$CURL(4,8) = \$CURL(4,6) * 3.0 * XX2 - E(1,2) * 6.0 * XX1$

1. Subroutine Name: \$ØLVE
2. Purpose: To solve for lambdas as functions of XI.
 i.e. $\lambda = f(XI)$
3. Equations and Procedures: The routine uses algebraic techniques to arrive at a solution.
 eg.)

$$LAM2 = \frac{\cos \left(A1 + \frac{XI}{RP} \right)}{R1 - RP * \left(\sin(A1) + \sin \left(A1 + \frac{XI}{RP} \right) \right)}$$
 where A1, R1, RP are constants
 LAM3 and LAM4 are similar
4. Input Arguments: $\left. \begin{array}{l} A1 \\ R1 \\ RP \\ XI \end{array} \right\} \text{Variables used for calculation of the lambdas}$
5. Output Arguments: $\left. \begin{array}{l} LAM2 \\ LAM3 \\ LAM4 \end{array} \right\} \text{: Output values}$
6. Error Returns: None
7. Calling Sequence: (A1, R1, RP, XI, LAM2, LAM3, LAM4, CØNT)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: PLUG 5
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: QUADI
2. Purpose: To evaluate integrals by an enclosed quadrature formula
3. Equations and Procedures:

Given the integrals of the form

$$I_2^j = \int_0^s \frac{\xi^j}{r_1 + \xi \cos \alpha_1} d\xi$$

when it is true that

$$\frac{s \cos \alpha_1}{r_1} < 1$$

it follows that

$$I_2^j \approx \frac{s^{j+1}}{r_1} \sum_{m=1}^{j+1} \frac{(-1)^{m-1}}{j+m} \left(\frac{s \cos \alpha_1}{r_1} \right)^{m-1}$$

where E_n is the error term.

The formula converges when

$$|E_n| \leq \frac{1}{j+m+1} \left(\frac{s \cos \alpha_1}{r_1} \right)^m$$

4. Input Arguments:

R1: Change in coordinates (distance)
 S: Upper bound of integration
 N: Number of integral ($N = j + 1$)
 CTRM: Criteria for convergence $CTRM = \frac{S \cos \alpha_1}{r_1}$

5. Output Arguments:

XI: Value of approximation

6. Error Returns: If the quadrature doesn't converge after 1000 iterations, the program terminates.

7. Calling Sequence:

CALL QUADI (R1, S, N, CTRM, XI)

8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: PLUG5
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PRINT5
2. Purpose: To print, as output, the intermediate matrices and single valued variables, generated in Plug 5.
3. Equations and Procedures: The routine contains the proper write and format statements.
4. Input Arguments: All the variables to be printed.
(See calling sequence)
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence: (C, DM, DB, PHIB, RP, \$, BB, RT, P\$I1, P\$I2, CØ\$1, \$IN1, XI, YI, B, D, W, W1, H, ALF2, ALF1, W3, R1, R2, Z1, Z2, EP, ET, VPT, AXI, ABETA, T1I, T1Ø, T2I, T2Ø, LAM1, AIK, AJK, ETØ, ET1, ALTØ, ALT1, E, FMEØ, FME1, FFEØ, FFE1, FTEL, P1, P2, CON\$T1, CØN\$T2, FPCQ, FPEL, \$CURL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: W(10,18), W1(18,18), XI(6,12), YI(6,12), B(10,18), D(10,10), FTEL(18,1), FMEØ (10,2), FME1(10,2), FFEØ(10,2), FFE1(10,2), E(2,2), AIK(2,2), AJK(2,2), ETØ(2,1), ET1(2,1), ALTØ(2,1), ALT1(2,1), FPEL(18), FPCQ(10,1), W(18,18), \$CURL(15,10)
12. Subroutine User: PLUG 5
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: PLUG 14
2. Purpose: To compute the element stiffness, stress and diagonal mass matrix.
3. Equations and Procedures: The routine first generates the transformation matrix, PH, and prints it out (using P14PRT) if option is in effect. It then calculates the stress matrix depending on input code KI $\neq$ 2. It now calculates the stiffness matrix, transforms it to system coordinates using MULTF, and expands it using P00F. If KI = 2, the routine will then calculate the lumped mass matrix and expand it using P00F.
4. Input Arguments:
 - IPL: Plug Number (must equal 14)
 - NN0: Number of node points (must equal 4)
 - X,Y,Z: Three vectors of length four each having the X,Y,Z coordinates of the 4 node points.
 - N0RD: Order of stiffness and mass matrix (must equal 24)
 - KI: Selective calculation code
 - MAT: Material properties array
 - MAT (2) = E - Young's Modulus
 - MAT (5) = u - Poisson Ratio
 - MAT (22) = DENSM - mass density
 - MAT (23) = C0NT - print control
 - EXTRA: Extra input array (EXTRA (1) = T = thickness)
5. Output Arguments:
 - NERR: Error return code
 - N0INK: Number of elements in symmetric stiffness matrix (equals 300)
 - AKELXP: Singly subscripted array of element stiffness matrix (symmetric lower half by rows)
 - SELXP: Singly subscripted array of element stress matrix of size 1 x 24
 - AMASS: Singly subscripted array of element mass matrix (symmetric lower half by rows)
 - NRSEL: Number of rows in stress matrix (equals 1)
 - NMASS: Number of elements in symmetric mass matrix (equals 300)
 - NSELXP: Number of elements in stress matrix (equals 24)
 - TSELXP: Thermal stress vector of length 1 is set to zero.
 - TPELXP: Applied load vector of length 24 is set to zero.
6. Error Returns: If NERR $\neq$ 0 then error was detected in input arguments. (See ELPLUG)
7. Calling Sequence:

(IPL, NN0, X,Y,Z, TEMP, P, QS, IP, N0RD, N0INK, AKELXP, ANEL, TPELXP, SELXP, TSELXP, AMASS, NDMPV, NDMPS, NSELXP, KI, KF, KS, KM, KDS, KDV, KN, IUSEL, EPSI0, S0, MAT, EXTRA, GPAXEL, NDIR, NDEG, IC0NT)

8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
- 11: Storage Required: 505 decimal locations of work storage used from unlabeled common block.
- 12: Subroutine User: ELPLUG
- 13: Subroutines Required:
ELTEST, P14PRT, MULTF, PØØF
- 14: Remarks: All arguments in calling sequence not defined were not used in subroutine.

1. Subroutine Name: MULTF
2. Purpose: To perform the matrix multiplication B transpose times A times B, where A is a symmetric matrix and B is a rectangular matrix.
3. Equations and Procedures:
$$C = B (\text{transpose}) * A * B$$

The routine first generates the product of a row of B transpose times each column of A and stores this in a temporary storage V. It then multiplies V times the appropriate columns of B to generate the corresponding row of C.
4. Input Arguments:
 - A : The symmetric input matrix doubly dimensioned 8x8 with only symmetric lower half needed.
 - NA: Order of A must be less than 9.
 - B: The rectangular input matrix doubly dimensioned 8x12 with size NA x NBC
 - NBC: Number of columns of B (less than 12)
 - V: A work storage vector of length NA.
5. Output Arguments:
 - C: The results of the multiplication, doubly dimensioned 12x12 with only symmetric lower half returned. Size is NBC x NBC.
6. Error Returns: None
7. Calling Sequence:
(A, NA, B, NBC, V, C)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: PLUG 14
13. Subroutines Require: None
14. Remarks: None

1. Subroutine Name: POOF
2. Purpose: Expand element stiffness matrix (lower symmetric by rows or upper symmetric by columns) and element thermal load vector and add the components into the expanded matrix and vector in their appropriate positions.
3. Procedure: Using the decoding vector determine the locations of the components of the element stiffness matrix in the new expanded (assembled stiffness) matrix and add these old element components into their new positions. The same procedure is used for the thermal load vector.
4. Input Arguments:
 - LIST: Decoding vector consisting of NORD components the subscript of each component gives the old (element) row or column and the component itself gives the row or column in the new expanded matrix.
 - NORD: Order of old element stiffness matrix (AKEL) also length of old thermal load vector (FTEL).
 - AKEL: Old element stiffness matrix (upper symmetric by columns) .
 - FTEL: Old thermal load vector.
5. Output Arguments:
 - AK: Expanded stiffness matrix (upper symmetric by columns) .
 - FCOL: Expanded thermal load vector.
6. Error Returns: None
7. Calling Sequence: (LIST, NORD, AKEL, FTEL, AK, FCOL)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required:
12. Subroutine Users: PLUG 8
13. Subroutines Required: None
14. Remarks: None

1. Subroutine Name: P14PRT
2. Purpose: To print out on the system output unit the variables in the input argument list.
3. Equations and Procedures:
4. Input Arguments:

D12:	variable printed out and labled	L12
D15:	" " " " "	L15
D35:	" " " " "	L35
ALX:	" " " " "	LAMX
ALY:	" " " " "	LAMY
ALZ:	" " " " "	LAMZ
PX:	" " " " "	PSIX
PY:	" " " " "	PSIY
PZ:	" " " " "	PSIZ
XP4:	" " " " "	XP4
YP4:	" " " " "	YP4
PH:	An 8 x 12 matrix printed out and labled ELEMENT TRANSFORMATION MATRIX	
5. Output Arguments: None
6. Error Returns: None
7. Calling Sequence:

(D12, D15, D35, ALX, ALY, ALZ, PX, PY, PZ, XP4, YP4, PH)
8. Input Tapes: None
9. Output Tapes: None
10. Scratch Tapes: None
11. Storage Required: None
12. Subroutine User: PLUG 14
13. Subroutines Required: None
14. Remarks: None

APPENDIX VI

SUBSYS DOCUMENTATION

A. INTRODUCTION

The following section consists almost wholly of information contained in the distributed documentation supplied by SHARE regarding SUBSYS. Alterations have been made to enable one version of SUBSYS to be compatible on a stand alone 7090/94 or on a Direct Couple System 7040/7090 or 7044/7094.

Recognition for the bulk of the documentation is deserved by Mr. David E. Bluett of Westinghouse Electric Corporation, author of the original SUBSYS documentation.

This report describes a package of programs which will operate upon any FORTRAN IV program in such a way as to produce a program tape. The programs may be Overlay or non-Overlay, and the program tape may contain any number of such programs. The tape may then be used as a mounted program library (similar to a CHAIN tape in FORTRAN II) or may be edited directly onto the system tape to produce executable subsystem(s) under IBSYS.

B. BACKGROUND

The need for a package such as SUBSYS arose out of a desire to put some high-activity, high-load-time Overlay codes somewhere within the framework of IBSYS to provide increased accessibility and decreased load and peripheral times. An attempt was first made to insert a large Overlay code into IBLIB, with the intention of still going through IBLDR, but eliminating the large object deck. This method of attack ran into considerable troubles, the greatest of which was due to the limited size of the Subroutine Name and Dependency Tables when doing a Librarian edit. It became obvious that the most desirable situation would be the ability to say:

\$EXECUTE XXXXXX

thereby completely eliminating the need for input decks and any connection with IBJØB. Examination of the IBSYS manual showed that a subsystem under IBSYS should be an absolute assembly and obey certain rules. It seemed that a FORTRAN program, operating under IBJØB, already obeyed these rules

more or less by definition, since IBJOB is itself a subsystem. The only problem seemed to be the conversion of the FORTRAN code to an absolute assembly - a somewhat formidable task. However, it soon became obvious that the main link of an Overlay job (including all the Library) was itself an "absolute assembly" once it was loaded, and that the link tape, once written, was also in absolute scatter-loading format. The problem was now reduced to three parts: (1) dumping out the main link after it was loaded by IBLDR, (2) modifying the Overlay tape to correspond to proper subsystem rules, and (3) combining these two entities into one, ready for editing onto the system tape for use as a subsystem under IBSYS.

To solve part 1, a small program called CPYLKO (copy Link 0) was written which receives control immediately after execution and merely writes the main link out on tape. For convenience, this program has been made part of .LØVRY, which also had to be modified to properly control the new subsystems.

Parts 2 and 3 were solved by a separate program, LNKSTK (Link Stack), which modifies and combines the main link (as written by CPYLKO) and the Overlay tape (as written by IBLDR) to form a two-file program tape.

Tests were performed, and it was proved that the output tapes from LNKSTK could be edited onto the system tape and successfully used as subsystems under IBSYS. Even though these subsystems were placed on the system tape after IBJOB and SØRT, load time was reduced by about a factor of 4, and peripheral time (for input) reduced to essentially zero. Card shuffling errors in binary decks (a large source of lost runs) were eliminated as was a large portion of the total job setup time. Since LNKSTK has the ability to pack all of the record, execution time was usually improved, except in the cases of excessive link tape rewinding (.LØVRY now must do a "backspace file" instead of a "rewind").

Once this part of the package was operational, it was realized that the program tapes produced by LNKSTK could be mounted and operate just as well by themselves as they did as subsystems on the system tape. Since more than one complete program may reside on the program tape, all that was needed was a small loading routine to perform the functions of SYSLDR, with the added feature of program selection. To provide this function, the SEARCH routine was written, and, in addition to subsystem generation, the SUBSYS package now provided the long-sought solution to the saving of Overlay tapes. It should be noted that the ability to save Overlay tapes came about essentially as a by-product of the process for subsystem generation.

C. INSTRUCTIONS FOR USE AND DETAILS ON THE SUBSYS PACKAGE

Assume that a User wishes to make a program tape from an existing PORTRAN IV Overlay program. Whether this tape will later be edited over as a subsystem or merely used as a "chain" tape is immaterial, since the technique for making the tape is the same in either case.

The special deck for .LØVRY (with CPYLKO) is inserted somewhere in the main link of the program, and the job is submitted for running in the following way:

- (1) Any desired combination of \$ATTACH or \$SWITCH cards if needed.
- (2) GØ (and any other options desired or needed) on the \$IBJØB card.
- (3) Only one link tape specified on the \$ØRIGIN cards.
- (4) The normal \$ENTRY card (if any).
- (5) No data (an end-of-file should immediately follow the \$DATA card).

The program will load (the Overlay tape being written where directed by the \$ØRIGIN cards) and execute by transferring to the pre-execution initialization section (PREEX). The first instruction in PREEX is TSX SYSIDR, 4, but a TTR to CPYLKO has been originated at SYSIDR in the IBSYS nucleus. The CPYLKO section of .LØVRY is thus entered immediately via SYSIDR.

The main link will now be written out as one big record on SYSCK2. If SYSCK2 is already the Overlay link tape, the output tape must be changed by altering an assembly parameter in CPYLKO. The size of the main link depends, of course on the last location used by this link, and this location is calculated in CPYLKO. The main link will be written from SYSIØC through this last word, preceded by a few communication and pointer words, and followed by an end-of-file. The output tape from CPYLKO is left un-rewound, and control returns to IBSYS via SYSRET. The cell SYSIDR in the nucleus has been altered by CPYLKO, so the next two cards in the deck (and the last two of this first phase of the job) must be:

\$IBSYS

\$RESTØRE

The next phase of the process is the combination of the

output tape from CPYLK0 with the Overlay tape to form the final program tape. This combination is made by the Link Stack program, which will normally be the next job on the input tape. It is strongly suggested that LNKSTK (Link Stack) itself be made a subsystem under IBSYS, since this greatly simplifies the deck setup and eliminates the need for protecting the Overlay tape during the loading of LNKSTK. Instructions for making LNKSTK a subsystem are included as Appendix VIII, and this description will proceed on the assumption that this has been done.

After the \$RESTORE card, the cards are as follows:

\$JOB

\$EXECUTE LNKSTK

(LNKSTK data card, giving name, tapes, and options)

End-of-File card

Next may come either a return to the monitor for signing off, a system tape edit, or a test run on the new program tape using the SEARCH routine.

Once LNKSTK is loaded, it will read its data card containing the program name and the tape information required (the data card format is detailed in Appendix VII), and perform the following operations:

1. Rewind all pertinent units and read the main link as written by CPYLK0.
2. Modify this link into proper scatter-loading format and write it as one record on the specified output tape, followed by an end-of-file.
3. Read the Overlay tape, modifying the link records appropriately.
4. Write the modified links on the specified output tape, followed by an end-of-file.
5. Print a map showing input and output record counts, word counts, etc.
6. Rewind all pertinent units and exit.

The program tape is written and ready for use. It may be edited onto the system tape, loaded by means of the SEARCH program or dismounted for later use.

As a summary by way of example, assume that it is desired to make a program tape from a FORTRAN IV Overlay code called TSTJØB, edit this onto the system tape immediately following IBJØB, and then run a sample case. The deck set-up would be as follows:

```

1      8              16
$IBSYS
$JØB
$EXECUTE          IBJØB
$IBJØB jobnam      GØ,MAP, etc.
$IBLDR DECK1
      (start of decks for main link of program "TSTJØB"
      Compiles and/or assemblies may be done in this run).

.
.
.
$IBLDR .LØVRY
      (special deck of .LØVRY with CPYLKO included somewhere
      in main link).

.
.
.
$ØRIGIN (start of link 1)

.
.
.
      (remainder of program)

.
.
.
($ENTRY card if normally included)
$DATA
End-of-file card
$IBSYS
$RESTØRE
$JØB
$EXECUTE          LNKSTK
                  (Link Stack data card, explained in Appendix VII)

```

End-of-file card

\$IBSYS

\$IBEDT

```
*EDIT      MAP,MØDS
*PLACE     TSTJØB,2,1,2
*REMARK    NØW IN NAME TABLE AS 2ND SUBSYSTEM, 2 FILES
*REMARK    PØSITIØN TAPE AFTER IBJØB
FILE *AFTER IBJØB
*REMARK    DUP IN TSTJØB FRØM SYSxxx
*DUP       SYSxxx,SYSUT1,2
*REMARK    ALL DØNE
```

End-of-file card

\$IBSYS

\$PAUSE SET UP NEW SYSTEM TAPE, etc.

.

.

\$IBSYS

\$JØB TSTJOB MAY NOW BE USED AS A SUBSYSTEM

\$EXECUTE TSTJØB

(sample data deck for TSTJØB)

End-of-file card

Obviously, any number of subsystems may be DUPed on in one edit, providing the proper *PLACE, *AFTER, and *DUP cards are used. In the IBSYS edit, the unit SYSxxx will be the LNKSTK output tape, which is one of the data card parameters.

As an alternate possibility, assume the activity of TSTJØB is not sufficiently high to warrant its inclusion as a subsystem, but that the load time is high enough to allow significant savings from the use of a program tape. The user therefore desires to make a program tape to be mounted on SYSLB2. It should be noted that program tapes produced by LNKSTK can only be mounted on one drive due to the changed structure of .LØVRY. In other words, if the program tape for TSTJØB is made to run on B5, then it must always run on B5. This so-called "running link tape" is one of the parameters on the LNKSTK data card and must be SYSLB2 for this version of SUBSYS. The following example illustrates the use of the SEARCH routine in conjunction with a program tape. The deck set-up is exactly the same as before, up through and including the EOF after the LNKSTK data card. The last three identical cards will be re-listed for continuity.

1 8

16

\$EXECUTE

LNKSTK

(Link Stack data card)

End-of-file card

```
$JOB
$EXECUTE          IBJOB
$IBJOB            GØ,MAP
$IBFTC CALL
C
```

CALL SEARCH (6HTSTJOB)

C

```
STØP
END
```

\$DATA

(Sample data deck for TSTJOB)

End-of-file card

This example assumes that SEARCH has been placed on the IBJOB library (IBLIB). If this is not the case, the binary deck for SEARCH would follow the END card of the FORTRAN program above. Note that the calling sequence to SEARCH is similar to that used for CHAIN in FORTRAN II, except that the tape to be searched is omitted since it is assumed to be SYSLB2.

Search finds the specified program on SYSLB2 by name and scatter-loads it right on top of itself, leaving only enough to execute a transfer to SYSTRA which will commence execution of the desired program. The time saved when running with a mounted program tape and using SEARCH is obviously most dependent on the time used to hang the tape. The time taken to load SEARCH and its calling routine and to find and load the program is usually no more than .004 hours.

D. SUMMARY

The SUBSYS package consisting of .LØVRY with CPYLKO, LNKSTK, and SEARCH can provide considerable savings in setup, peripheral, and main-frame time when used with 7090, 7094, 7094/2 FORTRAN IV Overlay and non-Overlay codes.

Since no modifications are involved to IBSYS or IBJOB, SUBSYS should be more "version independent" than other packages available which do involve system mods. SUBSYS has been tested on both version 12 and version 13 installations. This is a tape-oriented package, and its value to a disk-oriented user is questionable. It is left to the disk user to make such an evaluation.

More detailed descriptions of the three routines and their uses may be found in the Appendices. A sincere attempt has been made to thoroughly comment all the listings, so that, for detailed analyses, the documentation may be said to be an integral part of the coding.

APPENDIX VII
DETAILS ON LNKSTK

A. INTRODUCTION

The information needed by LNKSTK to produce a program tape is supplied by two sources: the communication words passed on by CPYLKO, and the LNKSTK data card. The communication words are obtained by LNKSTK when it reads the main link from tape, and are described in Appendix IX.

B. LNKSTK DATA CARD FORMAT

Field	Columns	Contents
1	1 - 6	The program name as it will appear in the first record of the program and on the \$EXECUTE card or SEARCH argument. The name must be BCD, 6 character max., left adjusted in the field with trailing blanks if less than 6 characters. If this program is to become a subsystem, the name must be different from any other system record name.
2	8 - 13	Input tape on which LNKSTK may expect to find the main link as written by CPYLKO. This unit must be specified as SYSxxx, and would be SYSCK2 if running with the distributed version of CPYLKO which uses SYSCK2 for its output.
3	15 -20	Input tape containing the Overlay links as written by IBLDR. This unit, which must also be specified by its SYSUNI name, is the tape presently containing the Overlay links, regardless of what SYSUNI it may have been (due to \$ATTACH and \$SWITCH cards) when the program was loaded. If this is not an Overlay job, the word "NOLINK" must be inserted in this field.
4	22 - 27	This field is another SYSUNI name which specifies the "running link tape", or the unit on which the program tape must be mounted when running with the SEARCH program and must be SYSLB2 for the distributed version of LNKSTK and SEARCH.

Field	Columns	Contents
5	29 - 34	If this is not an Overlay job, the word "NØLINK" may be inserted in this field. Output tape for LNKSTK, also a SYSUNI name. This name may be the same as that in Field 2, but may not be the same as the Overlay link tape in Field 3. It may be, but is not necessarily the same as the "running link tape" in Field 4. If the record packing option is desired, this field should contain the word "PACK". If PACK is specified, all records for each Overlay link (written as 464 words by IBLDR) will be combined to form one long record. The .LRECT table generated by the loader is modified by LNKSTK to reflect the new positions of the links on tape. So called "remote sections" specified by \$INCLUDE cards cannot be handled by LNKSTK*. This feature means that considerably less tape is used for the link section of the program, due to fewer record gaps. Link loading is considerably faster, usually resulting in an overall improvement in execution time. If this option is not specified, the records produced will be 465 words, a BCD name being added to each record (standard system record format). This option is meaningless for a non-Overlay job.
6	36 - 39	
7	41 - 45 (if Field 6 was present)	Rewind options applying to the LNKSTK output tape. Either RB and/or RA, in either order, may occupy this field, or the field may be null.

* See Version 13 IBJØB manual, C28-6389-0, page 43.

Field	Columns	Contents
7	36 - 40 (if Field 6 was absent)	To permit the stacking of more than one program on the output tape, the rewinds are strictly controlled by these options. If RB (rewind before writing this program) is specified, LNKSTK will perform a rewind on the output tape immediately before it attempts to write the modified main link. If RA (rewind after writing this program) is specified, LNKSTK will finish writing the last Overlay link, write an EOF, and write a 3 word trailer record containing the word "ENDTPE". It will then rewind the output tape. This trailer record will cause the word ENDTPE, to scatter-load into SYSFAZ, enabling the SEARCH routine to recognize the end of the program tape. RB must be specified for the first (or only) program to be put on the output tape, while RA must be specified for the last (or only) program. If more than two programs are to be stacked on the output tape, any "middle" programs would have neither option specified to insure that no rewinds are performed.

All fields are separated by commas (or any other non-blank delimiter). The remainder of the card after col. 45 is available for comments. Fields 1 - 5 must be present in the columns assigned, while the last two fields are optional.

C. EXAMPLES

col. 1
*

```
TSTJØB,SYSC2,SYST2,SYSLB2,SYCT5,PACK,RB,RA
SIFT ,SYSC2,SYSLB3,SYSLB2,SYSC2,PACK
SMALJB,SYSC2,NØLINK,NØLINK,SYSLB2,RB
BIGJØB,SYST3,SYSC2,SYSLB2,SYST7,PACK,RA
```

The setup for stacking more than one program on the output tape is merely an extension of the case for one program. The order of jobs in the deck would be similar to the following:

```

First Program
LNKSTK run (with RB on the data card)
Second Program
LNKSTK run (with no rewind options)
.
.
.
(nth Program)
(nth LNKSTK run, no rewinds)
.
.
.
Last Program
LNKSTK run (with RA on the data card)

```

The output tape would, of course, be the same on all these LNKSTK data cards, while the other options may be as desired. Overlay and non-Overlay programs may be stacked on the same tape. A double EOF will follow a non-Overlay program, so that each program will be 2 files for the SEARCH routine (see Appendix X). If the system rewinds the LNKSTK output tape between jobs or job segments, these rewinds must be circumvented if more than one program is to be stacked on a given output tape.

In addition to writing the main link in scatter-load format, LNKSTK provides entries for the following communication cells:

1. Location 2 - TTR .LXSTR
2. Location 10₈ - TTR .FPTRP
3. Location 230₈* - A corrected skew-check mask.
4. SYSTRA - TTR PREEX (start of pre-execution initialization)
5. SYSGET - "IBSxec"
6. SYSFAZ - program name from data card

- * A "feature" has been added in IBSYS Version 13 such that any IØCP with a word count greater than 37777₈ which enters SYSTCH causes the record to be treated as if it were redundant. Entry 3 above corrects this, but is only done if LNKSTK is assembled for Version 13. See "Assembly Parameters".

- 7. SYSLØC - zero
- 8. .JLIN (line ctr.) - zero
- 9. SYSCUR - name of each record (main or Overlay as it is loaded)

The program name enters SYSFAZ and SYSCUR when the main link is loaded, and the name remains in SYSFAZ throughout the run. Each link record stores its name in SYSCUR as it is loaded, so that the contents of SYSCUR will always represent the last record read.

The link record name is a combination of the program name and the link number if record packing is in effect, or the program name, link number, and record number if packing is not in effect.

Examples from "TSTJØB":

Packing: TSTJØ4 (Link 4)

No Packing: TST721 (Link 7, Record 21)

All the link and record numbers will be BCD. The link will occupy 2 characters if it becomes greater than 9.

D. ASSEMBLY PARAMETERS

1. VRSION - assembled as 13 by a "SET". Pertains to the existence of SYSUT5-SYSUT9 and to skew-mask correction. See Appendix X, since the same parameter is contained in SEARCH to control I-Ø table assembly.
2. UNIT - assembled as SYSCK2. This is the output unit on which LNKSTK will dump itself if entered by a \$ENTRY CPYLNK card. It must therefore be specified as the input unit on the LNKSTK data card when producing a program tape from LNKSTK itself (see Appendix VIII).

E. ERROR MESSAGES

If any error is detected during a LNKSTK run, a message:

ERRØR IN LINK STACK AT RELATIVE LØC XXXXX ØCTAL (SEE LISTING).
CANNØT PRØCEED. is printed off-line. Examination of the comments on the listing will reveal the nature of the error.
The message:

ERROR IN LINK STACK. FLUSH ANY REMAINING PARTS ØF THIS
JØB HIT START TØ DUMP

ØPERATØR ACTION PAUSE

is printed on-line. Depressing the START key will cause a core dump via SYSDMP (AC, MQ, etc. are saved), but the operator is responsible for flushing the rest of the run.

F. DUMP FEATURE

If User modifications are made to LNKSTK, or if there seems to be trouble during a LNKSTK run, it may be desirable to obtain a core dump immediately after LNKSTK is through with its processing. To provide this facility, a feature has been added to LNKSTK such that the console entry keys are examined before LNKSTK returns to IBSYS via SYSRET. If any prefix key (S, 1, or 2) or any combination of prefix keys is down, LNKSTK will exit via SYSDMP rather than via SYSRET.

The operator must, of course, be informed that the key(s) are to be set before the termination of the LNKSTK run.

G. RESTRICTIONS

The fact that LNKSTK cannot handle "remote" sections specified on \$INCLUDE cards has already been mentioned, as has the fact that only one link tape may be called for on the \$ORIGIN cards.

Other problems may arise from certain record size limitations are imposed by SUBSYS and the systems which it must use. LNKSTK has a buffer size of 28000₁₀ words (66540₈), and this represents the maximum size of any link record (the main link would usually be the largest record, since it contains all the library routines and possibly some named COMMON). When running strictly from a program tape, the SEARCH routine can load a record in excess of the LNKSTK maximum (actually 28246₁₀ or 67126₈). However, things are not so simple when using the system editor. At the time of this writing, no documentation of any record size limitation has been found in either the coding for EDITOR or the IBSYS manual, but examination of the actual I-Ø command in EDITOR shows the following limits:

IBSYS Ver. 12 (EDITOR Ver.5) - 24607₁₀ or 60037₈
IBSYS Ver. 13 (EDITOR Ver.6) - 23840₁₀ or 56440₈

Analysis of a LOGIC or MAP will show whether a program is within these limits. Insertion of one or two redundant \$ØRIGIN cards in the main link is usually all that is needed to bring the program back into line with LNKSTK and EDITOR.

In IBSYS Ver. 13, SYSLDR has been changed to check for skew-errors by insisting that no bits enter bit position 3, 19, and 20 of any scatter load IØCP. This has been corrected by the skew-mask described previously, which effectively allows SYSLDR to load a record of any size. Regardless of the method used, the practical size limit is still SYSEND-SYSØRG.

APPENDIX VIII

LNKSTK AS AN EXECUTABLE SUBSYSTEM

A. INTRODUCTION

If LNKSTK is loaded from a binary deck, not only is the deck setup for making a program tape somewhat more complicated, but certain SYSUTx files (which might contain the Overlay links) must be protected during the loading of LNKSTK. The ideal situation is to have LNKSTK reside on the system tape as an executable subsystem. This adds no appreciable "bulk" to the system tape, since LNKSTK is only one file, consisting of one 2200 word record, and the deck setup of:

```
$JOB
$EXECUTE    LNKSTK
            (LNKSTK data card)
```

is certainly as compact and simple as could be desired.

B. INSTRUCTIONS FOR MAKING LNKSTK ITSELF A SUBSYSTEM

LNKSTK has its own built-in equivalent of CPYLKO, called CPYLNK, which may be entered in the case where it is desired to have LNKSTK operate on itself. Since this entry is not the general case, it is not made automatically as it is in the CPYLKO section of .LØVRY, but must be made by a \$ENTRY card. The deck setup to make a program tape from LNKSTK itself and edit it over to the system tape immediately after IBJØB is as follows:

```
1      8      16

$IBSYS
$JOB
$EXECUTE    IBJØB
$IBJØB      GØ,MAP
$IBLDR LNKSTK
            (LNKSTK binary deck)
$DKEND LNKSTK
$ENTRY      CPYLNK
$DATA
```

	<u>1</u>	<u>8</u>	<u>16</u>

data card: LNKSTK,SYSC2,NØLINK,NØLINK,SYSC2,RB,RA
End-of-file card
\$IBSYS
\$JØB
\$IBEDT
*EDIT MAP,MØDS
*PLACE LNKSTK,1,1,2
FILE *AFTER IBJØB
*DUP SYSC2,SYST1,1
End-of-file card
\$IBSYS

In this example, the \$ENTRY card will cause LNKSTK to write itself out on SYSC2 (this tape is an assembly parameter in LNKSTK) before transferring to its normal entry. It will then read its data card and proceed as it would on any non-Overlay job. Note the following on the data card:

1. The program name is specified as LNKSTK (similarly on the *PLACE card), and this is the name that must be specified on the \$EXECUTE card when using the sub-system.
2. The input tape for the main (and only) link is specified as SYSC2.
3. The NØLINK feature is specified in place of the normal link tape designations, signifying that this is not an Overlay job.
4. SYSC2 is also used for the output tape, illustrating the fact that the output tape for LNKSTK may be the same as the main link output tape.
5. The PACK option is not specified, since this would be meaningless for a non-Overlay job.
6. Since this is the only program to be put on the output tape, both the RB (rewind before) and RA (rewind after)

The new system tape, containing LNKSTK as the 2nd subsystem under IBSYS, will be produced on whatever unit is attached as SYST1.

APPENDIX IX

DESCRIPTION OF THE MODIFIED .LØVRY WITH CPYLKO

The standard IBM routine .LØVRY, whose function is the loading of Overlay links, has been somewhat modified for use with the SUBSYS package. The largest change, of course, is the addition of the CPYLKO routine, which is discussed elsewhere in this write-up. Other changes are as follows:

1. The table of legal link tapes (UNITAB) has been reduced to one location, since now only one link tape is used, whether running as a subsystem or as a program tape. All general references to UNITAB (as a table) have been removed, and the UNITAB index in the .LRECT table is no longer examined. The single UNITAB cell in .LØVRY is now set by LNKSTK during its processing of the main link, the desired "running" link tape being specified on the LNKSTK data card. Since only one link tape may now be used, certain codes which have an extremely high activity of link loading and link tape rewinding may run considerably longer under this system, possibly enough to negate its worth. This is something that is best determined empirically.
2. All disk and hypertape coding has been removed for simplicity, since SUBSYS is a tape oriented package.
3. The IBSYS Version 13 Mod. which adds the skew error check is not included, since this has not proved to be troublesome in our installation. It may easily be inserted by the User if desired.
4. The subsystem (or program tapes) are now two files, the main link being one, and the Overlay links the second. .LØVRY must then skip over the EOF after the main link on the first entry and after each BSF. BSF's now replace rewinds when a rewind is requested by REW on the \$ØRIGIN card.

5. If the PACK option is specified on the LNKSTK data card, all records for one Overlay link will be packed into one long record, thereby reducing the length of tape needed for the program and shortening the time for link loading. However, the .LRECT table produced by the loader will no longer reflect the correct record counts and tape positions for each link. This table is automatically modified in LNKSTK to reflect the true "one record per link" status of the link file on the tape. No change to .LØVRY is involved here.

Aside from these changes, .LØVRY is essentially the same. The number of words removed is about the same as the number of words added by the addition of CPYLKO. In the process of writing the main link from SYSLOC through its last word, CPYLKO also passes on to LNKSTK:

1. The address of PREEX
2. The addresses of .LXSTR and .FPTRP
3. The address and length of the .LRECT table.
4. The address of UNITAB in .LØVRY.

All other information needed by LNKSTK is present on the data card.

The length of the main link is calculated at execution time in CPYLKO. A search is performed from SYSEND-1000 backward (towards location 0), looking for the first word that is not an STR 0,,0. This is assumed to be the last word of the main link. This is reliable as long as IBLDR performs as it is supposed to in its final section, and this method is certainly preferable to using an assembly parameter as was formerly done.

The standard error message in .LØVRY is written on first entry if the UCB's for the unit specified in UNITAB and SYSLB1 show both these units at load point.

APPENDIX X

DESCRIPTION OF THE SEARCH ROUTINE

A. INTRODUCTION

The general function of the SEARCH routine has been described earlier in this manual. The scatter-load and redundancy-checking routine is originated at 72000₈ to prevent it from being destroyed as a large main link is scatterloading in. The initialization and table sections of the program will be destroyed in this process, since they are needed only once. The search for the program is dependent on the BCD name supplied in the calling sequence. The program name from tape will be scatter-loaded into SYSFAZ. When SYSFAZ becomes non-zero, SEARCH compares its contents with the name from the CALL. If they are the same, the scatter-load is allowed to continue, and, if not redundant, control then passes to the main link via SYSTRA. If the names are not the same, the scatter-load is immediately terminated and 2 files are skipped. The process is then repeated until either the program is found or the word "ENDTPE" enters SYSFAZ, signifying the end of the tape. If this trailer label is encountered, an error message is printed and the program exits via SYSDMP.

B. CALLING SEQUENCE

In FORTRAN or MAP: CALL SEARCH (Arg1)

where Arg1 is the program name as 6Hxxxxxx

It is strongly suggested that SEARCH be edited onto the IBJØB library as soon as it has been reassembled for the particular installation.

C. ASSEMBLY PARAMETERS

1. VRSIØN is assembled as 13 by a "SET", and represents the version of IBSYS in use. It pertains only to the existence of SYSUT5 - SYSUT9 and is used with IFT's and IFF's to control the assembly of the I-Ø tables.
2. BCDTAB - table of BCD SYSUNI names.
3. SYSTAB - table of SYSUNI indices.
4. RDSTAB - tables of read selects.

All of these I-Ø tables must be examined and made to conform to the installation I-Ø configuration.

D. ERROR MESSAGES

Due to a number of possible causes such as illegal tape designation, the word "ENDTPE" entering SYSFAZ, etc., the message:

```
PROGRAM 'XXXXXX' IS NOT ON SYSLB2 . . . SORRY
is printed on-line, followed by a dump. If the
arguments look all right, the cell SYSFAZ should be
examined.
```

If the main link record is still redundant after 10 tries, the message:

```
REDUNDANCY READING SYSLB2 . . . SEARCH DISCONTINUED is
printed on-line, followed by a dump.
```

APPENDIX XI
DECK NUMBERING SEQUENCE

A. SUBSYS DECKS

SS00	-	LODFMT
SS10	-	AFMTII
SS20	-	BFMTII
SS22	-	RESET
SS23	-	BEXEQ
SS30	-	USERO4
SS32	-	SRESET
SS33	-	SEXEQ

B. NEW FORMAT SYSTEM DECKS

F03A - MATSUP

C. STRUCTURAL SYSTEM DECKS

S000	-	US04
S001	-	NTEST
S002	-	REC1
S010	-	LOGFLO
S100	-	US04A
S102	-	INDECK
S104	-	CONTRL
S106	-	COPYDK
S108	-	INFUT
S110	-	FRED
S112	-	BOUND
S114	-	ELEM
S116	-	MATCH
S118	-	LAG
S120	-	FGR LDS
S122	-	FMAT
S124	-	SHIFT
S126	-	REFORM
S128	-	PHASE1
S130	-	LATCH
S132	-	FORMIN
S134	-	PHASE2
S136	-	OPEN
S138	-	CHEK
S140	-	OUTINT
S142	-	FLOADS
S144	-	FTR
S200	-	US04B
S202	-	ININT
S204	-	DEFLEX
S206	-	FELEM
S208	-	SQUISH
S210	-	ELPLUG
S212	-	REC3
S214	-	REC4
S216	-	MINV
S218	-	AXTRA2
S220	-	MAB
S222	-	MSB
S224	-	BCB
S226	-	MATB
S228	-	SYMPRT
S230	-	LOC
S232	-	ELTEST
S234	-	MPRD
S236	-	TPRD
S238	-	AI
S240	-	BINT

S242	-	AK
S244	-	AM
S246	-	IFAC
S248	-	FJAB
S250	-	F6219
S252	-	F6211
S254	-	AJ
S256	-	COEF
S258	-	F89
S260	-	FF100
S262	-	AXTRA1
S264	-	AXTRA3
S266	-	ELPRT
S268	-	OUTMAT
S270	-	US461
S272	-	US462
S274	-	US463

D. QUADRILATERAL THIN SHELL ELEMENT DECKS

B100	-	PLUG1
B102	-	CC21
B104	-	MABC
B106	-	NEWFT
B108	-	CDELPQ
B110	-	CHDELL
B112	-	PIPRTA
B114	-	CK11
B116	-	CT11
B118	-	MATI60
B120	-	CTOGM
B122	-	CTGRM
B124	-	CC1
B126	-	CMMASS
B128	-	CSTM
B130	-	CDM
B132	-	CFMTS
B134	-	CFMV
B136	-	PRT1
B138	-	CK22
B140	-	CTGB
B142	-	MATI70
B144	-	CTOGB
B146	-	CTGRB
B148	-	CC2
B150	-	CFP
B152	-	CFPB
B154	-	CSTF
B156	-	CDF
B158	-	CDFX
B160	-	CDFY
B162	-	CFFTS
B164	-	CFFV
B166	-	CFMAS

E. TRIANGULAR THIN SHELL ELEMENT DECKS

B200	-	PLUG2
B202	-	ASSY2
B204	-	DCD
B206	-	DTAPR
B208	-	MATPR
B210	-	NEWFT1
B212	-	PTBM
B214	-	PTMGS
B216	-	DPQINT
B218	-	PKM
B220	-	PSTM
B222	-	PFMTS
B224	-	PFMV1
B226	-	APRT
B228	-	PTFGS
B230	-	PKF
B232	-	CCB
B234	-	PFP
B236	-	PSTF
B238	-	PFFTS
B240	-	PFFV1
B242	-	PNC1D
B244	-	PNG1D
B246	-	EPRT
B248	-	PLAS2D
B250	-	PTBF

F. TOROIDAL RING ELEMENT DECKS

B500	-	PLUG5
B502	-	MSTR
B504	-	ROMBER
B506	-	F4
B508	-	F5
B510	-	F6
B511	-	QUADI
B512	-	BMATRX
B514	-	DMATRX
B516	-	GAMMAT
B518	-	FCURL
B520	-	PLMX
B522	-	SCRLM
B524	-	SOLVE
B526	-	PRINT5

G. TRIANGULAR RING ELEMENT DECKS

B600	-	PLUG6
B602	-	EXPCOL
B604	-	EXPSIX
B606	-	TRAIC
B608	-	TESTJ
B610	-	TRCPRT
B612	-	TRAIE
B614	-	TIEPRT
B616	-	TRAIK
B618	-	TIKPRT
B620	-	TRAIFP
B622	-	TFPRT
B624	-	TRAIFT
B626	-	TFTPRT
B628	-	TRAIS
B630	-	TISPRT
B632	-	TRAITS
B634	-	TTSPT
B636	-	TRAIM
B638	-	TIMPRT
B640	-	TRAIFS
B642	-	TFSPT
B644	-	TRAIST
B646	-	TSTPRT
B648	-	PLCPRT

H. FRAME ELEMENT DECKS

B700	-	PLUG7
B702	-	INCRF
B704	-	P7PRT
B706	-	CTS
B708	-	CTCQ
B710	-	CECC

I. SHEAR PANEL ELEMENT DECKS

B300	-	PLUG14
B302	-	MULTF
B304	-	POOF
B306	-	P14PRT

APPENDIX XII
REVISIONS TO FORMAT SYSTEM DECKS

Deck Number: F010

Subroutine Name: PREP

Purpose of Revision: Provide the capability for suppressing input matrices in an abstraction instruction

Method: Fortran statement number 200 was changed to initialize the variable NUMSUP to zero. NUMSUP was added to the calling sequence to subroutine INST and upon return will contain the number of input suppressed matrices located during compilation of the input abstraction instructions. If NUMSUP is non-zero upon return from INST, then subroutine MATSUP is called to introduce the input suppressed matrices into the Format system.

Deck Number: F030

Subroutine Name: INST

Purpose of Revision: Provide distinct names for suppressed matrices and record the number of input suppressed matrices encountered while compiling the abstraction instructions

Method: The variable NUMSUP was added to the calling sequence of INST and inserted into the calling sequence for INST90 to record the number of input suppressed matrices located. The variable KOUNT was initialized in INST as zero and inserted in the calling sequence to INST90 to be used as a counter to ensure the generation of unique suppressed matrix names.

Deck Number: F043

Subroutine Name: INST90

Purpose of Revision: Introduce unique matrix names into the Format system for both output and input suppressed matrices for the .USERXX. form input abstraction instruction

Method: The variables KOUNT and NUMSUP were added to the calling sequence for subroutine INST90, KOUNT to indicate the next unique suppressed matrix name and NUMSUP to record the number of input suppressed matrices encountered. Whether input or output, a suppressed matrix is located and a name assigned to it by the same procedure. All blanks have been removed from the input instruction by subroutine PUTL1. The instruction is scanned, first the output side, then the input side. Whenever a matrix position has length zero, i.e. the matrix name was blank, the suppressed name is created by inserting four slashes for the first four characters and adding one to KOUNT and inserting that value as the last two characters. The sign of the matrix is set to plus. If the suppressed matrix was an input matrix, i.e. was encountered on the right sign of the equal sign, then NUMSUP is incremented by one.

Deck Number: F044

Subroutine Name: MATR

Purpose of Revision: Provide the capability of placing card input matrices on the same data set as input suppressed matrices, if necessary

Method: If card input matrices are present then subroutine MATR is called to place these matrices on NDATA, the data set selected by the Format pre-processor for that purpose. However, if input suppressed matrices were present then they already exist on NDATA at the time that MATR is called. Therefore MATR had to be revised to check NUMD, the variable indicating the number of matrices already on NDATA, before recording card input matrices on NDATA. If NUMD is zero then NDATA is rewound and a data set header written and the card input matrices recorded. If NUMD is non-zero, then NDATA is searched until the data set trailer is located, then backspaced over the data set trailer and then the card input matrices are recorded.

Deck Number: F050

Subroutine Name: ALOC

Purpose of Revision: Pass the value of IPRINT, the Format system print control, to subroutine ALOC4 for transmittal when operating under SUBSYS control

Method: The variable IPRINT was added to the calling sequence for ALOC and inserted into the call statement to ALOC4.

Deck Number: F061

Subroutine Name: ALOC31

Purpose of Revision: Indicate to the Format system the number of scratch data sets required to execute the .USER04. instruction

Method: The variable MINSCR(94) was set equal to four.

Deck Number: F062

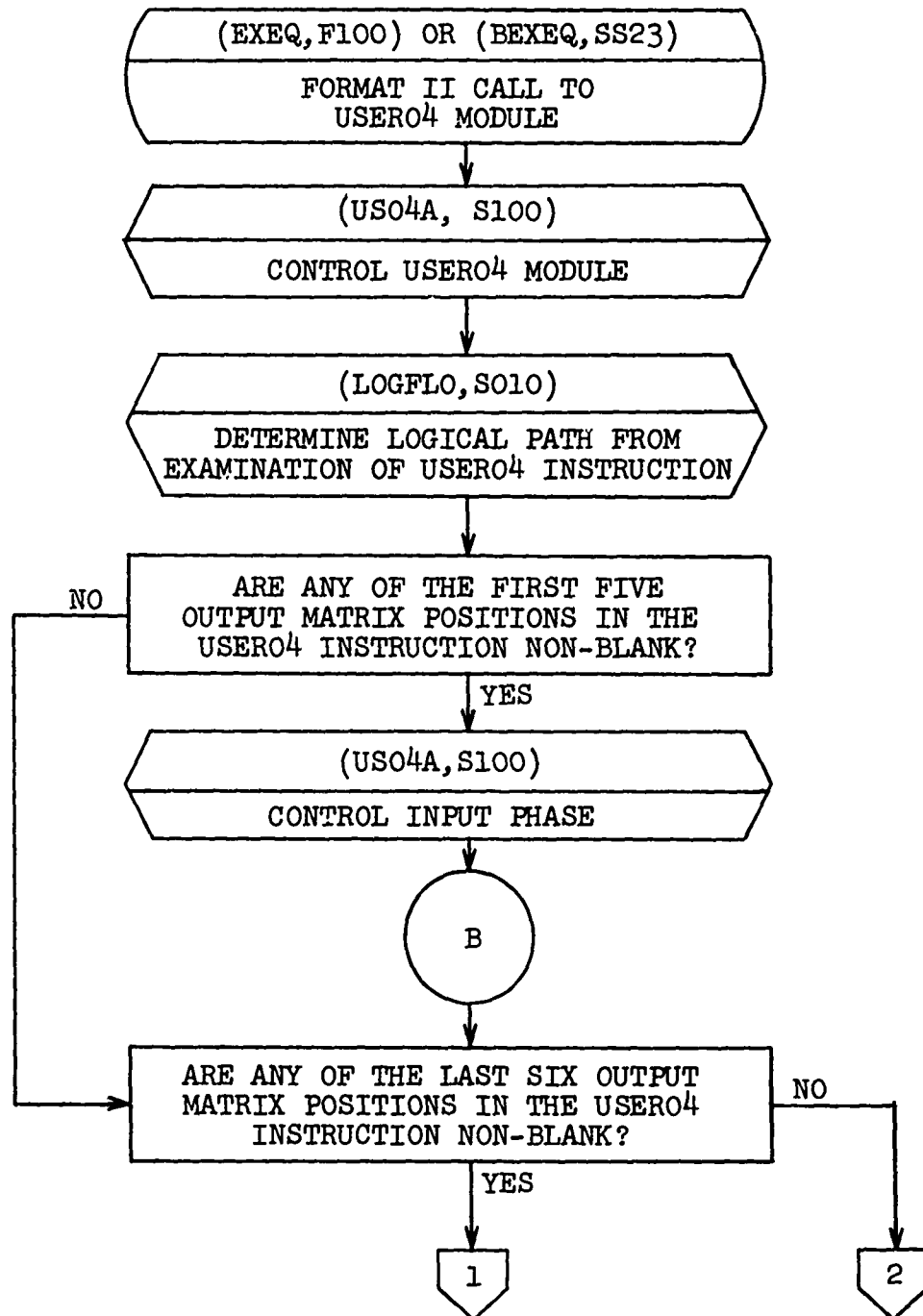
Subroutine Name: ALOC4

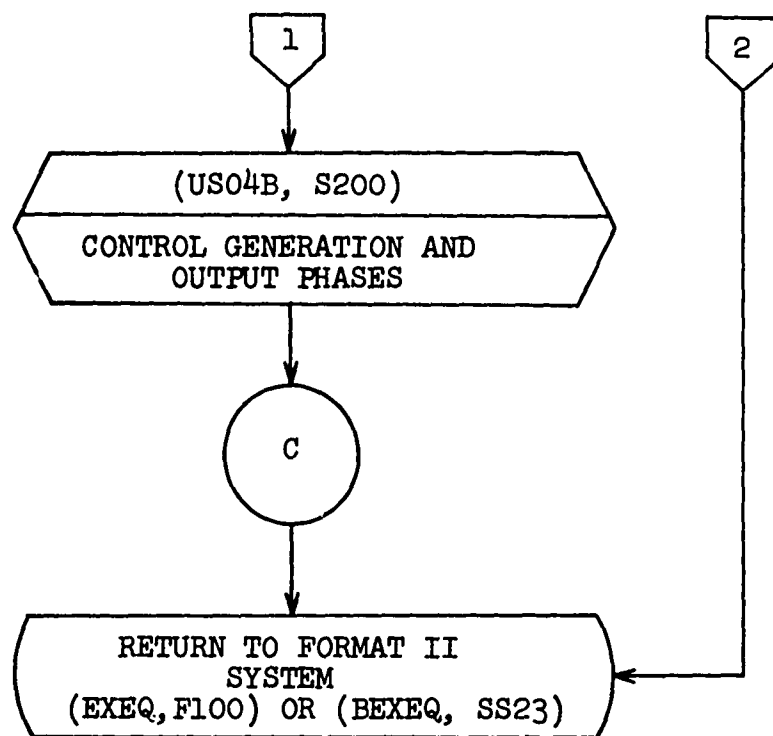
Purpose of Revision: Store on the instruction data set, NINST, the necessary data for re-initialization of program constants for operation under Subsys control

Method: When proceeding from program to program under Subsys control, the necessary system parameters must be reset at the start of each program. The values of the parameters are obtained as follows: NPIT, the system input unit, NPOT, the system output unit, KONST, the maximum matrix size capability and NWORK, the number of available work storages are obtained via the COMMON statement in ALOC4. The value of IPRINT is received through the calling sequence of ALOC4. These five system parameters, NPIT, NPOT, KONST, NWORK and IPRINT, are added as extra words to the return instruction recorded on NINST.

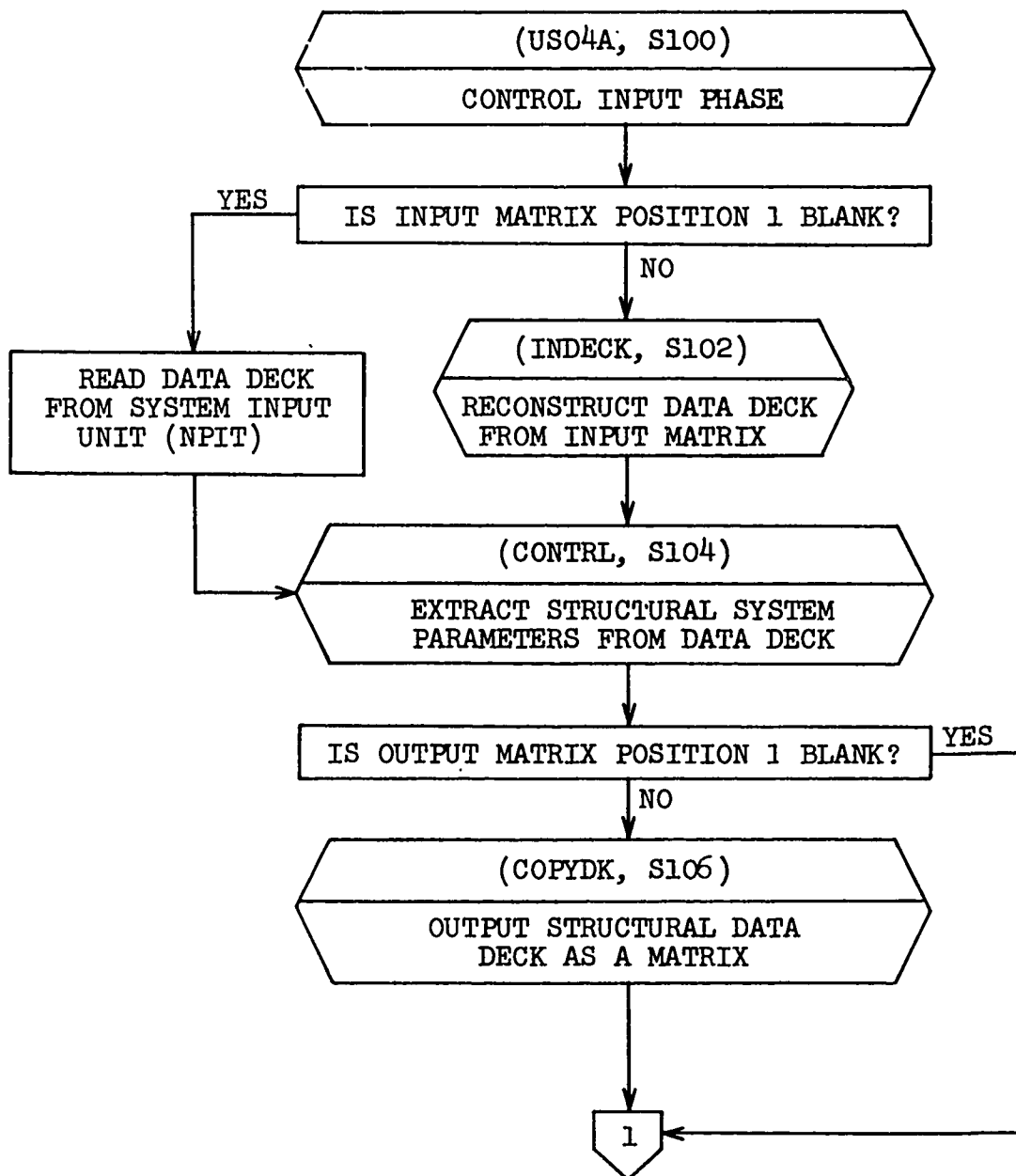
APPENDIX XIII
LOGICAL FLOWCHARTS

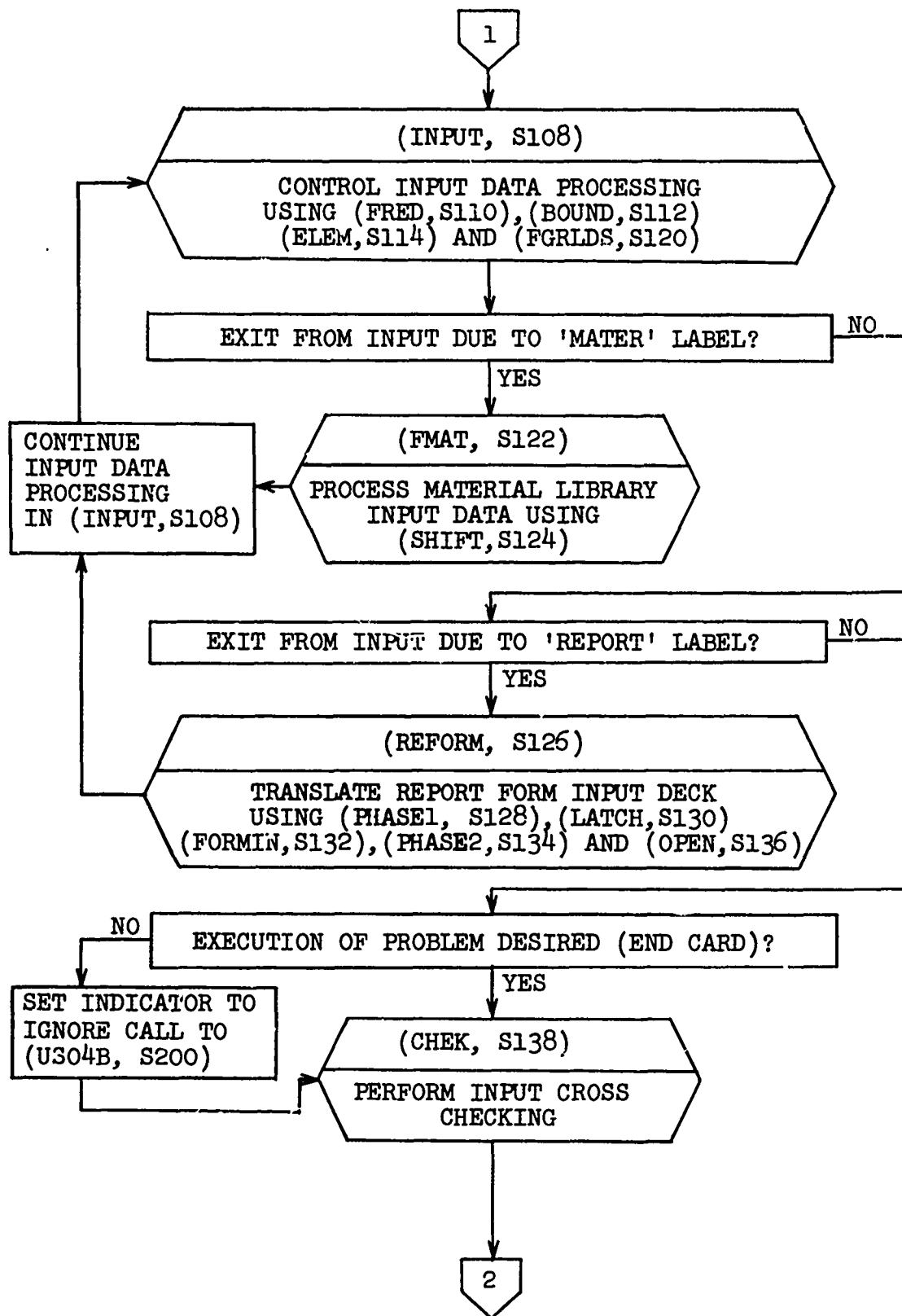
A. STRUCTURAL GENERATIVE SYSTEM LOGIC FLOW

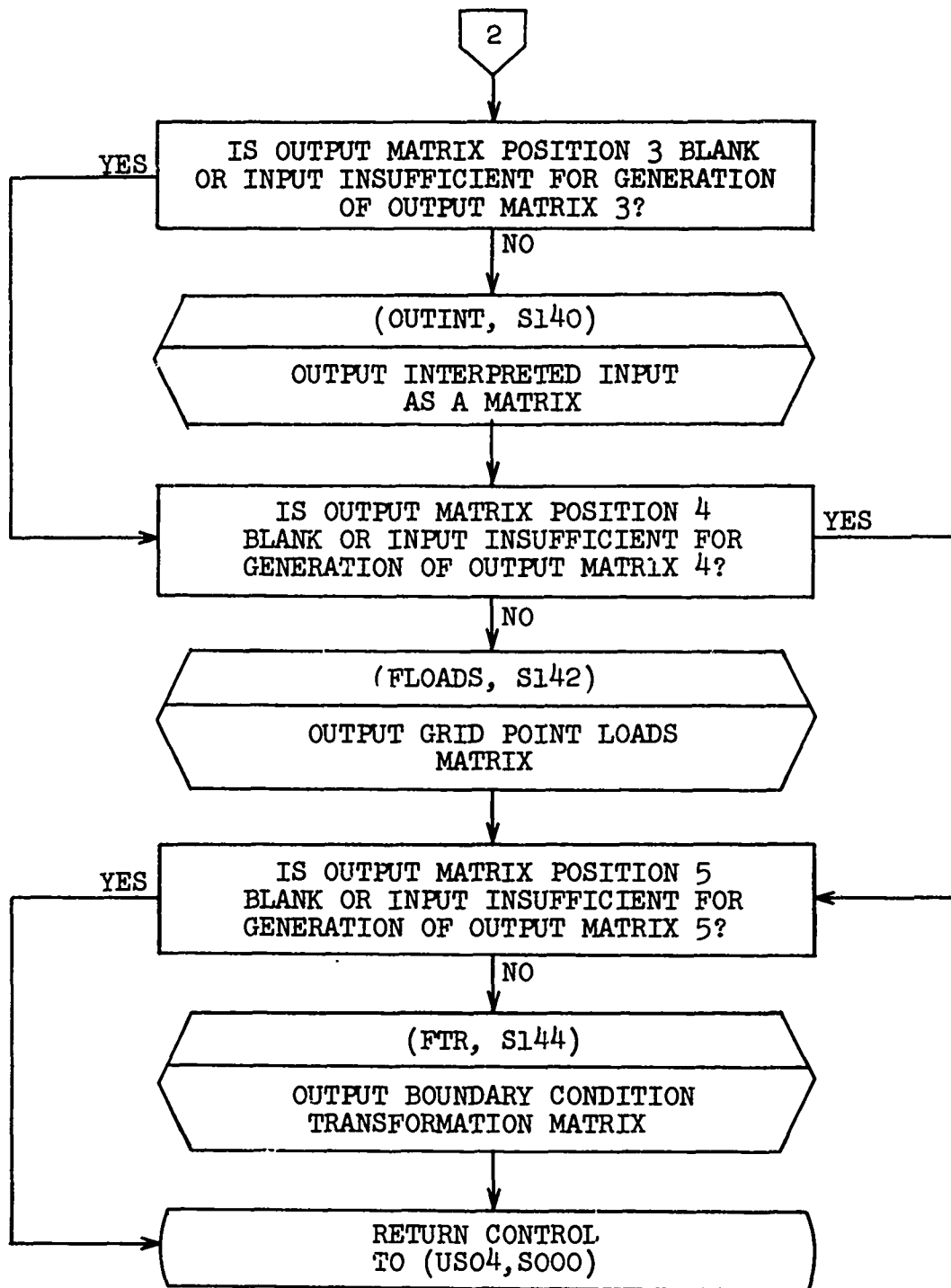




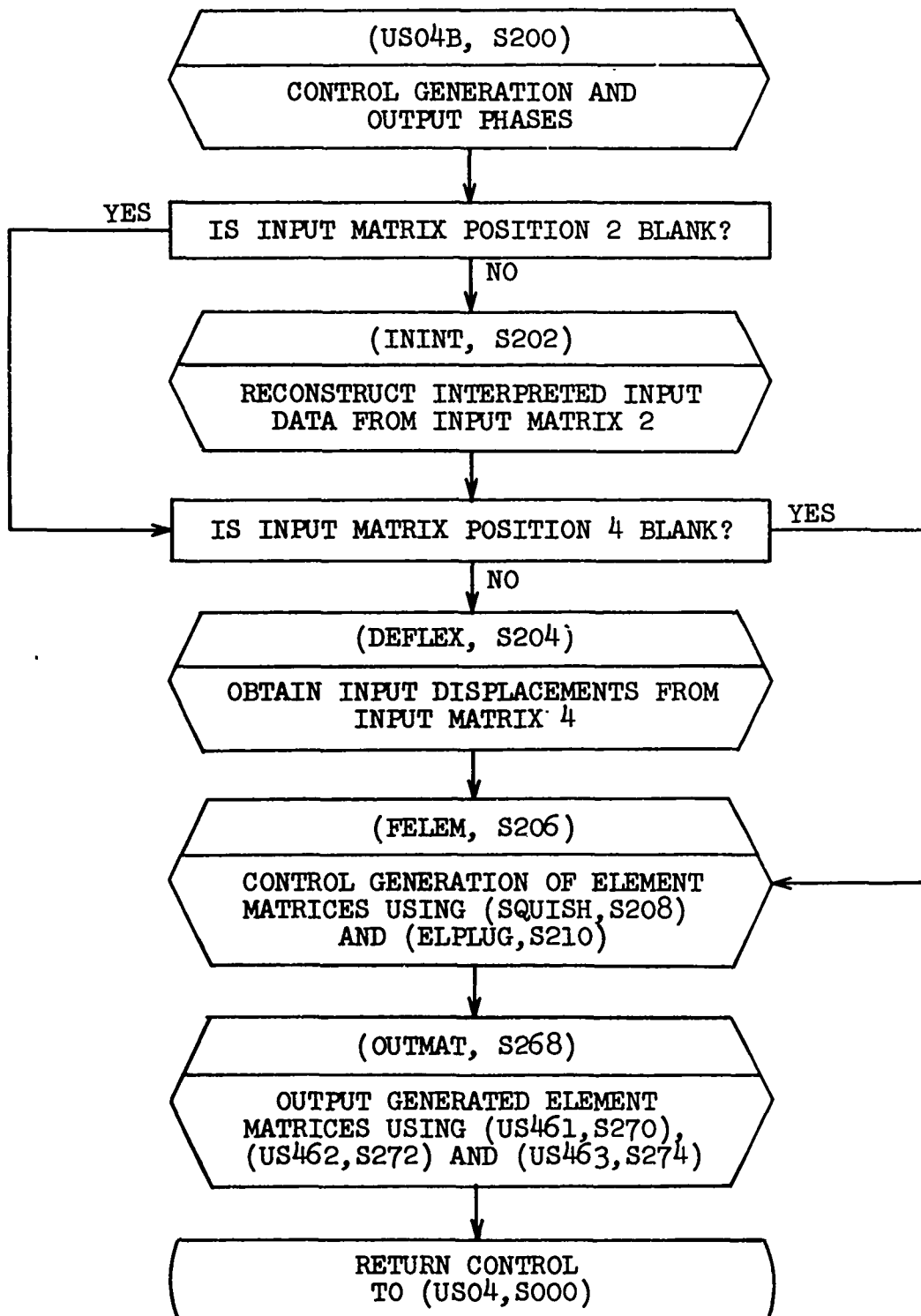
B. INPUT PHASE LOGIC FLOW







C. GENERATION AND OUTPUT PHASES LOGIC FLOW



UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D		
(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)		
1. ORIGINATING ACTIVITY (Corporate author)		2a. REPORT SECURITY CLASSIFICATION
Bell Aerosystems a Textron Company		Unclassified
		2b. GROUP
		N/A
3. REPORT TITLE		
MAGIC - An Automated General Purpose System for Structural Analysis Volume III - Programmers Manual		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates)		
Final Report		
5. AUTHOR(S) (First name, middle initial, last name)		
DeSantis, Daniel		
6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
January 1969	368	None
8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)	
AF33(615)-67-C-1505	AFFDL-TR-68-56, Vol. III	
b. PROJECT NO.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
1467	None	
c.		
Task No. 146702		
d.		
10. DISTRIBUTION STATEMENT		
This document has been approved for public release and sale. Its distribution is unlimited.		
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY
None		Air Force Flight Dynamics Laboratory Research & Technology Division Wright-Patterson AF Base, Ohio
13. ABSTRACT		
An automated general purpose system for analysis is presented. This system, identified by the acronym "MAGIC" for "Matrix Analysis via Generative and Interpretive Computations," provides a flexible framework for implementation of the finite element analysis technology. Powerful capabilities for displacement, stress and stability analyses are included in the subject MAGIC System for structural analysis. The matrix displacement method of analysis based upon finite element idealization is employed throughout. Six versatile finite elements are incorporated in the finite element library. These are: frame, shear panel, triangular cross-section ring, toroidal thin shell ring, quadrilateral thin shell and triangular thin shell elements. These finite element representations include matrices for stiffness, incremental stiffness, prestrain load, thermal load, distributed mechanical load and stress. Documentation of the MAGIC System is presented in three parts; namely, Volume I: Engineer's Manual, Volume II: User's Manual and Volume III: Programmer's Manual. The subject Volume, ^{THE} Volume III, is designed to facilitate implementation, operation, modification, and extension of the MAGIC System.		

DD FORM 1 NOV 65 1473

UNCLASSIFIED

Security Classification

UNCLASSIFIED

Security Classification

14.	KEY WORDS	LINK A		LINK B		LINK C	
		ROLE	WT	ROLE	WT	ROLE	WT
	1. Structural analysis 2. Matrix methods 3. Matrix abstraction 4. Digital computer methods						

UNCLASSIFIED

Security Classification